Σχολή Θετικών Επιστημών και Τεχνολογίας

Τμήμα Πληροφορικής

Πτυχιακή Εργασία

Σχεδιασμός και Υλοποίηση πληροφοριακού συστήματος διαχείρισης ωρολογίου προγράμματος Πανεπιστημίου

Ιωάννης Δόσκορης

Επιβλέπων καθηγητής: Μηνάς Δασυγένης

Πάτρα, Ιούλιος 2026

ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Σχεδιασμός και υλοποίηση πληροφοριακού συστήματος διαχείρισης ωρολογίου προγράμματος Πανεπιστημίου

Ιωάννης Δόσκορης

Επιτροπή Επίβλεψης Πτυχιακής Εργασίας

Επιβλέπων Καθηγητής:
Δασυγένης Μηνάς
Αναπληρωτής Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, ΠΔΜ

Συν-Επιβλέπων Καθηγητής:
Καραλιόπουλος Μερκούριος
Ερευνητής, Intracom Defense και ΕΚΕΤΑ

Συν-Επιβλέπων Καθηγητής:
Αναγνωστόπουλος Χρήστος-Νικόλαος
Καθηγητής Πανεπιστημίου Αιγαίου
ΣΕΠ, ΘΕ ΠΛΗ-31, ΕΑΠ

Πάτρα, Ιούλιος 2026

Ευχαριστώ τη σύζυγό μου και την κόρη μου, καθώς η υποστήριξη, η υπομονή και η κατανόησή τους αποτέλεσαν καθοριστικό παράγοντα τόσο κατά τη διάρκεια των σπουδών μου όσο και κατά την εκπόνηση της παρούσας πτυχιακής εργασίας. Χωρίς τη στήριξή τους, η ολοκλήρωση αυτής της προσπάθειας δεν θα ήταν εφικτή.

Θα ήθελα επίσης να ευχαριστήσω τη μητέρα μου για τη συνεχή υποστήριξη και ενθάρρυνσή της και ιδιαίτερα τον πατέρα μου, ο οποίος με στήριξε έμπρακτα, βοηθώντας με πολλές φορές στις επαγγελματικές μου υποχρεώσεις κατά τη διάρκεια των σπουδών και της εκπόνησης της εργασίας.

Ευχαριστώ τον επιβλέποντα καθηγητή μου Μηνά Δασυγένη για την καθοδήγηση, τη συνεργασία και τις παρατηρήσεις του κατά την εκπόνηση της εργασίας, καθώς και τους συν-επιβλέποντες καθηγητές Μερκούριο Καραλιόπουλο και Χρήστο Αναγνωστόπουλο για τα σχόλια και τις υποδείξεις τους που συνέβαλαν στη βελτίωση της εργασίας.

Περίληψη

Η παρούσα πτυχιακή εργασία αφορά την ανάλυση, σχεδίαση και ανάπτυξη μίας διαδικτυακής πλατφόρμας διαχείρισης και προβολής ωρολογίου προγράμματος Πανεπιστημίου. Στόχος της εργασίας είναι η δημιουργία ενός ευέλικτου πληροφοριακού συστήματος, το οποίο να υποστηρίζει τη διαχείριση μαθημάτων, διδασκόντων, αιθουσών, ακαδημαϊκών εξαμήνων και προγραμμάτων σπουδών, παρέχοντας παράλληλα εύχρηστη και διαδραστική διεπαφή για διαφορετικές κατηγορίες χρηστών.

Η πλατφόρμα αναπτύχθηκε ως διαδικτυακή εφαρμογή, βασισμένη σε αρχιτεκτονική διαχωρισμού frontend και backend, με χρήση PHP, JavaScript, HTML, CSS και SQL. Ιδιαίτερη έμφαση δόθηκε στην ασφάλεια, στη συντηρησιμότητα του κώδικα, στην επεκτασιμότητα της αρχιτεκτονικής και στη συμμόρφωση με σύγχρονες πρακτικές ανάπτυξης λογισμικού.

Το σύστημα υποστηρίζει τη διαχείριση Σχολών, Τμημάτων, μαθημάτων, διδασκόντων, αιθουσών και ακαδημαϊκών εξαμήνων, καθώς και τη δυναμική δημιουργία και επεξεργασία ωρολογίων προγραμμάτων και ημερολογίων μαθημάτων. Παρέχονται λειτουργίες ειδοποιήσεων, εσωτερικής επικοινωνίας και άμεσης ενημέρωσης των δημοσιευμένων προγραμμάτων σε περιπτώσεις αλλαγών ή αναπληρώσεων μαθημάτων. Επιπλέον, υποστηρίζεται εξαγωγή δεδομένων σε μορφές PDF, Excel και iCalendar, δυνατότητα φιλτραρίσματος και αναζήτησης, καθώς και διασύνδεση με εξωτερικά συστήματα μέσω REST API. Παράλληλα, υλοποιήθηκε υποστηρικτικός μηχανισμός αυτόματης δημιουργίας προτεινόμενων ωρολογίων προγραμμάτων μέσω γενετικού αλγορίθμου.

Για την αξιολόγηση της εφαρμογής πραγματοποιήθηκαν δοκιμές λειτουργικότητας, απόδοσης και στατικής ανάλυσης κώδικα με χρήση εξειδικευμένων εργαλείων. Τα αποτελέσματα έδειξαν ότι η πλατφόρμα παρουσιάζει σταθερή λειτουργία, ικανοποιητική απόδοση και δυνατότητα μελλοντικής επέκτασης.

Λέξεις – Κλειδιά

Πληροφοριακό Σύστημα, Ωρολόγιο Πρόγραμμα, Πανεπιστήμιο, Διαχείριση Μαθημάτων, Γενετικός Αλγόριθμος, Διαδικτυακή Εφαρμογή, PHP, Βάση Δεδομένων, REST API, Προσαρμοστικός Σχεδιασμός Διεπαφής, Βελτιστοποίηση, Ασφάλεια Εφαρμογών

Abstract

This thesis concerns the analysis, design and development of a web-based platform for the management and presentation of university timetables. The objective of the project is the development of a flexible information system capable of supporting the management of courses, instructors, classrooms, academic semesters and study programs, while providing a user-friendly and interactive interface for different categories of users.

The platform was developed as a web application based on frontend / backend separation architecture, using PHP, JavaScript, HTML, CSS and SQL. Particular emphasis was placed on security, code maintainability, architectural extensibility and compliance with modern software development practices.

The system supports the management of Faculties, Departments, courses, professors, classrooms and academic semesters, as well as the dynamic creation and editing of timetables and course calendars. It also provides notification and internal communication features, along with immediate updates of published schedules in cases of course changes or rescheduled lectures. In addition, the platform supports data export in PDF, Excel and iCalendar formats, filtering and search capabilities, as well as integration with external systems through REST APIs. Furthermore, a supportive mechanism for the automatic generation of proposed timetables based on a genetic algorithm was implemented.

For the evaluation of the application, functionality, performance and static code analysis tests were carried out using specialized tools. The results showed that the platform demonstrates stable operation, satisfactory performance and potential for future expansion.

Keywords

Information System, University Timetable Management, Course Management, Genetic Algorithm, Web Application, PHP, Database, REST API, Responsive Web Design, Optimization, Web Application Security

Περιεχόμενα

Κατάλογος Εικόνων / Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες & Ακρωνύμια

2FA	Έλεγχος ταυτότητας δύο παραγόντων – Two Factor Authentication
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface – Διεπαφή Προγραμματισμού Εφαρμογών
CSS	Cascading Style Sheets
CSV	Comma Separated Values
DRY	Don't Repeat Yourself – Αρχή «Μη Επανάληψης Κώδικα»
EAA	European Accessibility Act – Ευρωπαϊκός Κανονισμός Προσβασιμότητας
ECTS	European Credit Transfer and Accumulation System
ERD	Entity-Relationship Diagram – Διάγραμμα Οντοτήτων-Συσχετίσεων
GA	Genetic Algorithm – Γενετικός Αλγόριθμος
GDPR	General Data Protection Regulation – Κανονισμός Προστασίας Δεδομένων
GUI	Graphical User Interface – Διεπαφή Χρήστη
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol – Πρωτόκολλο Μεταφοράς Υπερκειμένου
JSON	JavaScript Object Notation
MS	Microsoft
OOP	Object-Oriented Programming – Αντικειμενοστραφής Προγραμματισμός
PDO	PHP Data Objects
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language – Δομημένη Γλώσσα Ερωτημάτων
SSO	Single-Sign-On – Ενιαία Σύνδεση Χρήστη
SWOT	Strengths, Weaknesses, Opportunities, Threats
UML	Unified Modeling Language
UX	User Experience – Εμπειρία Χρήστη
XSS	Cross-Site Scripting
ΑΕΙ	Ανώτατα Εκπαιδευτικά Ιδρύματα
ΕΑΠ	Ελληνικό Ανοικτό Πανεπιστήμιο
ΔΠΧ	Διάγραμμα Περιπτώσεων Χρήσης
ΠΕ	Πτυχιακή Εργασία
ΠΧ	Περίπτωση Χρήσης – Use Case

1. Εισαγωγή

Στην παρούσα Πτυχιακή Εργασία έχει σχεδιαστεί και αναπτυχθεί μία διαδικτυακή πλατφόρμα διαχείρισης και παρουσίασης του ωρολογίου προγράμματος Πανεπιστημίων. Το σύστημα μπορεί να αξιοποιηθεί τόσο από Ελληνικά, όσο και από Διεθνή Πανεπιστημιακά Ιδρύματα, προσφέροντας στη γραμματεία και στους διδάσκοντες σύγχρονους τρόπους δημιουργίας, ορισμού, διαχείρισης και προβολής του ωρολογίου προγράμματος μαθημάτων εξαμήνου και εξετάσεων κάθε Τμήματος. Παράλληλα, οι φοιτητές έχουν τη δυνατότητα να παρακολουθούν το προσωπικό, εξατομικευμένο πρόγραμμά τους και να ενημερώνονται άμεσα για τυχόν έκτακτες αλλαγές.

1.1 Υφιστάμενη Κατάσταση

Η τριτοβάθμια εκπαίδευση στην Ελλάδα αλλά και διεθνώς, βρίσκεται τα τελευταία χρόνια σε στάδιο εντατικού ψηφιακού μετασχηματισμού, ενώ τα Πανεπιστημιακά Ιδρύματα καλούνται να ανταποκριθούν στις αυξανόμενες απαιτήσεις ψηφιοποίησης, με στόχο τη βελτίωση της οργάνωσης, της διαφάνειας και των παρεχόμενων υπηρεσιών προς τους φοιτητές και το διδακτικό προσωπικό. Σε ευρωπαϊκό επίπεδο, ο Οργανισμός Οικονομικής Συνεργασίας και Ανάπτυξης (OECD), παρουσιάζει αναλυτικά τους δείκτες ψηφιοποίησης των εκπαιδευτικών ιδρυμάτων συμπεριλαμβανομένης και της τριτοβάθμιας εκπαίδευσης, επισημαίνοντας ότι τα ψηφιακά εργαλεία, που αξιοποιούν τα εκπαιδευτικά ιδρύματα, παρέχονται κατά κύριο λόγο μέσω κρατικών υπηρεσιών. Παράλληλα, σε πολλές χώρες υλοποιούνται συστηματικά προγράμματα ψηφιοποίησης και αναβάθμισης των παρεχόμενων ψηφιακών υπηρεσιών [1].

Στην Ελλάδα, το έργο «Ηλεκτρονικό Πανεπιστήμιο» περιλαμβάνει ειδικές λειτουργίες που σχετίζονται με τη διαχείριση ωρολογίων προγραμμάτων, την οργάνωση μαθημάτων και τον χρονικό προγραμματισμό αιθουσών, γεγονός που αναδεικνύει την ανάγκη περαιτέρω αυτοματοποίησης αυτών των διαδικασιών στα ΑΕΙ [2]. Παρά την πρόοδο των τελευταίων ετών μέσω αυτού του έργου, ένα σημαντικό ποσοστό Πανεπιστημιακών Τμημάτων εξακολουθεί να χρησιμοποιεί χειρωνακτικά ή ημι-αυτοματοποιημένα συστήματα, στατικά αρχεία PDF και λογιστικά φύλλα για την οργάνωση και δημοσίευση των ωρολογίων προγραμμάτων.

Κατόπιν ανεπίσημης διερεύνησης και επικοινωνίας με διάφορα Δημόσια Ελληνικά Πανεπιστημιακά Ιδρύματα, διαπιστώνεται ότι μέχρι σήμερα, αρκετές Σχολές και Τμήματα εξακολουθούν να χρησιμοποιούν συμβατικές και μη αυτοματοποιημένες μεθόδους για την εκπόνηση, διαμόρφωση και παρουσίαση του ωρολογίου προγράμματος. Πιο συγκεκριμένα, σε πολλές σχολές το ωρολόγιο πρόγραμμα εκπονείται χειρωνακτικά με χρήση υπολογιστικών φύλλων, όπως Microsoft Excel ή OpenOffice Calc και παρουσιάζεται προς τους φοιτητές μέσω αρχείων PDF, Excel ή Word, με ανάρτησή τους στον επίσημο ιστοχώρο του Πανεπιστημίου ή του Τμήματος [3–6]. Επιπλέον, η διαδικασία δημιουργίας του ωρολογίου προγράμματος βασίζεται συχνά στην προσωπική εμπειρία ή κρίση των διοικητικών και ακαδημαϊκών στελεχών που την αναλαμβάνουν, γεγονός που ενδέχεται να δημιουργεί περιορισμούς στην τυποποίηση, την ευελιξία και την αποτελεσματική διαχείριση αλλαγών. Τα ποιοτικά χαρακτηριστικά, η μεθοδολογία και αναλυτικά στοιχεία παρουσιάζονται στο Παράρτημα Α, όπου τεκμηριώνεται ο τρόπος συλλογής και αξιολόγησης των δεδομένων της διερεύνησης.

Οι εν λόγω πρακτικές οδηγούν σε καθυστερήσεις, δυσκολίες στον έλεγχο συγκρούσεων μαθημάτων, περιορισμένη δυνατότητα εξατομικευμένης πληροφόρησης προς τους φοιτητές, δυσχέρεια στις έκτακτες αλλαγές και στον εντοπισμό ημερολογιακών κενών και διαθέσιμων αιθουσών για αναπλήρωση μαθημάτων, καθώς και σε περιορισμένη δυνατότητα εξαγωγής στατιστικών στοιχείων. Παράλληλα, η έλλειψη αυτοματοποιημένων μηχανισμών επικαιροποίησης έχει ως αποτέλεσμα οι αλλαγές στο πρόγραμμα να μην γνωστοποιούνται άμεσα στους φοιτητές, δημιουργώντας αναστάτωση στη φοιτητική κοινότητα.

Επιπλέον, σύμφωνα με την έκθεση της European University Association, πολλά ευρωπαϊκά ανώτατα εκπαιδευτικά ιδρύματα αντιμετωπίζουν προκλήσεις που σχετίζονται με μη ενοποιημένα ή αποσπασματικά ψηφιακά συστήματα, τα οποία δυσχεραίνουν τη διαλειτουργικότητα, ενώ επηρεάζουν την έγκαιρη και αποτελεσματική ενημέρωση των φοιτητών [7]. Αυτό έχει ως αποτέλεσμα η ενημέρωση για αλλαγές προγραμμάτων σπουδών, ωρολογίων προγραμμάτων και εξετάσεων να πραγματοποιείται συχνά μέσω ανεπίσημων καναλιών, όπως προφορικές ανακοινώσεις ή κοινωνικά δίκτυα, αντί για θεσμικά πληροφοριακά συστήματα.

Σε εθνικό επίπεδο, τα Πανεπιστημιακά Ιδρύματα της χώρας έχουν ψηφιοποιήσει βασικές υπηρεσίες, όπως μητρώα, ηλεκτρονικές εγγραφές φοιτητών και ψηφιακές τάξεις (eClass). Σημαντικές όμως διοικητικές λειτουργίες, όπως η διαμόρφωση ωρολογίων προγραμμάτων

και η κατανομή αιθουσών, εξακολουθούν να εκτελούνται με ημι-αυτοματοποιημένα ή χειρωνακτικά μέσα, γεγονός που επιβαρύνει τις γραμματείες και μειώνει την αποτελεσματικότητα της ακαδημαϊκής διοίκησης.

Η κατάσταση αυτή δημιουργεί πολλαπλά προβλήματα λειτουργικότητας, όπως:

- Αδυναμία γρήγορης και εύκολης αναπροσαρμογής του προγράμματος σε περιπτώσεις έκτακτων αλλαγών.
- Έλλειψη ακριβούς καταγραφής στατιστικών δεδομένων, όπως η κατανομή διδακτικών ωρών ανά διδάσκοντα ή η χρήση αιθουσών.
- Αύξηση διοικητικού φόρτου για τις γραμματείες, που αφιερώνουν σημαντικό χρόνο στη διασταύρωση στοιχείων και στην ενημέρωση φοιτητών.
- Μη προσωποποιημένο ωρολόγιο πρόγραμμα προς τους φοιτητές, γεγονός που δυσχεραίνει την ακαδημαϊκή οργάνωση.

Ανακεφαλαιώνοντας, καθίσταται σαφής η ανάγκη για ένα ενιαίο, αυτοματοποιημένο και παραμετροποιήσιμο πληροφοριακό σύστημα που θα εξυπηρετεί όλες τις κατηγορίες χρηστών (γραμματεία, διδάσκοντες, φοιτητές) και θα υποστηρίζει δυναμικά ωρολόγια προγράμματα και εξεταστικές περιόδους, με λειτουργίες αυτόματης ενημέρωσης, στατιστικής ανάλυσης και εξατομικευμένης πρόσβασης.

1.2 Διατύπωση Προβλήματος

Παρά την πρόοδο στον ψηφιακό μετασχηματισμό της τριτοβάθμιας εκπαίδευσης, η πλειονότητα των Πανεπιστημιακών Ιδρυμάτων συνεχίζει να αντιμετωπίζει σημαντικές δυσκολίες στη διαχείριση των ωρολογίων προγραμμάτων και των προγραμμάτων εξετάσεων. Τα περισσότερα διαθέσιμα συστήματα παρουσιάζουν περιορισμένη αυτοματοποίηση, απουσία αλγοριθμικής επίλυσης συγκρούσεων, έλλειψη εξατομικευμένης πληροφόρησης ανά χρήστη, καθώς και αδυναμία συνδυασμού όλων των απαιτούμενων εργαλείων και υπηρεσιών. Η διαχείριση πραγματοποιείται συχνά με χειρωνακτικές διαδικασίες, γεγονός που οδηγεί σε λάθη, καθυστερήσεις, επαναλαμβανόμενες εργασίες και αναθεωρήσεις. Το βασικό πρόβλημα που αναδεικνύεται είναι η αδυναμία των υφιστάμενων εργαλείων να προσαρμόζονται δυναμικά στις ανάγκες κάθε πανεπιστημιακού ρόλου, προσφέροντας ένα ενοποιημένο περιβάλλον που να υποστηρίζει:

- Αυτόματη δημιουργία και ενημέρωση ωρολογίων προγραμμάτων με βάση κανόνες διαθεσιμότητας αιθουσών, διδασκόντων και αποφυγής συγκρούσεων.
- Παραμετροποίηση ανά φοιτητή ή διδάσκοντα, με δυνατότητα προβολής προσωπικού ωρολογίου προγράμματος.
- Δυνατότητα στατιστικής εποπτείας και εξαγωγής αναφορών για τη λήψη αποφάσεων.
- Διευκόλυνση των αλλαγών και άμεση δημοσίευσή τους με ειδοποιήσεις σε πραγματικό χρόνο.
- Διασύνδεση με άλλες πανεπιστημιακές πλατφόρμες (π.χ. συστήματα μητρώου φοιτητών, eClass, γραμματειακά συστήματα).

Η γραμματεία της εκάστοτε Σχολής δεν έχει άμεση πρόσβαση και δυνατότητα παρακολούθησης στατιστικών στοιχείων σχετικά με τα μαθήματα, τις διδακτικές ώρες ανά καθηγητή και άλλα ενδεχομένως χρήσιμα στοιχεία, χάνοντας ένα σημαντικό μέρος πληροφορίας που αφορά σε βασικές λειτουργίες κάθε Τμήματος και γενικότερα του Πανεπιστημίου. Οι καθηγητές δεν έχουν σε πολλές περιπτώσεις έναν άμεσο και εύχρηστο τρόπο ενημέρωσης της γραμματείας τόσο για τον αρχικό σχεδιασμό του ωρολογίου προγράμματος όσο και για έκτακτες αλλαγές κατά τη διάρκεια του εξαμήνου. Ταυτόχρονα, δεν υπάρχει ενοποιημένη παρουσίαση των ωρολογίων προγραμμάτων όλων των εξαμήνων, ώστε να διευκολύνεται η αναζήτηση κενών ωρών και αιθουσών για περιπτώσεις αναπλήρωσης μαθημάτων.

Περαιτέρω, οι υπάρχουσες λύσεις δεν εκμεταλλεύονται σύγχρονες μεθόδους που διευκολύνουν την εκπόνηση ωρολογίου προγράμματος και προγράμματος εξετάσεων, οι οποίες συνιστούν διεργασίες με υψηλό βαθμό πολυπλοκότητας. Η δημιουργία του ωρολογίου προγράμματος, είναι μια πολυσύνθετη διαδικασία λόγω του αυξημένου αριθμού παραμέτρων, όπως:

α) διαθεσιμότητα αιθουσών,

β) περιορισμοί και προτιμήσεις των διδασκόντων,

γ) διαμοιρασμός αιθουσών και κοινή χρήση από διαφορετικά Τμήματα,

δ) επίσημες αργίες,

ε) συμπλήρωση των ελάχιστων απαιτούμενων ωρών διδασκαλίας ανά μάθημα,

στ) προσθήκη / αφαίρεση / αλλαγή κάποιου εξαμηνιαίου μαθήματος,

ζ) αποφυγή σύγκρουσης μαθημάτων στα οποία δεν επιτρέπεται η χρονική επικάλυψη,

η) αποφυγή χρονικής επικάλυψης διαφορετικών μαθημάτων που διδάσκει ο ίδιος διδάσκων,

θ) συνδυασμός μαθημάτων θεωρίας και εργαστηρίων,

ι) ημερήσια ωράρια έναρξης και λήξης μαθημάτων,

ια) πιθανή διακοπή μαθημάτων κατά τις ώρες σίτισης,

ιβ) ισορροπία κατανομής μαθημάτων στη διάρκεια της εβδομάδας.

Τα παραπάνω καθιστούν σαφές ότι η δημιουργία και η διαχείριση του ωρολογίου προγράμματος συνιστά ένα σύνθετο πρόβλημα βελτιστοποίησης με πολλαπλούς περιορισμούς.

Παρόμοια πολυπλοκότητα και δυσκολία παρατηρείται και κατά την εκπόνηση του προγράμματος εξετάσεων των Τμημάτων με κύριους περιορισμούς, όπως:

α) η διαθεσιμότητα αιθουσών,

β) η χρονική απόσταση ανά εξέταση μαθήματος,

γ) ο συνδυασμός μαθημάτων διαφορικών εξαμήνων,

δ) η ισορροπία κατανομής εξετάσεων σε όλη τη διάρκεια της εξεταστικής περιόδου.

Εν συντομία, οι φοιτητές δεν έχουν πρόσβαση σε εξατομικευμένη και επικαιροποιημένη πληροφορία, οι διδάσκοντες αντιμετωπίζουν δυσκολίες στη διαχείριση του διδακτικού τους φορτίου και οι γραμματείες επιβαρύνονται με χρονοβόρες διαδικασίες, αυξάνοντας την πιθανότητα λαθών.

Το πρόβλημα που καλείται να αντιμετωπίσει η παρούσα Πτυχιακή Εργασία έγκειται στην ανάγκη για ένα ολοκληρωμένο, ευέλικτο, αυτοματοποιημένο και παραμετροποιήσιμο πληροφοριακό σύστημα διαχείρισης ωρολογίων προγραμμάτων και προγραμμάτων εξετάσεων, το οποίο να εξαλείφει τις συγκρούσεις, να ενισχύει την αποδοτικότητα και να προσφέρει εξατομικευμένη εμπειρία χρήσης σε φοιτητές, διδάσκοντες και διοικητικό προσωπικό.

1.3 Στόχος Έργου, Παραδοτέο, Καινοτομίες και Αναμενόμενη Συνεισφορά

Η ανάλυση του προβλήματος καταδεικνύει ότι η υφιστάμενη κατάσταση χαρακτηρίζεται από περιορισμούς, μεμονωμένες και ξεπερασμένες πρακτικές, καθώς και ελλείψεις στη διαχείριση και παρουσίαση των ωρολογίων προγραμμάτων. Για την αντιμετώπιση αυτών των αδυναμιών, η παρούσα εργασία επικεντρώνεται στην ανάπτυξη μιας ενιαίας, ευέλικτης και τεχνολογικά σύγχρονης λύσης. Στις υποενότητες που ακολουθούν παρουσιάζονται: ο στόχος του έργου, το παραδοτέο προϊόν, οι καινοτομίες που το διαφοροποιούν από υπάρχουσες προσεγγίσεις, καθώς και η αναμενόμενη ακαδημαϊκή και πρακτική συνεισφορά του έργου.

1.3.1 Στόχος Έργου

Η παρούσα εργασία στοχεύει να καλύψει τα κενά που εντοπίζονται στην εκπόνηση, διαχείριση και παρουσίαση των ωρολογίων προγραμμάτων των Πανεπιστημίων με την ανάπτυξη μιας ολοκληρωμένης, αυτοματοποιημένης και παραμετροποιήσιμης διαδικτυακής πλατφόρμας με υποστήριξη πολλαπλών ρόλων (φοιτητές, διδάσκοντες, γραμματεία τμήματος). Το σύστημα παρέχει δυναμική δημιουργία και διαχείριση ωρολογίων και εξεταστικών προγραμμάτων, εξατομικευμένη και άμεση ενημέρωση φοιτητών, ευέλικτη διαχείριση από τη γραμματεία και τους διδάσκοντες, υποστηρικτικές δυνατότητες εποπτείας και εξαγωγής αναφορών, καθώς και ενσωμάτωση αλγοριθμικών μεθόδων για βελτιστοποίηση του χρονοπρογραμματισμού. Επιπλέον, η πλατφόρμα σχεδιάστηκε με όρους ευχρηστίας, παρέχοντας φιλική διεπαφή χρήστη και είναι προσβάσιμη μέσω σύγχρονων τεχνολογιών web με προσαρμοστική σχεδίαση (responsive design) για χρήση από συσκευές διαφορετικών διαστάσεων, όπως υπολογιστές, tablets και κινητά. Δίνεται έτσι μια λύση που συνδυάζει τεχνολογικό εκσυγχρονισμό με λειτουργικότητα, προσαρμοσμένη στις ανάγκες της ακαδημαϊκής κοινότητας.

1.3.2 Παραδοτέο Έργου

Στο πλαίσιο του έργου υλοποιήθηκε ένα ολοκληρωμένο, αυτοματοποιημένο, εύχρηστο, ασφαλές και επεκτάσιμο διαδικτυακό σύστημα, για την εκπόνηση, διαχείριση και παρουσίαση του ωρολογίου προγράμματος και του προγράμματος εξετάσεων των Τμημάτων των Πανεπιστημιακών Ιδρυμάτων, με αξιοποίηση σύγχρονων τεχνολογιών.

Το παραδοτέο έργο αποτελείται από μια πλήρως λειτουργική διαδικτυακή πλατφόρμα, η οποία μπορεί να εγκατασταθεί σε οποιονδήποτε σύγχρονο διακομιστή (server) και να λειτουργήσει αυτόνομα, ενώ υποστηρίζει διασύνδεση με υπάρχοντα συστήματα Πανεπιστημίων, βάσει καθορισμένων προδιαγραφών ανταλλαγής δεδομένων και μηχανισμών ενοποιημένης πρόσβασης. Περιλαμβάνει οδηγίες εγκατάστασης, τόσο του διαδικτυακού (web) περιβάλλοντος όσο και της βάσης δεδομένων, ενώ συνοδεύεται από τεκμηρίωση για τη χρήση της πλατφόρμας από τους τελικούς χρήστες, καθώς και για τη διάθεση και αξιοποίηση των παρεχόμενων διεπαφών (APIs). Επιπλέον, στην πλατφόρμα έχουν ενσωματωθεί βασικές λειτουργίες προσβασιμότητας, σύμφωνα με τις αρχές της Ευρωπαϊκής Πράξης για την Προσβασιμότητα (European Accessibility Act – EAA)[1] , όπως δυνατότητα εναλλαγής θέματος εμφάνισης και προσαρμογής της αναγνωσιμότητας, βελτιώνοντας τη χρηστικότητα και διευκολύνοντας την πρόσβαση από ευρύτερο σύνολο χρηστών. Παρέχεται πολυγλωσσική υποστήριξη με κύριες γλώσσες ελληνικά και αγγλικά, ενώ μπορεί να ενσωματωθεί εύκολα βιβλιοθήκη λεξιλογίου και άλλων γλωσσών.

1.3.3 Καινοτομίες

Πέραν της κάλυψης των βασικών απαιτήσεων, έχουν ενσωματωθεί καινοτόμες λειτουργίες με στόχο την ικανοποίηση αναγκών που δεν καλύπτονται επαρκώς από τα υφιστάμενα συστήματα, όπως:

α) στατιστικά στοιχεία μαθημάτων και διδακτικών ωρών καθηγητών,

β) online δήλωση περιορισμών και προτιμήσεων διδασκόντων για το ωρολόγιο πρόγραμμα,

γ) online άμεση τροποποίηση προγράμματος με έκτακτες αλλαγές από τους διδάσκοντες,

δ) αναζήτηση σε πραγματικό χρόνο για διαθέσιμες ώρες και αίθουσες, προκειμένου ο διδάσκων να ορίσει αναπλήρωση μαθήματος,

ε) άμεση ειδοποίηση διαχειριστή κατά τη δημιουργία ωρολογίων προγραμμάτων για συγκρούσεις και παραβιάσεις περιορισμών,

στ) αυτόματη ειδοποίηση και ενημέρωση φοιτητών για κάθε αλλαγή στο πρόγραμμα,

ζ) εξατομικευμένη προβολή ωρολογίου προγράμματος για φοιτητές και διδάσκοντες,

[1] https://www.wcag.com/compliance/european-accessibility-act/

η) αυτόματη δημιουργία προτάσεων ωρολογίου προγράμματος εξαμήνου και εξετάσεων με χρήση γενετικού αλγορίθμου,

θ) ανεξαρτησία και πλήρης παραμετροποίηση βάσει των αναγκών κάθε Ιδρύματος.

1.3.4 Αναμενόμενη Ακαδημαϊκή και Πρακτική Συνεισφορά

Η συμβολή της εργασίας έγκειται στην κάλυψη κενών σε ψηφιακά εργαλεία διαχείρισης πανεπιστημιακών ωρολογίων προγραμμάτων και στη γεφύρωση του χάσματος μεταξύ θεωρητικών μοντέλων χρονοπρογραμματισμού και της πραγματικής εφαρμογής τους στο ελληνικό πανεπιστημιακό περιβάλλον. Συγκεκριμένα, αναμένεται:

- Μείωση του διοικητικού φόρτου για τις γραμματείες τμημάτων κατά σημαντικό ποσοστό.
- Αύξηση της ακρίβειας και της αξιοπιστίας των ωρολογίων προγραμμάτων.
- Βελτίωση της φοιτητικής εμπειρίας και της δυνατότητας ακαδημαϊκού προγραμματισμού.
- Ενίσχυση της διαφάνειας μέσω της καταγραφής και παρακολούθησης των αλλαγών.
- Δημιουργία προτύπου ψηφιακού εργαλείου, το οποίο μπορεί να υιοθετηθεί και να εξελιχθεί περαιτέρω από ακαδημαϊκούς φορείς.

Με την υλοποίηση των ανωτέρω, η εργασία συμβάλλει στην ενίσχυση της αποτελεσματικότητας της ακαδημαϊκής διοίκησης και στη διαμόρφωση ενός σύγχρονου και ευέλικτου περιβάλλοντος μάθησης.

1.4 Μεθοδολογία και Εργαλεία Ανάπτυξης Έργου

1.4.1 Μεθοδολογία Έργου

Για τη σχεδίαση και ανάπτυξη της διαδικτυακής πλατφόρμας εφαρμόστηκε συνδυασμός αντικειμενοστρεφούς και συναρτησιακής προσέγγισης προγραμματισμού. Στο αρχικό στάδιο, ο σχεδιασμός των περιπτώσεων χρήσης (use cases), καθώς και των σχέσεων μεταξύ οντοτήτων – καθώς και η αντίστοιχη σχεδίαση της βάσης δεδομένων – πραγματοποιήθηκε με αντικειμενοστρεφή προσέγγιση, ως καταλληλότερη για αναλυτική μοντελοποίηση και σταδιακή επέκταση κατά την εξέλιξη του έργου. Σε επίπεδο υλοποίησης και συγγραφής κώδικα έγινε συνδυαστική χρήση αντικειμενοστρεφούς και συναρτησιακού προγραμματισμού, όπως υποστηρίζεται από την κύρια γλώσσα που χρησιμοποιήθηκε (PHP).

Για τη διαχείριση του κύκλου ζωής του λογισμικού χρησιμοποιήθηκε το γενικό μοντέλο κύκλου ζωής, με βασικά πρότυπα το σπειροειδές μοντέλο και το μοντέλο καταρράκτη, ενώ για τη διαχείριση απαιτήσεων χρησιμοποιήθηκε η ευέλικτη προσέγγιση (Agile). Το μοντέλο καταρράκτη (Waterfall Model) είναι ένα μοντέλο ανάπτυξης λογισμικού με διαδοχικά στάδια (ανάλυση, σχεδίαση, υλοποίηση, δοκιμή, συντήρηση), όπου κάθε στάδιο ολοκληρώνεται, προτού ξεκινήσει το επόμενο [8]. Το σπειροειδές μοντέλο ανάπτυξης λογισμικού (Spiral Model) προτάθηκε από τον Boehm [9] και συνδυάζει στοιχεία του μοντέλου καταρράκτη και της επαναληπτικής ανάπτυξης. Βασίζεται σε κύκλους εργασιών (σπείρες), όπου κάθε κύκλος περιλαμβάνει καθορισμό στόχων, ανάλυση ρίσκων, ανάπτυξη και αξιολόγηση. Έτσι, παρέχει ευελιξία για αναθεωρήσεις και προσαρμογές κατά τη διάρκεια της ανάπτυξης, διατηρώντας παράλληλα τον έλεγχο στην ποιότητα και διαχείριση κινδύνων [8,10,11]. Η υιοθέτηση του γενικού μοντέλου προσέφερε προσαρμοστικότητα στις ανάγκες του έργου. Το μοντέλο καταρράκτη χρησιμοποιήθηκε στη γενική ανάλυση και παρουσίαση του έργου ως μέρος Πτυχιακής Εργασίας, όπου απαιτείται η τήρηση συγκεκριμένων σταδίων ανάπτυξης. Η χρήση του σπειροειδούς μοντέλου ήταν επιτακτική, αφού οι απαιτήσεις του συστήματος δεν είναι μοναδικές και συγκεκριμένες, λόγω των ιδιαζόντων περιπτώσεων ανά Πανεπιστήμιο και η ανάπτυξη απαιτούσε αρκετούς ελέγχους και αναθεωρήσεις.

Για τη λειτουργία του αυτόματου χρονοπρογραμματισμού χρησιμοποιήθηκε η μέθοδος αναζήτησης βέλτιστων λύσεων μέσω γενετικού αλγορίθμου (GA), καθώς προσφέρει ευελιξία στη διαχείριση και βελτιστοποίηση πολλαπλών περιορισμών. Η μέθοδος παρουσιάζεται αναλυτικά στην Ενότητα 2.9.

Βήματα Μεθοδολογικής Προσέγγισης:

1) Ανάλυση απαιτήσεων ρόλων: επισκέπτες, φοιτητές, διδάσκοντες διαχειριστές.
2) Καθορισμός λειτουργικών και μη λειτουργικών προδιαγραφών.
3) Μοντελοποίηση συστήματος μέσω UML διαγραμμάτων (Use Cases, User Stories) και ERD διαγράμματος για τη βάση δεδομένων.
4) Επαναληπτική ανάπτυξη (Iterative Development) με προτεραιότητα στις βασικές λειτουργίες του συστήματος, όπως: ασφάλεια, πρότυπο συγγραφής, ρύθμιση παραμέτρων, διαχείριση ρόλων, ειδοποιήσεις, διεπαφή χρήστη, έλεγχοι περιορισμών, αυτόματος χρονοπρογραμματισμός.
5) Έλεγχος και αξιολόγηση (testing) με βάσει σενάρια χρήσης και κριτήρια αποδοχής (acceptance criteria).
6) Τελική αξιολόγηση της λειτουργικότητας και τεκμηρίωση συμπερασμάτων.

1.4.2 Εργαλεία Ανάπτυξης Έργου

Για την ανάπτυξη της πλατφόρμας αξιοποιήθηκε ένα σύνολο εργαλείων και τεχνολογιών, τα οποία συνέβαλαν τόσο στην υλοποίηση όσο και στην τεκμηρίωση του συστήματος.

Γλώσσες και τεχνολογίες

Η βασική γλώσσα προγραμματισμού ήταν η PHP 8.2, ενώ για τις διεπαφές χρήστη χρησιμοποιήθηκαν HTML5, CSS3, JavaScript και JSON. Η οπτική παρουσίαση επιτεύχθηκε με προσαρμοσμένη (custom) μορφοποίηση μέσω CSS3, σε συνδυασμό με το πλαίσιο (framework) Bootstrap 5 (CSS και JavaScript bundle/minified) για προσαρμοστική διεπαφή (responsive interface), διασφαλίζοντας τη σωστή προβολή σε κινητές και επιτραπέζιες συσκευές. Για την αποθήκευση και διαχείριση δεδομένων χρησιμοποιήθηκαν οι βάσεις δεδομένων MySQL και MariaDB [12–20].

Περιβάλλον ανάπτυξης

Η ανάπτυξη κώδικα πραγματοποιήθηκε στο Visual Studio Code, ενώ για απλές επεξεργασίες αρχείων κειμένου, JSON και λοιπών βοηθητικών αρχείων χρησιμοποιήθηκε και το Notepad++. Για τη σχεδίαση και ανάπτυξη της Βάσης Δεδομένων χρησιμοποιήθηκε το εργαλείο phpMyAdmin [21–23].

Διαχείριση αρχείων και έκδοσης

Για την αποθήκευση και τον συγχρονισμό αρχείων τεκμηρίωσης και συνοδευτικού υλικού (όπως αρχεία MS Word, ZIP κ.ά.) χρησιμοποιήθηκε το Microsoft OneDrive. Η διαχείριση και ο έλεγχος εκδόσεων του πηγαίου κώδικα πραγματοποιήθηκαν μέσω του GitHub [24,25], ενώ αναπτύχθηκε και ένα προσαρμοσμένο (custom) σύστημα για εσωτερικούς ελέγχους κώδικα και αναζητήσεις εκδόσεων.

Δοκιμές και λειτουργία

Η πλατφόρμα δοκιμάστηκε σε περιβάλλον IIS Express, καθώς και σε Apache / XAMPP, ώστε να επιτευχθεί συμβατότητα με διαφορετικούς διακομιστές [26–28]. Για τον έλεγχο ποιότητας του κώδικα εφαρμόστηκαν διαδικασίες αποσφαλμάτωσης, κανονικοποίησης και ασφάλειας με τη χρήση εργαλείων στατικής ανάλυσης κώδικα, όπως PHPStan, PHP CodeSniffer (phpcs), PHP Mess Detector (PHPMD), PHP Metrics και Psalm (βλ. υποενότητα 2.6.2), με στόχο την τήρηση βέλτιστων πρακτικών και προτύπων προγραμματισμού. Επιπλέον, για τον έλεγχο απόδοσης και βιωσιμότητας χρησιμοποιήθηκε το εργαλείο Apache JMeter (π.χ. προσομοίωση λειτουργίας με 50+ ταυτόχρονους χρήστες, συνθετικό φορτίο) (βλ. υποενότητα 2.8.5). Τα αποτελέσματα των ελέγχων της τελευταίας έκδοσης της πλατφόρμας παρουσιάζονται στην ενότητα 5.2 του κεφαλαίου 5.

Σχεδίαση και τεκμηρίωση

Ο σχεδιασμός και η ανάλυση του συστήματος τεκμηριώθηκαν με διαγράμματα UML και ERD, τα οποία αναπτύχθηκαν χρησιμοποιώντας το εργαλείο Visual Paradigm [29]. Ο κώδικας (source code) περιλαμβάνει πλήρες και αναλυτικό συνοδευτικό τεκμηρίωσης (PHPDoc), ενώ στο Παράρτημα Β περιλαμβάνονται οι οδηγίες εγκατάστασης της πλατφόρμας. Το έργο συνοδεύεται από οδηγό χρήσης σε μορφή PDF.

1.5 Σχετική Έρευνα και Παρόμοια Συστήματα

Η σχεδίαση και η ανάπτυξη ενός πληροφοριακού συστήματος για τη διαχείριση και παρουσίαση του ωρολογίου προγράμματος Πανεπιστημίου δεν αποτελεί καινούρια ερευνητική πρόκληση. Στην Ελλάδα και κυρίως στο εξωτερικό, έχουν προταθεί και υλοποιηθεί λύσεις που κινούνται από απλούς αλγόριθμους χρονοπρογραμματισμού, έως και πλήρη συστήματα με πολυεπίπεδη λειτουργικότητα.

1.5.1 Πανεπιστημιακές Εργασίες και έργα στην Ελλάδα

Βάσει βιβλιογραφικής ανασκόπησης έχουν παρουσιαστεί αρκετές προπτυχιακές και μεταπτυχιακές εργασίες με αντικείμενο τον χρονοπρογραμματισμό. Ενδεικτικά αναφέρονται:

- στο Πανεπιστήμιο Δυτικής Μακεδονίας, το 2012 υλοποιήθηκε πληροφοριακό σύστημα για το ωρολόγιο πρόγραμμα με χρήση ευρετικών αλγορίθμων σε C++, με έμφαση στις συγκρούσεις μαθημάτων και στις αίθουσες [30].
- στο Πανεπιστήμιο Στερεάς Ελλάδας, το 2012 αναπτύχθηκε εργαλείο αυτόματης δημιουργίας ωρολογίων προγραμμάτων με χρήση γενετικών αλγορίθμων, παράλληλα με χρήση του λογισμικού FET [31].

Οι παραπάνω εργασίες εστιάζουν κυρίως στον αλγοριθμικό πυρήνα του χρονοπρογραμματισμού. Αντίθετα, η παρούσα εργασία προσεγγίζει το πρόβλημα συνολικά, μέσω μίας ολοκληρωμένης διαδικτυακής πλατφόρμας διαχείρισης και προβολής ακαδημαϊκού ωρολογίου προγράμματος. Πιο συγκεκριμένα, το σύστημα υποστηρίζει:

- διακριτούς ρόλους χρηστών,
- δυναμική διαχείριση μαθημάτων, αιθουσών και διδασκόντων,
- άμεσες τροποποιήσεις και ενημερώσεις προγράμματος,
- online υποβολή περιορισμών και διαθεσιμοτήτων,
- προσωποποιημένη προβολή προγράμματος,
- εξαγωγή αναφορών και στατιστικών στοιχείων,
- καθώς και δυνατότητα ενσωμάτωσης αλγοριθμικών τεχνικών βελτιστοποίησης.

Ενδεικτικές πρόσφατες εργασίες που προσεγγίζουν το υπό μελέτη σύστημα είναι:

- στο Χαροκόπειο Πανεπιστήμιο αναπτύχθηκε το 2021 διαδικτυακή εφαρμογή ορισμού ωρολογίου προγράμματος από πολλούς χρήστες και με ορισμό συγκεκριμένων περιορισμών [32].
- στο Πανεπιστήμιο Δυτικής Μακεδονίας υλοποιήθηκε το 2025 διαδικτυακός ιστότοπος διαχείρισης προγράμματος Πανεπιστημίου με ρόλους διαχειριστή, γραμματείας, διδασκόντων και φοιτητών, αξιοποιώντας PHP, HTML, CSS και MySQL [33].

Παρότι οι παραπάνω εφαρμογές καλύπτουν επιμέρους ανάγκες διαχείρισης και προβολής προγράμματος, η παρούσα εργασία διαφοροποιείται ως προς τον συνδυασμό λειτουργιών που παρέχει σε ενιαίο περιβάλλον, δίνοντας έμφαση στην ευελιξία διαχείρισης, στην άμεση ενημέρωση των χρηστών, στη δυνατότητα συνεχών τροποποιήσεων και στην επεκτασιμότητα της πλατφόρμας.

Επιπλέον, στην ελληνική αγορά διατίθεται το λογισμικό «ΩΡΟΛΟΓΙΟ» της ANTINOOS Software Applications. Πρόκειται για ένα εργαλείο που έχει αναπτυχθεί ειδικά για τη σύνταξη ωρολογίων προγραμμάτων – σύμφωνα με την εταιρική παρουσίαση, επιτρέπει τη γρήγορη και αυτόματη δημιουργία ωρολογίων προγραμμάτων διδασκόντων [34]. Η συγκεκριμένη εφαρμογή εστιάζει κυρίως στη διαδικασία του χρονοπρογραμματισμού και λειτουργεί ως εφαρμογή τοπικής εγκατάστασης (desktop). Αντίθετα, το σύστημα που αναπτύσσεται στην παρούσα εργασία προσεγγίζει το πρόβλημα ως μία ολοκληρωμένη διαδικτυακή πλατφόρμα, με δυνατότητες διαχείρισης διαφορετικών κατηγοριών χρηστών, άμεσης ενημέρωσης και υποστήριξης δυναμικών αλλαγών σε ενιαίο περιβάλλον.

Παράλληλα, σύγχρονες διαδικτυακές εφαρμογές που χρησιμοποιούνται σε ελληνικά Πανεπιστήμια, όπως η πλατφόρμα παρουσίασης ωρολογίου προγράμματος του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης[2], παρέχουν δημόσια προβολή των προγραμμάτων μαθημάτων μέσω διαδικτυακής διεπαφής. Οι εφαρμογές αυτές εστιάζουν κυρίως στην παρουσίαση και αναζήτηση του προγράμματος προς τους τελικούς χρήστες, χωρίς να ενσωματώνουν εκτεταμένες δυνατότητες διαχείρισης χρηστών, δυναμικών αλλαγών ή αυτοματοποίησης του χρονοπρογραμματισμού, όπως επιχειρεί το υπό μελέτη σύστημα.

[2] https://courses.auth.gr/schedule/

1.5.2 Διεθνή συστήματα και ανοιχτό λογισμικό

Σε διεθνές επίπεδο, το πρόβλημα του πανεπιστημιακού χρονοπρογραμματισμού έχει τυποποιηθεί σε σενάρια όπως το *Post-Enrollment Course Timetabling* (PE-CTT) και το *Curriculum-Based Course Timetabling* (CB-CTT). Τα δύο αυτά σενάρια χρησιμοποιήθηκαν ως επίσημες κατηγορίες προβλημάτων στο International Timetabling Competition 2007 (ITC-2007) [35], και αποτελούν καθιερωμένα σύνολα αναφοράς στη σχετική βιβλιογραφία (PATAT Conference) [36]. Σε αυτό το πλαίσιο έχουν αναπτυχθεί:

- UniTime: ολοκληρωμένη πλατφόρμα ανοικτού κώδικα σε Java/J2EE, που υποστηρίζει ωρολόγια προγράμματα μαθημάτων, εξετάσεων και κατανομές φοιτητών. Χρησιμοποιείται από πολλά Πανεπιστήμια διεθνώς και έχει αποσπάσει διακρίσεις στον διαγωνισμό ITC [37].
- FET: ελεύθερο λογισμικό για αυτόματη δημιουργία ωρολογίων προγραμμάτων με δυνατότητες εισαγωγής/εξαγωγής δεδομένων σε XML/CSV και δημοσίευσης σε HTML. Αν και αρκετά διαδεδομένο σε σχολεία και Πανεπιστήμια, εστιάζει περισσότερο στον αλγόριθμο και λιγότερο στη διαχείριση χρηστών ρόλων [38].
- CELCAT Timetabler: εμπορική λύση με έμφαση στην πολυκαναλική διάθεση (π.χ. Outlook, iCal, SMS) και στη διασύνδεση με άλλα πληροφοριακά συστήματα. Η λειτουργικότητα της προσεγγίζει τη φιλοσοφία της παρούσας πλατφόρμας, αλλά με κλειστό κώδικα και υψηλό κόστος αδειοδότησης [39].
- Wise Timetable (wtimetable.com): Εμπορικό λογισμικό για ακαδημαϊκά ιδρύματα, με δυνατότητα αυτόματης δημιουργίας προγράμματος βάσει περιορισμών, συνδυασμό αυτόματης και χειρωνακτικής δημιουργίας, διαχείριση διαθεσιμότητας διδασκόντων και αιθουσών, καθώς και εργαλεία δημοσίευσης του προγράμματος σε web και mobile εφαρμογές [40].

Σε αντίθεση με το σύστημα που αναπτύσσεται στην παρούσα εργασία, τα παραπάνω συστήματα δεν καλύπτουν με τον ίδιο συνδυαστικό τρόπο το σύνολο των λειτουργιών που επιδιώκει να προσφέρει η προτεινόμενη πλατφόρμα, καθώς είτε εστιάζουν κυρίως στην επίλυση του προβλήματος του χρονοπρογραμματισμού είτε αποτελούν τοπικής χρήσης (desktop) εφαρμογές με περιορισμένη προσβασιμότητα και επεκτασιμότητα, ενώ τα τελευταία έτη παρατηρείται στροφή προς web-based και cloud πλατφόρμες με λειτουργίες διαχείρισης χρηστών και διασύνδεσης με άλλα πληροφοριακά συστήματα.

1.5.3 Επιστημονικές μέθοδοι, αλγόριθμοι και διεπαφή χρήστη

Βιβλιογραφικά, έχουν προταθεί ποικίλες μέθοδοι επίλυσης του προβλήματος χρονοπρογραμματισμού, όπως γενετικοί αλγόριθμοι (Genetic Algorithms – GA) [41], ευρετικές μέθοδοι, όπως η Tabu Search [42], τεχνικές Variable Neighborhood Search [43], καθώς και προσεγγίσεις μικτού ακέραιου προγραμματισμού (Mixed-Integer Programming – MIP) [44]. Οι προσεγγίσεις αυτές εφαρμόζονται συχνά σε εξειδικευμένα συστήματα «επιλυτών» (solvers), τα οποία επικεντρώνονται κυρίως στη βελτιστοποίηση του χρονοπρογραμματισμού.

Παράλληλα, στη βιβλιογραφία συναντώνται προσεγγίσεις που επιχειρούν να συνδυάσουν αλγορίθμους επίλυσης με διαδικτυακές πλατφόρμες διαχείρισης, ώστε ο αλγόριθμος να αποτελεί υποστηρικτικό εργαλείο και όχι τον κύριο άξονα του συστήματος (π.χ. UniTime). Σχετικές εργασίες σε ελληνικά ΑΕΙ, όπως αυτές που παρουσιάζονται στην υποενότητα 1.5.1, έχουν αναπτύξει web εφαρμογές που υποστηρίζουν βασικούς ρόλους χρηστών, καταχώριση μαθημάτων, διαχείριση αιθουσών και δημοσίευση ωρολογίων προγραμμάτων.

Παρότι χρήσιμες, οι εφαρμογές αυτές δεν ενσωματώνουν προηγμένους αυτοματισμούς ούτε λειτουργίες άμεσης ενημέρωσης, δυναμικής τροποποίησης ή online υποβολής περιορισμών από τους διδάσκοντες. Επιπλέον, τέτοιου είδους εφαρμογές συχνά αναπτύσσονται με βάση τις ανάγκες συγκεκριμένων Ιδρυμάτων ή Τμημάτων, γεγονός που περιορίζει τη γενικευμένη αξιοποίησή τους και την επεκτασιμότητά τους σε ευρύτερο ακαδημαϊκό περιβάλλον.

Σε αντίθεση με τα παραπάνω, η παρούσα εργασία δεν στοχεύει στη δημιουργία ενός ακόμη solver ή μίας διαχειριστικής πλατφόρμας, αλλά στην ανάπτυξη ενός πλήρους, web-based συστήματος διαχείρισης ωρολογίων προγραμμάτων. Στόχος του είναι να προσφέρει αυτοματοποίηση διαδικασιών, υποβολή και επεξεργασία περιορισμών από τους διδάσκοντες, εύκολη πραγματοποίηση έκτακτων αλλαγών από γραμματεία και καθηγητές, άμεση παροχή εξατομικευμένων προγραμμάτων προς τους φοιτητές και δυνατότητα διασύνδεσης με υφιστάμενα πανεπιστημιακά συστήματα. Ο αλγοριθμικός μηχανισμός (GA) λειτουργεί υποστηρικτικά μέσα σε αυτό το περιβάλλον, ως εργαλείο βελτιστοποίησης και όχι ως αυτοτελής επιστημονικός πυρήνας.

1.5.4 Συνοπτική αποτίμηση

Κατά την αποτίμηση των σχετικών εργασιών εντοπίστηκαν και προσεγγίσεις που αξιοποιούν Γενετικούς Αλγορίθμους για το πρόβλημα δημιουργίας ωρολογίου προγράμματος Πανεπιστημίου, κυρίως μέσω συναρτήσεων καταλληλότητας με σκληρούς και μαλακούς περιορισμούς, καθώς και τεχνικών επιλογής, διασταύρωσης και μετάλλαξης χρωμοσωμάτων. Επιπλέον, στη διεθνή βιβλιογραφία συναντώνται και web-based εφαρμογές που ενσωματώνουν Γενετικούς Αλγορίθμους ως υποστηρικτικό μηχανισμό δημιουργίας προγράμματος [45,46].

Η παρούσα εργασία αξιοποιεί αντίστοιχες αλγοριθμικές αρχές ως προαιρετικό υποστηρικτικό εργαλείο μέσα σε μία ευρύτερη διαδικτυακή πλατφόρμα διαχείρισης ωρολογίων προγραμμάτων και όχι ως αυτόνομο σύστημα επιλυτή. Ο επιλυτής λειτουργεί μέσα από τη διαδικτυακή πλατφόρμα, δημιουργεί προτάσεις προγράμματος για όλα τα εξάμηνα της επιλεγμένης περιόδου, επιτρέπει την προσωρινή αποθήκευση και επεξεργασία των λύσεων με μορφή αντίστοιχη του κανονικού προγράμματος, ενώ υποστηρίζει και διαδικασία μερικής επίλυσης, συμπληρώνοντας τμήματα του προγράμματος που δεν έχουν ακόμη οριστεί.

Συνολικά, η ανασκόπηση δείχνει ότι, ενώ υπάρχουν εργαλεία και έρευνες για τη διαχείριση ωρολογίων προγραμμάτων Πανεπιστημίων και τον χρονοπρογραμματισμό, δεν εντοπίστηκε στη βιβλιογραφική ανασκόπηση ένα ολοκληρωμένο σύστημα που να συνδυάζει:

1) εξατομικευμένα ωρολόγια προγράμματα ανά φοιτητή,
2) παραμετροποίηση για κάθε Τμήμα,
3) αυτόματες ειδοποιήσεις και online δηλώσεις προτιμήσεων και αλλαγών,
4) υποστήριξη διασύνδεσης με υπάρχοντα διαχειριστικά συστήματα Πανεπιστημίων,
5) δυνατότητα εξαγωγής δεδομένων προγραμμάτων.

Η προσέγγιση αυτή διαφοροποιεί το προτεινόμενο σύστημα από τις υπάρχουσες λύσεις, καθώς δίνει έμφαση όχι μόνο στον χρονοπρογραμματισμό, αλλά και στη συνολική διαχείριση, παρουσίαση και ευέλικτη προσαρμογή του προγράμματος σε ένα ενιαίο διαδικτυακό περιβάλλον, με υποστηρικτική αξιοποίηση αλγοριθμικών μεθόδων όπου απαιτείται.

1.6 Δομή Εργασίας

Η παρούσα Πτυχιακή Εργασία ακολουθεί οργανωμένη ακαδημαϊκή δομή, προκειμένου να παρουσιαστεί με σαφήνεια η θεωρητική θεμελίωση, η ανάλυση απαιτήσεων και η υλοποίηση του συστήματος. Η διάρθρωση της εργασίας είναι η εξής:

- **Κεφάλαιο 1**: Εισαγωγή: Παρουσιάζεται η γενική περιγραφή της πλατφόρμας, η υφιστάμενη κατάσταση, το πρόβλημα, ο σκοπός, οι στόχοι της εργασίας, η μεθοδολογία και τα εργαλεία που χρησιμοποιήθηκαν, οι σχετικές έρευνες και εφαρμογές, τα προβλήματα στον τομέα του χρονοπρογραμματισμού στα Πανεπιστήμια και οι λύσεις που προτείνει η πλατφόρμα και η αναμενόμενη συνεισφορά.
- **Κεφάλαιο 2**: Θεωρητικό Υπόβαθρο: Παρουσιάζεται το θεωρητικό υπόβαθρο της εργασίας, οι τεχνολογίες και τα εργαλεία ανάπτυξης της διαδικτυακής πλατφόρμας, τα πρότυπα και οι μεθοδολογίες που αξιοποιήθηκαν, ενώ αναλύονται οι βασικές αρχές που διέπουν τον σχεδιασμό και την υλοποίηση διαδικτυακών εφαρμογών.
- **Κεφάλαιο 3**: Προδιαγραφές Συστήματος: Περιγράφονται οι λειτουργικές και μη λειτουργικές απαιτήσεις, οι ρόλοι χρηστών και οι προδιαγραφές του συστήματος, οι ιστορίες χρηστών (user stories), τα σενάρια περιπτώσεων χρήσης (use case), το διάγραμμα οντοτήτων-συσχετίσεων (ERD) και το διάγραμμα περιπτώσεων χρήσης (UML).
- **Κεφάλαιο 4**: Σχεδίαση και Υλοποίηση: Παρουσιάζεται η αρχιτεκτονική του συστήματος, η βάση δεδομένων, η διεπαφή χρήστη και η υλοποίηση των βασικών λειτουργιών.
- **Κεφάλαιο 5**: Αξιολόγηση και Αποτελέσματα: Παρουσιάζονται τα αποτελέσματα των δοκιμών, η αξιολόγηση του συστήματος και η σύγκρισή του με υφιστάμενες λύσεις.
- **Κεφάλαιο 6**: Συμπεράσματα και Προτάσεις για Μελλοντικές Επεκτάσεις: Συνοψίζονται τα ευρήματα και προτείνονται πιθανές επεκτάσεις και μελλοντικές κατευθύνσεις.

2. Θεωρητικό υπόβαθρο

Στο κεφάλαιο αυτό παρουσιάζεται το θεωρητικό πλαίσιο και οι βασικές αρχές που υποστηρίζουν την ανάπτυξη της διαδικτυακής πλατφόρμας. Εξετάζονται οι τεχνολογίες, τα εργαλεία και τα πρότυπα που αξιοποιήθηκαν για τη σχεδίαση και την υλοποίηση του συστήματος, καθώς και οι θεωρητικές προσεγγίσεις που καθοδήγησαν τον σχεδιασμό της αρχιτεκτονικής και της λειτουργικότητας του λογισμικού.

Επιπλέον, γίνεται αναφορά στις έννοιες και τις τεχνικές που σχετίζονται με την ανάπτυξη διαδικτυακών εφαρμογών, τη δομή των βάσεων δεδομένων, την ασφάλεια πληροφοριών και τη διαχείριση χρηστών, ώστε να τεκμηριωθεί επιστημονικά η επιλογή των χρησιμοποιούμενων τεχνολογιών.

2.1 Αρχιτεκτονική Web Εφαρμογών

2.1.1 Διαδίκτυο και αρχιτεκτονική client–server

Το Διαδίκτυο αποτελεί ένα παγκόσμιο, κατανεμημένο σύστημα διασύνδεσης υπολογιστικών δικτύων, το οποίο βασίζεται στα πρωτόκολλα TCP/IP και επιτρέπει την ανταλλαγή δεδομένων μεταξύ ετερογενών συστημάτων. Η λειτουργία του δεν περιορίζεται σε μία ενιαία υποδομή, αλλά υλοποιείται μέσω ανεξάρτητων δικτύων που συνεργάζονται με κοινά, τυποποιημένα πρωτόκολλα, εξασφαλίζοντας τη διαλειτουργικότητα [47].

Στο πλαίσιο των εφαρμογών Παγκόσμιου Ιστού (World Wide Web), κυρίαρχο μοντέλο επικοινωνίας αποτελεί η αρχιτεκτονική client–server, σύμφωνα με το οποίο ο client – συνήθως ένας φυλλομετρητής ιστού, γνωστός ως web browser – αποστέλλει αιτήματα προς έναν διακομιστή (server), ο οποίος επεξεργάζεται τα αιτήματα και επιστρέφει τις αντίστοιχες απαντήσεις. Ο διαχωρισμός αυτός επιτρέπει τη συγκέντρωση της επιχειρησιακής λογικής και των δεδομένων στον διακομιστή, ενώ ο client περιορίζεται στην παρουσίαση και στην αλληλεπίδραση με τον χρήστη [47].

Ο φυλλομετρητής ιστού (web browser) λειτουργεί ως το λογισμικό διεπαφής του χρήστη με τον Ιστό. Είναι υπεύθυνος για την αποστολή HTTP αιτημάτων (HyperText Transfer Protocol – Πρωτόκολλο Μεταφοράς Υπερκειμένου), την ερμηνεία των απαντήσεων και την απόδοση του περιεχομένου (HTML, CSS, JavaScript). Αντίστοιχα, ο web server διαχειρίζεται τα εισερχόμενα αιτήματα, εκτελεί server-side κώδικα όπου απαιτείται και επιστρέφει το

παραγόμενο περιεχόμενο και τα δεδομένα [47,48]. Η παρούσα πλατφόρμα εντάσσεται πλήρως στο μοντέλο αυτό, καθώς υλοποιείται ως διαδικτυακή εφαρμογή με σαφή διαχωρισμό ρόλων μεταξύ πελάτη (client) και διακομιστή (server), αξιοποιώντας τις δυνατότητες του Διαδικτύου για απομακρυσμένη πρόσβαση και ταυτόχρονη χρήση από πολλαπλούς χρήστες.

2.1.2 Ιστοσελίδες, Ιστότοποι και Web Εφαρμογές

Μία ιστοσελίδα αποτελεί ένα μεμονωμένο έγγραφο του Παγκόσμιου Ιστού, συνήθως γραμμένο σε HTML και προσβάσιμο μέσω μίας μοναδικής διεύθυνσης (URL). Αντίθετα, ένας ιστότοπος (website) είναι ένα οργανωμένο σύνολο ιστοσελίδων που συνδέονται μεταξύ τους και εξυπηρετούν έναν κοινό σκοπό ή μία θεματική ενότητα [47]. Η διάκριση αυτή είναι ουσιώδης, καθώς ο σχεδιασμός και η υλοποίηση ενός ιστοτόπου προϋποθέτουν συνολική αρχιτεκτονική και όχι απλώς μεμονωμένες σελίδες.

Οι ιστότοποι μπορούν να κατηγοριοποιηθούν σε στατικούς και δυναμικούς. Στους στατικούς ιστότοπους το περιεχόμενο παραμένει αμετάβλητο και παραδίδεται στον χρήστη χωρίς επεξεργασία στον server. Αντίθετα, οι δυναμικοί ιστότοποι παράγουν περιεχόμενο κατά την εκτέλεση, συνήθως με δεδομένα από βάσεις δεδομένων και παραμέτρων εισόδου, επιτρέποντας εξατομίκευση και σύνθετη αλληλεπίδραση του χρήστη [47].

Σε πιο προχωρημένο επίπεδο, οι web εφαρμογές αποτελούν δυναμικούς ιστότοπους με αυξημένη λειτουργικότητα, οι οποίοι προσεγγίζουν τη συμπεριφορά αυτόνομων πληροφοριακών συστημάτων. Τέτοιες εφαρμογές υποστηρίζουν αυθεντικοποίηση χρηστών, διαχείριση δεδομένων, έλεγχο πρόσβασης και πολύπλοκες ροές εργασίας [47]. Η πλατφόρμα που αναπτύχθηκε στο πλαίσιο της παρούσας πτυχιακής εργασίας κατατάσσεται σε αυτή την κατηγορία, καθώς παρέχει ολοκληρωμένες λειτουργίες διαχείρισης και όχι απλή παρουσίαση πληροφορίας.

2.1.3 Πρωτόκολλο HTTP/HTTPS και ασφαλής επικοινωνία

Η επικοινωνία μεταξύ client και server στον Παγκόσμιο Ιστό βασίζεται στο Πρωτόκολλο Μεταφοράς Υπερκειμένου (HTTP). Το HTTP ορίζει τη μορφή και τον τρόπο ανταλλαγής μηνυμάτων μέσω ενός μηχανισμού αιτήματος – απάντησης (request – response). Κάθε

αίτημα περιλαμβάνει πληροφορίες όπως τη μέθοδο (π.χ. GET, POST), τις απαραίτητες επικεφαλίδες (headers) και προαιρετικά σώμα δεδομένων, ενώ η απάντηση του server περιέχει κωδικό κατάστασης και το αντίστοιχο περιεχόμενο. Οι κωδικοί κατάστασης απόκρισης HTTP (HTTP response status codes) υποδεικνύουν εάν ένα συγκεκριμένο αίτημα HTTP έχει ολοκληρωθεί επιτυχώς και ομαδοποιούνται στις παρακάτω πέντε κατηγορίες:

- 1xx: Πληροφοριακές αποκρίσεις
- 2xx: Επιτυχείς αποκρίσεις
- 3xx: Ανακατευθύνσεις
- 4xx: Σφάλματα πελάτη
- 5xx: Σφάλματα διακομιστή

Βασικό χαρακτηριστικό του HTTP είναι η stateless φύση του. Κάθε αίτημα αντιμετωπίζεται ανεξάρτητα, χωρίς το πρωτόκολλο να διατηρεί από μόνο του πληροφορία κατάστασης μεταξύ διαδοχικών αιτημάτων. Η ιδιότητα αυτή απλοποιεί τον σχεδιασμό και διευκολύνει την αποδοτική εξυπηρέτηση μεγάλου αριθμού αιτημάτων, καθιστώντας όμως αναγκαία τη χρήση συμπληρωματικών μηχανισμών για τη διατήρηση της κατάστασης χρήστη, όπως τα cookies και οι συνεδρίες (sessions) [48].

Για την προστασία της ακεραιότητας των δεδομένων, καθώς και την εμπιστευτικότητα, το HTTP χρησιμοποιείται στην ασφαλή του εκδοχή, το HTTPS. Το HTTPS βασίζεται στο πρωτόκολλο TLS (Transport Layer Security), το οποίο παρέχει κρυπτογράφηση της επικοινωνίας και μηχανισμούς πιστοποίησης του εξυπηρετητή. Ο όρος SSL συναντάται συχνά στη βιβλιογραφία και στην πρακτική, ωστόσο αναφέρεται σε παλαιότερες, πλέον καταργημένες εκδόσεις, καθώς στις σύγχρονες εφαρμογές χρησιμοποιείται αποκλειστικά το TLS [49].

Οι επικεφαλίδες (headers), τα cookies και οι ανακατευθύνσεις (redirects) αποτελούν θεμελιώδη στοιχεία της επικοινωνίας στον Ιστό. Τα headers μεταφέρουν μεταδεδομένα που καθορίζουν τον τρόπο επεξεργασίας των αιτημάτων και των απαντήσεων, τα cookies επιτρέπουν την αποθήκευση μικρών τμημάτων πληροφορίας στον client, ενώ οι ανακατευθύνσεις χρησιμοποιούνται για τη δρομολόγηση της πλοήγησης και της ροής της εφαρμογής [48]. Η παρούσα πλατφόρμα αξιοποιεί τους μηχανισμούς αυτούς στο πλαίσιο της

ασφαλούς και ορθής λειτουργίας της, εντός του καθιερωμένου μοντέλου επικοινωνίας του Παγκόσμιου Ιστού.

2.1.4 Ηλεκτρονική Επικοινωνία μέσω SMTP

Η ηλεκτρονική επικοινωνία μέσω ηλεκτρονικού ταχυδρομείου αποτελεί βασικό μηχανισμό αλληλεπίδρασης στις σύγχρονες web εφαρμογές και χρησιμοποιείται σε λειτουργίες όπως οι ειδοποιήσεις, οι επιβεβαιώσεις ενεργειών και η αποστολή αυτοματοποιημένων μηνυμάτων. Η αποστολή ηλεκτρονικής αλληλογραφίας βασίζεται στο πρωτόκολλο SMTP (Simple Mail Transfer Protocol). Με το πρωτόκολλο αυτό, ορίζεται ο τρόπος μεταφοράς μηνυμάτων μεταξύ mail servers μέσω των δικτύων TCP/IP [50]. Το SMTP λειτουργεί συμπληρωματικά με άλλα πρωτόκολλα ηλεκτρονικού ταχυδρομείου και είναι υπεύθυνο αποκλειστικά για τη διαδικασία αποστολής μηνυμάτων. Στις σύγχρονες εφαρμογές, η επικοινωνία μέσω SMTP πραγματοποιείται συνήθως με τη χρήση ασφαλών καναλιών (π.χ. μέσω TLS), ώστε να διασφαλίζεται η ακεραιότητα των δεδομένων και η εμπιστευτικότητά τους κατά τη μεταφορά.

2.2 Τεχνολογίες Front-end

Οι front-end τεχνολογίες αφορούν το τμήμα της web εφαρμογής με το οποίο αλληλεπιδρά άμεσα ο χρήστης. Περιλαμβάνουν τις γλώσσες και τα εργαλεία που εκτελούνται στον φυλλομετρητή ιστού και είναι υπεύθυνα για τη δομή, την εμφάνιση και τη δυναμική συμπεριφορά της διεπαφής χρήστη. Στην παρούσα πλατφόρμα χρησιμοποιούνται καθιερωμένες τεχνολογίες του Παγκόσμιου Ιστού, οι οποίες υποστηρίζονται από όλα τα σύγχρονα προγράμματα περιήγησης και ακολουθούν διεθνή πρότυπα [51].

2.2.1 HTML5

Η HTML (HyperText Markup Language) αποτελεί την τυπική και θεμελιώδη γλώσσα σήμανσης (markup language) για τη δομή και τη δημιουργία ιστοσελίδων. Η HTML5 αποτελεί την πλέον σύγχρονη έκδοση της γλώσσας, εισάγοντας ένα σύγχρονο και επεκτάσιμο σύνολο στοιχείων, το οποίο ευνοεί την οργάνωση του περιεχομένου και τη βελτίωση της σημασιολογίας.

Ιδιαίτερη σημασία στην HTML5 έχουν τα semantic elements – όπως header, nav, main, section, article, footer –, τα οποία περιγράφουν το νόημα του περιεχομένου και όχι μόνο την εμφάνισή του. Η χρήση τους διευκολύνει την κατανόηση της δομής της σελίδας τόσο από τους φυλλομετρητές όσο και από βοηθητικές τεχνολογίες, συμβάλλοντας στη βελτίωση της προσβασιμότητας και της συντηρησιμότητας του κώδικα [20].

Η HTML5 αποτελεί το θεμέλιο πάνω στο οποίο βασίζονται οι υπόλοιπες front-end τεχνολογίες, καθώς ορίζει τη βασική δομή του περιεχομένου που στη συνέχεια μορφοποιείται με CSS και εμπλουτίζεται με JavaScript.

2.2.2 CSS3 και Responsive Σχεδίαση

Η CSS (Cascading Style Sheets) χρησιμοποιείται για τον καθορισμό της εμφάνισης και της διάταξης του περιεχομένου που ορίζεται στις σελίδες HTML. Η έκδοση CSS3 εισάγει σύγχρονους μηχανισμούς μορφοποίησης, όπως media queries, variables και ευέλικτα μοντέλα διάταξης, επιτρέποντας τη δημιουργία προσαρμοζόμενων διεπαφών χρήστη [51].

Η προσαρμοστική σχεδίαση (responsive design) στοχεύει στην ορθή απεικόνιση της εφαρμογής σε διαφορετικά μεγέθη οθόνης συσκευών. Μέσω τεχνικών όπως τα media queries και τα ευέλικτα πλέγματα (grid / flex), διασφαλίζεται η ομοιόμορφη προσαρμογή των στοιχείων που εμφανίζονται σε κάθε σελίδα και διατηρείται η συνέπεια της εμπειρίας χρήστη ανεξαρτήτως του μέσου πρόσβασης [52].

2.2.3 JavaScript, AJAX και JSON

Η JavaScript αποτελεί τη βασική γλώσσα προγραμματισμού που εκτελείται στον φυλλομετρητή ιστού και επιτρέπει την υλοποίηση δυναμικής συμπεριφοράς στις ιστοσελίδες. Παρέχει τη δυνατότητα αλληλεπίδρασης του χρήστη με τη διεπαφή της εφαρμογής, προσφέροντας λειτουργικότητα τόσο σε επίπεδο εμφάνισης, όπως η δυναμική τροποποίηση περιεχομένου, όσο και σε επίπεδο επικοινωνίας με τον εξυπηρετητή. Μέσω της JavaScript είναι δυνατή η εκτέλεση βασικών ελέγχων εγκυρότητας δεδομένων και η αποστολή αιτημάτων προς τον εξυπηρετητή, πριν την περαιτέρω επεξεργασία των δεδομένων στο back-end της εφαρμογής [53].

Η τεχνική AJAX (Asynchronous JavaScript and XML) επιτρέπει την ασύγχρονη ανταλλαγή δεδομένων μεταξύ client και server, βελτιώνοντας την απόκριση και τη διαδραστικότητα των web εφαρμογών. Μέσω της AJAX είναι δυνατή η εκτέλεση ελέγχων και η ανταλλαγή δεδομένων στο παρασκήνιο, χωρίς να απαιτείται πλήρης επαναφόρτωση της ιστοσελίδας σε κάθε ενέργεια του χρήστη. Με τον τρόπο αυτό επιτυγχάνεται ομαλότερη λειτουργία της εφαρμογής και βελτιωμένη εμπειρία χρήσης. Τα δεδομένα που ανταλλάσσονται είναι κυρίως σε μορφή JSON [54].

Η JSON (JavaScript Object Notation) αποτελεί μία ελαφριά μορφή αναπαράστασης και ανταλλαγής δεδομένων, η οποία βασίζεται σε δομή ζευγών κλειδιού–τιμής και είναι ταυτόχρονα κατανοητή από τον άνθρωπο και εύκολα αξιοποιήσιμη από το εκάστοτε λογισμικό. Χρησιμοποιείται ευρέως στις σύγχρονες web εφαρμογές ως μέσο μεταφοράς δεδομένων μεταξύ client και server, καθώς είναι ανεξάρτητη από τη γλώσσα προγραμματισμού και υποστηρίζεται από όλα τα σύγχρονα περιβάλλοντα εκτέλεσης. Λόγω της απλότητας και της αποδοτικότητάς της, η JSON έχει καθιερωθεί ως η κύρια μορφή ανταλλαγής δεδομένων σε ασύγχρονες επικοινωνίες, όπως αυτές που υλοποιούνται μέσω AJAX, αντικαθιστώντας σε μεγάλο βαθμό παλαιότερες μορφές όπως το XML [55].

2.2.4 Bootstrap 5 (CSS και JavaScript)

Το Bootstrap[3] αποτελεί ένα διαδεδομένο front-end framework (βλ. υποενότητα 2.3.5), το οποίο παρέχει έτοιμα πρότυπα CSS και JavaScript για τη δημιουργία responsive και ομοιόμορφων διεπαφών χρήστη. Η έκδοση Bootstrap 5 βασίζεται σε σύγχρονες προδιαγραφές και δεν απαιτεί άλλες εξωτερικές βιβλιοθήκες [17]. Το framework διατίθεται τόσο ως πλήρες πακέτο, όσο και με δυνατότητα επιλεκτικής χρήσης επιμέρους τμημάτων του, όπως συγκεκριμένων CSS κλάσεων ή JavaScript components. Η προσέγγιση αυτή επιτρέπει την προσαρμογή του στις ανάγκες της εφαρμογής και συμβάλλει στη μείωση του περιττού φόρτου, αποφεύγοντας την ενσωμάτωση λειτουργιών που δεν είναι απαραίτητες. Η χρήση του Bootstrap επιταχύνει την ανάπτυξη της διεπαφής και συμβάλλει στη διατήρηση κοινής αισθητικής και λειτουργικότητας σε ολόκληρη την εφαρμογή.

[3] https://getbootstrap.com/

2.3 Back-end Τεχνολογίες και PHP

Οι back-end τεχνολογίες αφορούν το τμήμα της web εφαρμογής που εκτελείται στον εξυπηρετητή και είναι υπεύθυνο για την επεξεργασία των αιτημάτων του client, τη διαχείριση των δεδομένων και την υλοποίηση της επιχειρησιακής λογικής. Το back-end λειτουργεί ως ενδιάμεσο επίπεδο μεταξύ της διεπαφής χρήστη και των συστημάτων αποθήκευσης και υπηρεσιών στον server, διασφαλίζοντας την ορθότητα, την ασφάλεια και τη συνέπεια της λειτουργίας της εφαρμογής [56].

Στο πλαίσιο των web εφαρμογών, το back-end επικοινωνεί με τον client μέσω τυποποιημένων πρωτοκόλλων – κυρίως HTTP/HTTPS – και ανταλλάσσει δεδομένα σε καθορισμένες μορφές, όπως η JSON. Η κεντρική λειτουργία της επιχειρησιακής λογικής (business logic) μέσω του εξυπηρετητή, επιτρέπει τον αποτελεσματικό έλεγχο πρόσβασης, την ομοιόμορφη εφαρμογή των κανόνων σε ολόκληρη την εφαρμογή και την ευκολότερη συντήρηση του συστήματος.

2.3.1 PHP ως Server-Side Γλώσσα Προγραμματισμού

Η PHP[4] αποτελεί μία ευρέως διαδεδομένη γλώσσα προγραμματισμού για την ανάπτυξη server-side εφαρμογών, ειδικά στον χώρο των διαδικτυακών εφαρμογών και ιστοτόπων. Εκτελείται στον εξυπηρετητή και επιτρέπει τη δυναμική δημιουργία περιεχομένου, την επεξεργασία δεδομένων εισόδου και την αλληλεπίδραση με βάσεις δεδομένων και άλλες υπηρεσίες [12].

Οι σύγχρονες εκδόσεις της PHP (8.x και νεότερες) εισάγουν βελτιώσεις σε επίπεδο απόδοσης, ασφάλειας και δομής κώδικα, ενισχύοντας τη δυνατότητα ανάπτυξης αξιόπιστων και επεκτάσιμων web εφαρμογών. Η γλώσσα υποστηρίζει τόσο διαδικαστικό όσο και αντικειμενοστρεφές προγραμματιστικό μοντέλο, επιτρέποντας την προσαρμογή της σε διαφορετικές ανάγκες και αρχιτεκτονικές.

2.3.2 Αντικειμενοστρεφής υποστήριξη στην PHP

Η PHP παρέχει ολοκληρωμένη υποστήριξη αντικειμενοστρεφούς προγραμματισμού (Object-Oriented Programming – OOP), μέσω μηχανισμών όπως κλάσεις, αντικείμενα και διεπαφές

[4] https://www.php.net/

(interfaces). Η προσέγγιση αυτή επιτρέπει τη μοντελοποίηση σύνθετων εννοιών και την οργάνωση του κώδικα σε διακριτές οντότητες με καθορισμένες ευθύνες. Στις πρόσφατες εκδόσεις της PHP έχει προστεθεί και η υποστήριξη enumerations (enums), οι οποίες επιτρέπουν τον ορισμό περιορισμένων συνόλων τιμών με αυξημένη σαφήνεια και ασφάλεια τύπων. Η χρήση αντικειμενοστραφών μηχανισμών συμβάλλει στη βελτίωση της αναγνωσιμότητας, της συντηρησιμότητας και της επεκτασιμότητας των εφαρμογών.

2.3.3 Διαχείριση Συνεδρίας σε Web Εφαρμογές

Λόγω της stateless φύσης του πρωτοκόλλου HTTP, οι web εφαρμογές απαιτούν πρόσθετους μηχανισμούς για τη διατήρηση της κατάστασης χρήστη. Η έννοια της συνεδρίας (session) χρησιμοποιείται για τη συσχέτιση των αιτημάτων κάθε χρήστη, επιτρέποντας λειτουργίες όπως η αυθεντικοποίηση και η προσωρινή αποθήκευση δεδομένων. Οι συνεδρίες υλοποιούνται συνήθως σε συνδυασμό με cookies, τα οποία αποθηκεύονται στον client και λειτουργούν ως αναγνωριστικά της αντίστοιχης συνεδρίας στον server, γεγονός που αποτελεί καθιερωμένη πρακτική στις web εφαρμογές για τη διαχείριση της κατάστασης χρήστη [57].

2.3.4 Πρόσβαση σε Βάσεις Δεδομένων μέσω PDO

Για την επικοινωνία με συστήματα διαχείρισης βάσεων δεδομένων, η PHP παρέχει το επίπεδο πρόσβασης PDO (PHP Data Objects). Το PDO λειτουργεί ως ενιαία διεπαφή, ανεξάρτητη από το εκάστοτε σύστημα βάσης δεδομένων, παρέχοντας φορητότητα και συνέπεια στον κώδικα. Η χρήση PDO διευκολύνει τη διαχείριση των συνδέσεων, την εκτέλεση ερωτημάτων και τον χειρισμό σφαλμάτων, συμβάλλοντας στη σταθερότητα και την ασφάλεια των web εφαρμογών. Για τους λόγους αυτούς, το PDO έχει καθιερωθεί ως βασική επιλογή στη σύγχρονη ανάπτυξη εφαρμογών με PHP, καθώς παρέχει ενιαίο και τυποποιημένο τρόπο πρόσβασης σε βάσεις δεδομένων. Επιπλέον, μέσω της υποστήριξης μηχανισμών όπως τα prepared statements, συμβάλλει ουσιαστικά στη μείωση του κινδύνου επιθέσεων τύπου SQL Injection, όταν χρησιμοποιείται σύμφωνα με τις προτεινόμενες πρακτικές ασφάλειας [12].

2.3.5 Frameworks

Τα frameworks αποτελούν οργανωμένα σύνολα βιβλιοθηκών και υποδομών ανάπτυξης, τα οποία παρέχουν έτοιμες λύσεις για συνηθισμένες ανάγκες web εφαρμογών, όπως δρομολόγηση αιτημάτων, διαχείριση προτύπων εμφάνισης (templates), πρόσβαση σε βάσεις δεδομένων, μηχανισμούς ασφάλειας και υποστήριξη αρχιτεκτονικών προτύπων.. Η χρήση τους συμβάλλει στην ταχύτερη ανάπτυξη εφαρμογών και στην ομοιομορφία του κώδικα, απαιτεί όμως προσαρμογή του προγραμματιστή στους κανόνες, στον τρόπο οργάνωσης και στις συμβάσεις που επιβάλλει το εκάστοτε framework.

Στην παρούσα εργασία επιλέχθηκε προσαρμοσμένη υλοποίηση χωρίς χρήση πλήρους back-end framework, με αξιοποίηση επιλεγμένων εξωτερικών βιβλιοθηκών για επιμέρους λειτουργίες (π.χ. αποστολή email και διαχείριση αρχείων υπολογιστικών φύλλων), καθώς και front-end framework (Bootstrap) για την ομοιόμορφη και responsive διεπαφή, ώστε να επιτευχθεί μεγαλύτερος έλεγχος της αρχιτεκτονικής και της λειτουργικότητας της πλατφόρμας.

2.4 Βάσεις Δεδομένων και Πρόσβαση Δεδομένων

Οι βάσεις δεδομένων αποτελούν βασικό δομικό στοιχείο των σύγχρονων web εφαρμογών, καθώς χρησιμοποιούνται για την αποθήκευση, οργάνωση και ανάκτηση μεγάλου όγκου δεδομένων με αξιόπιστο και αποδοτικό τρόπο. Στο πλαίσιο αυτό, χρησιμοποιούνται συνήθως σχεσιακά συστήματα διαχείρισης βάσεων δεδομένων (Relational Database Management Systems – RDBMS), τα οποία επιτρέπουν τη διατήρηση της ακεραιότητας των δεδομένων, τον έλεγχο πρόσβασης και την υποστήριξη ταυτόχρονης χρήσης από πολλαπλούς χρήστες [58]. Στις web εφαρμογές, η βάση δεδομένων λειτουργεί ως κεντρικό αποθετήριο πληροφορίας, το οποίο επικοινωνεί με το back-end της εφαρμογής μέσω καθορισμένων μηχανισμών πρόσβασης και μέσω της γλώσσας SQL (Structured Query Language), διασφαλίζοντας την ορθή, συνεπή και ασφαλή διαχείριση των δεδομένων [59].

2.4.1 Σχεσιακές Βάσεις Δεδομένων

Οι σχεσιακές βάσεις δεδομένων βασίζονται στο σχεσιακό μοντέλο, σύμφωνα με το οποίο τα δεδομένα οργανώνονται σε πίνακες με εγγραφές ανά σειρά και πεδία τιμών ανά στήλες.

Κάθε πίνακας περιγράφει μία συγκεκριμένη οντότητα, ενώ οι σχέσεις μεταξύ πινάκων υλοποιούνται μέσω πρωτευόντων και ξένων κλειδιών. Παράλληλα, υλοποιούνται περιορισμοί για την ασφαλή εισαγωγή και διαχείριση των δεδομένων, όπως μοναδικότητα τιμής πεδίου ή συνδυασμού τιμών πεδίων και δυνατότητα απαγόρευσης διαγραφής εγγραφής εάν υπάρχει σχέση πεδίου ως ξένο κλειδί σε άλλο πίνακα.

Σημαντική έννοια στον σχεδιασμό σχεσιακών βάσεων δεδομένων αποτελεί η κανονικοποίηση (normalization), η οποία αποσκοπεί στη μείωση του πλεονασμού και στην αποφυγή ασυνεπειών κατά την αποθήκευση και ενημέρωση των δεδομένων. Οι βασικές Κανονικές Μορφές (Normal Forms) περιγράφουν κριτήρια οργάνωσης των δεδομένων με στόχο τη μείωση του πλεονασμού και την αποφυγή ανωμαλιών εισαγωγής, ενημέρωσης και διαγραφής. Η Πρώτη Κανονική Μορφή (1NF) απαιτεί οι τιμές των πεδίων να είναι ατομικές (να μην περιέχουν επαναλαμβανόμενες ομάδες ή λίστες) και κάθε εγγραφή να είναι μοναδικά προσδιορισμένη. Η Δεύτερη Κανονική Μορφή (2NF) επεκτείνει την 1NF και απαιτείται για κάθε πεδίο που δεν αποτελεί τμήμα του πρωτεύοντος κλειδιού, να εξαρτάται πλήρως από το πρωτεύον κλειδί του. Η Τρίτη Κανονική Μορφή (3NF) απαιτεί τα πεδία να εξαρτώνται μόνο από το πρωτεύον κλειδί και όχι μεταξύ τους, οδηγώντας σε σχήματα με μικρότερη πιθανότητα ασυνεπειών. Η Τέταρτη Κανονική Μορφή (4NF) στοχεύει στην αντιμετώπιση πολλαπλών εξαρτήσεων τιμών (multivalued dependencies). Απαιτείται η Βάση να βρίσκεται σε 3NF και επιπλέον να μην υπάρχουν πίνακες όπου ερωτήματα προς κάποια στήλη αυτού του πίνακα θα επιστρέψουν ανεξάρτητα ζεύγη πολλαπλών τιμών.

Μια κατ' ελάχιστον καλή πρακτική κανονικοποίησης οφείλει να εφαρμόζεται τουλάχιστον έως την Τρίτη Κανονική Μορφή (Third Normal Form – 3NF), η οποία θεωρείται επαρκής για τα περισσότερα πληροφοριακά συστήματα, εξισορροπώντας την ακεραιότητα των δεδομένων και την απόδοση [59]. Οι σχεσιακές βάσεις δεδομένων παραμένουν ιδιαίτερα διαδεδομένες στις web εφαρμογές, λόγω της συνέπειας και της ισχυρής ακεραιότητας δεδομένων που προσφέρουν, της σταθερότητας και της εκτεταμένης υποστήριξης από πλήθος εργαλείων και τεχνολογιών ανάπτυξης. Παράλληλα, μπορούν να συνυπάρχουν και να ενσωματώνονται με νεότερες τεχνολογίες, ανάλογα με τις απαιτήσεις της εφαρμογής.

2.4.2 MySQL / MariaDB

Η MySQL[5] και η MariaDB[6] αποτελούν δημοφιλή συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS), τα οποία χρησιμοποιούνται ευρέως σε web εφαρμογές. Παρέχουν υποστήριξη για τη γλώσσα SQL, μηχανισμούς συναλλαγών (transactions) για την ασφαλή και συνεπή εκτέλεση πολλαπλών σχετιζόμενων ενεργειών, δείκτες (indexes) και ελέγχους ακεραιότητας δεδομένων. Ιδιαίτερη σημασία έχει η υποστήριξη Unicode μέσω κωδικοποίησης UTF-8, η οποία επιτρέπει την αποθήκευση και ορθή διαχείριση πολυγλωσσικού περιεχομένου. Επιπλέον, τα συστήματα αυτά προσφέρουν υψηλή απόδοση και αξιοπιστία, καθιστώντας τα κατάλληλα για εφαρμογές με αυξημένες απαιτήσεις ταυτόχρονης πρόσβασης χρηστών ή μεγάλου όγκου δεδομένων. Η MariaDB, ως διάδοχος της MySQL, διατηρεί πλήρη συμβατότητα σε επίπεδο λειτουργικότητας, ενώ εξελίσσεται ανεξάρτητα με έμφαση στην ανοιχτή ανάπτυξη και τη διαφάνεια [18,19].

2.5 Ασφάλεια Web Εφαρμογών

Η ασφάλεια αποτελεί κρίσιμο παράγοντα στον σχεδιασμό και τη λειτουργία σύγχρονων web εφαρμογών, καθώς τα συστήματα αυτά είναι εκτεθειμένα σε δημόσια δίκτυα και διαχειρίζονται συχνά ευαίσθητα δεδομένα. Κατά τον σχεδιασμό και την ανάπτυξη απαραίτητη προϋπόθεση αποτελεί η αναγνώριση πιθανών απειλών και η εφαρμογή γενικών αρχών προστασίας σε όλα τα επίπεδα της εφαρμογής, από τη διεπαφή χρήστη έως το back-end και τη βάση δεδομένων. Στο πλαίσιο αυτό, διεθνείς οργανισμοί και πρότυπα, όπως το OWASP (Open Worldwide Application Security Project), παρέχουν κατευθυντήριες οδηγίες και ταξινομήσεις απειλών, οι οποίες χρησιμοποιούνται ευρέως ως σημείο αναφοράς για την ανάπτυξη ασφαλών web εφαρμογών [60].

2.5.1 Συνήθεις απειλές σε web εφαρμογές

Μεταξύ των συχνότερων απειλών που αντιμετωπίζουν οι web εφαρμογές συγκαταλέγονται οι επιθέσεις SQL Injection, κατά τις οποίες κακόβουλος κώδικας εισάγεται σε ερωτήματα βάσης δεδομένων με σκοπό την αλλοίωση ή την υποκλοπή δεδομένων. Αντίστοιχα, οι επιθέσεις Cross-Site Scripting (XSS) στοχεύουν στην εκτέλεση μη αξιόπιστου κώδικα στον

[5] https://www.mysql.com/
[6] https://mariadb.org/

φυλλομετρητή του χρήστη, επηρεάζοντας την ακεραιότητα της διεπαφής και την ασφάλεια της πληροφορίας. Μία ακόμη συνηθισμένη κατηγορία επιθέσεων είναι το Cross-Site Request Forgery (CSRF), όπου ο επιτιθέμενος εκμεταλλεύεται μια ενεργή συνεδρία ήδη αυθεντικοποιημένου χρήστη, προκαλώντας ανεπιθύμητες ενέργειες χωρίς τη γνώση ή τη συγκατάθεσή του. Επιπλέον, ζητήματα που σχετίζονται με τη διαχείριση συνεδριών, όπως το Session Fixation, αποτελούν δυνητικό μηχανισμό παραβίασης της αυθεντικοποίησης χρηστών, εάν δεν εφαρμοστούν κατάλληλα μέτρα προστασίας σε επίπεδο διαχείρισης συνεδρίας [60,61].

2.5.2 Γενικές αρχές ασφάλειας web εφαρμογών

Η αντιμετώπιση των παραπάνω απειλών βασίζεται στην εφαρμογή γενικών και καθιερωμένων αρχών ασφάλειας. Κεντρικό ρόλο διαδραματίζει ο έλεγχος και η επικύρωση των δεδομένων εισόδου, ώστε να αποτρέπεται η εισαγωγή μη έγκυρου ή κακόβουλου περιεχομένου στην εφαρμογή. Παράλληλα, η υιοθέτηση μηχανισμών ελέγχου πρόσβασης διασφαλίζει ότι οι χρήστες έχουν πρόσβαση μόνο στις λειτουργίες και στα δεδομένα που αντιστοιχούν στα δικαιώματά τους [61].

Εξίσου σημαντική είναι η χρήση ασφαλούς επικοινωνίας, μέσω πρωτοκόλλων όπως το HTTPS, για την προστασία και την κρυπτογράφηση των δεδομένων κατά τη μεταφορά τους στο διαδίκτυο. Η αρχή του security by design, σύμφωνα με την οποία η ασφάλεια λαμβάνεται υπόψη από τα πρώτα στάδια του σχεδιασμού της εφαρμογής, συμβάλλει στη μείωση των κινδύνων και στην ανάπτυξη πιο ανθεκτικών συστημάτων. Η εφαρμογή των παραπάνω αρχών δεν εξαλείφει πλήρως τις απειλές, αλλά μειώνει σημαντικά τα πιθανά σημεία επίθεσης και ενισχύει το συνολικό επίπεδο ασφάλειας μιας web εφαρμογής.

2.6 Έλεγχος Ποιότητας και Στατική Ανάλυση (Static Analysis)

Ο έλεγχος ποιότητας λογισμικού αποτελεί αναπόσπαστο μέρος της σύγχρονης ανάπτυξης web εφαρμογών, καθώς στοχεύει στη βελτίωση της αξιοπιστίας, της συντηρησιμότητας και της ασφάλειας του παραγόμενου κώδικα. Εκτός από τις δοκιμαστικές εκτελέσεις της web εφαρμογής (runtime testing), ιδιαίτερη σημασία έχει και η ανάλυση του κώδικα χωρίς την εκτέλεσή του, μέσω τεχνικών στατικής ανάλυσης (static analysis). Η στατική ανάλυση

επιτρέπει τον εντοπισμό πιθανών σφαλμάτων, ασυνεπειών και παραβιάσεων προτύπων κώδικα σε πρώιμο στάδιο, συμβάλλοντας στη μείωση του χρόνου διόρθωσης και κατ' επέκταση, του αντίστοιχου κόστους, καθώς και στη βελτίωση της συνολικής ποιότητας του λογισμικού [62,63].

2.6.1 Έννοια της Στατικής Ανάλυσης (Static Analysis)

Η στατική ανάλυση αναφέρεται στη διαδικασία ανάλυσης και ελέγχου του πηγαίου κώδικα χωρίς την εκτέλεσή του. Μέσω αυτής της προσέγγισης είναι δυνατόν να εντοπιστούν προβλήματα όπως συντακτικά σφάλματα, πιθανά λογικά λάθη, ασυμβατότητες τύπων, καθώς και αποκλίσεις από καθιερωμένα πρότυπα συγγραφής κώδικα [62]. Σε αντίθεση με τις δυναμικές δοκιμές, οι οποίες απαιτούν εκτέλεση του προγράμματος, επομένως πλήρη λειτουργικότητα, η στατική ανάλυση μπορεί να ενσωματωθεί και να αξιοποιηθεί σε όλα τα στάδια του κύκλου ζωής της ανάπτυξης λογισμικού και συγγραφής κώδικα, προσφέροντας άμεση ανατροφοδότηση στον προγραμματιστή.

2.6.2 Εργαλεία Static Analysis στην PHP

Στο περιβάλλον προγραμματισμού της PHP έχουν αναπτυχθεί εξειδικευμένα εργαλεία static analysis ανοιχτού κώδικα και με διαρκείς ενημερώσεις, τα οποία χρησιμοποιούνται ευρέως για τον έλεγχο ποιότητας και τη διασφάλιση της ορθότητας του κώδικα. Παρακάτω αναφέρονται ενδεικτικά κάποια ευρέως διαδεδομένα εργαλεία, τα οποία χρησιμοποιήθηκαν κατά την ανάπτυξη της παρούσας εφαρμογής.

Το PHPStan[7] αποτελεί εργαλείο που εστιάζει στον έλεγχο τύπων και στη λογική συνέπεια του κώδικα, επιτρέποντας τον εντοπισμό σφαλμάτων τα οποία δεν είναι πάντα εμφανή κατά την εκτέλεση. Αντίστοιχα, το Psalm[8] προσφέρει προηγμένους μηχανισμούς ανάλυσης τύπων και ενισχύει την ασφάλεια και την προβλεψιμότητα της συμπεριφοράς της εφαρμογής.

Για τη διατήρηση ενιαίου ύφους και τη συμμόρφωση με πρότυπα συγγραφής κώδικα, χρησιμοποιείται το PHP_CodeSniffer[9], το οποίο ελέγχει τον κώδικα ως προς τα καθιερωμένα standards, όπως το PSR-12, συμβάλλοντας στη βελτίωση της αναγνωσιμότητας και της

[7] https://phpstan.org/
[8] https://psalm.dev/
[9] https://github.com/squizlabs/PHP_CodeSniffer

συντηρησιμότητας. Το PSR-12 (PHP Standards Recommendation 12) αποτελεί επίσημη σύσταση μορφοποίησης και συγγραφής κώδικα για τη γλώσσα PHP. Ορίζει κανόνες σχετικά με την εσοχή (indentation), τη χρήση κενών, τη διάταξη εντολών ελέγχου, τη δήλωση συναρτήσεων και κλάσεων, καθώς και τη γενική δομή των αρχείων. Η τήρησή του εξασφαλίζει ομοιογένεια στη συγγραφή, βελτιώνει την αναγνωσιμότητα και διευκολύνει τη συντήρηση του κώδικα, ιδιαίτερα σε έργα μεσαίας ή μεγάλης κλίμακας [64].

Επιπλέον, χρησιμοποιήθηκε το εργαλείο PHP Mess Detector[10] (PHPMD), για τον εντοπισμό πιθανών προβληματικών σημείων στον κώδικα, όπως υπερβολικά μεγάλες συναρτήσεις, επαναλαμβανόμενα τμήματα κώδικα ή μη χρησιμοποιούμενες μεταβλητές και παράμετροι. Το εργαλείο αυτό βοηθά στον εντοπισμό σχεδιαστικών ή δομικών σημείων που ενδέχεται να δυσχεραίνουν τη συντήρηση και την επέκταση του λογισμικού.

Συμπληρωματικά, το εργαλείο PHP Metrics[11] παρέχει μετρικές που σχετίζονται με το μέγεθος, τη δομή και την κυκλωματική πολυπλοκότητα του κώδικα, προσφέροντας συνολική εικόνα της ποιότητας και διευκολύνοντας την αξιολόγηση της αρχιτεκτονικής της εφαρμογής. Η κυκλωματική πολυπλοκότητα (cyclomatic complexity) αποτελεί μετρική που εκφράζει τον αριθμό των ανεξάρτητων διαδρομών εκτέλεσης ενός τμήματος κώδικα και χρησιμοποιείται για την εκτίμηση της δομικής πολυπλοκότητας και της συντηρησιμότητας [65].

Η συνδυαστική χρήση εργαλείων static analysis ενισχύει τον προληπτικό έλεγχο του κώδικα και συμβάλλει στη δημιουργία πιο αξιόπιστων και συντηρήσιμων web εφαρμογών.

2.6.3 Ανάλυση Εκτέλεσης Κώδικα με το εργαλείο QCacheGrind

Το εργαλείο QCacheGrind[12] χρησιμοποιείται για την ανάλυση της εκτέλεσης ενός προγράμματος, παρέχοντας γραφική απεικόνιση των κλήσεων συναρτήσεων και του χρόνου εκτέλεσής τους, επιτρέποντας τον εντοπισμό σημείων με υψηλό υπολογιστικό κόστος και τη βελτίωση της απόδοσης εκτέλεσης. Επίσης, διευκολύνει την ανάλυση και την κατανόηση της συμπεριφοράς του συστήματος, μέσω της ιεραρχικής προβολής των κλήσεων και των σχετικών μετρικών απόδοσης κατά την εκτέλεση.

[10] https://phpmd.org/
[11] https://www.phpmetrics.org/
[12] https://github.com/KDE/kcachegrind

2.6.4 Συμβάσεις Ονοματοδοσίας και Αναγνωσιμότητα Κώδικα

Οι συμβάσεις ονοματοδοσίας (naming conventions), όπως οι μορφές «snake_case» και «camelCase», αποτελούν καθιερωμένη πρακτική στη συγγραφή λογισμικού, καθώς συμβάλλουν στη συνέπεια, την αναγνωσιμότητα και τη μείωση λαθών. Η μορφή «snake_case» περιλαμβάνει μόνο πεζούς λατινικούς χαρακτήρες και αριθμούς (όχι όμως στην αρχή του ονόματος), με διαχωρισμό των λέξεων μέσω του χαρακτήρα underscore «_». Αντίστοιχα, η μορφή «camelCase» εκκινεί πάντα με πεζό γράμμα, δεν επιτρέπει τη χρήση underscore και ο διαχωρισμός των λέξεων πραγματοποιείται με κεφαλαίο το πρώτο γράμμα κάθε επόμενης λέξης και πεζά τα υπόλοιπα. Η επιλογή συγκεκριμένης σύμβασης, εξαρτάται από το εκάστοτε περιβάλλον και το επίπεδο υλοποίησης (π.χ. βάσεις δεδομένων, μεταβλητές, συναρτήσεις, κλάσεις). Η συνεπής εφαρμογή τους διευκολύνει τη συντήρηση και την κατανόηση του κώδικα από τρίτους προγραμματιστές.

2.7 Σύστημα Ελέγχου Εκδόσεων (Git και GitHub)

Τα συστήματα ελέγχου εκδόσεων (Version Control Systems – VCS) χρησιμοποιούνται για την παρακολούθηση των αλλαγών του πηγαίου κώδικα και την αποδοτικότερη διαχείριση κατά τη διάρκεια της ανάπτυξης του λογισμικού. Μέσω αυτών, καθίσταται δυνατή η καταγραφή της εξέλιξης ενός έργου, η επαναφορά σε προηγούμενες εκδόσεις και η ασφαλής συνεργασία πολλαπλών προγραμματιστών στο ίδιο σύνολο αρχείων [66].

2.7.1 Git

Το Git[13] αποτελεί ένα κατανεμημένο σύστημα ελέγχου εκδόσεων, το οποίο έχει σχεδιαστεί με κύριους άξονες την υψηλή απόδοση, την αξιοπιστία και την ευελιξία. Σε αντίθεση με τα κεντρικοποιημένα συστήματα, κάθε αντίγραφο ενός Git αποθετηρίου περιλαμβάνει πλήρες ιστορικό των αλλαγών, επιτρέποντας την εργασία ακόμη και χωρίς σύνδεση σε εξυπηρετητή. Βασική έννοια στο Git αποτελεί το *commit*, το οποίο αντιπροσωπεύει μία καταγεγραμμένη κατάσταση του έργου σε συγκεκριμένο χρονικό σημείο. Μέσω των *commits* είναι δυνατή η ιχνηλάτηση αλλαγών και η τεκμηρίωση της εξέλιξης του κώδικα ενός έργου. Επιπλέον, το

[13] https://git-scm.com/

Git υποστηρίζει μηχανισμούς όπως τα *branches*, επιτρέποντας την παράλληλη ανάπτυξη διαφορετικών λειτουργιών χωρίς να επηρεάζεται η κύρια έκδοση του έργου. Η χρήση του Git έχει καθιερωθεί στη βιομηχανία λογισμικού, αποτελώντας βασικό εργαλείο για την οργανωμένη και την ελεγχόμενη ανάπτυξη εφαρμογών.

2.7.2 GitHub

Το GitHub[14] αποτελεί διαδικτυακή πλατφόρμα φιλοξενίας αποθετηρίων Git και παρέχει πολλαπλές δυνατότητες συνεργασίας και διαχείρισης έργων λογισμικού. Ο προγραμματιστής μπορεί να αποθηκεύει αποθετήρια Git, να μοιράζεται τον κώδικά του και να παρακολουθεί την εξέλιξη ενός έργου σε πραγματικό χρόνο. Η πλατφόρμα προσφέρει λειτουργίες όπως η ορατότητα του ιστορικού αλλαγών, η τεκμηρίωση μέσω αρχείων όπως το README και CHANGELOG και η υποστήριξη συνεργατικής εργασίας. Επιπλέον, η διαχείριση των αποθετηρίων μπορεί να πραγματοποιείται τόσο μέσω της διαδικτυακής διεπαφής της πλατφόρμας όσο και τοπικά από το περιβάλλον εργασίας του χρήστη, αξιοποιώντας εργαλεία που υποστηρίζουν το σύστημα ελέγχου εκδόσεων Git. Η χρήση του GitHub ενισχύει τη διαφάνεια και τη συνεργασία, ενώ παράλληλα επιτρέπει την οργανωμένη παρουσίαση και διαχείριση του πηγαίου κώδικα μιας εφαρμογής [25].

Στο πλαίσιο της παρούσας εργασίας, το σύστημα ελέγχου εκδόσεων χρησιμοποιείται ως βασικό εργαλείο οργάνωσης και παρακολούθησης της ανάπτυξης της πλατφόρμας, εξασφαλίζοντας τη συνέπεια και την ιχνηλασιμότητα των αλλαγών στον κώδικα.

[14] https://github.com/

2.8 Εργαλεία Ανάπτυξης Λογισμικού

Κατά την ανάπτυξη σύγχρονων web εφαρμογών, η χρήση κατάλληλων εργαλείων ανάπτυξης αποτελεί καθοριστικό παράγοντα για την αποδοτικότητα, την ποιότητα και την ορθότητα του παραγόμενου λογισμικού. Τα εργαλεία αυτά υποστηρίζουν τον προγραμματιστή σε διαφορετικά στάδια του κύκλου ζωής του έργου, όπως η συγγραφή κώδικα, η σχεδίαση, η διαχείριση βάσεων δεδομένων και ο έλεγχος απόδοσης. Στο πλαίσιο της παρούσας εργασίας αξιοποιήθηκαν καθιερωμένα και ευρέως διαδεδομένα εργαλεία, τα οποία υποστηρίζονται ενεργά και χρησιμοποιούνται σε ακαδημαϊκό, αλλά και σε επαγγελματικό περιβάλλον.

2.8.1 Visual Studio Code

Το Visual Studio Code[15] αποτελεί ένα σύγχρονο και επεκτάσιμο περιβάλλον επεξεργασίας κώδικα, το οποίο υποστηρίζει πληθώρα γλωσσών προγραμματισμού και τεχνολογιών web. Παρέχει δυνατότητες όπως σύνταξη με επισήμανση (syntax highlighting), έλεγχο σφαλμάτων, ενσωμάτωση εργαλείων version control και επεκτάσεις για ανάλυση κώδικα, συμβάλλοντας στη βελτίωση της παραγωγικότητας και της ποιότητας του κώδικα.

2.8.2 Notepad++

Ο επεξεργαστής κειμένου Notepad++[16] προσφέρει ένα ελαφρύ και γρήγορο περιβάλλον για απλή επεξεργασία διαφόρων αρχείων ρυθμίσεων, καταγραφής και δεδομένων (π.χ. JSON). Η απλότητά του τον καθιστά χρήσιμο σε περιπτώσεις άμεσων διορθώσεων και ελέγχων.

2.8.3 Visual Paradigm

Για τη σχεδίαση και την οπτική αναπαράσταση της δομής του συστήματος χρησιμοποιήθηκε το Visual Paradigm[17], ένα εργαλείο μοντελοποίησης που υποστηρίζει διαγράμματα UML και άλλες τεχνικές σχεδίασης πληροφοριακών συστημάτων. Η χρήση του εργαλείου διευκολύνει την κατανόηση της αρχιτεκτονικής, την τεκμηρίωση του συστήματος και την παρουσίαση της σχεδίασης με σαφή και τυποποιημένο τρόπο.

[15] https://code.visualstudio.com/
[16] https://notepad-plus-plus.org/
[17] https://www.visual-paradigm.com/

2.8.4 phpMyAdmin και HeidiSQL για MariaDB

Για τη διαχείριση και την εποπτεία της βάσης δεδομένων αξιοποιήθηκε το εργαλείο phpMyAdmin[18], το οποίο παρέχει ένα εύχρηστο γραφικό περιβάλλον για τη σχεδίαση πινάκων, την εκτέλεση ερωτημάτων SQL, τον έλεγχο δεδομένων και τη συντήρηση της βάσης δεδομένων. Παράλληλα χρησιμοποιήθηκε και το GUI περιβάλλον διαχείρισης βάσεων δεδομένων HeidiSQL[19] που παρέχεται μαζί με το δημοφιλές MariaDB) (βλ. υποενότητα 2.4.2), για την υλοποίηση και τη διαχείριση της Βάσης Δεδομένων της πλατφόρμας σε τοπικό επίπεδο.

2.8.5 Apache JMeter

Για τη δοκιμή συμπεριφοράς της εφαρμογής υπό συνθήκες αυξημένου φόρτου χρησιμοποιήθηκε το Apache JMeter[20], ένα εργαλείο δημιουργίας σεναρίων προσομοίωσης χρήσης. Το εργαλείο αυτό βοηθάει στην αξιολόγηση της απόκρισης του συστήματος, τον εντοπισμό σημείων συμφόρησης και την εκτίμηση απόδοσης της εφαρμογής.

2.8.6 IIS Express

Το IIS Express αποτελεί ελαφριά έκδοση του web server IIS[21] και χρησιμοποιείται κυρίως για τοπική ανάπτυξη και δοκιμή εφαρμογών σε περιβάλλον Windows. Παρέχει βασική υποστήριξη HTTP/HTTPS και PHP μέσω FastCGI, επιτρέποντας τη γρήγορη εκκίνηση και τον έλεγχο εφαρμογών χωρίς την ανάγκη πλήρους εγκατάστασης παραγωγικού εξυπηρετητή.

2.8.7 Apache HTTP Server / XAMPP

Το Apache HTTP Server, σε συνδυασμό με εργαλεία όπως το XAMPP[22], χρησιμοποιείται ευρέως ως περιβάλλον τοπικής ανάπτυξης. Το XAMPP παρέχει ολοκληρωμένο πακέτο που περιλαμβάνει web server, υποστήριξη PHP και σύστημα διαχείρισης βάσεων δεδομένων, διευκολύνοντας τη γρήγορη ρύθμιση και δοκιμή web εφαρμογών σε ελεγχόμενο περιβάλλον.

[18] https://www.phpmyadmin.net/
[19] https://www.heidisql.com/
[20] https://jmeter.apache.org/
[21] https://www.iis.net/
[22] https://www.apachefriends.org/

2.9 Γενετικοί Αλγόριθμοι και Χρονοπρογραμματισμός

Οι γενετικοί αλγόριθμοι (Genetic Algorithms – GA) αποτελούν ευριστικές στοχαστικές μεθόδους βελτιστοποίησης, εμπνευσμένες από τις αρχές της φυσικής εξέλιξης και της φυσικής επιλογής. Η βασική τους ιδέα στηρίζεται στη λογική της σταδιακής βελτίωσης ενός πληθυσμού πιθανών λύσεων μέσω επαναληπτικών διαδικασιών επιλογής και αναπαραγωγής, με στόχο την εύρεση βέλτιστων ή σχεδόν βέλτιστων λύσεων σε προβλήματα μεγάλης πολυπλοκότητας.

Η λειτουργία των γενετικών αλγορίθμων βασίζεται στην αναπαράσταση των υποψήφιων λύσεων μέσω χρωμοσωμάτων (chromosomes), τα οποία αποτελούνται από επιμέρους στοιχεία που ονομάζονται γονίδια (genes). Προσομοιώνουν τη διαδικασία της φυσικής εξέλιξης των έμβιων οργανισμών, εφαρμόζοντας μηχανισμούς αντίστοιχους με την αναπαραγωγή, τον συνδυασμό γενετικού υλικού και τις τυχαίες μεταλλάξεις. Μέσω της διαδικασίας επιλογής, ευνοούνται σταδιακά οι καταλληλότερες λύσεις, ανάλογα με τα κριτήρια αξιολόγησης του προβλήματος.

Στους γενετικούς αλγόριθμους, τα χρωμοσώματα κωδικοποιούν τις μεταβλητές του προβλήματος και υφίστανται μεταβολές κατά τη διάρκεια της εκτέλεσης του αλγορίθμου, με στόχο τη βελτίωση της ποιότητας των λύσεων. Η διαδικασία εκτέλεσης ενός γενετικού αλγορίθμου περιλαμβάνει συγκεκριμένα στάδια. Αρχικά δημιουργείται ένας αρχικός πληθυσμός λύσεων, συνήθως με τυχαίο τρόπο, ώστε να καλύπτεται όσο το δυνατόν μεγαλύτερο μέρος του χώρου αναζήτησης. Στη συνέχεια, κάθε λύση αξιολογείται μέσω μιας συνάρτησης καταλληλότητας (fitness function), η οποία εκφράζει το πόσο καλά ικανοποιεί τους στόχους του προβλήματος [67]. Με βάση τις τιμές καταλληλότητας, εφαρμόζονται οι βασικοί γενετικοί τελεστές:

- επιλογή (selection), κατά την οποία επιλέγονται οι καταλληλότερες λύσεις για αναπαραγωγή,
- διασταύρωση (crossover), όπου συνδυάζονται τμήματα διαφορετικών λύσεων για τη δημιουργία νέων,
- μετάλλαξη (mutation), η οποία εισάγει μικρές τυχαίες αλλαγές ώστε να διατηρείται η ποικιλία στον πληθυσμό,

- επιδιόρθωση (repair), όπου εφαρμόζεται προαιρετικά όταν απαιτείται η διόρθωση μη έγκυρων λύσεων,
- δημιουργία νέου πληθυσμού (replacement), κατά την οποία ο νέος πληθυσμός αντικαθιστά τον προηγούμενο και ακολουθεί εκ νέου αξιολόγηση.

Τα παραπάνω στάδια επαναλαμβάνονται διαδοχικά, οδηγώντας σε σταδιακή βελτίωση του πληθυσμού μέχρι να ικανοποιηθεί κάποιο κριτήριο τερματισμού, όπως ένας μέγιστος αριθμός επαναλήψεων ή η επίτευξη αποδεκτής λύσης.

Σε αντίθεση με απλές τυχαίες μεθόδους αναζήτησης, οι γενετικοί αλγόριθμοι συνδυάζουν τυχαιότητα και κατευθυνόμενη εξερεύνηση, επιτρέποντας την αποδοτική αναζήτηση σε μεγάλα και πολύπλοκα πεδία λύσεων.

Οι γενετικοί αλγόριθμοι έχουν εφαρμοστεί εκτενώς σε προβλήματα βελτιστοποίησης, όπως ο χρονοπρογραμματισμός, όπου απαιτείται η ικανοποίηση πολλαπλών περιορισμών (π.χ. διαθεσιμότητα πόρων, αποφυγή συγκρούσεων) και ταυτόχρονα η χρήση κριτηρίων ποιότητας για τη βελτιστοποίηση των λύσεων. Η ευελιξία τους και η δυνατότητα προσαρμογής τους σε διαφορετικά προβλήματα τους καθιστά ιδιαίτερα κατάλληλους για την αντιμετώπιση σύνθετων προβλημάτων, όπως η δημιουργία ωρολογίων προγραμμάτων σε ακαδημαϊκά περιβάλλοντα.

2.10 Σύνοψη Κεφαλαίου

Στο παρόν κεφάλαιο παρουσιάστηκε το θεωρητικό υπόβαθρο που διέπει την ανάπτυξη και τη λειτουργία της παρούσας web εφαρμογής. Αναλύθηκαν βασικές έννοιες που σχετίζονται με την αρχιτεκτονική web εφαρμογών, τα πρωτόκολλα επικοινωνίας και τις τεχνολογίες front-end και back-end που χρησιμοποιήθηκαν.

Επιπλέον, εξετάστηκαν θέματα που αφορούν τις βάσεις δεδομένων, την ασφάλεια των web εφαρμογών, τον έλεγχο ποιότητας μέσω static analysis, καθώς και τα εργαλεία ανάπτυξης και ελέγχου εκδόσεων που υποστηρίζουν τη διαδικασία υλοποίησης. Τέλος, παρουσιάστηκαν τα διάφορα εργαλεία που χρησιμοποιήθηκαν κατά το στάδιο της σχεδίασης και ανάπτυξης, της συγγραφής κώδικα και των δοκιμών της εφαρμογής.

Στο επόμενο κεφάλαιο ακολουθεί η ανάλυση των προδιαγραφών του πληροφοριακού συστήματος που αναπτύχθηκε. Παρουσιάζονται οι λειτουργικές απαιτήσεις της πλατφόρμας,

τα σενάρια περιπτώσεων χρήσης και οι ιστορίες χρηστών, καθώς και τα αντίστοιχα διαγράμματα που αποτυπώνουν τη δομή και τη λειτουργία του συστήματος. Η μετάβαση αυτή από το θεωρητικό υπόβαθρο στις προδιαγραφές επιτρέπει τη σαφή κατανόηση των απαιτήσεων πάνω στις οποίες βασίστηκε ο σχεδιασμός και η υλοποίηση της εφαρμογής.

Στον Πίνακα 2-1 παρουσιάζονται συνοπτικά οι βασικές τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής, μαζί με τον κύριο σκοπό χρήσης τους.

Υποενότητα	**Τεχνολογία / Εργαλείο**	**Σκοπός χρήσης**
2.2.1	HTML5	Δομή και σημασιολογική οργάνωση των ιστοσελίδων της εφαρμογής.
2.2.2	CSS3	Μορφοποίηση, σχεδιασμός διεπαφής και responsive εμφάνιση σε διαφορετικές συσκευές.
2.2.3	JavaScript, AJAX, JSON	Διαδραστικότητα διεπαφής, ασύγχρονη επικοινωνία με τον server.
2.2.4	Bootstrap 5	Front-end framework για responsive σχεδίαση και επαναχρησιμοποιήσιμα στοιχεία διεπαφής.
2.3.1	PHP	Γλώσσα προγραμματισμού για την υλοποίηση της λογικής της εφαρμογής από πλευράς διακομιστή.
2.3.3	Μηχανισμοί Session PHP	Διαχείριση συνεδριών χρηστών και έλεγχος ταυτοποίησης.
2.3.4	PDO	Ασφαλής πρόσβαση και επικοινωνία με τη βάση δεδομένων μέσω prepared statements.
2.4.2	MariaDB / MySQL	Σύστημα διαχείρισης σχεσιακής βάσης δεδομένων για αποθήκευση και διαχείριση δεδομένων.
2.6.2	PHPStan, Psalm	Static analysis εργαλεία για έλεγχο τύπων και εντοπισμό πιθανών σφαλμάτων στον κώδικα.
2.6.2	PHP_CodeSniffer (PSR-12)	Έλεγχος συμμόρφωσης του κώδικα με πρότυπα μορφοποίησης και συγγραφής.

2.6.2	PHP Metrics	Ανάλυση δομής κώδικα και υπολογισμός μετρικών όπως η κυκλωματική πολυπλοκότητα.
2.6.3	QCacheGrind	Ανάλυση εκτέλεσης και εντοπισμός σημείων υψηλού υπολογιστικού κόστους.
2.7.2	GitHub	Διαχείριση εκδόσεων κώδικα.
2.8.1	Visual Studio Code	Κύριο περιβάλλον ανάπτυξης κώδικα (IDE).
2.8.2	Notepad++	Ελαφρύς επεξεργαστής κώδικα για γρήγορη επεξεργασία αρχείων.
2.8.3	Visual Paradigm	Σχεδίαση διαγραμμάτων UML και μοντελοποίηση της αρχιτεκτονικής του συστήματος.
2.8.4	phpMyAdmin, HeidiSQL	Διαχείριση και επεξεργασία της βάσης δεδομένων.
2.8.5	Apache JMeter	Έλεγχος απόδοσης και προσομοίωση φόρτου χρηστών.
2.8.6	IIS Express	Τοπικός web server για ανάπτυξη και δοκιμή της εφαρμογής.
2.8.7	Apache HTTP Server / XAMPP	Περιβάλλον τοπικής φιλοξενίας της εφαρμογής και εκτέλεσης PHP.

Πίνακας 2-1: Σύνοψη τεχνολογιών και εργαλείων που χρησιμοποιήθηκαν

3. Προδιαγραφές Πληροφοριακού Συστήματος

Στο κεφάλαιο αυτό, αναλύονται οι προδιαγραφές της διαδικτυακής πλατφόρμας, οι λειτουργικές απαιτήσεις και τα σενάρια περιπτώσεων χρήσης (use case scenarios), ενώ παρατίθενται και ενδεικτικές ιστορίες χρηστών ως μέσο παρουσίας της προστιθέμενης αξίας που λαμβάνουν οι χρήστες από το σύστημα [68]. Παράλληλα, περιλαμβάνονται και τα σχετικά διαγράμματα περιπτώσεων χρήσης (UML), καθώς και το διάγραμμα οντοτήτων-συσχετίσεων (Entity-Relationship Diagram – ERD) για την εννοιολογική αναπαράσταση σχέσεων μεταξύ των διαφόρων οντοτήτων σε ένα σύστημα βάσης δεδομένων.

3.1 Χαρακτηριστικά και Στόχοι Συστήματος

Το πληροφοριακό σύστημα υποστηρίζει τον πλήρη κύκλο διεργασιών για την εκπόνηση, διαχείριση και παρουσίαση ωρολογίων προγραμμάτων και προγραμμάτων εξετάσεων για Πανεπιστημιακά Ιδρύματα, με προσαρμογή στις ιδιαιτερότητες κάθε Τμήματος. Παρέχει εξατομικευμένες προβολές για τους φοιτητές, διευκολύνει τη γραμματεία και τους διδάσκοντες στη δημιουργία και τροποποίηση προγραμμάτων, ενώ ενσωματώνει μηχανισμούς ειδοποιήσεων για έκτακτες αλλαγές. Παράλληλα, προσφέρει στατιστικά και εποπτικές λειτουργίες που υποβοηθούν τη λήψη αποφάσεων και τη βελτιστοποίηση της λειτουργίας του τμήματος.

3.1.1 Λειτουργικά χαρακτηριστικά (σύνοψη)

Συνοπτικά, η πλατφόρμα:

- Δημιουργεί και τροποποιεί ωρολόγια προγράμματα ανά εξάμηνο και ανά τμήμα.
- Διαχειρίζεται περιορισμούς (αίθουσες, προτιμήσεις διδασκόντων, αργίες, σύγκρουση μαθημάτων, ώρες διδασκαλίας κ.ά.).
- Παρέχει εξατομικευμένο πρόγραμμα ανά φοιτητή (περιλαμβανομένων μαθημάτων προηγούμενων εξαμήνων που παρακολουθούνται).
- Υποστηρίζει δηλώσεις προτιμήσεων ή μη διαθεσιμότητας διδασκόντων και καταγράφει έκτακτες αλλαγές.
- Αποστέλλει ειδοποιήσεις και ενημερώνει τους χρήστες για έκτακτες αλλαγές.
- Τηρεί ιστορικό εκδόσεων εξαμήνων και καταγραφή ενεργειών χρηστών.

- Εισάγει δεδομένα από αρχεία CSV/Excel (όπως λίστα καθηγητών, λίστα κατάστασης φοιτητών κ.ά.).
- Παράγει πρόγραμμα εξετάσεων με ελέγχους σύγκρουσης και ισορροπημένης κατανομής.
- Παρέχει αναφορές και στατιστικά (ώρες ανά μάθημα/διδάσκοντα, χρήση αιθουσών).
- Εξάγει δεδομένα σε διάφορες μορφές αρχείων (PDF, Excel) των ωρολογίων προγραμμάτων, καθώς και άλλων στοιχείων, όπως αναφορές και στατιστικά.
- Παρουσιάζει τα ωρολόγια προγράμματα και τα προγράμματα εξετάσεων σε ψηφιακή web μορφή με δυναμικές λειτουργίες.
- Παρέχει δυνατότητα διασύνδεσης με άλλα συστήματα για τη λήψη πληροφοριών και δεδομένων (λίστες φοιτητών, δηλώσεις μαθημάτων κ.λπ.).
- Επιπλέον, το σύστημα υποστηρίζει μηχανισμό αυτόματης παραγωγής προτεινόμενων ωρολογίων προγραμμάτων μέσω αλγοριθμικής προσέγγισης βελτιστοποίησης, με στόχο την υποβοήθηση της διαδικασίας αρχικής κατανομής μαθημάτων.

3.1.2 Μη λειτουργικά χαρακτηριστικά (ποιότητα)

- Χρηστικότητα & προσβασιμότητα: Φιλική διεπαφή προς τον χρήστη, προσαρμοστική σχεδίαση (responsive design) συμβατή με σύγχρονες συσκευές (υπολογιστές desktop, φορητές συσκευές tablet και κινητά) και εφαρμογή βασικών πρακτικών προσβασιμότητας σύμφωνα με τις κατευθυντήριες αρχές της Ευρωπαϊκής Πράξης για την Προσβασιμότητα (EAA) [69].
- Απόδοση: Χρόνοι απόκρισης κατάλληλοι για μεγάλο αριθμό ταυτόχρονων χρηστών που αντιστοιχεί στην εκτιμώμενη κλίμακα του Ιδρύματος.
- Ασφάλεια: Έλεγχος πρόσβασης ανά ρόλο, ασφαλής χειρισμός δεδομένων, καταγραφή ενεργειών (logging), κρυπτογράφηση ευαίσθητων δεδομένων, χρήση μοναδικών κλειδιών και διακριτικών (tokens), χρήση πρωτοκόλλου SSL, πολυεπίπεδος έλεγχος εισερχόμενων δεδομένων.
- Επεκτασιμότητα & παραμετροποίηση: Δυνατότητα προσαρμογής σε κανόνες και πολιτικές του κάθε Τμήματος χωρίς αλλαγές στον βασικό κορμό κώδικα.
- Διαλειτουργικότητα: Τεκμηριωμένα API για ανταλλαγή δεδομένων με υφιστάμενα πληροφοριακά συστήματα.

- Συντηρησιμότητα: Υιοθέτηση σύγχρονων προτύπων κώδικα, στατική ανάλυση και δομημένη τεκμηρίωση.

3.1.3 Περιορισμοί και παραδοχές

Περιορισμοί:

- Τεχνικοί: διαθεσιμότητα και χωρητικότητα εξυπηρετητών, επιδόσεις βάσης δεδομένων, όρια αποστολής email/SMTP, πολιτικές ασφαλείας ιδρύματος (π.χ. firewalls, SSL/TLS), υποστήριξη τελευταίων εκδόσεων λογισμικού.
- Οργανωτικοί / Λειτουργικοί: περιορισμένη διαθεσιμότητα / χωρητικότητα αιθουσών και εξοπλισμού, ωράρια λειτουργίας, διαθεσιμότητα διδασκόντων, κανονισμοί εξετάσεων και ειδικές πολιτικές τμημάτων (μη ταυτόχρονη διεξαγωγή συγκεκριμένων μαθημάτων κ.ά.), απαιτούμενες ώρες ανά μάθημα.
- Δεδομένα: ασυνέπειες ή ελλείψεις σε δηλώσεις μαθημάτων / λίστες φοιτητών, καθυστερημένες ενημερώσεις από τρίτα συστήματα, ανάγκη ευθυγράμμισης κωδικοποιήσεων (π.χ. κωδικοί μαθημάτων και τμημάτων), ανομοιογένεια στη μορφοποίηση των δεδομένων εισαγωγής (π.χ. κεφαλαία/πεζά, χρήση τόνων).
- Ασφάλεια & κανονιστική συμμόρφωση: διαχείριση προσωπικών δεδομένων (GDPR), δικαιώματα ανά ρόλο, πιστοποίηση και ταυτοποίηση χρηστών.

Παραδοχές:

- Υπάρχει έγκυρος μηχανισμός ταυτοποίησης χρηστών (ιδρυματικοί λογαριασμοί) και επαρκής πολιτική ελέγχου από τη γραμματεία.
- Τα δεδομένα εισόδου (δηλώσεις, διαθέσιμες αίθουσες, ωράρια) παρέχονται έγκαιρα και με ορθότητα από τα αρμόδια τμήματα.
- Είναι διαθέσιμη υποδομή email/ειδοποιήσεων (SMTP ή ισοδύναμο) και εγκρίνονται οι αυτοματοποιημένες αποστολές.
- Σε περιπτώσεις ασυνέπειας δεδομένων, προηγούνται οι επίσημες εγγραφές (π.χ. η γραμματεία έχει τη «δεσμευτική» πηγή ορθών δεδομένων).
- Το Πανεπιστημιακό Ίδρυμα έχει πολιτική υποστήριξης, συντήρησης και αναβάθμισης του συστήματος μετά την εγκατάσταση.

3.1.4 Στόχοι συστήματος

Στο στάδιο αυτό, ορίζονται ενδεικτικοί δείκτες, που αποτυπώνουν τους στόχους και τα αναμενόμενα οφέλη του συστήματος. Η ποσοτική αποτίμηση μπορεί να πραγματοποιηθεί σε μεταγενέστερη πιλοτική εφαρμογή.

- Μείωση χρόνου σύνταξης ωρολογίου προγράμματος ανά εξάμηνο.
- Μείωση συγκρούσεων μαθημάτων / αιθουσών που επηρεάζουν τους φοιτητές.
- Χρόνος ενημέρωσης αλλαγών (δημοσίευση / ειδοποίηση άμεσα με την έγκριση).
- Πληρότητα και ποιότητα δεδομένων μέσω υποχρεωτικών πεδίων, ελέγχων εγκυρότητας και αναφορών πληρότητας.
- Ικανοποίηση χρηστών (γραμματεία / διδάσκοντες / φοιτητές) μέσω ερωτηματολογίου σε μορφή ανατροφοδότησης (feedback) κατά την πιλοτική εφαρμογή.

3.2 Ανάλυση Χρηστών και Ρόλων

3.2.1 Κατηγορίες Χρηστών

- Μη Πιστοποιημένος Χρήστης: Οποιοσδήποτε επισκέπτης του ιστοτόπου χωρίς να έχει πραγματοποιήσει είσοδο στην πλατφόρμα. Έχει περιορισμένη πρόσβαση σε βασικές λειτουργίες, όπως η προβολή γενικών ωρολογίων προγραμμάτων και δημόσιων πληροφοριών.
- Πιστοποιημένος Χρήστης: Κάθε χρήστης που διαθέτει λογαριασμό στην πλατφόρμα, έχει ενεργά προσωπικά διαπιστευτήρια και έχει πραγματοποιήσει είσοδο στην πλατφόρμα. Στην κατηγορία αυτή περιλαμβάνονται οι Διαχειριστές[23] (Κύριος Διαχειριστής και Διαχειριστής Τμήματος), ο Διδάσκων και ο Φοιτητής.
- Κύριος Διαχειριστής: Ο κεντρικός διαχειριστής της πλατφόρμας με πλήρη δικαιώματα. Διαχειρίζεται τις βασικές λειτουργίες και ρυθμίσεις του συστήματος, έχει την ευθύνη για τη δημιουργία και επεξεργασία λογαριασμών Διαχειριστών Τμημάτων και Διδασκόντων, καθώς και για την παραμετροποίηση των Σχολών, των Τμημάτων και των Αιθουσών. Ο λογαριασμός του δημιουργείται αυτόματα κατά την αρχική εγκατάσταση και ενεργοποιείται με τον ορισμό προσωπικού κωδικού πρόσβασης.

[23] Όπου αναγράφεται ο γενικός ρόλος «Διαχειριστής», νοείται τόσο ο Κύριος όσο και ο Διαχειριστής Τμήματος. Σε περίπτωση που απαιτείται διαχωρισμός εννοιών/ρόλων γίνεται ρητή αναφορά στο είδος Διαχειριστή.

- Διαχειριστής Τμήματος: Πιστοποιημένος χρήστης με ρόλο διαχειριστή, στον οποίο έχουν εκχωρηθεί συγκεκριμένα δικαιώματα. Ο λογαριασμός του δημιουργείται από τον Κύριο Διαχειριστή και ενεργοποιείται μετά από πρόσκληση μέσω email και τη δημιουργία προσωπικού κωδικού. Χειρίζεται λειτουργίες αποκλειστικά του Τμήματος που ανήκει, όπως στοιχεία Φοιτητών, διαχείριση μαθημάτων Τμήματος, εκπόνηση ωρολογίου προγράμματος Τμήματος.
- Διδάσκων: Καθηγητές και μέλη του διδακτικού προσωπικού του Ιδρύματος. Ο λογαριασμός του δημιουργείται από τους Διαχειριστές και ενεργοποιείται είτε μέσω ορισμού προσωπικού κωδικού είτε μέσω ιδρυματικής σύνδεσης (π.χ. SSO, εφόσον παρέχεται από το Πανεπιστήμιο, βλ. 3.2.7). Ο ρόλος περιλαμβάνει την καταχώρηση διαθεσιμότητας και προτιμήσεων για τον σχεδιασμό ωρολογίου προγράμματος, καθώς και τη δήλωση έκτακτων αλλαγών στο πρόγραμμα μαθημάτων του.
- Φοιτητής: Οι φοιτητές των Σχολών του Ιδρύματος που είτε έχουν ολοκληρώσει την εγγραφή τους στην πλατφόρμα είτε τους παρέχεται πρόσβαση μέσω ιδρυματικής σύνδεσης (π.χ. SSO, εφόσον παρέχεται από το Πανεπιστήμιο). Μέσω του λογαριασμού τους έχουν πρόσβαση στο εξατομικευμένο ωρολόγιο πρόγραμμα, σε ειδοποιήσεις έκτακτων αλλαγών, καθώς και σε επιπρόσθετες λειτουργίες, όπως η παρακολούθηση του ωρολογίου προγράμματος άλλων μαθημάτων ενδιαφέροντος.
- API Client: Εξωτερικό πληροφοριακό σύστημα με δυνατότητα διασύνδεσης στην πλατφόρμα για την αποστολή και λήψη δεδομένων (π.χ. λίστες φοιτητών, δηλώσεις μαθημάτων, λίστες διδασκόντων, ωρολόγια προγράμματα). Αποτελεί τεχνικό ρόλο και όχι φυσικό χρήστη, διασφαλίζοντας τη διαλειτουργικότητα με υφιστάμενα πληροφοριακά συστήματα του Ιδρύματος.
- Identity Provider: Εξωτερικό σύστημα πιστοποίησης χρηστών μέσω της υπηρεσίας Single-Sign-On (SSO), εφόσον αυτή παρέχεται από το Πανεπιστημιακό Ίδρυμα. Αποτελεί τεχνικό ρόλο και όχι φυσικό χρήστη, προσφέροντας δυνατότητα εισόδου σε υφιστάμενους, ενεργούς χρήστες του Πανεπιστημίου (Φοιτητές και Διδάσκοντες). Παρέχει επίσης τα βασικά δεδομένα ταυτοποίησης και χαρακτηριστικά του χρήστη, όπως, έτος εισαγωγής, τμήμα, ρόλο, αριθμό μητρώου και ιδρυματικό email, δεδομένα που αξιοποιούνται από το σύστημα για την ορθή ανακατεύθυνση και καταχώριση του χρήστη στην πλατφόρμα.

- Message Client: Εξωτερικό σύστημα αποστολής μηνυμάτων μέσω Email (SMTP) ή SMS (Υπηρεσία Third-Party) ή Telegram. Η κύρια λειτουργία αφορά την αποστολή email με σκοπό την ενημέρωση των χρηστών με αυτόματες ειδοποιήσεις, την ενεργοποίηση λογαριασμού χρήστη και την ταυτοποίηση με δεύτερο παράγοντα (2FA), ενώ τα μηνύματα μέσω SMS και Telegram χρησιμοποιούνται για την αποστολή κωδικών μίας χρήσης και ταυτοποίησης με δεύτερο παράγοντα (2FA).

3.2.2 Λειτουργίες Μη Πιστοποιημένων Χρηστών

Ως Μη Πιστοποιημένος χρήστης νοείται κάθε χρήστης που δεν διαθέτει ενεργό λογαριασμό ή δεν έχει πρόσβαση σε λειτουργίες της πλατφόρμας που απαιτούν είσοδο με διαπιστευτήρια (ιδρυματικό email και κωδικό πρόσβασης).

Αντίστοιχα, ως Μη Πιστοποιημένος χρήστης νοείται ο Διαχειριστής Τμήματος ή ο Διδάσκων, ο οποίος δεν έχει ενεργοποιήσει τον λογαριασμό του και δεν έχει δημιουργήσει προσωπικό κωδικό εισόδου ή δεν έχει ενεργοποιηθεί ο λογαριασμός του από τον Κύριο Διαχειριστή, οπότε και δεν έχει ακόμα πρόσβαση στις λειτουργίες Πιστοποιημένων Χρηστών της πλατφόρμας.

Ο Μη Πιστοποιημένος χρήστης έχει πρόσβαση σε δημόσιες πληροφορίες της πλατφόρμας, όπως είναι το γενικό ωρολόγιο πρόγραμμα κάθε Σχολής και Τμήματος ανά εξάμηνο και έτος ή πιθανές δημόσιες ανακοινώσεις, χωρίς προσωποποιημένα στοιχεία.

Για την είσοδό του στην πλατφόρμα με χρήση τοπικής πιστοποίησης, ο χρήστης επιλέγει τον σχετικό σύνδεσμο στην αρχική σελίδα και εισάγει τα στοιχεία εισόδου του (ιδρυματικό email και κωδικό πρόσβασης). Για λόγους ασφαλείας, σε περίπτωση που εντοπιστούν πολλαπλές ανεπιτυχείς προσπάθειες εισόδου σε σύντομο χρονικό διάστημα, είτε για τον ίδιο λογαριασμό είτε από την ίδια διεύθυνση IP, ενεργοποιείται μηχανισμός προσωρινού κλειδώματος. Το κλείδωμα αυτό αποσκοπεί στην αποτροπή κακόβουλων προσπαθειών πρόσβασης και αίρεται αυτόματα μετά την παρέλευση προκαθορισμένου χρονικού διαστήματος. Κατά την πρώτη του είσοδο – μέσω τοπικής πιστοποίησης από μια νέα συσκευή – το σύστημα απαιτεί ταυτοποίηση με δεύτερο παράγοντα (2FA), αποστέλλοντας προσωρινό κωδικό μέσω email, sms ή Telegram (κατόπιν επιλογής από τον χρήστη).

Ο χρήστης μπορεί να επιλέξει την επαναφορά (επαναδημιουργία) του κωδικού πρόσβασης σε περίπτωση απώλειας, επιλέγοντας τον σχετικό σύνδεσμο στη φόρμα εισόδου και εισάγοντας το ιδρυματικό του email. Στη συνέχεια, λαμβάνει μέσω email ένα μοναδικό σύνδεσμο επαναδημιουργίας του κωδικού του. Για την είσοδο στην πλατφόρμα ως Διδάσκων ή Φοιτητής, προσφέρεται επιπρόσθετα στον χρήστη η δυνατότητα εισόδου μέσω της υπηρεσίας Single-Sign-On (SSO), εφόσον αυτή παρέχεται από το Πανεπιστημιακό Ίδρυμα. Ο χρήστης σε αυτή την περίπτωση επιλέγει τον σχετικό σύνδεσμο στη φόρμα εισόδου και μεταφέρεται στο περιβάλλον πιστοποίησης του Πανεπιστημίου. Μετά την επιτυχή επαλήθευση των διαπιστευτηρίων που τηρούνται στο Πανεπιστήμιο, επιστρέφει αυτόματα στο περιβάλλον της πλατφόρμας σε κατάσταση συνδεδεμένου Πιστοποιημένου Χρήστη.

3.2.3 Λειτουργίες Πιστοποιημένων Χρηστών

Ως Πιστοποιημένος Χρήστης νοείται κάθε χρήστης, ο οποίος διαθέτει ενεργό λογαριασμό στην πλατφόρμα και έχει πραγματοποιήσει είσοδο με διαπιστευτήρια είτε τοπικά από το σύστημα εισόδου της πλατφόρμας – εισάγοντας το ιδρυματικό του email και τον κωδικό πρόσβασης που δημιούργησε για την πλατφόρμα – είτε μέσω της υπηρεσίας πιστοποίησης SSO, που παρέχεται από το Πανεπιστημιακό Ίδρυμα, όπου γίνεται ανακατεύθυνση στη σελίδα εισόδου του Ιδρύματος.

Πιστοποιημένος Χρήστης μπορεί να είναι: ο Κύριος Διαχειριστής, ο Διαχειριστής Τμήματος, ο Διδάσκων και ο Φοιτητής. Μετά την είσοδό του στην πλατφόρμα, ο Πιστοποιημένος Χρήστης έχει τη δυνατότητα διαχείρισης του προσωπικού του προφίλ (εκτός των σταθερών ιδρυματικών στοιχείων του, όπως είναι το ιδρυματικό του email, ο αριθμός μητρώου κ.λπ.). Επίσης, έχει τη δυνατότητα αλλαγής κωδικού πρόσβασης μέσω του προφίλ του.

Η πρόσβαση στις λειτουργίες και τις παρεχόμενες δυνατότητες της πλατφόρμας καθορίζεται σύμφωνα με τον ρόλο του εκάστοτε χρήστη, όπως αυτοί αναλύονται στις επόμενες υποενότητες.

Οι πιστοποιημένοι χρήστες έχουν πρόσβαση σε εσωτερικό σύστημα ανταλλαγής μηνυμάτων με τη γραμματεία του Τμήματος. Μέσω της λειτουργίας αυτής μπορούν να αποστέλλουν μηνύματα για τεχνικά ή ακαδημαϊκά ζητήματα που σχετίζονται με την πλατφόρμα ή το Τμήμα, ενώ η γραμματεία μπορεί να απαντά και να διατηρείται ιστορικό της επικοινωνίας.

3.2.4 Λειτουργίες Διαχειριστών

Κύριος Διαχειριστής:

Ο Κύριος Διαχειριστής δημιουργείται αυτόματα κατά την εγκατάσταση της πλατφόρμας σε ένα νέο περιβάλλον, με συγκεκριμένη διεύθυνση email, που έχει παρασχεθεί από το Πανεπιστημιακό Ίδρυμα, και προσωρινό κωδικό εισόδου. Κατά την πρώτη του είσοδο στην πλατφόρμα με χρήση του προσωρινού κωδικού, ζητείται από τον Κύριο Διαχειριστή να δημιουργήσει έναν νέο προσωπικό κωδικό εισόδου (ο προσωρινός καταργείται αυτόματα).

Ο Κύριος Διαχειριστής έχει την κεντρική διαχείριση της πλατφόρμας με βασικές λειτουργίες, όπως: α) ρυθμίσεις συστήματος, β) διαχείριση στοιχείων Σχολών, Τμημάτων, Αιθουσών, γ) έγκριση ενεργειών Διαχειριστών Τμημάτων, όπου απαιτείται (two-person rule), δ) δημιουργία και διαχείριση λογαριασμών Διαχειριστών Τμημάτων, ε) δημιουργία και διαχείριση λογαριασμών Διδασκόντων, στ) προβολή και εξαγωγή αναφορών και στατιστικών.

Ο Κύριος Διαχειριστής ορίζει τα βασικά στοιχεία του Πανεπιστημιακού Ιδρύματος, όπως την ονομασία, το λογότυπο (logo), τα στοιχεία επικοινωνίας και το κύριο ιδρυματικό domain. Επίσης, ορίζει τις Σχολές, τα Τμήματα και τις Αίθουσες. Ρυθμίζει σε κάθε εξάμηνο τις όποιες αλλαγές μπορεί να υπάρχουν στα παραπάνω δεδομένα. Δηλώνει τις ημερομηνίες και τα όρια για τη δημιουργία, έναρξη και λήξη των προς διαχείριση εξαμήνων και εξεταστικών περιόδων ανά εξάμηνο, δηλαδή από ποια ημερομηνία και έπειτα θα έχουν πρόσβαση στη διαχείριση επερχόμενων εξαμήνων οι Διαχειριστές Τμημάτων.

Εφόσον υπάρχει απαίτηση για αυξημένο έλεγχο κάποιας ενέργειας από άλλο πιστοποιημένο χρήστη, ο Κύριος Διαχειριστής εγκρίνει την ενέργεια αυτή ως διπλή επιβεβαίωση (two-person rule) για αποφυγή ανεπιθύμητων πράξεων.

Ο Κύριος Διαχειριστής μπορεί να δημιουργήσει νέους Διαχειριστές Τμημάτων (γραμματεία Τμήματος). Κατά τη δημιουργία του λογαριασμού, αποστέλλεται μέσω email (στο ιδρυματικο email του χρήστη) ένας μοναδικός προσωρινός σύνδεσμος για τη δημιουργία του προσωπικού του κωδικού και την ενεργοποίηση του λογαριασμού του. Επίσης, ο Κύριος Διαχειριστής έχει τη δυνατότητα απενεργοποίησης ενός λογαριασμού Διαχειριστή Τμήματος, καθώς και ενημέρωσης των στοιχείων του (εκτός από το ιδρυματικό του email και τον κωδικό πρόσβασης).

Ο Κύριος Διαχειριστής διαθέτει πρόσβαση στο σύνολο των πιστοποιημένων Διδασκόντων και μπορεί να προβεί στην καταχώρηση νέου Διδάσκοντα με δημιουργία νέου λογαριασμού, καθώς και στην τροποποίηση στοιχείων, όπως η κατάσταση λογαριασμού, η βαθμίδα, τα μαθήματα που διδάσκουν, η οργανική θέση κ.ά. Έχει τη δυνατότητα να ανεβάσει και να καταχωρήσει μαζικά μέσω αρχείων Excel/CSV λίστες με δεδομένα Διδασκόντων για τη δημιουργία ή ενημέρωση λογαριασμών. Η αντιστοίχιση των δεδομένων της λίστας με τα δεδομένα της πλατφόρμας γίνεται αποκλειστικά βάσει του ιδρυματικού email κάθε χρήστη. Πραγματοποιείται έλεγχος αντιστοίχισης και εγκυρότητας δεδομένων, πριν την οριστική μαζική υποβολή, για αποφυγή εσφαλμένης καταχώρησης.

Ο Κύριος Διαχειριστής ορίζει τις διαθέσιμες αίθουσες κάθε Σχολής, το είδος της αίθουσας (όπως θεωρίας, εργαστηρίων, μικτή, αμφιθέατρο), καθώς και τα Τμήματα, που μπορούν να κάνουν χρήση της εκάστοτε αίθουσας.

Διαχειριστής Τμήματος:

Ο Διαχειριστής Τμήματος έχει ρόλο γραμματείας Τμήματος και η λειτουργία του επικεντρώνεται στη διαχείριση των φοιτητών και των διδασκόντων του Τμήματος, τον καθορισμό των μαθημάτων εξαμήνου και των παραμέτρων για τη δημιουργία των ωρολογίων προγραμμάτων και των εξεταστικών προγραμμάτων ανά εξάμηνο, καθώς και την εκπόνηση των προγραμμάτων αυτών.

Ο λογαριασμός κάθε Διαχειριστή Τμήματος δημιουργείται από τον Κύριο Διαχειριστή. Για την ενεργοποίηση του λογαριασμού του, ο χρήστης λαμβάνει μήνυμα ηλεκτρονικού ταχυδρομείου με μοναδικό προσωρινό σύνδεσμο (link), μέσω του οποίου δημιουργεί τον προσωπικό του κωδικό εισόδου.

Οι κύριες λειτουργίες και οι ενέργειες που εκτελεί ο Διαχειριστής Τμήματος είναι:

- Διαχείριση πιστοποιημένων Φοιτητών, προβολή και τροποποίηση στοιχείων:
  Ο Διαχειριστής Τμήματος διαθέτει πρόσβαση στο σύνολο των πιστοποιημένων Φοιτητών και μπορεί να προβεί στην τροποποίηση των στοιχείων τους και την ενημέρωση της κατάστασης φοίτησης, καθώς και στην προβολή ή εξαγωγή λίστας Φοιτητών με φίλτρα επιλογής (κατάσταση, εξάμηνο, Τμήμα, μάθημα κ.λπ.).
- Διαχείριση Διδασκόντων και επεξεργασία στοιχείων:

Ο Διαχειριστής Τμήματος προβάλλει το σύνολο των πιστοποιημένων Διδασκόντων και των στοιχείων τους, όπως η κατάσταση λογαριασμού, η βαθμίδα, τα μαθήματα που διδάσκουν, η οργανική θέση κ.ά., χωρίς όμως δυνατότητα επεξεργασίας. Επίσης, έχει πρόσβαση για προβολή και επεξεργασία σε δηλωθείσες ημέρες και ώρες διδασκαλίας των Διδασκόντων και μπορεί να εξάγει λίστα με βάση επιλεγμένα κριτήρια (κατάσταση, βαθμίδα, Τμήμα, μάθημα κ.λπ.).

- Εισαγωγή λίστας Φοιτητών για μαζική καταχώρηση δεδομένων:
  Ο Διαχειριστής Τμήματος έχει τη δυνατότητα να ανεβάσει και να καταχωρήσει μαζικά μέσω αρχείων Excel/CSV λίστες δεδομένων Φοιτητών για τη δημιουργία ή ενημέρωση λογαριασμών. Μπορεί να πραγματοποιήσει μαζική ενημέρωση στοιχείων και κατάστασης φοίτησης. Η αντιστοίχιση των δεδομένων της λίστας με τα δεδομένα της πλατφόρμας γίνεται αποκλειστικά με βάσει το ιδρυματικό email κάθε χρήστη. Για τη διασφάλιση της ορθότητας των δεδομένων, το σύστημα διαθέτει πρότυπο αρχείο Excel με έλεγχο τιμών πεδίων και επικύρωση δεδομένων, το οποίο μπορεί να χρησιμοποιήσει προς διευκόλυνσή της η γραμματεία για την ορθή συμπλήρωση των δεδομένων. Κατά το ανέβασμα του αρχείου, πραγματοποιείται αυτόματος έλεγχος και επισήμανση σφαλμάτων τιμών και αντιστοιχίσεων, ενώ παρέχεται προεπισκόπηση και έλεγχος ορθότητας πριν την οριστική καταχώρηση.

- Διαχείριση Μαθημάτων Τμήματος:
  Ο Διαχειριστής Τμήματος ελέγχει και διαμορφώνει τα προσφερόμενα μαθήματα ανά Τμήμα, όπως αυτά έχουν οριστεί από το Πανεπιστημιακό Ίδρυμα στο πρόγραμμα σπουδών, ορίζει την ονομασία και τον κωδικό αναφοράς κάθε μαθήματος, το είδος (θεωρητικό, εργαστηριακό, μεικτό), το προσφερόμενο εξάμηνο (ή εξάμηνα), τον τύπο (υποχρεωτικό, επιλογής, κ.ο.κ.), τις απαιτούμενες εβδομαδιαίες ώρες διδασκαλίας ανά είδος (θεωρία, εργαστήριο, άσκηση), τους διαθέσιμους Διδάσκοντες. Ο ορισμός της σχέσης Μάθημα – Διδάσκων γίνεται μέσω της διαχείρισης Μαθημάτων.

- Ορισμός Αιθουσών ανά Μάθημα:
  Ο Διαχειριστής Τμήματος ορίζει τις αίθουσες για κάθε μάθημα του Τμήματος, ενώ παράλληλα, όσον αφορά τις κοινόχρηστες αίθουσες που χρησιμοποιούνται από διάφορα Τμήματα της Σχολής, καταχωρεί συγκεκριμένες ημέρες χρήσης, προς αποφυγή επικαλύψεων κατά τη δημιουργία των ωρολογίων προγραμμάτων.

- Δήλωση και καταχώρηση διαθέσιμων ημερών/ωρών διδασκαλίας Διδασκόντων:
  Ο Διαχειριστής Τμήματος μπορεί να ορίσει προτιμώμενες ημέρες και ώρες διδασκαλίας για Διδάσκοντες, καθώς και περιορισμούς σε ημέρες/ώρες που υπάρχει αδυναμία διδασκαλία. Αντίστοιχες δηλώσεις μπορούν να καταχωρούν και οι Διδάσκοντες μέσω του συστήματος. Οι δηλωθείσες ώρες και ημέρες λαμβάνονται αυστηρά υπόψη για τη δημιουργία του ωρολογίου προγράμματος. Σε κάθε νέα δήλωση ή αλλαγή, αποστέλλεται αυτόματο ενημερωτικό email προς τον Διδάσκοντα.

- Καθορισμός παραμέτρων δημιουργίας ωρολογίου προγράμματος:
  Ο Διαχειριστής Τμήματος ορίζει τις διάφορες παραμέτρους που απαιτούνται για τη δημιουργία του ωρολογίου προγράμματος κάθε εξαμήνου ή των εξεταστικών περιόδων. Οι παράμετροι αυτοί περιλαμβάνουν δεδομένα και στοιχεία, που έχουν ήδη οριστεί (π.χ. διαθέσιμες ώρες/ημέρες Διδασκόντων, απαιτούμενες εβδομαδιαίες ώρες ανά μάθημα, διαθέσιμες αίθουσες), καθώς και ημερομηνία έναρξης/λήξης, επιθυμητά διαλείμματα (π.χ. μεσημεριανή σίτιση) και αργίες. Οι παραπάνω παράμετροι χρησιμοποιούνται τόσο για την αυτόματη δημιουργία προτάσεων προγραμμάτων όσο και για τη χειρωνακτική διαμόρφωση με έλεγχο παραβίασης περιορισμών.

- Δημιουργία και διαχείριση ωρολογίου προγράμματος εξαμήνου:
  Το σύστημα παρέχει δυνατότητα αυτόματης δημιουργίας προτάσεων ωρολογίου προγράμματος, βάσει παραμέτρων και περιορισμών που έχει ορίσει ο Διαχειριστής Τμήματος. Κατά τον αυτόματο υπολογισμό, λαμβάνονται υπόψη οι πιθανές συγκρούσεις και οι περιορισμοί, ενώ δημιουργείται ωρολόγιο πρόγραμμα για όλα τα Τμήματα. Ο Διαχειριστής Τμήματος μπορεί να προβεί σε διορθώσεις και αλλαγές. Η δημιουργία του προγράμματος γίνεται σε εβδομαδιαίο ημερολόγιο, με χρονικές θέσεις (slots) ανά ώρα για κάθε μάθημα. Κατά τη χειρωνακτική δήλωση ωρών, το σύστημα ελέγχει αυτόματα για συγκρούσεις ή παραβίαση περιορισμών και ενημερώνει τον Διαχειριστή Τμήματος με αναδυόμενο μήνυμα (pop-up message). Ο Διαχειριστής Τμήματος μπορεί να προβάλει το πλήρες ημερολογιακό πρόγραμμα με συγκεντρωτικά στοιχεία ωρών ανά μάθημα και διδάσκοντα, για τον έλεγχο συμπλήρωσης των απαιτούμενων ωρών ανά μάθημα. Μετά την ολοκλήρωση δημιουργίας του ωρολογίου προγράμματος και την οριστικοποίηση, αναρτάται στην πλατφόρμα με εξατομικευμένη προβολή αλλά και δημόσια κοινοποίηση με πρόσβαση από κάθε ενδιαφερόμενο, ενώ μπορεί να εξαχθεί σε μορφές αρχείων Excel ή PDF.

- Δημιουργία και διαχείριση προγράμματος εξετάσεων εξαμήνου:
  Κατά ανάλογο τρόπο με το ωρολόγιο πρόγραμμα, το σύστημα παρέχει αυτόματες προτάσεις για το πρόγραμμα εξετάσεων, λαμβάνοντας υπόψη περιορισμούς σύγκρουσης εξετάσεων μαθημάτων, χρονικές αποστάσεις μεταξύ μαθημάτων, διαθεσιμότητα αιθουσών και ισορροπημένη κατανομή στο σύνολο της περιόδου. Υποστηρίζεται χειρωνακτική προσαρμογή και έλεγχο εγκυρότητας, πριν την οριστικοποίηση και ανάρτηση.

- Προβολή στατιστικών στοιχείων:
  Ο Διαχειριστής Τμήματος έχει τη δυνατότητα εξαγωγής δεδομένων και στατιστικών ανά εξάμηνο, όπως: ώρες διδασκαλίας ανά Διδάσκοντα, πλήθος μαθημάτων ανά Διδάσκοντα, αριθμός αλλαγών/τροποποιήσεων προγράμματος, δεσμευμένες ώρες, χρήση αιθουσών κ.ά. Τα δεδομένα κάθε εξαμήνου και ακαδημαϊκού έτους διατηρούνται σε ιστορικό εκδόσεων (versioning) και οι χρήστες έχουν δυνατότητα προβολής προγραμμάτων προηγούμενων εξαμήνων και ετών. Αυτό παρέχει στον Διαχειριστή τη δυνατότητα προβολής και εξαγωγής στατιστικών προηγούμενων εξαμήνων, καθώς και συγκεντρωτικών καταστάσεων ή συγκριτικών αναφορών.

3.2.5 Λειτουργίες Διδασκόντων

Ο Διδάσκων δηλώνει τις προτιμώμενες ημέρες και ώρες διδασκαλίας, καθώς και τις ημέρες ή ώρες αδυναμίας διδασκαλίας (μη διαθεσιμότητας), κατά την περίοδο δημιουργίας του ωρολογίου προγράμματος από τους Διαχειριστές Τμημάτων. Οι δηλώσεις αυτές λαμβάνονται αυστηρά υπόψη κατά τη δημιουργία του ωρολογίου προγράμματος.

Ο Διδάσκων έχει τη δυνατότητα προβολής του προσωπικού του προγράμματος σε εβδομαδιαία ή ημερολογιακή μορφή, καθώς και πρόσβαση στο συνολικό πρόγραμμα του Τμήματος με την εφαρμογή φίλτρων (όπως ημερομηνία, αίθουσα, μάθημα και τύπος δραστηριότητας). Επιπλέον, ο Διδάσκων λαμβάνει ειδοποιήσεις, μέσω email αλλά και μέσω του εσωτερικού συστήματος μηνυμάτων, για οποιεσδήποτε αλλαγές στο πρόγραμμα που τον αφορούν.

Ο Διδάσκων έχει τη δυνατότητα δήλωσης αναβολής μαθήματος σε συγκεκριμένη ημερομηνία, οπότε ενημερώνεται αυτόματα η αλλαγή στο ημερολογιακό πρόγραμμα μαθημάτων. Παράλληλα, μπορεί να δηλώσει την αναπλήρωση μαθήματος σε άλλη διαθέσιμη

ημερομηνία και ώρα, με αναζήτηση διαθέσιμης αίθουσας και ημέρας/ώρας και αυτόματους ελέγχους διαθεσιμότητας αιθουσών και συγκρούσεων με άλλα μαθήματα. Σε περιπτώσεις αδυναμίας πρόσβασης στο σύστημα, η αλλαγή μπορεί να καταχωρηθεί από Διαχειριστή Τμήματος, με αυτόματη ενημέρωση του Διδάσκοντα μέσω email. Για κάθε αλλαγή ή τροποποίηση του προγράμματος, αναρτάται σχετική ανακοίνωση με δημόσια κοινοποίηση, καθώς και εξατομικευμένη εσωτερική ειδοποίηση προς τους ενδιαφερόμενους Φοιτητές.

Ο Διδάσκων δύναται να τροποποιήσει ή να ενημερώσει ορισμένα στοιχεία του προφίλ του (π.χ. στοιχεία επικοινωνίας). Τα σταθερά δεδομένα (όπως μαθήματα, οργανική θέση κ.ά.) μπορούν να τροποποιηθούν μόνο από τον Κύριο Διαχειριστή.

3.2.6 Λειτουργίες Φοιτητών

Ο Φοιτητής προβάλλει το εξατομικευμένο πρόγραμμά του (τρέχον και προηγούμενα εξάμηνα) σε μορφή εβδομαδιαίου ημερολογίου, με πληροφορίες αιθουσών, αργιών και έκτακτων αλλαγών, αναβολών και αναπληρώσεων.

Ο Φοιτητής λαμβάνει ειδοποιήσεις μέσω εσωτερικών μηνυμάτων ή email, για κάθε αλλαγή (αναβολή ή αναπλήρωση) σε μαθήματα που παρακολουθεί ή έχει δηλώσει ενδιαφέρον.

Ο Φοιτητής μπορεί να επιλέξει μαθήματα του Τμήματός του που τον ενδιαφέρουν – πέραν των μαθημάτων του τρέχοντος εξαμήνου –, ώστε να εμφανίζονται στο εξατομικευμένο του πρόγραμμα (λειτουργία «Τα Μαθήματά μου», αντίστοιχη με τη λειτουργία watchlist που παρέχουν πολλές ιστοσελίδες). Επιπλέον, μπορεί με παρόμοιο τρόπο να επιλέξει από λίστα των μη υποχρεωτικών μαθημάτων του τρέχοντος εξαμήνου (μαθήματα επιλογής), εκείνα που τον ενδιαφέρουν και επιθυμεί να εμφανίζονται στο εξατομικευμένο του πρόγραμμα και να λαμβάνει σχετικές ειδοποιήσεις.

Στον Φοιτητή διατίθεται ιστορικό εκδόσεων των ωρολογίων προγραμμάτων, ώστε να συμβουλεύεται προγράμματα προηγούμενων εξαμήνων και ετών.

Ο Φοιτητής δύναται να τροποποιήσει / ενημερώσει ορισμένα στοιχεία του προφίλ του (π.χ. στοιχεία επικοινωνίας), ενώ σταθερά δεδομένα (όπως μαθήματα, ονοματεπώνυμο κ.ά.) τροποποιούνται αποκλειστικά από τον Διαχειριστή Τμήματος.

3.2.7 Λειτουργίες Εξωτερικών Συστημάτων (API)

API Client:

Ο ρόλος αντιστοιχεί σε εξωτερική εφαρμογή ή διεπαφή χρήστη που αξιοποιεί το REST API της πλατφόρμας για την ανάκτηση δεδομένων. Η πλατφόρμα παρέχει τεκμηριωμένο REST API μέσω του οποίου είναι δυνατή η λήψη δεδομένων που σχετίζονται με το ωρολόγιο και το ημερολογιακό πρόγραμμα μαθημάτων, συμπεριλαμβανομένων δεδομένων που μπορούν να αξιοποιηθούν για τη λειτουργία εξωτερικών μηχανισμών επίλυσης (solvers).

Η πρόσβαση στα δεδομένα πραγματοποιείται με χρήση παραμέτρων και φίλτρων, όπως Σχολή, Τμήμα, ακαδημαϊκή περίοδος, εξάμηνο, ημερομηνία, διδάσκων και αίθουσα. Τα αποτελέσματα επιστρέφονται σε δομημένη μορφή (JSON), με ελέγχους ασφάλειας για τον περιορισμό υπερβολικών αιτήσεων. Το API μπορεί να αξιοποιηθεί για την ανάπτυξη εναλλακτικών διεπαφών χρήστη ή εφαρμογών που απαιτούν πρόσβαση σε δεδομένα του συστήματος.

Identity Provider:

Αφορά σε εξωτερική υπηρεσία που παρέχεται μέσω μηχανισμού Single-Sign-On (SSO) (SAML 2.0 ή OAuth 2.0), που διαθέτει το εκάστοτε Πανεπιστημιακό Ίδρυμα. Η επιτυχής διασύνδεση πραγματοποιείται με τεκμηριωμένη παραμετροποίηση της πλατφόρμας, για την αντιστοίχιση των πεδίων μεταξύ των δύο συστημάτων και των απαιτούμενων δεδομένων προς λήψη.

Η ταυτοποίηση και εξουσιοδότηση γίνεται μέσω ασφαλών μηχανισμών (π.χ. tokens, ιδιωτικό κλειδί), ώστε να διασφαλίζεται η διαλειτουργικότητα με υφιστάμενες υποδομές του Πανεπιστημιακού Ιδρύματος. Η υπηρεσία παρέχεται ως επιπρόσθετη δυνατότητα εισόδου, πέραν του τοπικού συστήματος εισόδου, για την είσοδο Φοιτητών και Διδασκόντων στην πλατφόρμα, οι οποίοι διαθέτουν ενεργό λογαριασμό στο Πανεπιστημιακό Ίδρυμα.

Message Client:

Ο ρόλος αντιστοιχεί σε εξωτερικές υπηρεσίες αποστολής μηνυμάτων, όπως Mail Server για αποστολή email μέσω SMTP, πάροχοι υπηρεσιών αποστολής SMS για κινητές συσκευές, καθώς και υπηρεσίες αποστολής μηνυμάτων, όπως Telegram ή Viber. Η πλατφόρμα έχει ενσωματωμένα API για επικοινωνία και αποστολή μηνυμάτων μέσω των παραπάνω υπηρεσιών.

3.3 Ιστορίες Χρηστών (User Stories) & Κριτήρια Αποδοχής

Οι ιστορίες χρηστών είναι σύντομα κείμενα, στα οποία περιγράφεται η αναμενόμενη αλληλεπίδραση του χρήστη σε συγκεκριμένες λειτουργίες της πλατφόρμας. Προσομοιάζουν τις προδιαγραφές απαιτήσεων λογισμικού, παραλείποντας τεχνικές λεπτομέρειες και εστιάζοντας στο τι πρέπει να κάνει το σύστημα από την οπτική του τελικού χρήστη [68]. Συμβάλλουν στη σχεδίαση και ανάπτυξη του συστήματος, καθώς και στην περαιτέρω ανάλυση των απαιτήσεων, προσδίδοντας ευελιξία (agile) στο έργο. Η τεχνική αυτή χρησιμοποιείται εκτενώς σε ευέλικτα πλαίσια ανάπτυξης, όπως το Scrum, ωστόσο εφαρμόζεται αποτελεσματικά και σε ατομικά έργα με σκοπό τη βελτίωση της οργάνωσης και του σχεδιασμού. Η σύνταξη των ιστοριών ακολούθησε το πρότυπο που προτείνει ο M. Cohn [70], σύμφωνα με το οποίο κάθε ιστορία εκφράζεται με τη μορφή:

«Ως ένας [τύπος χρήστη], θέλω να [ενέργεια], ώστε να [σκοπός]».

Μετά από κάθε ιστορία χρήστη, αναλύονται τα κριτήρια αποδοχής (acceptance criteria), τα οποία αποτυπώνουν τις αναμενόμενες εκβάσεις της αντίστοιχης ενέργειας από την οπτική του χρήστη.

Παρακάτω, παρατίθενται ορισμένες ενδεικτικές ιστορίες χρηστών, οι οποίες επιλέχθηκαν για να παρουσιαστούν εντός του κυρίως κειμένου. Ο πλήρης κατάλογος με τις ιστορίες χρηστών περιλαμβάνεται στο Παράρτημα Γ: «Ιστορίες Χρηστών & Κριτήρια Αποδοχής».

3.3.1 Ιστορίες Χρηστών (Μη Πιστοποιημένος Χρήστης / Επισκέπτης)

Τίτλος: Προβολή γενικού ωρολογίου προγράμματος ανά Τμήμα
Ως επισκέπτης **Θέλω να** δω το ωρολόγιο πρόγραμμα Τμήματος για συγκεκριμένο εξάμηνο **Ώστε** να ενημερωθώ για τις ώρες και αίθουσες διδασκαλίας κάθε μαθήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Ωρολόγιο Πρόγραμμα» από το κεντρικό μενού. • Επιλέγει από αναπτυσσόμενη λίστα τη Σχολή και το Τμήμα που τον ενδιαφέρει.

• Επιλέγει από αναπτυσσόμενη λίστα το εξάμηνο που τον ενδιαφέρει (προεπιλεγμένα εμφανίζεται το τρέχον εξάμηνο. • Το σύστημα εμφανίζει το εβδομαδιαίο ωρολόγιο πρόγραμμα του επιλεγμένου Τμήματος, με τις ημέρες, ώρες και αίθουσες διδασκαλίας για κάθε μάθημα.

Πίνακας 3-1-US: Προβολή γενικού ωρολογίου προγράμματος ανά Τμήμα

3.3.2 Ιστορίες Χρηστών (Κοινές Λειτουργίες Πιστοποιημένων Χρηστών)

Τίτλος: Είσοδος στην πλατφόρμα με χρήση SSO (Single-Sign-On)
Ως πιστοποιημένος χρήστης **Θέλω να** συνδεθώ στην πλατφόρμα με χρήση του ιδρυματικού μου λογαριασμού **Ώστε** να έχω πρόσβαση σε λειτουργίες πιστοποιημένων χρηστών της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος με SSO» στη σελίδα. • Ο χρήστης μεταφέρεται στην κεντρική υπηρεσία πιστοποίησης χρηστών του Ιδρύματος. • Εισάγει έγκυρα διαπιστευτήρια (ιδρυματικό email και κωδικό πρόσβασης). **Σενάριο 1: Επιβεβαίωση διαπιστευτηρίων** • Η σύνδεση ολοκληρώνεται και ο χρήστης μεταφέρεται στην πλατφόρμα. **Σενάριο 1α: Επιτυχής λήψη επιβεβαίωσης και δεδομένων μέσω SSO** • Ο χρήστης μεταφέρεται στην κεντρική σελίδα πιστοποιημένων χρηστών. • Ο χρήστης ενημερώνεται ότι μπορεί να δημιουργήσει τοπικό κωδικό εισόδου, εάν δεν το έχει ήδη πράξει, για είσοδο απευθείας από την πλατφόρμα. **Σενάριο 1β: Αποτυχία λήψη επιβεβαίωσης ή δεδομένων μέσω SSO ή διακοπή** • Ο χρήστης μεταφέρεται στην αρχική σελίδα εισόδου, χωρίς να έχει πραγματοποιηθεί σύνδεση/είσοδος και εμφανίζεται μήνυμα αποτυχίας εισόδου και παρότρυνση επαναπροσπάθειας.

Σενάριο 2: Απόρριψη διαπιστευτηρίων από το Ίδρυμα
• Εμφανίζεται στον χρήστη μήνυμα αποτυχίας, όπως αυτό έχει οριστεί στη σελίδα SSO εισόδου. • Ο χρήστης μπορεί να επιλέξει εκ νέου προσπάθεια εισόδου.

Πίνακας 3-2-US: Είσοδος στην πλατφόρμα με χρήση SSO (Single-Sign-On)

3.3.3 Ιστορίες Χρηστών (Διαχειριστής)

Τίτλος: Προβολή και διαχείριση ενεργών και επιβεβαιωμένων Διδασκόντων
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω λίστα Διδασκόντων και να διαχειριστώ τον λογαριασμό Διδάσκοντα **Ώστε** να επεξεργαστώ τα στοιχεία του, τα μαθήματα που διδάσκει, να αλλάξω την κατάσταση λογαριασμού και να ελέγξω ιστορικό ενεργειών του.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Διδασκόντων» από το κεντρικό μενού. • Εμφανίζεται λίστα με τους ενεργούς και επιβεβαιωμένους λογαριασμούς Διδασκόντων. • Η λίστα περιλαμβάνει το ονοματεπώνυμο του Διδάσκοντα, το Τμήμα που έχει οριστεί, τα μαθήματα που διδάσκει και την κατάσταση λογαριασμού. • Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά. • Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Κατάσταση Λογαριασμού»: ενεργός / ανενεργός / όλοι ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων ή «Μάθημα»: επιλογή από λίστα διαθέσιμων μαθημάτων), οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα. • Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, Τμήμα, μάθημα,

τηλέφωνο, email, αριθμό μητρώου), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν. • Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα. • Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διδάσκοντα. • Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διδάσκοντα με όλα τα σχετικά πεδία. • Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού, όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, οργανική θέση / Τμήμα που ανήκει, μαθήματα που διδάσκει, κατάσταση λογαριασμού (ενεργός / ανενεργός). • Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Ο χρήστης μπορεί να επιλέξει την προβολή ιστορικού κινήσεων εισόδου/εξόδου και ενεργειών που έχει πραγματοποιήσει ο συγκεκριμένος Διδάσκων, επιλέγοντας το κουμπί «Προβολή Ιστορικού Χρήστη». • Μπορεί να επιλέξει τη λήψη του ιστορικού χρήστη σε αρχείο CSV και τη διαγραφή του ιστορικού παλαιότερων μηνών, με ορισμό χρονικού ορίου διαγραφής (π.χ. 1, 2, 3 μήνες) από λίστα (το ιστορικό διατηρείται κατόπιν σε εξωτερικό αρχείο στον διακομιστή). • Ο χρήστης επιστρέφει στη λίστα ενεργών λογαριασμών, επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας 3-3-US: Προβολή και διαχείριση ενεργών και επιβεβαιωμένων Διδασκόντων

Τίτλος: Διαμόρφωση ωρολογίου προγράμματος
Ως Διαχειριστής Τμήματος **Θέλω να** ορίσω σε εβδομαδιαία βάση τα Μαθήματα, τις Αίθουσες και τους Διδάσκοντες **Ώστε** να διαμορφώσω το ωρολόγιο πρόγραμμα για το επερχόμενο εξάμηνο.
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον έχουν καθοριστεί οι παράμετροι του επερχόμενου εξαμήνου, εμφανίζεται στον χρήστη η επιλογή «Διαμόρφωση Ωρολογίου Προγράμματος».

- Ο χρήστης επιλέγει τον σύνδεσμο «Διαμόρφωση Ωρολογίου Προγράμματος» από το κεντρικό μενού.
- Εμφανίζεται ημερολόγιο εβδομάδας σε μορφή πίνακα (Δευτέρα έως Κυριακή, ανάλογα με τις παραμέτρους εξαμήνου) και ώρες με θέσεις (slots) ανά μία ώρα.
- Εφόσον υπάρχουν προσωρινά αποθηκευμένα δεδομένα από προηγούμενη χρήση, αυτά εμφανίζονται στον πίνακα του ημερολογίου εβδομάδας.
- Ο χρήστης επιλέγει μία θέση στον πίνακα.
- Εμφανίζεται αναδυόμενο παράθυρο με πεδία: Μάθημα (επιλογή από λίστα), διάρκεια μαθήματος σε ώρες (επιλογή από λίστα 1, 2, 3 κ.ο.κ.), Διδάσκων (επιλογή από λίστα), Αίθουσα (επιλογή από λίστα).
- Οι επιλογές από λίστα έχουν τιμές, οι οποίες υπολογίζονται αυτόματα, σύμφωνα με τα δεδομένα που έχουν καταχωρηθεί στο σύστημα, καθώς και με τα δηλωθέντα στο τρέχον ωρολόγιο πρόγραμμα δεδομένα, για αποφυγή παραβίασης περιορισμών.
- Σε περίπτωση επιλογής που παραβιάζει κάποιον περιορισμό, ο οποίος δεν υπολογίζεται αυτόματα ή υπολογίζεται κατά τον ορισμό τιμής πεδίου, εμφανίζεται σχετικό μήνυμα.
- Σε παραβίαση αυστηρού περιορισμού, όπως π.χ. επικάλυψη μαθημάτων κορμού, εμφανίζεται προειδοποιητικό μήνυμα και δεν επιτρέπεται οριστική αποθήκευση.
- Σε παραβίαση προαιρετικού περιορισμού, όπως π.χ. δήλωση ωρών επιθυμητής διακοπής σίτισης, εμφανίζεται ενημερωτικό μήνυμα, ενώ επιτρέπεται οριστική αποθήκευση.
- Στον χρήστη εμφανίζονται γενικά στοιχεία, που αφορούν τα δηλωθέντα δεδομένα στο ωρολόγιο πρόγραμμα για την διευκόλυνση της διαμόρφωσης, όπως: εκκρεμείς ώρες συμπλήρωσης ελάχιστου ορίου ανά Μάθημα, συνολικές ώρες ανά Μάθημα, συνολικές ώρες ανά Διδάσκοντα, χρησιμοποιούμενες Αίθουσες.
- Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης και τον χρήστη που πραγματοποίησε την τροποποίηση (ονοματεπώνυμο Διαχειριστή Τμήματος).
- Ο χρήστης μπορεί να επιλέξει το κουμπί «Προσωρινή Αποθήκευση».

Σενάριο 1: Αποτυχία προσωρινής αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση της αιτίας.

• Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης και ο χρήστης μπορεί να συνεχίσει με τη διαμόρφωση του προγράμματος. • Εφόσον έχουν συμπεριληφθεί όλα τα Μαθήματα εξαμήνου του Τμήματος και έχουν συμπληρωθεί οι ελάχιστες απαιτούμενες ώρες ανά Μάθημα, εμφανίζεται το κουμπί «Οριστική Αποθήκευση Ωρολογίου Προγράμματος». • Ο χρήστης επιλέγει το κουμπί «Οριστική Αποθήκευση Ωρολογίου Προγράμματος». **Σενάριο 3: Αποτυχία οριστικής αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση της αιτίας. • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 4: Επιτυχής οριστική αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Προβάλλεται στον χρήστη το οριστικό ωρολόγιο πρόγραμμα, χωρίς δυνατότητα περαιτέρω επεξεργασίας.

Πίνακας 3-4-US: Διαμόρφωση ωρολογίου προγράμματος

3.3.4 Ιστορίες Χρηστών (Διδάσκων)

Τίτλος: Δήλωση προτιμήσεων και διαθεσιμότητας ωρών/ημερών διδασκαλίας
Ως Διδάσκων **Θέλω να** δηλώσω τις ημέρες και ώρες της εβδομάδας στις οποίες προτιμώ ή δεν μπορώ να διδάξω **Ώστε** να να ληφθούν υπόψη οι αντίστοιχες πληροφορίες κατά τη δημιουργία του ωρολογίου προγράμματος από τη Γραμματεία του Τμήματος.

Κριτήρια Αποδοχής (Acceptance criteria):

- Όταν βρίσκεται σε ισχύ η περίοδος καθορισμού του ωρολογίου προγράμματος, εμφανίζεται στον χρήστη ενεργό το κουμπί «Δήλωση Ωρών Διδασκαλίας».
- Ο χρήστης επιλέγει την ενότητα «Δήλωση Ωρών Διδασκαλίας» μέσω του κεντρικού μενού.
- Εμφανίζεται εβδομαδιαίο ημερολόγιο με ωριαίες χρονικές θέσεις (slots).
- Ο χρήστης επιλέγει μία συγκεκριμένη χρονική θέση και εμφανίζεται λίστα επιλογών: «Προτιμώμενη Ώρα» και «Μη Διαθέσιμη Ώρα».
- Ο χρήστης επιλέγει μία τιμή από τη λίστα επιλογών.
- Η επιλεγμένη ωριαία θέση εμφανίζεται με κατάλληλο χρώμα και το κείμενο της επιλογής.
- Ο χρήστης έχει τη δυνατότητα να επεξεργαστεί μία ήδη καταχωρημένη ωριαία θέση και να αναιρεί τη δήλωσή του.
- Ο χρήστης μπορεί να ορίσει προτίμηση ή μη διαθεσιμότητα για ολόκληρη ημέρα, χρησιμοποιώντας τη σχετική επιλογή που εμφανίζεται πάνω από τη συγκεκριμένη στήλη του ημερολογίου.
- Ο χρήστης πατά το κουμπί «Αποθήκευση Επιλογών» για να καταχωρήσει τις προτιμήσεις και τις μη διαθέσιμες ώρες.
- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης της δήλωσης.
- Η δήλωση μπορεί να τροποποιείται καθ’ όλη τη διάρκεια της περιόδου που έχει οριστεί από τη Διεύθυνση του Τμήματος, ακολουθώντας την ίδια διαδικασία.

Σενάριο 1: Αποτυχία αποθήκευσης

- Εμφανίζεται μήνυμα που ενημερώνει τον χρήστη ότι η καταχώριση δεν ολοκληρώθηκε και του ζητά να επαναλάβει την προσπάθεια.

Πίνακας 3-5-US: Δήλωση προτιμήσεων και διαθεσιμότητας ωρών/ημερών διδασκαλίας

3.3.5 Ιστορίες Χρηστών (Φοιτητής)

Τίτλος: Προσθήκη μαθήματος σε λίστα ενδιαφέροντος (watchlist)
Ως Φοιτητής **Θέλω να** προσθέσω ένα μάθημα προηγούμενου εξαμήνου στη λίστα ενδιαφέροντος **Ώστε** να εμφανίζεται στο εξατομικευμένο μου ωρολόγιο πρόγραμμα και να ενημερώνομαι για τυχόν έκτατες αλλαγές.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης ανοίγει από το μενού την επιλογή «Προβολή Ωρολογίου Προγράμματος». • Προβάλλεται το εξατομικευμένο πρόγραμμα του τρέχοντος εξαμήνου, μαζί με τυχόν μαθήματα που έχουν ήδη προστεθεί στη λίστα ενδιαφέροντος. • Ο χρήστης επιλέγει διαφορετικό εξάμηνο από το τρέχον μέσω σχετικής λίστας επιλογών. • Προβάλλεται το εβδομαδιαίο ωρολόγιο πρόγραμμα για το επιλεγμένο εξάμηνο. • Ο χρήστης επιλέγει μάθημα, πατώντας πάνω στη θέση του στο ημερολόγιο. • Εμφανίζεται παράθυρο με αναλυτικές πληροφορίες (τίτλος, εξάμηνο, διδάσκων, αίθουσα) και κουμπί «Προσθήκη στη λίστα Ενδιαφέροντος». • Με την επιλογή «Προσθήκη στη λίστα Ενδιαφέροντος», το μάθημα καταχωρείται στη λίστα του χρήστη και το κουμπί αλλάζει σε «Αφαίρεση από τη λίστα Ενδιαφέροντος». • Ο χρήστης επιστρέφει στο εξατομικευμένο πρόγραμμα για το εξάμηνο που παρακολουθεί, στο οποίο εμφανίζεται με ξεχωριστό χρωματισμό το μάθημα που πρόσθεσε στη λίστα ενδιαφέροντος.

Πίνακας 3-6-US: Προσθήκη μαθήματος σε λίστα ενδιαφέροντος (watchlist)

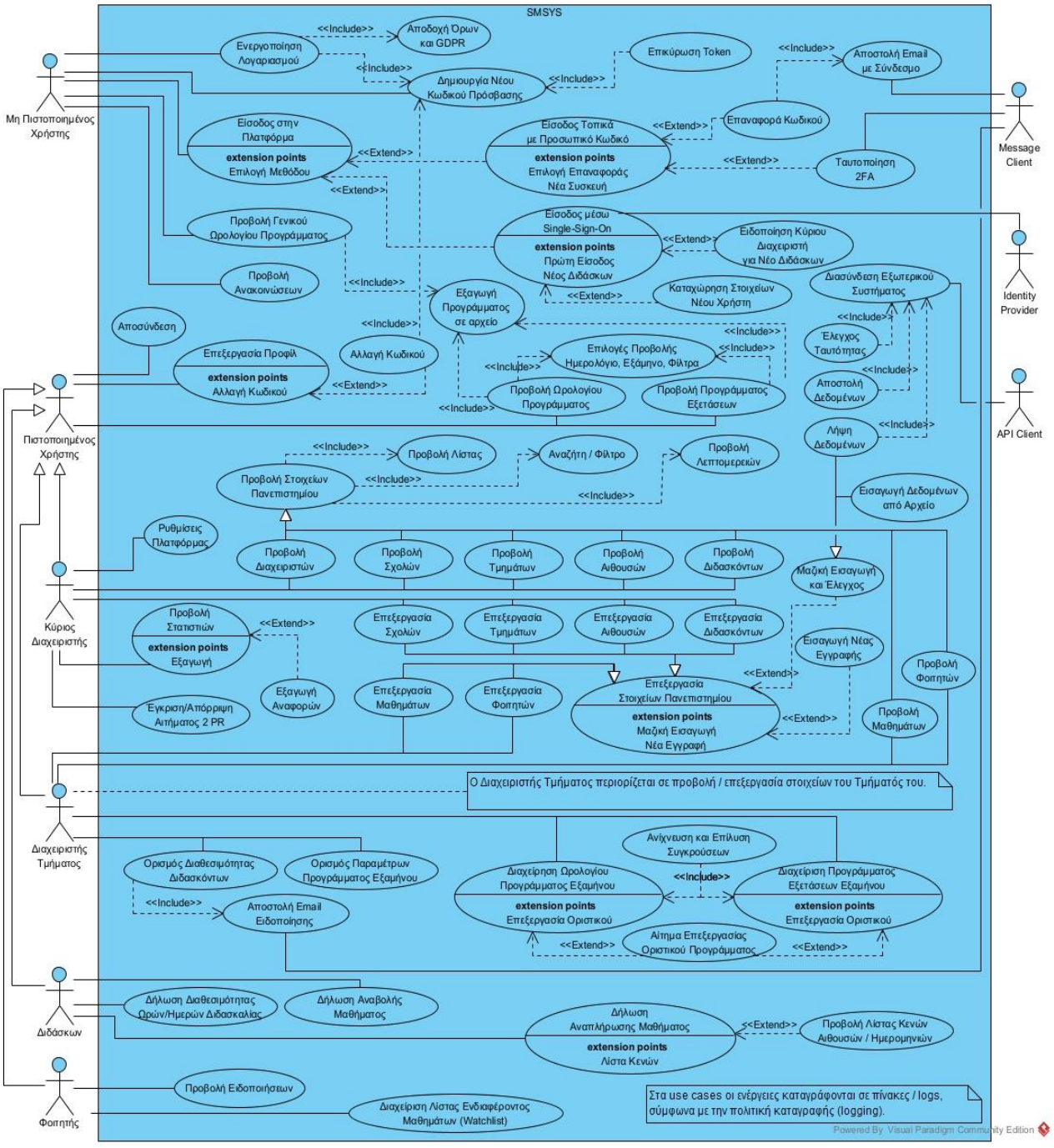
3.4 Περιπτώσεις Χρήσης (Use Case Scenarios) & Διάγραμμα

Οι Περιπτώσεις Χρήσης (Use Cases) αποτελούν αναλυτικές περιγραφές των λειτουργιών και ενεργειών που εκτελεί το λογισμικό σε αλληλεπίδραση με τον χρήστη ή άλλο εξωτερικό σύστημα, ώστε να ολοκληρωθεί μια λειτουργική απαίτηση [71]. Σε αντιδιαστολή με τις Ιστορίες Χρηστών που αποτυπώνουν τις λειτουργίες από την οπτική του χρήστη, οι Περιπτώσεις Χρήσης εστιάζουν στο τι πρέπει να κάνει το σύστημα για να ολοκληρωθεί κάποια ενέργεια, διατηρώντας απλή και οργανωμένη δομή σεναρίων, ώστε να προκύπτει σαφής εικόνα των λειτουργιών και των ορίων του. Παράλληλα και βάσει των περιγραφών τους, παρατίθεται και το αντίστοιχο διάγραμμα περιπτώσεων χρήσης (UML), το οποίο συμπληρώνει τη μοντελοποίηση των σχέσεων χρηστών–συστήματος και διευκολύνει τον έλεγχο πληρότητας και συνέπειας των απαιτήσεων.

Σημειώνεται ότι σε κάθε Περίπτωση Χρήσης η οποία έχει ως αποτέλεσμα: α) την τροποποίηση δεδομένων, β) την εκτέλεση ενέργειας από και προς τον χρήστη (π.χ. αποστολή email, επιτυχής είσοδος, αποτυχία επικοινωνίας, κ.ο.κ.) ή γ) την πρόκληση κάποιου σφάλματος, το σύστημα πραγματοποιεί αναλυτική καταγραφή (logging) των ενεργειών, των σφαλμάτων και των αλλαγών δεδομένων. Οι καταγραφές αυτές τηρούνται τόσο σε πίνακες της βάσης δεδομένων όσο και σε αρχεία κειμένου (.log), σε περίπτωση που η επικοινωνία με τη βάση δεδομένων δεν είναι διαθέσιμη. Για λόγους σαφήνειας και ευκρίνειας των διαγραμμάτων, οι μηχανισμοί καταγραφής δεν απεικονίζονται στα διαγράμματα ή στις επιμέρους περιγραφές των περιπτώσεων χρήσης.

Στις επόμενες υποενότητες παρουσιάζονται ενδεικτικά βασικές περιγραφές περιπτώσεων χρήσης του συστήματος, ακολουθώντας τυποποιημένη φόρμα περιγραφής UP (τίτλος, id, σύντομη περιγραφή, χειριστές, προϋποθέσεις, κύρια ροή, εναλλακτικές ροές, μετακατάσταση, κατάσταση εξόδου), όπως προτείνεται στο συνοδευτικό υλικό του μαθήματος ΠΛΗ24 του ΕΑΠ για τον σχεδιασμό λογισμικού [72]. Άλλα βασικά σενάρια περιπτώσεων χρήσης περιλαμβάνονται στο Παράρτημα Δ.

3.4.1 Διάγραμμα Περιπτώσεων Χρήσης (UML)



Σχήμα 1 - Διάγραμμα Περιπτώσεων Χρήσης (UML)

3.4.2 Περιγραφές Περιπτώσεων Χρήσης (φόρμες UP)

Τίτλος: Διαχείριση Ωρολογίου Προγράμματος Εξαμήνου **ID:** UC1
Σύντομη περιγραφή: Η Περίπτωση Χρήσης επιτρέπει στον χειριστή τη δημιουργία/επεξεργασία ωρολογίου προγράμματος εξαμήνου του Τμήματος που διαχειρίζεται, εμφανίζοντας σε εβδομαδιαίο ημερολόγιο με θέσεις (slots) ανά ώρα και ημέρα επιλογές ορισμού μαθημάτων, διδασκόντων και αιθουσών. Εξειδικεύει τις λειτουργίες δημιουργίας και επεξεργασίας ωρολογίου προγράμματος εξαμήνου.
Χειριστές: Διαχειριστής Τμήματος
Κατάσταση Εισόδου (Προϋποθέσεις): Υπάρχει καταχωρημένο πρόγραμμα μαθημάτων στο εξάμηνο. Έχουν δηλωθεί διαθέσιμες αίθουσες και διδάσκοντες. Έχει ανοίξει η περίοδος επεξεργασίας επερχόμενου εξαμήνου από τον Κύριο Διαχειριστή. Αν το ωρολόγιο πρόγραμμα είναι πρόχειρο, ο χρήστης μπορεί να το δημιουργήσει ή επεξεργαστεί απευθείας. Αν το ωρολόγιο πρόγραμμα είναι οριστικοποιημένο, έχει προηγηθεί επιτυχής ολοκλήρωση της Περίπτωση Χρήσης UC3 «Αίτημα Επεξεργασίας Οριστικού Προγράμματος» (extend), και το σύστημα επιτρέπει την επεξεργασία.
Βασική ροή: 1. Ο χρήστης επιλέγει «Διαχείριση Ωρολογίου Προγράμματος». 2. Ο χρήστης επιλέγει εξάμηνο από λίστα επιλογής (Α, Β, Γ, ...). 3. Το σύστημα εμφανίζει εβδομαδιαίο ημερολόγιο με τις διαθέσιμες χρονικές ζώνες ανά ώρα και ημέρα, καθώς και τα ήδη καταχωρημένα μαθήματα εάν υπάρχουν. 4. Ο χρήστης επιλέγει θέση χρονικής ζώνης για εισαγωγή/επεξεργασία μαθήματος. 5. Ο χρήστης επιλέγει μάθημα, διδάσκοντα και αίθουσα για εισαγωγή στη συγκεκριμένη χρονική ζώνη. 6. Το σύστημα ενεργοποιεί την Περίπτωση Χρήσης UC2 «Ανίχνευση και Επίλυση Συγκρούσεων».

7. Το σύστημα καταχωρεί το μάθημα στη θέση στο ωρολόγιο πρόγραμμα.
8. Ο χρήστης επαναλαμβάνει τα βήματα 4 έως 7 μέχρι την ολοκλήρωση όλων των μαθημάτων εξαμήνου και τη συμπλήρωση των απαιτούμενων ωρών.
9. Εφόσον έχουν συμπληρωθεί τα απαιτούμενα μαθήματα ο χρήστης επιλέγει την οριστική δημιουργία και αποθήκευση του ωρολογίου προγράμματος.
10. Το σύστημα αποθηκεύει το ωρολόγιο πρόγραμμα.
11. Το σενάριο τερματίζεται.

Εναλλακτική ροή 1 – «Επεξεργασία θέσης»:

5α. Προβάλλεται παράθυρο με τις επιλογές στη συγκεκριμένη ωριαία θέση (μάθημα, διδάσκοντα και αίθουσα).

5β. Ο χρήστης τροποποιεί τις συγκεκριμένες επιλογές του μαθήματος στην ωριαία θέση.

5γ. Η ροή μεταφέρεται στο βήμα 6 της βασικής ροής.

Εναλλακτική ροή 2 – «Αφαίρεση μαθήματος από θέση»:

5α. Προβάλλεται παράθυρο με τις επιλογές στη συγκεκριμένη ωριαία θέση και κουμπί αφαίρεσης μαθήματος από τη θέση εφόσον έχει οριστεί μάθημα.

5β. Ο χρήστης επιλέγει το κουμπί «Αφαίρεση Μαθήματος από ωριαία θέση».

5γ. Το μάθημα αφαιρείται από την ωριαία θέση και ο χρήστης επανέρχεται σε προβολή του ημερολογίου.

5δ. Η ροή μεταφέρεται στο βήμα 3 της βασικής ροής.

Εναλλακτική ροή 3 – «Εντοπισμός συγκρούσεων με Παράκαμψη»:

7α. Αν εντοπιστεί σύγκρουση, εμφανίζεται Προειδοποίηση και πιθανές Προτάσεις διαθέσιμων εναλλακτικών.

7β. Ο χρήστης επιλέγει «Παράκαμψη», εφόσον η σύγκρουση δεν αφορά σε αυστηρό περιορισμό, όπως αυτοί έχουν οριστεί στα δεδομένα του συστήματος (π.χ. διπλή ανάθεση διδάσκοντα ή χρήση αίθουσας σε ίδια ώρα από άλλο μάθημα).

7γ. Η ροή μεταφέρεται στο βήμα 8 της βασικής ροής.

Εναλλακτική ροή 4 – «Εντοπισμός συγκρούσεων χωρίς Παράκαμψη»:

7α. Αν εντοπιστεί σύγκρουση, εμφανίζεται Προειδοποίηση και πιθανές Προτάσεις διαθέσιμων εναλλακτικών.

7β. Η σύγκρουση αφορά σε αυστηρό περιορισμό, όπως αυτοί έχουν οριστεί στα δεδομένα του συστήματος (π.χ. διπλή ανάθεση διδάσκοντα ή χρήση αίθουσας σε ίδια ώρα από άλλο μάθημα) και δεν υπάρχει δυνατότητα παράκαμψης.

7γ. Ο χρήστης έχει τη δυνατότητα τροποποίησης των επιλογών για τη συγκεκριμένη θέση ή διαγραφής των επιλογών της θέσης. 7δ. Η ροή μεταφέρεται στο βήμα 5 της βασικής ροής.
Μη λειτουργικές απαιτήσεις: Η περίοδος δημιουργίας και επεξεργασίας επερχόμενου εξαμήνου είναι ενεργή, όπως έχει ορίσει ο Κύριος Διαχειριστής.
Κατάσταση εξόδου: Το ωρολόγιο πρόγραμμα του εξαμήνου έχει αποθηκευτεί. Είναι διαθέσιμο για προβολή από όλους τους ρόλους χρηστών.

Πίνακας 3-7-UC1: Διαχείριση Ωρολογίου Προγράμματος Εξαμήνου

Τίτλος: Ανίχνευση και Επίλυση Συγκρούσεων Ωρολογίου Προγράμματος **ID:** UC2
Σύντομη περιγραφή: Η Περίπτωση Χρήσης καλύπτει τον αυτόματο έλεγχο του συστήματος για πιθανές συγκρούσεις κατά την τοποθέτηση μαθημάτων στο ωρολόγιο πρόγραμμα, καθώς και την παρουσίαση διαθέσιμων εναλλακτικών για την επίλυσή τους.
Χειριστές: Διαχειριστής Τμήματος
Κατάσταση Εισόδου (Προϋποθέσεις): Το σύστημα βρίσκεται στη διαδικασία δημιουργίας ωρολογίου προγράμματος. Υπάρχουν καταχωρημένα μαθήματα, διδάσκοντες και αίθουσες. Ο χρήστης έχει εισαγάγει δεδομένα σε επιλεγμένη χρονική ζώνη του ημερολογίου (ημέρα, ώρα, μάθημα, διδάσκον, αίθουσα).
Βασική ροή: 1. Το σύστημα πραγματοποιεί έλεγχο για πιθανές συγκρούσεις και περιορισμούς. 2. Αν δεν εντοπιστούν συγκρούσεις, επιστρέφεται η τιμή «Χωρίς σύγκρουση». **Εναλλακτική ροή 1 – «Εντοπισμός σύγκρουσης»:** 2α. Το σύστημα εντοπίζει μία ή περισσότερες συγκρούσεις (π.χ. διδάσκων με περιορισμό την συγκεκριμένη ημέρα/ώρα, δεσμευμένη αίθουσα, παράλληλο μάθημα ίδιου

εξαμήνου). 2β. Εμφανίζονται διαθέσιμες εναλλακτικές επιλογές (αλλαγή ώρας, αίθουσας, διδάσκων) ανάλογα με τις συγκρούσεις και περιορισμούς που εντοπίστηκαν. 2γ. Ο χρήστης επιλέγει μία από τις προτεινόμενες λύσεις ή ακυρώνει την καταχώριση. 2δ. Η ροή μεταφέρεται στο βήμα 3.
Μη λειτουργικές απαιτήσεις: Το σύστημα βρίσκεται σε κατάσταση επεξεργασίας ωρολογίου προγράμματος.
Κατάσταση εξόδου: «Χωρίς σύγκρουση». «Σύγκρουση» με υπόδειξη, εναλλακτικές (π.χ. σύγκρουση μαθήματος, διδάσκοντα, αίθουσας) και είδος περιορισμού (αυστηρός ή προαιρετικός) .

Πίνακας 3-8-UC2: Ανίχνευση και Επίλυση Συγκρούσεων Ωρολογίου Προγράμματος

Τίτλος: Αίτημα Επεξεργασίας Οριστικού Προγράμματος **ID:** UC3
Σύντομη περιγραφή: Η παρούσα Περίπτωση Χρήσης περιγράφει τη διαδικασία με την οποία ο Διαχειριστής Τμήματος υποβάλλει αίτημα επεξεργασίας ωρολογίου προγράμματος ή εξετάσεων που έχει ήδη οριστικοποιηθεί και η περίοδος επεξεργασίας έχει λήξει. Το αίτημα αποστέλλεται προς τον Κύριο Διαχειριστή για έγκριση.
Χειριστές: Διαχειριστής Τμήματος
Κατάσταση Εισόδου (Προϋποθέσεις): Η περίοδος επεξεργασίας προγραμμάτων εξαμήνου έχει λήξει. Δεν υπάρχει ήδη αίτημα επεξεργασίας για το συγκεκριμένο πρόγραμμα σε εκκρεμότητα ή που έχει εγκριθεί.
Βασική ροή: 1. Ο χρήστης επιλέγει «Διαχείριση Ωρολογίου Προγράμματος» ή «Διαχείριση Προγράμματος Εξετάσεων». 2. Ο χρήστης επιλέγει εξάμηνο από λίστα επιλογής (1ο, 2ο, 3ο, ...).

3. Το σύστημα εμφανίζει το οριστικοποιημένο ωρολόγιο πρόγραμμα ή αντίστοιχα το πρόγραμμα εξετάσεων ανάλογα με την επιλογή. 4. Ο χρήστης επιλέγει το κουμπί «Υποβολή Αιτήματος Επεξεργασίας». 5. Το σύστημα καταχωρεί το αίτημα επεξεργασίας και εμφανίζει μήνυμα επιτυχούς αποστολής μαζί με την ημερομηνία και ώρα καταχώρησης. **Εναλλακτική ροή 1 – «Αποτυχία αποστολής αιτήματος»:** 2α. Το σύστημα εμφανίζει μήνυμα αποτυχίας υποβολής του αιτήματος και δυνατότητα νέας προσπάθειας. 2β. Η ροή επιστρέφει στο βήμα 4 της βασικής ροής.
Μη λειτουργικές απαιτήσεις: Δεν εφαρμόζεται σε αυτό το Use Case.
Κατάσταση εξόδου: Το αίτημα βρίσκεται σε κατάσταση «Αίτημα Επεξεργασίας Οριστικού Προγράμματος Εκκρεμές» και αναμένεται έγκριση από τον Κύριο Διαχειριστή.

Πίνακας 3-9-UC3: Αίτημα Επεξεργασίας Οριστικού Προγράμματος

Τίτλος: Προβολή Ωρολογίου Προγράμματος **ID:** UC4
Σύντομη περιγραφή: Η Περίπτωση Χρήσης επιτρέπει στον Πιστοποιημένο Χρήστη να εμφανίσει το ωρολόγιο πρόγραμμα εξαμήνου βάσει φίλτρων και επιλογών.
Χειριστές: Πιστοποιημένος Χρήστης
Κατάσταση Εισόδου (Προϋποθέσεις): Ο χρήστης έχει συνδεθεί στο σύστημα.
Βασική ροή: 1. Ο χρήστης επιλέγει «Προβολή Ωρολογίου Προγράμματος». 2. Το σύστημα εμφανίζει το ωρολόγιο πρόγραμμα του τρέχοντος εξαμήνου για το Τμήμα που είναι εγγεγραμμένος ο χρήστης. 3. Σε περίπτωση που ο χρήστης δεν έχει συγκεκριμένο Τμήμα, όπως για παράδειγμα ο

Κύριος Διαχειριστής, εμφανίζεται επιλογή Τμήματος από λίστα. 4. Ο χρήστης επιλέγει το είδος προβολής ημερολογίου (Εβδομαδιαίο ή Ημερολογιακό). 5. Ο χρήστης μπορεί να επιλέξει από λίστες επιλογής το Εξάμηνο και το Τμήμα για προβολή προγράμματος, ενώ μπορεί να αναζητήσει με λέξεις κλειδιά μαθήματα, διδάσκοντες, και αίθουσες. 6. Η προβολή του ημερολογίου ενημερώνεται ανάλογα με τις επιλογές του χρήστη. 7. Ο φοιτητής μπορεί να επιλέξει την εξαγωγή του προγράμματος σε αρχείο, πατώντας το ανάλογο κουμπί, οπότε ενεργοποιείται η Περίπτωση Χρήσης UC5. **Εναλλακτική ροή 1 – «Κενό Πρόγραμμα»:** 2α. Εμφανίζεται ενημερωτικό μήνυμα «Το πρόγραμμα δεν έχει ακόμα οριστικοποιηθεί». **Εναλλακτική ροή 2 – «Ο χρήστης είναι Φοιτητής»:** 2α. Στο ημερολόγιο εμφανίζεται το ωρολόγιο πρόγραμμα του εξαμήνου που φοιτά και του Τμήματος που ανήκει ο Φοιτητής. 2β. Το ημερολόγιο περιλαμβάνει μαθήματα που ο Φοιτητής έχει προσθέσει ως «Τα Μαθήματά μου» (watchlist). 2γ. Η ροή επιστρέφει στο βήμα 4.
Μη λειτουργικές απαιτήσεις: Δεν εφαρμόζεται σε αυτό το Use Case.
Κατάσταση εξόδου: Το ωρολόγιο πρόγραμμα έχει διατεθεί προς προβολή στον χρήστη.

Πίνακας 3-10-UC4: Προβολή Ωρολογίου Προγράμματος

Τίτλος: Εξαγωγή Προγράμματος σε αρχείο **ID:** UC5
Σύντομη περιγραφή: Η Περίπτωση Χρήσης επιτρέπει στον Πιστοποιημένο Χρήστη να εξάγει το επιλεγμένο ωρολόγιο πρόγραμμα εξαμήνου που προβάλλεται στην οθόνη του.
Χειριστές: Πιστοποιημένος Χρήστης
Κατάσταση Εισόδου (Προϋποθέσεις):

Ο χρήστης έχει συνδεθεί στο σύστημα. Ο χρήστης έχει επιλέξει κάποιο ωρολόγιο πρόγραμμα με βάσει τις επιλογές λίστας, φίλτρων ή αναζήτησης.
Βασική ροή: 1. Ο φοιτητής επιλέγει το κουμπί «Εξαγωγή σε PDF». 2. Το σύστημα δημιουργεί αρχείο PDF και το εμφανίζει στο χρήστη με δυνατότητα λήψης ή εκτύπωσης του αρχείου. **Εναλλακτική ροή 1 – «Εξαγωγή σε iCalendar»:** 2α. Ο φοιτητής επιλέγει «Εξαγωγή σε iCalendar». 2β. Το σύστημα δημιουργεί αρχείο κειμένου μορφής iCalendar (.ics) και παραπέμπει τον χρήστη για λήψη του αρχείου. **Εναλλακτική ροή 2 – «Εξαγωγή σε Excel»:** 2α. Ο φοιτητής επιλέγει «Εξαγωγή σε Excel». 2β. Το σύστημα δημιουργεί αρχείο Excel και παραπέμπει τον χρήστη για λήψη του αρχείου.
Μη λειτουργικές απαιτήσεις: Δεν εφαρμόζεται σε αυτό το Use Case.
Κατάσταση εξόδου: Το ωρολόγιο πρόγραμμα έχει δημιουργηθεί σε αρχείο για λήψη από τον χρήστη.

Πίνακας 3-11-UC5: Εξαγωγή Προγράμματος σε αρχείο

Τίτλος: Διαχείριση Προγράμματος Εξετάσεων Εξαμήνου **ID:** UC6
Σύντομη περιγραφή: Η περίπτωση χρήσης επιτρέπει στον χρήστη να δημιουργεί/επεξεργαστεί πρόγραμμα εξετάσεων με βάση τα μαθήματα του εξαμήνου, τις διαθέσιμες αίθουσες και το ωρολόγιο πρόγραμμα, εμφανίζοντας σε ημερολόγιο με θέσεις (slots) ανά ώρα και ημερομηνία επιλογές ορισμού μαθημάτων.
Χειριστές: Διαχειριστής Τμήματος
Κατάσταση Εισόδου (Προϋποθέσεις):

Υπάρχει καταχωρημένο πρόγραμμα μαθημάτων. Έχουν δηλωθεί διαθέσιμες αίθουσες και ημερομηνίες έναρξης/λήξης. Υπάρχει καταχωρημένο ωρολόγιο πρόγραμμα. Έχει ανοίξει η περίοδος επεξεργασίας επερχόμενου εξαμήνου από τον Κύριο Διαχειριστή. Αν το πρόγραμμα εξετάσεων είναι πρόχειρο, ο χρήστης μπορεί να το δημιουργήσει ή επεξεργαστεί απευθείας. Αν το πρόγραμμα εξετάσεων είναι οριστικοποιημένο, έχει προηγηθεί επιτυχής ολοκλήρωση της Περίπτωση Χρήσης UC3 «Αίτημα Επεξεργασίας Οριστικού Προγράμματος» (extend), και το σύστημα επιτρέπει την επεξεργασία.
Βασική ροή: 1. Ο χρήστης επιλέγει «Δημιουργία Προγράμματος Εξετάσεων». 2. Ο χρήστης επιλέγει εξάμηνο από λίστα επιλογής (1ο, 2ο, 3ο, ...). 3. Το σύστημα εμφανίζει ημερολόγιο με τις διαθέσιμες χρονικές ζώνες ανά ημερομηνία και αίθουσα. 4. Ο χρήστης επιλέγει θέση χρονικής ζώνης για εισαγωγή/επεξεργασία μαθήματος. 5. Ο χρήστης επιλέγει μάθημα και αίθουσα για τη χρονική ζώνη. 6. Το σύστημα ενεργοποιεί την Περίπτωση Χρήσης UC2 «Ανίχνευση και Επίλυση Συγκρούσεων». 7. Το σύστημα καταχωρεί το μάθημα στη θέση στο πρόγραμμα εξετάσεων. 8. Ο χρήστης επαναλαμβάνει τα βήματα 4 έως 7 μέχρι την ολοκλήρωση όλων των μαθημάτων εξαμήνου. 9. Το σύστημα αποθηκεύει το πρόγραμμα εξετάσεων. **Εναλλακτική ροή 1 – «Επεξεργασία θέσης»:** 5α. Προβάλλεται παράθυρο με τις επιλογές στη συγκεκριμένη θέση (μάθημα και αίθουσα). 5β. Ο χρήστης τροποποιεί τις συγκεκριμένες επιλογές του μαθήματος στην ωριαία θέση. 5γ. Η ροή μεταφέρεται στο βήμα 6 της βασικής ροής. **Εναλλακτική ροή 2 – «Αφαίρεση μαθήματος από θέσης»:** 5α. Προβάλλεται παράθυρο με τις επιλογές στη συγκεκριμένη ωριαία θέση και κουμπί αφαίρεσης μαθήματος από τη θέση εφόσον έχει οριστεί μάθημα. 5β. Ο χρήστης επιλέγει το κουμπί «Αφαίρεση Μαθήματος από ωριαία θέση». 5γ. Το μάθημα αφαιρείται από την ωριαία θέση και ο χρήστης επανέρχεται σε προβολή του ημερολογίου.

5δ. Η ροή μεταφέρεται στο βήμα 3 της βασικής ροής. **Εναλλακτική ροή 3 – «Εντοπισμός συγκρούσεων με Παράκαμψη»:** 7α. Αν εντοπιστεί σύγκρουση, εμφανίζεται Προειδοποίηση και πιθανές Προτάσεις διαθέσιμων εναλλακτικών. 7β. Ο χρήστης επιλέγει «Παράκαμψη», εφόσον η σύγκρουση δεν αφορά σε αυστηρό περιορισμό, όπως αυτοί έχουν οριστεί στο συστήματος (π.χ. ανάθεση μαθημάτων ίδιου εξαμήνου την ίδια ημέρα ή χρήση αίθουσας σε ίδια ώρα από άλλο μάθημα). 7γ. Η ροή μεταφέρεται στο βήμα 7 της βασικής ροής. **Εναλλακτική ροή 4 – «Εντοπισμός συγκρούσεων χωρίς Παράκαμψη»:** 7α. Αν εντοπιστεί σύγκρουση, εμφανίζεται Προειδοποίηση και πιθανές Προτάσεις διαθέσιμων εναλλακτικών. 7β. Η σύγκρουση αφορά σε αυστηρό περιορισμό, όπως αυτοί έχουν οριστεί στα δεδομένα του συστήματος (π.χ. ανάθεση μαθημάτων ίδιου εξαμήνου την ίδια ημέρα ή χρήση αίθουσας σε ίδια ώρα από άλλο μάθημα) και δεν υπάρχει δυνατότητα παράκαμψης. 7γ. Ο χρήστης έχει τη δυνατότητα τροποποίησης των επιλογών για τη συγκεκριμένη θέση ή διαγραφής των επιλογών της θέσης. 7δ. Η ροή μεταφέρεται στο βήμα 5 της βασικής ροής.
Μη λειτουργικές απαιτήσεις: Η περίοδος δημιουργίας και επεξεργασίας επερχόμενου εξαμήνου είναι ενεργή, όπως έχει ορίσει ο Κύριος Διαχειριστής.
Κατάσταση εξόδου: Το πρόγραμμα εξετάσεων έχει αποθηκευτεί. Είναι διαθέσιμο για προβολή από όλους τους ρόλους χρηστών.

Πίνακας 3-12-UC6: Διαχείριση Προγράμματος Εξετάσεων Εξαμήνου

Τίτλος: Επεξεργασία Στοιχείων Πανεπιστημίου **ID:** UC7
Σύντομη περιγραφή: Η Περίπτωση Χρήσης περιγράφει τη λειτουργία επεξεργασίας (εισαγωγή, τροποποίηση, διαγραφή) δεδομένων που αφορούν σε Σχολές, Τμήματα, Αίθουσες και Διδάσκοντες.
Χειριστές:

Κύριος Διαχειριστής
Κατάσταση Εισόδου (Προϋποθέσεις): Ο χρήστης διαθέτει κατάλληλα δικαιώματα.
Βασική ροή: 1. Ο χρήστης επιλέγει ενότητα δεδομένων (Διαχείριση Σχολών, Τμημάτων, Αιθουσών ή Διδασκόντων). 2. Το σύστημα εμφανίζει τη λίστα εγγραφών. 3. Ο χρήστης επιλέγει εγγραφή προς επεξεργασία πληροφοριών. 4. Εμφανίζονται λεπτομέρειες της εγγραφής. 5. Ο χρήστης τροποποιεί τα στοιχεία της εγγραφής. 6. Το σύστημα ελέγχει την εγκυρότητα – ορθότητα των πεδίων της εγγραφής. 7. Ο έλεγχος είναι επιτυχής και ο χρήστης επιλέγει αποθήκευση στοιχείων. 8. Το σύστημα αποθηκεύει τα στοιχεία της εγγραφής. 9. Η ροή επιστρέφει στο βήμα 2. **Εναλλακτική ροή 1 – «Εισαγωγή νέας εγγραφής»:** 3α. Εμφανίζεται κενή φόρμα με πεδία για τη νέα εγγραφή. 3β. Ο χρήστης συμπληρώνει τα πεδία της φόρμας. 3γ. Το σύστημα ελέγχει για συμπληρωμένα υποχρεωτικά πεδία. 3δ. Η ροή μεταφέρεται στο βήμα 6. **Εναλλακτική ροή 2 – «Διαγραφή εγγραφής»:** 5α. Ο χρήστης επιλέγει το κουμπί «Διαγραφή». 5β. Το σύστημα εμφανίζει μήνυμα επιβεβαίωσης της ενέργειας. 5γ. Το σύστημα ελέγχει για δυνατότητα διαγραφής της εγγραφής (δεσμευμένη εγγραφή σε σχέση, π.χ. το Τμήμα ανήκει σε Σχολή ή το Μάθημα είναι καταχωρημένο σε Τμήμα). αλλιώς απορρίπτει με μήνυμα σφάλματος. 5δ. Η διαγραφή της εγγραφής ολοκληρώνεται. 5ε. Η ροή μεταφέρεται στο βήμα 2. **Εναλλακτική ροή 3 – «Αποτυχία ελέγχου εγκυρότητας – ορθότητας τιμής πεδίου»:** 6α. Εμφανίζεται ενημερωτικό – προειδοποιητικό μήνυμα για τη διόρθωση – συμπλήρωση του πεδίου με ορθή τιμή. 6β. Ο χρήστης διορθώνει την τιμή του πεδίου. 6γ. Η ροή μεταφέρεται στο βήμα 6.

Μη λειτουργικές απαιτήσεις: Έχουν οριστεί οι πιθανές σχέσεις μεταξύ των εγγραφών στο σχήμα της βάσης δεδομένων.
Κατάσταση εξόδου: Τα δεδομένα έχουν ενημερωθεί και είναι διαθέσιμα στο σύστημα. Οι ενέργειες καταγράφονται σύμφωνα με την πολιτική καταγραφής (logging).

Πίνακας 3-13-UC7: Επεξεργασία Στοιχείων Πανεπιστημίου

Τίτλος: Δήλωση Διαθεσιμότητας Ωρών/Ημερών Διδασκαλίας **ID:** UC8
Σύντομη περιγραφή: Η Περίπτωση Χρήσης αναλύει τη διαδικασία δήλωσης ωρών/ημερών προτίμησης ή μη διαθεσιμότητας του Διδάσκοντα, για το επερχόμενο εξάμηνο, ώστε να ληφθούν υπόψη κατά τη δημιουργία του ωρολογίου προγράμματος.
Χειριστές: Διδάσκων
Κατάσταση Εισόδου (Προϋποθέσεις): Ο λογαριασμός Διδάσκοντα είναι ενεργός. Έχει ανοίξει η περίοδος δηλώσεων προτιμήσεων από τον Διαχειριστή Τμήματος. Είναι καταχωρημένα τα μαθήματα που διδάσκει ο Διδάσκων.
Βασική ροή: 1. Ο χρήστης επιλέγει «Δήλωση Διαθεσιμότητας». 2. Το σύστημα εμφανίζει πλέγμα εβδομάδας με διαθέσιμες ζώνες ώρας ανά ημέρα. 3. Ο χρήστης επιλέγει μία ωριαία θέση (slot) και δηλώνει είτε «προτιμώμενη» είτε «μη διαθεσιμότητα». 4. Το σύστημα επικυρώνει ότι η επιλογή δεν υπερκαλύπτει το ωράριο του Τμήματος. 5. Ο χρήστης επαναλαμβάνει τα βήματα 2 έως 4 για κάθε επιθυμητή επιλογή. 6. Ο χρήστης επιλέγει «Υποβολή». 7. Το σύστημα αποθηκεύει τις δηλώσεις και αποστέλλει επιβεβαίωση μέσω email. **Εναλλακτική ροή 1 – «Εντοπισμός συγκρούσεων»:** 3α. Αν εντοπιστεί σύγκρουση με ιδρυματικούς περιορισμούς (υπερκάλυψη ωραρίου

Τμήματος), εμφανίζεται προειδοποίηση για τροποποίηση της δήλωσης. 3β. Η ροή μεταφέρεται στο βήμα 3.
Μη λειτουργικές απαιτήσεις: Δεν εφαρμόζεται σε αυτό το Use Case.
Κατάσταση εξόδου: Οι προτιμήσεις / μη διαθεσιμότητες αποθηκεύτηκαν. Οι ενέργειες καταγράφονται σύμφωνα με την πολιτική καταγραφής (logging).

Πίνακας 3-14-UC8: Δήλωση Διαθεσιμότητας Ωρών/Ημερών Διδασκαλίας

3.5 Διάγραμμα Οντοτήτων-Συσχετίσεων (ER Diagram)

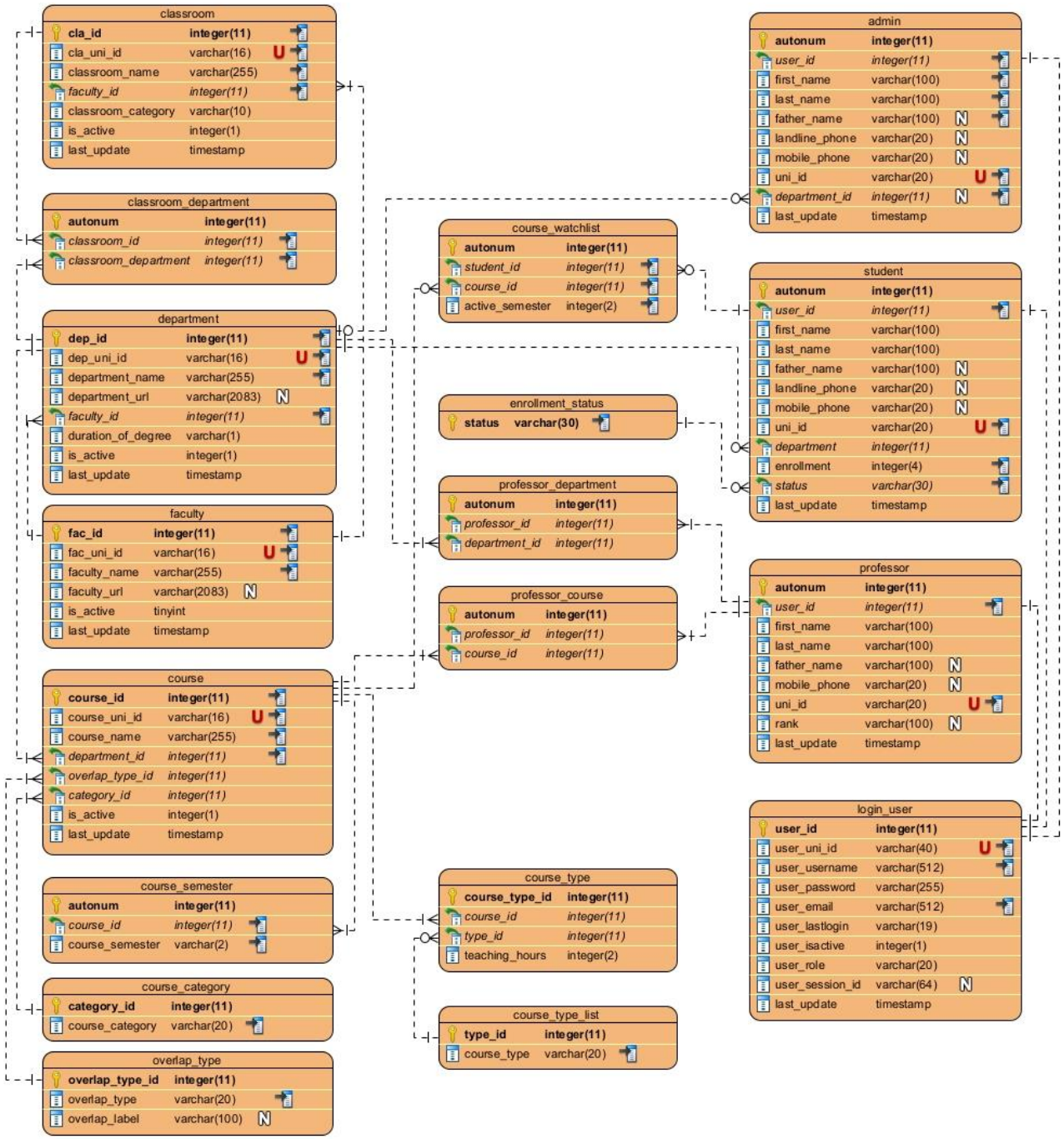
Η ενότητα αυτή παρουσιάζει το Διάγραμμα Οντοτήτων–Συσχετίσεων (Entity–Relationship Diagram, ERD) του πληροφοριακού συστήματος, το οποίο απεικονίζει τη δομή των δεδομένων και τις συσχετίσεις μεταξύ των βασικών οντοτήτων του.

Το ERD αποτελεί θεμελιώδες στάδιο του εννοιολογικού σχεδιασμού μιας βάσης δεδομένων και χρησιμοποιείται για να αποτυπώσει με σαφήνεια και τυπικότητα τις οντότητες (στοιχεία του πραγματικού κόσμου με αυτόνομη υπόσταση), τα κατηγορήματα (γνωρίσματα που καθορίζουν τα χαρακτηριστικά μίας οντότητας) και τις σχέσεις (relationships) που αναπτύσσονται μεταξύ οντοτήτων και καθορίζουν το σύνολο των συσχετισμών μεταξύ των εγγραφών τους [73]. Στο διάγραμμα αποδίδεται η λογική δομή της βάσης δεδομένων του συστήματος πριν από την υλοποίησή της σε φυσικό σχεσιακό επίπεδο.

Στη βιβλιογραφία, συνηθίζεται οι οντότητες να απεικονίζονται με ορθογώνια, οι σχέσεις με ρόμβους και τα γνωρίσματα με ελλείψεις, ωστόσο, στην παρούσα εργασία χρησιμοποιείται η τυποποιημένη συμβολική γραφή του Visual Paradigm, η οποία ευθυγραμμίζεται περισσότερο με τη σχεσιακή μοντελοποίηση και επιτρέπει σαφέστερη απεικόνιση, προσφέροντας μεγαλύτερη ευκρίνεια και ευρέως χρησιμοποιούμενη μορφοποίηση. Το ERD αποτελεί ενιαίο διάγραμμα, η παρουσίασή του, ωστόσο, έχει κατατμηθεί σε τρεις επιμέρους εικόνες, αποκλειστικά για λόγους ευκρίνειας και καλύτερης ανάγνωσης, χωρίς αυτό να αλλοιώνει τη συνοχή ή τη λογική του δομή.

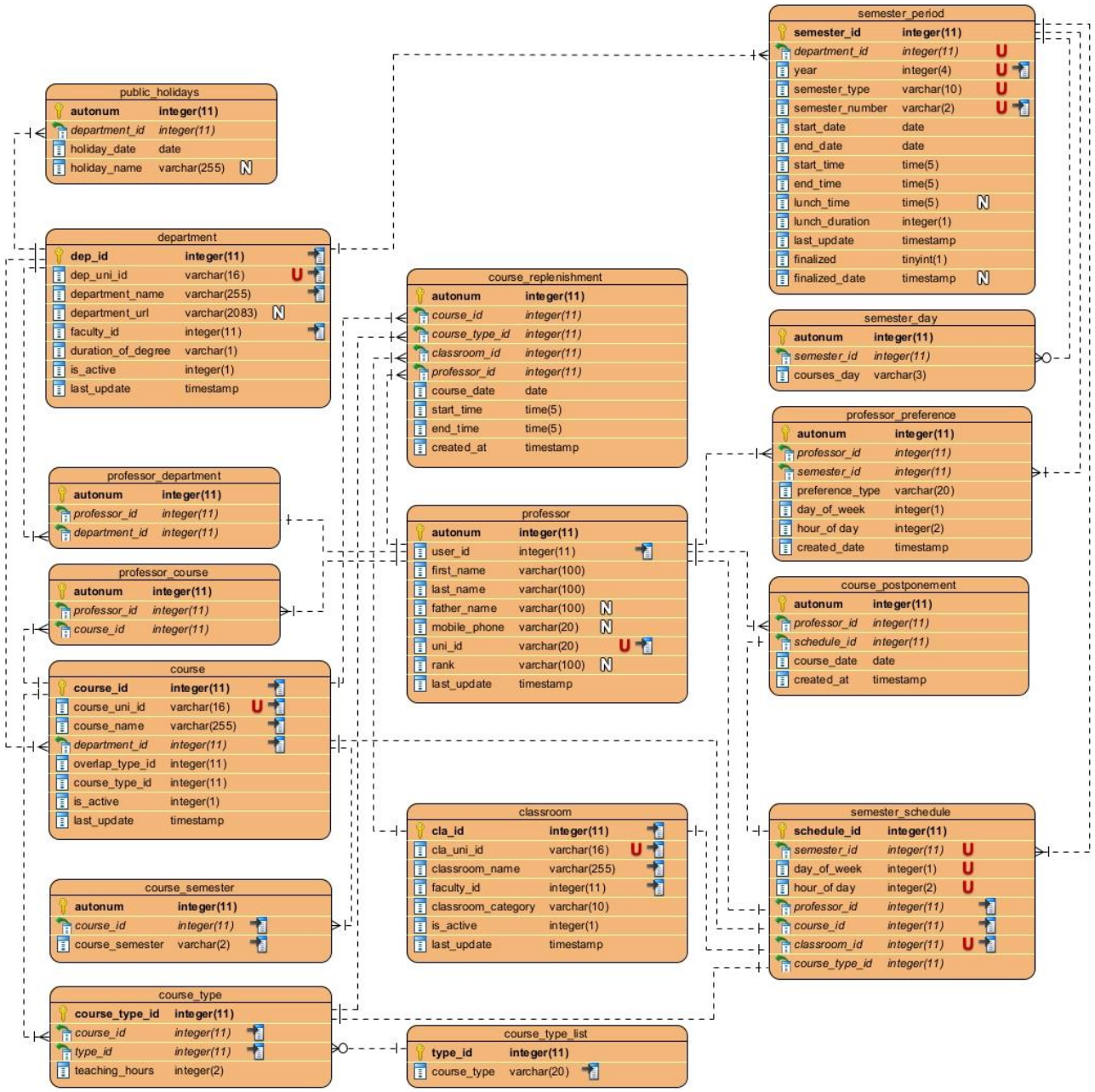
3.5.1 Διάγραμμα ERD

Διάγραμμα οργανωτικών οντοτήτων (χρήστες, εγκαταστάσεις και σχέσεις μεταξύ τους)



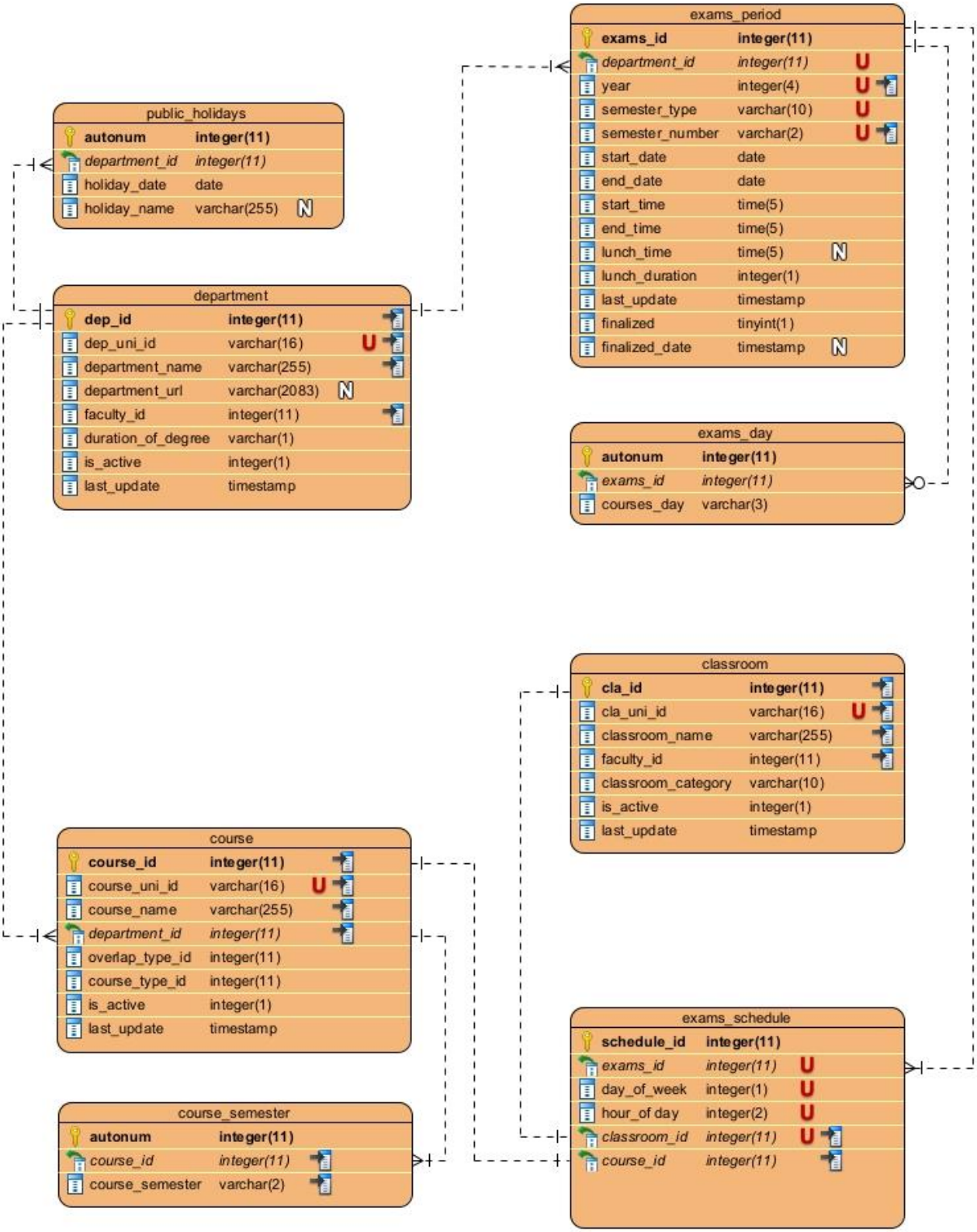
Σχήμα 2 - Διάγραμμα Οντοτήτων-Συσχετίσεων 1 (ERD)

Διάγραμμα λειτουργικών οντοτήτων (περιορισμοί, αναπλήρωση και ωρολόγιο πρόγραμμα)



Σχήμα 3 - Διάγραμμα Οντοτήτων-Συσχετίσεων 2 (ERD)

Διάγραμμα λειτουργικών οντοτήτων (πρόγραμμα εξετάσεων)



Σχήμα 4 - Διάγραμμα Οντοτήτων-Συσχετίσεων 3 (ERD)

3.5.2 Ανάλυση Οντοτήτων και Συσχετίσεων

Στους πίνακες της Βάσης Δεδομένων, καθώς και στο αντίστοιχο διάγραμμα ERD, ορισμένα πεδία έχουν υλοποιηθεί ως Enumerations. Τα πεδία αυτά μπορούν να λάβουν μόνο προκαθορισμένες τιμές από ένα συγκεκριμένο σύνολο (enum), καθώς το εύρος τιμών τους είναι σταθερό και δεν απαιτείται μεταβολή ή επέκταση. Χαρακτηριστικά παραδείγματα αποτελούν ο τύπος εξαμήνου (SPRING / FALL), η αριθμητική αναπαράσταση του εξαμήνου (1ο, 2ο, …, 12ο) και οι ημέρες της εβδομάδας (MON, TUE, etc.). Η χρήση πεδίων τύπου enum περιορίζει την ανάγκη για πρόσθετους πίνακες αναφοράς και απλοποιεί τη σύνταξη των SQL ερωτημάτων, μειώνοντας την πολυπλοκότητα των συσχετίσεων. Ως αποτέλεσμα, βελτιώνεται η συνολική απόδοση του συστήματος, χωρίς να επηρεάζεται η ποιότητα του κώδικα ή η καθαρότητα του σχήματος της ΒΔ.

Τα προσφερόμενα μαθήματα ανά Τμήμα και ανά εξάμηνο, ορίζονται μέσω του πίνακα «course» και «course_semester». Οι πίνακες των οντοτήτων, που χαρακτηρίζουν το Πανεπιστήμιο, τη δομή του, καθώς και τα προγράμματα των Σχολών, περιλαμβάνουν και πεδία για versioning, δηλαδή την τήρηση ιστορικού εκδόσεων ανά εξάμηνο και έτος. Στο διάγραμμα ERD κρίθηκε σκόπιμο να μην αποτυπωθούν τα κατηγορήματα και οι σχέσεις τήρησης ιστορικού εκδόσεων, για απλοποίηση της παρουσίασης και καλύτερη κατανόηση της κύριας δομής και λειτουργίας των οντοτήτων και των σχέσεών τους. Οι πίνακες και οι σχέσεις, περιλαμβάνονται με αναλυτικό τρόπο στο Κεφάλαιο 4 όπου αναλύεται η φυσική δομή και το σχήμα της ΒΔ.

Στις φυσικές οντότητες (όχι δηλαδή τις σχέσεις μεταξύ οντοτήτων), περιλαμβάνεται το κατηγόρημα με κατάληξη «uni_id» το οποίο αντιστοιχεί στον μοναδικό κωδικό αναφοράς που χρησιμοποιεί το εκάστοτε Πανεπιστήμιο για κάθε εγγραφή (π.χ. Αριθμός Μητρώου, Κωδικός Μαθήματος, Κωδικός Τμήματος κ.ά.).

Διάγραμμα ERD 1 (οργανωτικές οντότητες):

Οντότητα «faculty»: Οι Σχολές του Πανεπιστημίου.

Οντότητα «department»: Τα Τμήματα του Πανεπιστημίου. Εδώ ορίζεται η διάρκεια σπουδών και η Σχολή που ανήκει το Τμήμα (1-προς- N).

Οντότητα «classroom»: Οι διαθέσιμες αίθουσες ανά Σχολή. Κάθε αίθουσα ανήκει σε μία και μόνο Σχολή, αλλά μπορεί να εξυπηρετεί ένα ή περισσότερα Τμήματα της Σχολής.

Οντότητα «classroom_department»: Τα Τμήματα από τα οποία χρησιμοποιείται μια αίθουσα, έχει την έννοια Τμήμα-Έχει-Αίθουσα (Ν-προς-Ν).

Οντότητα «course»: Τα προσφερόμενα μαθήματα κάθε Τμήματος. Περιλαμβάνει τα κατηγορήματα «overlap_id» (εάν μπορεί να επικαλύψει άλλο μάθημα στο πρόγραμμα) και «category_id» (το είδος μαθήματος, υποχρεωτικό, επιλογής κ.ο.κ.).

Οντότητα «course_semester»: Ορίζει σε ποια εξάμηνα του προγράμματος σπουδών ανήκει κάθε μάθημα (μέσω enum με τιμές από 1 έως 12). Ενδέχεται ένα μάθημα να προσφέρεται σε περισσότερα από ένα εξάμηνα (σχέση 1-προς-Ν).

Οντότητα «course_category»: Λίστα τιμών με τα είδη μαθημάτων (Υποχρεωτικό, Κατ' επιλογήν υποχρεωτικό, Επιλογής, Σεμινάριο Δ.Ε.).

Οντότητα «course_type»: Ορίζει σε κάθε μάθημα τους τύπους και ώρες ανά τύπο.

Οντότητα «course_type_list»: Λίστα τιμών με τους τύπους μαθημάτων (Θεωρητικό, Εργαστηριακό, Άσκηση).

Οντότητα «overlap_type»: Λίστα τιμών με είδος επιτρεπόμενης επικάλυψης άλλου μαθήματος (Ναι, Ποτέ, Μόνο Επιλογής).

Οντότητα «login_user»: Περιλαμβάνει όλους τους πιστοποιημένους χρήστες με τα στοιχεία εισόδου στην πλατφόρμα, το μοναδικό αναγνωριστικό τους (ιδρυματικό email), τον ρόλο τους και το id που χρησιμοποιείται στο σχήμα της ΒΔ. Συνδέεται ως σχέση γενίκευσης με τους πίνακες των χρηστών του συστήματος (Διαχειριστές, Διδάσκοντες, Φοιτητές).

Οντότητα «admin»: Τα στοιχεία των Διαχειριστών του συστήματος. Κάθε Διαχειριστής μπορεί να ανήκει σε ένα και μόνο Τμήμα, εκτός του Κύριου Διαχειριστή, ο οποίος δεν ανήκει σε κάποιο Τμήμα.

Οντότητα «professor»: Τα στοιχεία των Διδασκόντων του Πανεπιστημίου. Κάθε Διδάσκων υπάγεται σε ένα ή περισσότερα Τμήματα, ενώ διδάσκει ένα ή περισσότερα μαθήματα.

Οντότητα «student»: Τα στοιχεία των Φοιτητών του Πανεπιστημίου. Κάθε Φοιτητής είναι εγγεγραμμένος σε ένα Τμήμα. Περιλαμβάνει το κατηγόρημα «enrollment» που αντιστοιχεί στο έτος έναρξης σπουδών, από το οποίο υπολογίζεται αυτόματα και το εξάμηνο φοίτησης.

Οντότητα «course_watchlist»: Τα δηλωθέντα από τον Φοιτητή μαθήματα για παρακολούθηση στο εξατομικευμένο ωρολόγιο πρόγραμμα. Στο κατηγόρημα «active_semester» αποθηκεύεται το τρέχον εξάμηνο (αριθμητικά, 1ο, 2ο, κ.ο.κ.) για το οποίο έγινε η δήλωση ενδιαφέροντος και υπολογίζεται αυτόματα σύμφωνα με το έτος εισαγωγής (enrollment) του Φοιτητή.

Οντότητα «professor_department»: Η σχέση Διδάσκων-Υπάγεται-Τμήμα (1-προς-Ν).

Οντότητα «professor_course»: Η σχέση Διδάσκων-Διδάσκει-Μάθημα (1-προς-Ν).

Οντότητα «enrollment_status»: Λίστα τιμών για την κατάσταση φοίτησης ενός Φοιτητή, όπως ενεργός, αποφοιτήσας, διαγραφείσας, κ.ο.κ.

Διάγραμμα ERD 2 (λειτουργικές οντότητες, περιορισμοί, αναπλήρωση, ωρολόγιο):

Οντότητα «professor»: Τα στοιχεία των Διδασκόντων του Πανεπιστημίου. Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για την παρουσίαση των επιπρόσθετων σχέσεων.

Οντότητα «department»: Τα Τμήματα του Πανεπιστημίου. Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για την παρουσίαση των επιπρόσθετων σχέσεων.

Οντότητα «classroom»: Οι διαθέσιμες αίθουσες ανά Σχολή. Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για την παρουσίαση των επιπρόσθετων σχέσεων.

Οντότητα «course»: Τα προσφερόμενα μαθήματα κάθε Τμήματος. Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για την παρουσίαση των επιπρόσθετων σχέσεων.

Οντότητα «professor_department»: Η σχέση Διδάσκων-Υπάγεται-Τμήμα (1-προς-Ν). Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για παρουσίαση σχέσεων.

Οντότητα «professor_course»: Η σχέση Διδάσκων-Διδάσκει-Μάθημα (1-προς-Ν). Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για παρουσίαση σχέσεων.

Οντότητα «course_semester»: Ορίζει σε ποια εξάμηνα του προγράμματος σπουδών ανήκει κάθε μάθημα (μέσω enum με τιμές από 1 έως 12). Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για παρουσίαση σχέσεων.

Οντότητα «course_type»: Ορίζει σε κάθε μάθημα τους τύπους και ώρες ανά τύπο. Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για παρουσίαση σχέσεων. Επιπλέον

παρουσιάζει τη συσχέτιση με το πρόγραμμα (semester_schedule) όπου ορίζεται εκτός από το μάθημα και ο τύπος μαθήματος και υπολογίζονται και οι απαιτούμενες ώρες διδασκαλίας.

Οντότητα «course_type_list»: Λίστα τιμών με τους τύπους μαθημάτων (Θεωρητικό, Εργαστηριακό, Άσκηση). Είναι η ίδια οντότητα με το ERD 1 και απεικονίζεται για παρουσίαση σχέσεων.

Οντότητα «public_holidays»: Οι ημερομηνίες των επίσημων αργιών που επηρεάζουν το πρόγραμμα ενός Τμήματος, με τίτλο/περιγραφή. Χρησιμοποιείται κατά την προβολή του αναλυτικού ημερολογίου του ωρολογίου προγράμματος.

Οντότητα «semester_day»: Λίστα των ημερών που είναι διαθέσιμες για ορισμό μαθημάτων στο ωρολόγιο πρόγραμμα εξαμήνου (Δευτέρα έως Κυριακή).

Οντότητα «professor_preference»: Οι περιορισμοί και προτιμήσεις των Διδασκόντων. Το κατηγόρημα «preference_type» είναι τύπου enum με τιμές «UNAVAILABLE», «PREFERRED», όπου δηλώνει αν η συγκεκριμένη ώρα της ημέρας αφορά σε μη διαθεσιμότητα ή προτίμηση του διδάσκοντα. Η επιλογή γίνεται για συγκεκριμένο εξάμηνο, ημέρα και ώρα.

Οντότητα «semester_period»: Αναπαριστά συγκεκριμένα ακαδημαϊκά εξάμηνα ανά τμήμα, έτος και τύπο (χειμερινό/εαρινό), με ημερομηνίες έναρξης/λήξης. Το κατηγόρημα «semester_type» είναι τύπου enum με τιμές «SPRING», «FALL», ενώ το κατηγόρημα «semester_number» παίρνει τιμές μέσω enum από 1 έως 12). Η αντιστοίχιση μεταξύ «θεωρητικού» εξαμήνου μαθήματος και «τρέχοντος» εξαμήνου υλοποιείται βάσει του αριθμού εξαμήνου (semester_number) και του τμήματος, όταν ορίζεται το ωρολόγιο πρόγραμμα κάθε περιόδου. Περιλαμβάνει κατηγορήματα ορισμού και άλλων παραμέτρων ή περιορισμών, όπως (προαιρετικά) διάλλειμα σίτισης, ώρες έναρξης και λήξης μαθημάτων, ημέρες που προσφέρονται τα μαθήματα, καθώς και σήμανση οριστικοποίησης.

Οντότητα «semester_schedule»: Το αναλυτικό ωρολόγιο πρόγραμμα, όπου κάθε εγγραφή αναπαριστά και μία ωριαία θέση (slot) στο ωρολόγιο πρόγραμμα. Κάθε θέση/εγγραφή περιλαμβάνει την ώρα, την ημέρα, το μάθημα, τον τύπο μαθήματος, τον διδάσκοντα και την αίθουσα, ενώ συνδέεται με ένα συγκεκριμένο πρόγραμμα εξαμήνου (semester_period). Το σχήμα περιλαμβάνει και περιορισμού μοναδικών πεδίων (συνδυαστικά) για έλεγχο διπλότυπων μη επιτρεπόμενων καταχωρήσεων, όπως για παράδειγμα η ίδια αίθουσα την ίδια ώρα και ημέρα, με διαφορετικό μάθημα.

Οντότητα «course_replenishment»: Ορίζει το μάθημα και τύπο μαθήματος προς αναπλήρωση, σε συγκεκριμένη αίθουσα, σε επιλεγμένη ημερομηνία και ώρα. Η διαθέσιμες ημέρες, ώρες και αίθουσες, υπολογίζονται από την εφαρμογή και δεν αναπαρίστανται σε διάγραμμα.

Οντότητα «course_postponement»: Ορίζει το μάθημα που έχει αναβληθεί (σύμφωνα με το ωρολόγιο πρόγραμμα ημέρα και ώρα, σε συγκεκριμένη ημερομηνία που ορίζεται στον πίνακα.

Οι περιορισμοί, η συμπλήρωση απαιτούμενων ωρών διδασκαλίας και οι πιθανές επικαλύψεις όπως έχει ορίσει στις παραμέτρους ο Διαχειριστής Τμήματος, υπολογίζονται εσωτερικά στην εφαρμογή σύμφωνα με τις σχέσεις και τα κατηγορήματα που ορίζονται στο διάγραμμα ERD. Στο αναλυτικό σχήμα της ΒΔ, περιλαμβάνονται όλες οι συσχετίσεις, καθώς και ιστορικό εκδόσεων (versioning) στους πίνακες των οντοτήτων όπου απαιτείται.

Διάγραμμα ERD 3 (πρόγραμμα εξετάσεων):

Οντότητα «exams_period»: Αναπαριστά συγκεκριμένες εξεταστικές περιόδους εξαμήνου ανά τμήμα, έτος και τύπο (χειμερινό/εαρινό), με ημερομηνίες έναρξης/λήξης. Παρόμοια χαρακτηριστικά με του διαγράμματος 2 και την οντότητα «semester_period».

Οντότητα «exams_schedule»: Το αναλυτικό πρόγραμμα εξετάσεων, όπου κάθε εγγραφή αναπαριστά και μία ωριαία θέση (slot) στο πρόγραμμα. Κάθε θέση/εγγραφή περιλαμβάνει την ώρα, την ημέρα, το μάθημα και την αίθουσα, ενώ συνδέεται με ένα συγκεκριμένο πρόγραμμα εξετάσεων (exams_period). Το σχήμα περιλαμβάνει και περιορισμούς μοναδικών πεδίων (συνδυαστικά) για έλεγχο διπλότυπων μη επιτρεπόμενων καταχωρήσεων, όπως για παράδειγμα η ίδια αίθουσα την ίδια ώρα και ημέρα, με διαφορετικό μάθημα.

Οντότητα «exams_day»: Λίστα των ημερών που είναι διαθέσιμες για ορισμό μαθημάτων στο πρόγραμμα εξετάσεων εξαμήνου (Δευτέρα έως Κυριακή).

Οι περιορισμοί και οι πιθανές επικαλύψεις, βάσει των παραμέτρους που όρισε ο Διαχειριστής Τμήματος, υπολογίζονται εσωτερικά στην εφαρμογή σύμφωνα με τις σχέσεις και τα κατηγορήματα που ορίζονται στο διάγραμμα ERD, ενώ στο σχήμα της ΒΔ περιλαμβάνονται αναλυτικότερες συσχετίσεις, καθώς και ιστορικό εκδόσεων (versioning).

3.6 Σύνοψη Κεφαλαίου

Ανακεφαλαιώνοντας, στο κεφάλαιο αυτό ορίστηκαν οι προδιαγραφές του πληροφοριακού συστήματος, όπως είναι: τα βασικά χαρακτηριστικά και οι στόχοι του, οι λειτουργικές και μη λειτουργικές απαιτήσεις, οι περιορισμοί και οι παραδοχές, καθώς και οι εμπλεκόμενοι ρόλοι χρηστών (διαχειριστές, διδάσκοντες, φοιτητές, εξωτερικά συστήματα). Μέσα από την ανάλυση χρηστών και ρόλων, τις ενδεικτικές ιστορίες χρηστών (user stories) με κριτήρια αποδοχής, τις περιπτώσεις χρήσης (use cases) και το αντίστοιχο διάγραμμα UML, καθώς και το διάγραμμα οντοτήτων–συσχετίσεων (ERD), αποτυπώθηκε πλήρως η διαδικασία λειτουργίας του συστήματος και ο τρόπος αλληλεπίδρασης με τους χρήστες και τα δεδομένα.

Σημειώνεται ότι τα παραπάνω διαγράμματα αποτυπώνουν τον αρχικό σχεδιασμό του συστήματος. Κατά την υλοποίηση προέκυψαν επιμέρους βελτιώσεις και επεκτάσεις, χωρίς να μεταβάλλεται η βασική δομή και λειτουργία.

Στο επόμενο Κεφάλαιο 4, με βάση τις παραπάνω προδιαγραφές, παρουσιάζονται η αρχιτεκτονική, η σχεδίαση και η υλοποίηση της πλατφόρμας, η δομή της βάσης δεδομένων, η διεπαφή χρήστη και τα βασικά υποσυστήματα, καθώς και ο τρόπος με τον οποίο οι απαιτήσεις του παρόντος κεφαλαίου μετατρέπονται σε λειτουργικό, web-base σύστημα.

4. Σχεδίαση και Υλοποίηση

Στο κεφάλαιο αυτό παρουσιάζεται η σχεδίαση και η υλοποίηση της πλατφόρμας, τόσο σε επίπεδο πηγαίου κώδικα (source code), όσο και σε επίπεδο Βάσης Δεδομένων (Database Schema). Αρχικά γίνεται αναφορά στις βασικές αρχές ορθού προγραμματισμού που ακολουθήθηκαν κατά την ανάπτυξη του συστήματος, οι οποίες επηρέασαν τη δομή, την οργάνωση και τη συντηρησιμότητα του κώδικα.

Στη συνέχεια αναλύεται η υλοποίηση του πηγαίου κώδικα, με διακριτό διαχωρισμό μεταξύ Frontend (διεπαφή χρήστη) και Backend (επίπεδο επιχειρησιακής λογικής), παρουσιάζοντας αντιπροσωπευτικά αποσπάσματα και τη λειτουργικότητά τους. Τέλος, περιγράφεται η σχεδίαση της Βάσης Δεδομένων και το σχήμα της, με έμφαση στη δομή των πινάκων, τις συσχετίσεις και τις σχεδιαστικές επιλογές που υποστηρίζουν τις απαιτήσεις της πλατφόρμας.

4.1 Ορθός Προγραμματισμός

Στην παρούσα ενότητα παρουσιάζονται οι βασικές αρχές ορθού προγραμματισμού που εφαρμόστηκαν κατά την ανάπτυξη της πλατφόρμας. Η προσέγγιση αυτή αφορά τη συγγραφή καθαρού, κατανοητού και συντηρήσιμου κώδικα, τη συμμόρφωση με αναγνωρισμένα πρότυπα και τη συστηματική χρήση εργαλείων ελέγχου ποιότητας, με στόχο τη μείωση σφαλμάτων και τη διευκόλυνση της μελλοντικής συντήρησης και επέκτασης του συστήματος.

4.1.1 Ονοματοδοσία, Συμβάσεις και Αναγνωσιμότητα Κώδικα

Τα ονόματα πινάκων και πεδίων στη βάση δεδομένων αποτελούνται αποκλειστικά από μικρούς λατινικούς χαρακτήρες, αριθμούς και τον χαρακτήρα underscore, ακολουθώντας τη μορφή «snake_case» (βλ. υποενότητα 2.6.4). Η επιλογή αυτή διασφαλίζει τη συνέπεια, την αναγνωσιμότητα και τη συμβατότητα μεταξύ διαφορετικών συστημάτων και εργαλείων διαχείρισης βάσεων δεδομένων. Έχουν επιλεγεί χαρακτηριστικά και περιγραφικά ονόματα, ενώ παράλληλα χρησιμοποιούνται προθέματα (prefixes) για την αποφυγή συγκρούσεων σε κοινότυπα αλλά απαραίτητα ονόματα, καθώς και για τον ευκολότερο εντοπισμό τους στον κώδικα. Ενδεικτικά, σε πεδία όπως το id, το οποίο απαντάται συχνά ως βασικό αναγνωριστικό, τοποθετείται πρόθεμα που δηλώνει το πεδίο χρήσης ή/και τον πίνακα στον

οποίο ανήκει. Ειδικά στους πίνακες, όπως φαίνεται στον Πίνακα 4-10, έχει προστεθεί το πρόθεμα «tbl_» για ευκολότερο εντοπισμό στα αρχεία του κώδικα και για σαφή διαχωρισμό από παρόμοια ονόματα μεταβλητών.

Οι μεταβλητές φέρουν επεξηγηματικά ονόματα που αποτυπώνουν με σαφήνεια τον σκοπό χρήσης τους. Για τοπικές μεταβλητές που χρησιμοποιούνται αποκλειστικά ως μετρητές σε δομές επανάληψης ή σε άλλες παρόμοιες περιπτώσεις, και οι οποίες έχουν καθαρά λειτουργικό χαρακτήρα, γίνεται χρήση ονομάτων όπως *$counter* ή σύντομων αναγνωριστικών τουλάχιστον τριών χαρακτήρων, βάσει καθιερωμένων συμβάσεων καλής πρακτικής προγραμματισμού. Σε σπάνιες περιπτώσεις, για συναρτήσεις που καλούνται πολύ συχνά, χρησιμοποιούνται ακρωνύμια (π.χ. η συνάρτηση ecn() στον Πίνακα 4-1, η οποία εκτελεί έξοδο με επιπρόσθετη νέα γραμμή "\n", ως συντομογραφία της εισαγωγής νέας γραμμής). Οι συναρτήσεις που εμφανίζουν έξοδο προς τον χρήστη φέρουν το πρόθεμα «e_» (ως έννοια του echo), ώστε να διευκολύνεται ο εντοπισμός και η διάκρισή τους. Όλα τα ονόματα μεταβλητών και συναρτήσεων ακολουθούν είτε την ονοματολογία camelCase είτε snake_case, ανάλογα με τη χρήση και την αναγνωσιμότητα στον κώδικα, όπως αναφέρεται στην υποενότητα 2.6.4. Βάσει σύμβασης καλής πρακτικής προγραμματισμού, οι σταθερές αναγράφονται με κεφαλαία γράμματα, ενώ οι τύποι enum αναγράφονται με κεφαλαίο το πρώτο γράμμα και μικρά τα υπόλοιπα, όπως φαίνεται ενδεικτικά στους Πίνακες 4-4 και 4-6 αντίστοιχα.

Έχουν πραγματοποιηθεί τακτικοί έλεγχοι στατικής ανάλυσης κώδικα, βάσει αυστηρών και προκαθορισμένων κανόνων, με τη χρήση εργαλείων όπως τα PHP CodeSniffer και PHP Mess Detector (βλ. υποενότητα 2.6.2). Οι έλεγχοι αυτοί διασφαλίζουν τη συνεπή τήρηση προτύπων συγγραφής κώδικα, ονοματοδοσίας και μορφοποίησης, καθώς και τη συμμόρφωση με το πρότυπο συγγραφής κώδικα PSR-12 της PHP (βλ. υποενότητα 2.6.2), συμβάλλοντας στη βελτίωση της αναγνωσιμότητας, της ποιότητας και της συντηρησιμότητας του κώδικα. Σε εξαιρετικές περιπτώσεις εφαρμόζονται ελεγχόμενες αποκλίσεις από το πρότυπο, αποκλειστικά για λόγους μείωσης της άσκοπης πολυπλοκότητας, χωρίς επίπτωση στη λειτουργικότητα ή την ασφάλεια του συστήματος.

Οι συναρτήσεις ακολουθούν την αρχή «μοναδική είσοδος – μοναδική έξοδος» (Single Entry, Single Exit – SESE), δηλαδή διαθέτουν ένα μόνο σημείο εισόδου και ένα μόνο σημείο επιστροφής (return) στο τέλος τους. Σε εξαιρετικές περιπτώσεις όπου απαιτείται πρόωρος τερματισμός του προγράμματος, γίνεται χρήση των εντολών exit ή die, η οποία ωστόσο δεν

θεωρείται τυπικός μηχανισμός εξόδου συνάρτησης. Τα παραπάνω παρουσιάζονται στον κώδικα του αρχείου στον Πίνακα 4-7.

Κάθε συνάρτηση τεκμηριώνεται με PHPDoc[24] και περιλαμβάνει τουλάχιστον τρία βασικά στοιχεία σχολιασμού:

1. Ρόλος / περιγραφή της συνάρτησης (ελεύθερο σχόλιο στην αρχή του PHPDoc block),
2. Είσοδοι και τύποι των παραμέτρων, με σχόλιο όπου απαιτείται (@param),
3. Έξοδος / επιστρεφόμενη τιμή και τύπος (@return).

Στις περιπτώσεις όπου η συνάρτηση δεν επιστρέφει τιμή (void), στο πεδίο (3) δηλώνεται ρητά η απουσία επιστρεφόμενης τιμής και, όπου ενδείκνυται, περιγράφονται οι ενέργειες ή τυχόν εξαιρέσεις που δύναται να εγείρει. Επιπλέον, σε κάθε συνάρτηση δηλώνεται ρητά ο τύπος επιστροφής στην υπογραφή της.

Κατά την ανάπτυξη του κώδικα ακολουθήθηκε η αρχή «Μη Επανάληψη Κώδικα» (Don't Repeat Yourself – DRY), σύμφωνα με την οποία δεν επιτρέπεται η ύπαρξη πανομοιότυπων ή σχεδόν πανομοιότυπων τμημάτων κώδικα σε δύο ή περισσότερα σημεία του ίδιου ή διαφορετικών αρχείων. Ειδικότερα, όταν συνεχόμενες γραμμές κώδικα είναι ίδιες ή παρόμοιες (με μόνη διαφορά, για παράδειγμα, τα ονόματα μεταβλητών), οι γραμμές αυτές αποσπώνται σε ανεξάρτητη συνάρτηση, η οποία στη συνέχεια καλείται από όλα τα σημεία όπου απαιτείται η ίδια λειτουργικότητα. Η πρακτική αυτή μειώνει δραστικά τον χρόνο συντήρησης και αποσφαλμάτωσης και διευκολύνει την ανάγνωση του κώδικα, καθώς το κύριο αρχείο περιέχει μόνο τον «λογικό κορμό» και όχι λεπτομέρειες υλοποίησης. Ενδεικτική υλοποίηση παρουσιάζεται στον Πίνακα Η-1. Η προσέγγιση αυτή εντάσσεται στη χρήση της αφαιρετικής τεχνικής[25] (abstraction). Εξαίρεση αποτελούν τμήματα κώδικα, κυρίως σε αρχεία διεπαφής (front-end), όπου κρίθηκε σκόπιμη η μερική επανάληψη για καλύτερη αναγνωσιμότητα του κώδικα. Τέλος, για λόγους οργάνωσης και συντήρησης, οι μεταβλητές που χρησιμοποιούνται καθολικά (global) δηλώνονται συγκεντρωτικά σε κοινό αρχείο επικεφαλίδας, όπως φαίνεται στον Πίνακα 4-4, ώστε να αποφεύγεται η διάσπαρτη δήλωσή τους σε διαφορετικά αρχεία. Η εφαρμογή της αρχής DRY συνεισφέρει επίσης στη μείωση της κυκλωματικής πολυπλοκότητας (cyclomatic complexity) (βλ. υποενότητα 2.6.2).

[24] Το PHPDoc είναι πρότυπο δομημένης τεκμηρίωσης για PHP, το οποίο χρησιμοποιείται για την περιγραφή συναρτήσεων, παραμέτρων και επιστρεφόμενων τιμών.
[25] Η αφαιρετική τεχνική (abstraction) επιτρέπει την απόκρυψη λεπτομερειών υλοποίησης και την έκθεση μόνο της απαραίτητης λειτουργικότητας, μειώνοντας την πολυπλοκότητα και βελτιώνοντας τη συντηρησιμότητα.

Όλα τα αρχεία κώδικα είναι αποθηκευμένα σε μορφή UTF-8 χωρίς BOM (No BOM), ώστε να αποφεύγεται η παρουσία ανεπιθύμητων χαρακτήρων στην αρχή τους. Επιπλέον, τα αρχεία HTML και PHP δηλώνουν πάντοτε ρητά την κωδικοποίηση UTF-8 στο αντίστοιχο header.

4.1.2 Σχεδίαση και Οργάνωση Βάσης Δεδομένων

Ο σχεδιασμός της Βάσης Δεδομένων βασίζεται σε αρχές κανονικοποίησης έως και την Τέταρτη Κανονική Μορφή (4NF) (βλ. υποενότητα 2.4.1), ιδίως για τους βασικούς λειτουργικούς πίνακες (οντότητες και συσχετίσεις), με στόχο την αποφυγή πλεονασμού και ανωμαλιών ενημέρωσης, ενώ στους πίνακες εφαρμόζεται διαχωρισμός μεταξύ μόνιμων και προσωρινών δεδομένων, όπως για παράδειγμα δεδομένα χρηστών και προσωρινά tokens. Οι πίνακες έχουν σχεδιαστεί με τρόπο ώστε να αποφεύγονται μεγάλα μεγέθη ανά γραμμή – σε επίπεδο σελίδας αποθήκευσης πρακτικά λίγο κάτω από ~8KB, παρότι το συνολικό θεωρητικό όριο μεγέθους γραμμής που επιβάλλεται από τη MySQL φτάνει έως 65.535 bytes [74] – βελτιώνοντας έτσι την απόδοση και τη συντηρησιμότητα της βάσης δεδομένων. Σε κάθε πίνακα έχουν δημιουργηθεί τα κατάλληλα και απαραίτητα indexes σε πεδία που χρησιμοποιούνται συχνά, ενώ αποφεύγεται η δημιουργία ευρετηρίων σε πεδία με περιορισμένη χρήση σε αναζητήσεις, ώστε να αποτρέπεται η άσκοπη κατανάλωση χώρου αποθήκευσης.

Κατά τη σύνδεση με τη βάση δεδομένων ορίζεται ρητά η κωδικοποίηση για πλήρη υποστήριξη χαρακτήρων Unicode (συμπεριλαμβανομένων των ελληνικών), με χρήση εντολών όπως SET NAMES utf8 ή utf8mb4_unicode_ci. Αντίστοιχα, η κωδικοποίηση στους πίνακες της βάσης δεδομένων έχει οριστεί με συνεπή τρόπο, όπως παρουσιάζεται στο σχήμα της ΒΔ στον Πίνακα 4-9.

4.1.3 Αρχιτεκτονική Εφαρμογής και Διαχωρισμός Ευθυνών

Για τη βελτίωση της αρχιτεκτονικής και της συντηρησιμότητας του συστήματος, οι συναρτήσεις και οι μονάδες κώδικα δεν τοποθετούνται στο κύριο αρχείο, αλλά διαχωρίζονται σε αυτόνομα αρχεία και ομαδοποιούνται ανά λειτουργία. Η σύνδεση με τη βάση δεδομένων υλοποιείται μέσω ανεξάρτητης κλάσης, η οποία παρουσιάζεται στον Πίνακα Η-8 ενώ πραγματοποιούνται τακτικοί έλεγχοι του κώδικα με εργαλεία στατικής ανάλυσης, όπως τα PHPStan, PHP Metrics και Psalm, τα οποία περιγράφονται στην

υποενότητα 2.6.2, διασφαλίζοντας την ποιότητα και την αξιοπιστία της εφαρμογής. Η οργάνωση αυτή ελαχιστοποιεί την επανάληψη κώδικα, διευκολύνει την αποσφαλμάτωση και επιτρέπει στοχευμένες αλλαγές χωρίς ανεπιθύμητες παρενέργειες.

Έχουν οριστεί βασικές και συχνά χρησιμοποιούμενες τιμές μεταβλητών ως σταθερές, ορισμένες σε ένα κεντρικό αρχείο παραμέτρων κώδικα, εκτός του web root για ασφάλεια (όπως κλειδιά για κρυπτογράφηση, διευθύνσεις email συστήματος, διακομιστής email, διαπιστευτήρια βάσης δεδομένων, διεύθυνση κύριας σελίδας, όρια και τιμές για μεγέθη αρχείων, χρόνους λήξης token και sessions, κ.ά.). Με αυτό τον τρόπο μπορεί εύκολα να τροποποιηθεί κάθε βασική παράμετρος του συστήματος, χωρίς να απαιτείται η αναζήτηση σε πολλαπλές σελίδες κώδικα.

Στην αρχιτεκτονική σχεδίαση εφαρμόζεται πλήρης διαχωρισμός μεταξύ αρχείων κώδικα που περιέχουν συναρτήσεις, αρχείων με εκτελέσιμες εντολές, handlers, αρχείων διεπαφής κ.λπ., ενώ έχει δοθεί ιδιαίτερη έμφαση στην επαναχρησιμοποίηση συναρτήσεων και κώδικα. Σε εξαιρετικές και σαφώς οριοθετημένες περιπτώσεις, επιτρέπεται περιορισμένη απόκλιση από τον διαχωρισμό αυτό, όταν η απομόνωση του κώδικα σε ξεχωριστές μονάδες θα οδηγούσε σε δυσανάλογη πολυπλοκότητα, δυσχεραίνοντας την αναγνωσιμότητα και τη συντηρησιμότητα. Στο επίπεδο διεπαφής (view / template) έχει υιοθετηθεί ένα σύγχρονο και διαδεδομένο πρότυπο συγγραφής, με ελάχιστη επιχειρησιακή λογική, χρήση inline PHP αποκλειστικά για δομές ελέγχου και επανάληψης, καθώς και χρήση της σύνταξης `<?=...?>` για την απόδοση εξόδου. Η επιλογή αυτή παρουσιάζεται στον Πίνακα 4-2 και τεκμηριώνεται ως εξής:

Πρότυπα και αναγνωσιμότητα: Η σύνταξη `<?=...?>` αποτελεί τυπική και προτεινόμενη πρακτική για έξοδο σε templates, καθώς είναι συντομότερη και πιο καθαρή από τη χρήση της echo, μειώνοντας τον «θόρυβο» στον κώδικα. Δεν χρησιμοποιούνται τα παλαιά short open tags `<?...?>`, αλλά τα `<?php...?>` και `<?=...?>`, τα οποία είναι συμβατά και ενεργά από την έκδοση PHP 5.4 και έπειτα.

Διαχωρισμός ευθυνών (Separation of Concerns): Στα templates δεν περιέχεται επιχειρησιακή λογική. Στα views χρησιμοποιείται αποκλειστικά ελαφριά λογική παρουσίασης (π.χ. δομές if, elseif, foreach), ενώ σε κάθε περίπτωση εφαρμόζεται escaping. Οι εναλλαγές `<?php ... ?>` εντός του HTML αφορούν μόνο δομές ελέγχου και όχι επιχειρησιακούς υπολογισμούς, διατηρώντας τη συντήρηση απλή και σαφή.

Συντηρησιμότητα και καθαρότητα: Η εναλλακτική σύνταξη ελέγχου (`if: endif;` και `foreach: endforeach;`) σε συνδυασμό με τη χρήση `<?= ... ?>` παράγει HTML-first templates, ευανάγνωστα ακόμη και από front-end προγραμματιστές. Παράλληλα, δίνεται έμφαση στη συνεπή χρήση εσοχών (indentation). Όπου εντοπίζονται επαναλαμβανόμενα τμήματα, αυτά αποσπώνται σε συναρτήσεις, ώστε να αποφεύγονται εκτενή μπλοκ κώδικα.

Ασφάλεια εξόδου: Εφαρμόζεται συνεπές escaping σε κάθε έξοδο, μέσω βοηθητικών συναρτήσεων (π.χ. `e_html()`), ανεξάρτητα από το αν η έξοδος πραγματοποιείται με `echo` ή `<?=...?>`. Σε συναρτήσεις οι οποίες επιστρέφουν μεγάλα αποσπάσματα HTML χρησιμοποιούνται ειδικοί helpers για την αποφυγή σφαλμάτων σε μεταβλητές και στο escaping ειδικών χαρακτήρων. Η χρήση HEREDOC περιορίζεται σε βοηθητικά αρχεία (helpers), ώστε να διατηρείται υψηλή αναγνωσιμότητα του HTML κώδικα.

Συμμόρφωση με καλές πρακτικές: Τηρείται το πρότυπο PSR-12 όσον αφορά το στυλ και τη μορφοποίηση του κώδικα. Δεν ενσωματώνεται επιχειρησιακή λογική στο επίπεδο του view, καθώς τα δεδομένα παρέχονται έτοιμα από τον handler ή το backend service. Ο σαφής διαχωρισμός ευθυνών και το συνεπές escaping συμβάλλουν περισσότερο στη βελτίωση της ποιότητας του κώδικα από τη χρήση ενός ενιαίου και εκτενούς PHP block.

Η μορφοποίηση (styling) του HTML υλοποιείται αποκλειστικά μέσω classes και ids, με χρήση ενιαίου αρχείου CSS που συγκεντρώνει όλες τις απαιτούμενες δηλώσεις μορφοποίησης. Στον ιστότοπο χρησιμοποιούνται κοινά αρχεία κεφαλίδας (header) και υποσέλιδου (footer), τα οποία ενσωματώνονται στις σελίδες του frontend με require_once. Εντός αυτών υλοποιείται η λογική προσαρμογής της εμφάνισης ανά ρόλο χρήστη (επισκέπτης, εγγεγραμμένος χρήστης, διαχειριστής).

4.1.4 Ασφάλεια Κώδικα, Δεδομένων και Πρόσβασης

Η πλατφόρμα υιοθετεί πρακτικές «security-by-design» (βλ. υποενότητα 2.5.2) ήδη από το στάδιο του αρχικού σχεδιασμού και σε όλα τα επίπεδα υλοποίησης, με την ασφάλεια του κώδικα να αποτελεί θεμελιώδη προτεραιότητα. Όλες οι είσοδοι και έξοδοι δεδομένων υπόκεινται σε ρητούς κανόνες επικύρωσης και εξυγίανσης (server-side validation, έλεγχος τύπων και ορίων, filtering), τόσο στο επίπεδο του frontend όσο και – κυρίως – στο επίπεδο του backend, ενώ εφαρμόζεται συστηματικό escaping κατά την έξοδο για την αποτροπή επιθέσεων τύπου XSS. Όλα τα δεδομένα που προέρχονται από τον χρήστη (POST, GET,

COOKIES, HEADERS, FILES), καθώς και κάθε άλλη παράμετρος που δύναται να αλλοιωθεί (π.χ. hidden fields ή HTTP headers), ελέγχονται συστηματικά βάσει ρητών κανόνων επικύρωσης.

Η πρόσβαση στη βάση δεδομένων πραγματοποιείται αποκλειστικά μέσω παραμετροποιημένων ερωτημάτων PDO (prepared statements με χειρισμό σφαλμάτων μέσω εξαιρέσεων) (βλ. υποενότητα 2.3.4), αποφεύγοντας πλήρως την απευθείας ενσωμάτωση τιμών σε εντολές SQL. Ευαίσθητα δεδομένα προστατεύονται με ισχυρό κατακερματισμό μονής κατεύθυνσης για κωδικούς πρόσβασης (password_hash/password_verify με ενσωματωμένο salt) και, όπου απαιτείται, με αμφίδρομη κρυπτογράφηση (όπως AES-256), με χρήση μοναδικού IV και κλειδιού.

Η αρχιτεκτονική της εφαρμογής βασίζεται σε σαφή διαχωρισμό ευθυνών (π.χ. κλάσεις όπως οι QueryValidator, ConstraintBuilder και Database, καθώς και μηχανισμούς ελέγχου πρόσβασης όπως τα αρχεία logincheck.php και _guard.php), μειώνοντας την πολυπλοκότητα και επιτρέποντας την επαναχρησιμοποίηση ελεγμένων και αξιόπιστων συναρτήσεων. Η πρόσβαση ακολουθεί την αρχή «default-deny» και την πολιτική ελαχίστων δικαιωμάτων, με έλεγχο ταυτοποίησης στην είσοδο κάθε σελίδας και μηχανισμούς προστασίας συνεδρίας (όπως session_regenerate_id(), έλεγχο User Agent και διεύθυνσης IP). Τα cookies ρυθμίζονται με τις παραμέτρους HttpOnly, Secure και SameSite, ενώ το περιεχόμενό τους αντιμετωπίζεται ως δεδομένα εισόδου και υπόκειται σε επικύρωση.

Για την υποβολή φορμών εφαρμόζεται προστασία CSRF και το μοτίβο Post/Redirect/Get (PRG)[26], σύμφωνα με το οποίο τα δεδομένα αποστέλλονται αρχικά μέσω αιτήματος POST, επεξεργάζονται στο backend και, μετά την ολοκλήρωση της ενέργειας, πραγματοποιείται ανακατεύθυνση σε νέα σελίδα μέσω αιτήματος GET, αποτρέποντας τη διπλή ή επαναλαμβανόμενη υποβολή φορμών.

Η ασφάλεια ενισχύεται πολυεπίπεδα μέσω προελέγχων περιορισμών πριν από εισαγωγές δεδομένων, χειρισμού σφαλμάτων με εξαιρέσεις και καταγραφής συμβάντων σε αρχεία καταγραφής (logs). Τέλος, πραγματοποιείται συστηματική αναγνώριση πιθανών κινδύνων (όπως SQL Injection, XSS, CSRF, Session Fixation και κλιμακωτά δικαιώματα πρόσβασης) (βλ. υποενότητα 2.5.1) και αντιστοίχισή τους με συγκεκριμένα αντίμετρα, ώστε η ασφάλεια

[26] Το POST χρησιμοποιείται για την αποστολή δεδομένων από τον πελάτη (client) προς τον διακομιστή (server), ενώ το GET για την ανάκτηση και προβολή δεδομένων από τον διακομιστή προς τον πελάτη, μετά από ελεγχόμενη ανακατεύθυνση.

να αποτελεί εγγενή ιδιότητα του συστήματος και όχι εκ των υστέρων προσθήκη. Τα στοιχεία σύνδεσης συγκεντρώνονται σε ενιαίο αρχείο ρυθμίσεων για όλη την πλατφόρμα, το οποίο βρίσκεται εκτός του κύριου φακέλου της εφαρμογής (web root) για λόγους ασφαλείας. Κάθε σύνδεση με τη βάση δεδομένων πραγματοποιείται αποκλειστικά με χρήση σταθερών μεταβλητών που ορίζονται στο συγκεκριμένο αρχείο, παρέχοντας τη δυνατότητα εύκολης και κεντρικής τροποποίησης των στοιχείων πρόσβασης χωρίς παρεμβάσεις σε πολλαπλά αρχεία κώδικα.

Σε κάθε αρχείο PHP πραγματοποιείται έλεγχος ώστε να αποφεύγεται η ύπαρξη κενού χαρακτήρα στην αρχή του κώδικα, ο οποίος θα μπορούσε να επηρεάσει την αποστολή των HTTP headers (μέσω στατικού ελέγχου με PHPCS). Κάθε σελίδα περιλαμβάνει ως εναρκτήρια εντολή τη δήλωση `declare(strict_types=1)` για αυστηρό έλεγχο τύπων και επιστρεφόμενων τιμών. Στη συνέχεια, κατά την πρώτη συμπερίληψη αρχείου (include/require), εκτελείται η `session_start()` μέσω κεντρικού μηχανισμού αρχικοποίησης συνεδρίας (αρχείο "_session_start.php", όπως παρουσιάζεται στον Πίνακα 4-5), στον οποίο εφαρμόζονται πρόσθετες ρυθμίσεις ασφαλείας, όπως προσαρμοσμένη ονοματοδοσία συνεδρίας, περιορισμός χρήσης σε HTTPS και περιοδική ανανέωση της συνεδρίας. Τα αρχεία κώδικα που προορίζονται αποκλειστικά για συμπερίληψη σε κύριες σελίδες, περιλαμβάνουν μηχανισμούς αποτροπής άμεσης πρόσβασης και, για τον λόγο αυτό, δεν εκκινούν αυτόνομα τη συνεδρία μέσω session_start().

Εφαρμόζεται μηχανισμός αυτόματης λήξης συνεδρίας βάσει χρονικού ορίου αδράνειας (idle timeout), καθώς και μέγιστη διάρκεια ζωής συνεδρίας (absolute timeout). Το χρονικό όριο αδράνειας ορίζεται κεντρικά στα σταθερά δεδομένα της πλατφόρμας, με τιμή προεπιλογής τα 30 λεπτά και ανανεώνεται σε κάθε έγκυρη ενέργεια του χρήστη. Ως αποτέλεσμα, η αποσύνδεση λόγω timeout πραγματοποιείται μόνο όταν παρέλθει προκαθορισμένο χρονικό διάστημα από την τελευταία καταγεγραμμένη ενέργεια.

Παράλληλα, υλοποιείται μηχανισμός περιοδικού ελέγχου κατάστασης της συνεδρίας μέσω ασύγχρονων αιτημάτων AJAX (βλ. υποενότητα 2.2.3), τα οποία λειτουργούν ως μηχανισμός τύπου heartbeat (ping) σε τακτά χρονικά διαστήματα (π.χ. κάθε ένα λεπτό), όσο ο χρήστης παραμένει ενεργός στη σελίδα. Σε περίπτωση παρατεταμένης αδράνειας της διεπαφής (π.χ. απώλεια εστίασης της καρτέλας ή απουσία αλληλεπίδρασης για προκαθορισμένο χρόνο), εμφανίζεται στον χρήστη μήνυμα επιβεβαίωσης παραμονής («Συνέχεια / Παραμονή σε σύνδεση»), παρέχοντάς του τη δυνατότητα συνέχισης της συνεδρίας ή ελεγχόμενης

αποσύνδεσης. Για τον σκοπό αυτό χρησιμοποιείται ειδική μεταβλητή κατάστασης συνεδρίας, η οποία υποδηλώνει ότι έχει ενεργοποιηθεί η διαδικασία επιβεβαίωσης παραμονής. Σε περίπτωση που η καρτέλα του φυλλομετρητή παραμείνει ανοιχτή αλλά ανενεργή, η εμφάνιση του μηνύματος βασίζεται στον χρόνο του τελευταίου έγκυρου ping. Αντιθέτως, εφόσον διαπιστωθεί ότι η καρτέλα έχει κλείσει ή ότι δεν υφίσταται πλέον ενεργή παρουσία χρήστη, η συνεδρία τερματίζεται αυτόματα κατά τον επόμενο έλεγχο, ακόμη και αν δεν έχει λήξει το χρονικό όριο αδράνειας.

Επιπλέον, εφαρμόζεται έλεγχος μοναδικής ενεργής συνεδρίας ανά χρήστη. Σε περίπτωση ανίχνευσης ταυτόχρονης σύνδεσης του ίδιου λογαριασμού από διαφορετική συσκευή ή παράθυρο περιήγησης, οι προηγούμενες συνεδρίες τερματίζονται αυτόματα και παραμένει ενεργή μόνο η πλέον πρόσφατη.

Κάθε εντολή (query) προς τη βάση δεδομένων εκτελείται αποκλειστικά μέσω προκατασκευασμένων εντολών PDO (prepared statements), με στόχο την ενίσχυση της ασφάλειας, όπως φαίνεται στον Πίνακα Η-8. Κάθε είσοδος δεδομένων προερχόμενη από τον χρήστη ή από εξωτερικές προς τον κώδικα πηγές, υπόκειται σε διαδικασίες εξυγίανσης (sanitization), ώστε να αποτρέπεται η εισαγωγή κακόβουλου κώδικα και οι επιθέσεις τύπου SQL Injection. Κατά τη χρήση του PDO εφαρμόζεται αποκλειστικά δέσμευση παραμέτρων (parameter binding), χωρίς καμία μεταβλητή να ενσωματώνεται απευθείας σε εντολή SQL, ενώ οι παράμετροι ορίζονται κυρίως ονομαστικά (named parameters).

Για τη μείωση της κατανάλωσης πόρων, δεν χρησιμοποιούνται μόνιμες (persistent) συνδέσεις με τη βάση δεδομένων. Κάθε σύνδεση ανοίγει μόνο κατά την εκτέλεση των απαιτούμενων εντολών και τερματίζεται αμέσως μετά, συμβάλλοντας στη μείωση της κατανάλωσης πόρων και στην αποφυγή παρατεταμένης διατήρησης συνδέσεων.

Η διαχείριση σφαλμάτων μέσω PDO έχει ρυθμιστεί σε λειτουργία εξαιρέσεων (ERRMODE_EXCEPTION). Με τον τρόπο αυτό, κάθε αποτυχία εκτέλεσης SQL εγείρει εξαίρεση τύπου PDOException, χωρίς να απαιτείται ρητός έλεγχος της επιστρεφόμενης τιμής των εντολών (π.χ. ενδεχόμενη τιμή false από την `execute()`). Όλες οι κλήσεις προς τη βάση δεδομένων εκτελούνται εντός δομών `try/catch` και, όπου απαιτείται, εντός συναλλαγών (transactions), ώστε να εξασφαλίζεται κεντρικός χειρισμός και καταγραφή σφαλμάτων. Συνεπώς, η μη χρήση ελέγχου της τιμής επιστροφής, όταν αυτό δεν είναι απαραίτητο, δεν υποβαθμίζει την αξιοπιστία του κώδικα αλλά αντιθέτως, απλοποιεί τη ροή

του, ενώ οι αναγκαίοι έλεγχοι (όπως ο αναμενόμενος αριθμός επηρεασμένων γραμμών ή η ύπαρξη αποτελεσμάτων) υλοποιούνται ρητά όπου απαιτείται.

Σε κάθε σελίδα όπου απαιτείται ταυτοποίηση χρήστη, ως πρώτη εντολή εκτελείται `require_once 'logincheck.php'`, όπως παρουσιάζεται στον Πίνακα Η-2. Το αρχείο αυτό εκκινεί τη συνεδρία (`session_start()`), ελέγχει αν ο χρήστης είναι συνδεδεμένος και, όπου απαιτείται, αν διαθέτει τα απαραίτητα δικαιώματα πρόσβασης για τη συγκεκριμένη σελίδα. Σε αντίθετη περίπτωση πραγματοποιείται ανακατεύθυνση (π.χ. προς την αρχική σελίδα μέσω `header('Location: ...')`) και η εκτέλεση του κώδικα τερματίζεται άμεσα με `die()`, συνοδευόμενη – όπου ενδείκνυται – από σύντομο μήνυμα ενημέρωσης, ώστε να αποτρέπεται κακόβουλη παράκαμψη της ανακατεύθυνσης λόγω μη ταυτοποίησης.

Επιπλέον, σε κάθε σελίδα δηλώνονται ως σχόλιο τα απαιτούμενα δικαιώματα πρόσβασης (AccessLevel), ενώ όπου χρειάζεται ορίζεται απαρίθμηση ρόλων (Roles enum) και πραγματοποιείται έλεγχος πρόσβασης μέσω φόρτωσης του αρχείου _guard.php, το οποίο παρουσιάζεται στον Πίνακα Η-3. Πέραν του ελέγχου δικαιωμάτων, το αρχείο _guard.php αποτρέπει την απευθείας πρόσβαση σε σελίδες του backend και σε handlers, τα οποία φορτώνονται αποκλειστικά μέσω άλλων σελίδων.

Η πλατφόρμα αποθηκεύει τα αρχεία στο σύστημα αρχείων και όχι στη βάση δεδομένων. Κατά τη διαδικασία μεταφόρτωσης αποδίδεται σε κάθε αρχείο μοναδικό και μη προβλέψιμο όνομα (τυχαίο αναγνωριστικό μήκους ≥ 128 bit με τυχαίο hash). Στη βάση δεδομένων καταχωρίζεται η αντιστοίχιση (stored_name, original_name), μαζί με βασικά μεταδεδομένα (τύπος MIME, μέγεθος, ιδιοκτήτης, κατάσταση). Στις διεπαφές εμφανίζεται το αρχικό όνομα του αρχείου, ενώ το αποθηκευμένο τυχαίο όνομα δεν εκτίθεται, αποτρέποντας τόσο συγκρούσεις ονομάτων (name collisions) όσο και επιθέσεις απαρίθμησης (enumeration attacks). Επιπλέον, εφαρμόζονται allow-list τύπων MIME, όρια μεγέθους αρχείων και καταγραφή συμβάντων (upload/delete) για σκοπούς ιχνηλασιμότητας.

Στον ιστοχώρο χρησιμοποιείται αποκλειστικά κρυπτογραφημένη πρόσβαση μέσω HTTPS. Σε περιβάλλον παραγωγής, κάθε σελίδα διεπαφής ελέγχει το χρησιμοποιούμενο πρωτόκολλο και, σε περίπτωση ανίχνευσης μη ασφαλούς HTTP, πραγματοποιείται αυτόματη ανακατεύθυνση σε HTTPS. Ο διακομιστής φιλοξενίας βασίζεται σε Windows Server και χρησιμοποιείται αρχείο web.config για ρυθμίσεις όπως ανακατευθύνσεις σφαλμάτων, περιορισμό πρόσβασης σε φακέλους και επανεγγραφή διευθύνσεων. Για λόγους

φορητότητας, διατηρούνται ισοδύναμες ρυθμίσεις και για Apache (.htaccess), ώστε ενδεχόμενη μετεγκατάσταση να μην απαιτεί τροποποιήσεις στον κώδικα. Ο κώδικας της εφαρμογής παραμένει ανεξάρτητος από το λειτουργικό σύστημα και τον διακομιστή (δεν πραγματοποιούνται κλήσεις system σε cmd.exe ή /bin/bash), ενώ οι ρυθμίσεις αυτές αφορούν αποκλειστικά τη βελτίωση της εμπειρίας χρήστη χωρίς να επηρεάζουν τη βασική λειτουργικότητα της πλατφόρμας.

Στον ιστοχώρο δεν υφίστανται «ορφανές» σελίδες, καθώς κάθε σελίδα διεπαφής είναι προσβάσιμη μέσω της κανονικής πλοήγησης. Κατ' εξαίρεση, η σελίδα εισόδου διαχειριστών έχει σχεδιαστεί και ως μεμονωμένης πρόσβασης, χωρίς να συνδέεται σε άλλες δημόσιες σελίδες, για κάλυψη ενδεχόμενης ζήτησης από το Ίδρυμα. Η πρακτική αυτή δεν αντιμετωπίζεται ως μέτρο ασφάλειας – η προστασία παρέχεται μέσω μηχανισμών αυθεντικοποίησης και εξουσιοδότησης – αλλά αποτρέπει την ευρετηρίαση και τη δημόσια δημοσίευση της διεύθυνσης. Για τον σκοπό αυτό χρησιμοποιούνται επιπλέον οδηγίες προς μηχανές αναζήτησης (π.χ. meta noindex), χωρίς να υποκαθιστούν τους ελέγχους πρόσβασης.

4.1.5 Διαχείριση Σφαλμάτων, Καταγραφή και Περιβάλλοντα Εκτέλεσης

Κατά την ανάπτυξη της πλατφόρμας, για σκοπούς αποσφαλμάτωσης και παρακολούθησης της εκτέλεσης, χρησιμοποιήθηκε η σταθερά IS_IN_PRODUCTION, η οποία ορίζεται κατά την εκκίνηση κάθε σελίδας και καθορίζει αν το σύστημα εκτελείται σε παραγωγικό (production) ή σε αναπτυξιακό/δοκιμαστικό (development/test) περιβάλλον. Όταν η σταθερά είναι ανενεργή, ενεργοποιείται η εμφάνιση αναλυτικών μηνυμάτων σφαλμάτων και καταγραφής λειτουργιών, διευκολύνοντας την αποσφαλμάτωση και την περαιτέρω ανάπτυξη. Σε κάθε περιβάλλον εκτέλεσης καταγράφονται αρχεία καταγραφής (logs) που αφορούν σφάλματα συστήματος και ενέργειες χρηστών, τα οποία αποθηκεύονται σε ειδικούς πίνακες της βάσης δεδομένων για σκοπούς ιχνηλασιμότητας και ανάλυσης, όπως παρουσιάζεται στους Πίνακες Η-5 και Η-6. Εφαρμόζεται πολιτική μηδενικών προειδοποιήσεων και σφαλμάτων σε όλο τον κώδικα. Σε δοκιμαστική λειτουργία ενεργοποιείται ρητά η προβολή σφαλμάτων (errors) και προειδοποιήσεων (warnings), ενώ σε παραγωγική λειτουργία η εμφάνισή τους απενεργοποιείται και τα σχετικά δεδομένα καταγράφονται στο αρχείο καταγραφής του συστήματος και του διακομιστή. Κατά την ολοκλήρωση της ανάπτυξης της πλατφόρμας πραγματοποιήθηκε πλήρης έλεγχος, χωρίς να

εντοπιστούν ενεργές αναφορές σφαλμάτων ή προειδοποιήσεων. Τα μηνύματα σφαλμάτων περιλαμβάνουν εξειδικευμένο περιγραφικό κείμενο και μοναδικό κωδικό αναφοράς (π.χ. [ERR0035]) για τον εύκολο εντοπισμό του αρχείου και της γραμμής σφάλματος.

Σε κάθε ενέργεια του συστήματος παρέχεται άμεσο και σαφές feedback προς τον χρήστη (μήνυμα επιτυχίας ή αποτυχίας), το οποίο βασίζεται στην έκβαση της αντίστοιχης κλήσης. Ειδικά στις λειτουργίες που αφορούν τη βάση δεδομένων, τα μηνύματα επιτυχίας εμφανίζονται μόνο όταν η εκτέλεση ολοκληρώνεται επιτυχώς και σε αντίθετη περίπτωση εμφανίζεται μήνυμα αποτυχίας με το σφάλμα να καταγράφεται για περαιτέρω ανάλυση.

4.1.6 Απόδοση, Βελτιστοποίηση και Cache

Για τη βελτίωση της απόδοσης της εφαρμογής υλοποιήθηκε μηχανισμός προσωρινής αποθήκευσης (cache) δεδομένων, αντίστοιχος σε επίπεδο λογικής με συστήματα όπως το Redis, ο οποίος όμως αναπτύχθηκε εξατομικευμένα. Η κύρια λειτουργία cache παρουσιάζεται στον Πίνακα Η-4. Η επιλογή αυτή διασφαλίζει τη μεταφερσιμότητα του κώδικα και την ανεξαρτησία της εφαρμογής από εξωτερικές υπηρεσίες. Ο μηχανισμός cache εφαρμόζεται αποκλειστικά σε σταθερά ή σπάνια μεταβαλλόμενα δεδομένα, τα οποία δεν περιλαμβάνουν προσωπικές πληροφορίες χρηστών. Τα δεδομένα που ανακτώνται από τη βάση δεδομένων αποθηκεύονται στον φάκελο cache σε μορφή JSON και επαναχρησιμοποιούνται σε επόμενα ισοδύναμα αιτήματα, εφόσον το αντίστοιχο αρχείο υφίσταται και κρίνεται έγκυρο. Σε διαφορετική περίπτωση, το σχετικό ερώτημα εκτελείται εκ νέου στη βάση δεδομένων και το αποτέλεσμα αποθηκεύεται εκ νέου στο cache. Παράλληλα, έχει υλοποιηθεί μηχανισμός ανακύκλωσης, μέσω του οποίου παλαιά ή μη έγκυρα αρχεία cache διαγράφονται περιοδικά, ώστε να διασφαλίζεται η επικαιροποίηση των δεδομένων. Η προσέγγιση αυτή μειώνει τον αριθμό των ερωτημάτων προς τη βάση δεδομένων και επιτρέπει στον διακομιστή να αξιοποιεί αποδοτικότερα τους διαθέσιμους υπολογιστικούς πόρους, ιδίως τη μνήμη.

4.1.7 Έλεγχος Ποιότητας και Εμπειρία Χρήστη

Κατά την ανάπτυξη της πλατφόρμας πραγματοποιήθηκε αναλυτική καταγραφή μετρικών με τη χρήση διαφόρων εργαλείων ελέγχου, ενώ παράλληλα υλοποιήθηκε ανάλυση διασφάλισης ποιότητας μέσω αυτοματοποιημένων ελέγχων και εργαλείων στατικής ανάλυσης (static

analysis). Τα αποτελέσματα τόσο των μετρικών όσο και της ανάλυσης ποιότητας έχουν συγκεντρωθεί και παρουσιαστεί στο Κεφάλαιο 5 «Αξιολόγηση και Αποτελέσματα».

Ο κώδικας έχει σχεδιαστεί με βάση την αρχή της ελάχιστης προσπάθειας από την πλευρά του χρήστη. Ζητούνται μόνο τα απολύτως απαραίτητα στοιχεία, όπως για παράδειγμα σύνδεση αποκλειστικά με όνομα χρήστη και κωδικό πρόσβασης, ενώ μηχανισμοί όπως το Captcha ενεργοποιούνται μόνο σε ανοικτές φόρμες για την αποτροπή αυτοματοποιημένων ενεργειών (bots). Ο ρόλος του χρήστη δεν δηλώνεται ρητά, αλλά αντλείται από τη βάση δεδομένων. Σε περίπτωση αποτυχίας επικύρωσης (validation), τα δεδομένα της φόρμας διατηρούνται μέσω $_SESSION και επανεμφανίζονται προσυμπληρωμένα, συνοδευόμενα από κατάλληλα μηνύματα σφάλματος στα αντίστοιχα πεδία, ώστε να διευκολύνεται η άμεση διόρθωση. Επιπλέον, συχνά χρησιμοποιούμενα δεδομένα ή σήματα κατάστασης (flags) αποθηκεύονται σε δομές $_SESSION με μοναδική και σαφή ονοματοδοσία, αποφεύγοντας την αποθήκευση ευαίσθητων πληροφοριών για λόγους ασφαλείας.

Οι φόρμες υποβάλλονται αποκλειστικά με τη μέθοδο POST, ενώ η μέθοδος GET χρησιμοποιείται μόνο για πλοήγηση στον ιστοχώρο και σε περιπτώσεις ανακατεύθυνσης. Με τον τρόπο αυτό αποφεύγεται η καταγραφή ευαίσθητων δεδομένων σε αρχεία καταγραφής (logs) του διακομιστή, σε περιπτώσεις όπου καταγράφονται αιτήματα POST/GET.

Η σχεδίαση και ανάπτυξη του ιστοχώρου πραγματοποιήθηκε με προσαρμοσμένο (custom) κώδικα σε PHP, χωρίς τη χρήση πλήρους πλαισίου ανάπτυξης (framework) (βλ. υποενότητα 2.3.5), και με αξιοποίηση επιλεγμένων εξωτερικών βιβλιοθηκών, όπως η PHPMailer[27] για αποστολή email μέσω SMTP (βλ. υποενότητα 2.1.4), η PHPSpreadsheet[28] για τη διαχείριση αρχείων Excel, καθώς και η βιβλιοθήκη FPDF[29] για τη δημιουργία εγγράφων PDF. Λόγω της παλαιότητας της τελευταίας, πραγματοποιήθηκε refactoring σε τμήματα του κώδικά της ώστε να διασφαλιστεί η συμβατότητα με νεότερες εκδόσεις της PHP (≥ 8.1) και η ομαλή ενσωμάτωσή της στην αρχιτεκτονική της εφαρμογής. Η επιλογή ανάπτυξης με προσαρμοσμένο (custom) κώδικα κρίθηκε καταλληλότερη για τις ανάγκες της παρούσας εργασίας, καθώς προσφέρει ελαφρύτερη και αποδοτικότερη υλοποίηση, μεγαλύτερη ευελιξία στη συντήρηση και την επεκτασιμότητα, καθώς και δυνατότητα πλήρους προσαρμογής του κώδικα στις απαιτήσεις του έργου. Επιπλέον, η προϋπάρχουσα εμπειρία

[27] https://github.com/PHPMailer/PHPMailer
[28] https://github.com/PHPOffice/PhpSpreadsheet
[29] https://www.fpdf.org/

στην ανάπτυξη εφαρμογών με custom κώδικα επέτρεψε ταχύτερη και αποτελεσματικότερη υλοποίηση, χωρίς την ανάγκη εκτεταμένης περιόδου εκμάθησης, όπως συμβαίνει συνήθως με ολοκληρωμένα frameworks.

Για τη διεπαφή της πλατφόρμας χρησιμοποιήθηκε, πέραν του προσαρμοσμένου κώδικα, το Bootstrap 5, τόσο σε επίπεδο CSS όσο και μέσω της ελαχιστοποιημένης JavaScript βιβλιοθήκης του (βλ. υποενότητα 2.2.4), με στόχο την υλοποίηση ενιαίου και πλήρως προσαρμοζόμενου (responsive) σχεδιασμού για κινητές συσκευές και υπολογιστές γραφείου, καθώς και τη βελτιστοποίηση της εμφάνισης στοιχείων διεπαφής, όπως παράθυρα, κουμπιά και επιλογές. Η διεπαφή ελέγχθηκε σε όλα τα στάδια ανάπτυξης ως προς την ορθή λειτουργία και εμφάνιση σε διαφορετικές διαστάσεις οθονών και συσκευές, με τη χρήση εργαλείων όπως τα Chrome DevTools, Mobile Web Switcher και Window Resizer, καθώς και μέσω δοκιμών σε πραγματικές συνθήκες χρήσης.

Στον σχεδιασμό της διεπαφής προστέθηκαν λειτουργίες για τη βελτίωση της προσβασιμότητας, ώστε η πλατφόρμα να μπορεί να χρησιμοποιηθεί αποτελεσματικά και από χρήστες που αξιοποιούν υποστηρικτικές τεχνολογίες (π.χ. screen readers) ή εναλλακτικούς τρόπους πλοήγησης (π.χ. πληκτρολόγιο). Για τον σκοπό αυτό αξιοποιήθηκαν σημασιολογικά στοιχεία HTML (semantic HTML) (βλ. υποενότητα 2.2.1), καθώς και προσθήκη γνωρισμάτων ARIA attributes (Accessible Rich Internet Applications), όπως το aria-label που φαίνεται στον Πίνακα 4-1, με σκοπό να παρέχονται σαφείς περιγραφές σε κουμπιά, πεδία και εικονίδια. Παράλληλα, χρησιμοποιήθηκαν συγκεκριμένες τεχνικές CSS για λειτουργίες αλλαγής εμφάνισης (φωτεινό ή σκοτεινό θέμα, αλλαγή μεγέθους κειμένων), όπως παρουσιάζεται στον Πίνακα 4-3, με εμφανή κουμπιά επιλογής στη διεπαφή χρήστη.

4.2 Υλοποίηση Κώδικα (Source Code)

Στην ενότητα αυτή αναλύεται η υλοποίηση του πηγαίου κώδικα της πλατφόρμας, με έμφαση στη δομή, την οργάνωση και τη λειτουργική διάκριση των επιμέρους τμημάτων. Παρουσιάζονται χαρακτηριστικά αποσπάσματα κώδικα, τα οποία συνοδεύονται από επεξήγηση της λογικής τους και των σχεδιαστικών επιλογών που υιοθετήθηκαν.

4.2.1 Δομή Αρχείων Κώδικα

Για την καλύτερη οργάνωση και τη διευκόλυνση κατά την υλοποίηση, τα αρχεία του κώδικα της εφαρμογής χωρίζονται σε διάφορα επίπεδα φακέλων, βάσει των λειτουργιών που εξυπηρετούν, όπως για παράδειγμα αρχεία κλάσεων, αρχεία διεπαφής χρήστη, αρχεία επιχειρησιακής λογικής, κοινόχρηστα αρχεία συστήματος, κ.ά. Η οργάνωση σε φακέλους παρουσιάζεται στο Σχήμα 5 παρακάτω:

```
root/
      ├── api/
      ├── backend/
      ├── cache/
      ├── classes/
      │     └── config/
      ├── Enum/
      ├── helpers/
      ├── images/
      ├── includes/
      ├── js/
      ├── libs/
      │     ├── phpmailer/
      │     └── phpspreadsheet/
      ├── pathtopem/
      ├── stats/
      │     └── logs/
      └── uploads/
            └── imports/
```

Σχήμα 5 - Ενδεικτική ιεραρχική δομή φακέλων της εφαρμογής

- Φάκελος «root»: Περιέχει τα κεντρικά αρχεία της διεπαφής χρήστη (frontend) της εφαρμογής.
- Φάκελος «api»: Περιλαμβάνει ανεξάρτητα αρχεία διεπαφών προγραμματισμού εφαρμογών (API), τα οποία εξυπηρετούν συγκεκριμένες λειτουργικές αιτήσεις της εφαρμογής, όπως η μεταφόρτωση και η ανάκτηση αρχείων εικόνας, αποστολή μηνυμάτων κ.ά.
- Φάκελος «backend»: Περιέχονται αρχεία χειρισμού αιτημάτων και ροών (handlers).
- Φάκελος «cache»: Περιέχει αρχεία JSON προσωρινής μνήμης, τα οποία αποθηκεύουν δεδομένα από τη βάση δεδομένων ώστε να αποφεύγεται η συχνή εκτέλεση ερωτημάτων για δεδομένα χαμηλής μεταβλητότητας.
- Φάκελος «classes»: Αρχεία κλάσεων για την πρόσβαση και τη διαχείριση δεδομένων στα οποία περιλαμβάνονται κλάσεις που συγκεντρώνουν τις PDO/SQL εντολές, κλάσεις ελέγχου δεδομένων για εξυγίανση (sanitize) και επικύρωση (validation) δεδομένων εισόδου από χρήστες, καθώς και άλλες βοηθητικές κλάσεις όπως μεταφράσεις σε επιλεγμένη γλώσσα.
- Φάκελος «config»: Βοηθητικά αρχεία κλάσεων που περιέχουν σταθερά δεδομένα, σχήματα και ονόματα συσχέτισης πεδίων ΒΔ με μεταβλητές του κώδικα.
- Φάκελος «Enum»: Φάκελος αρχείων δημιουργίας κύριου κορμού πλοήγησης καθώς και συσχέτισης πεδίων και μεταβλητών της διεπαφής χρήστη με τον κώδικα επιχειρησιακής λογικής.
- Φάκελος «helpers»: Περιλαμβάνει βοηθητικά αρχεία τα οποία χρησιμοποιούνται από το frontend και υλοποιούν κοινόχρηστες συναρτήσεις, συμπεριλαμβανομένης της δυναμικής παραγωγής HTML.
- Φάκελος «images»: Διαδρομή εγκατεστημένων αρχείων εικόνων.
- Φάκελος «includes»: Περιλαμβάνει αρχεία κύριας χρήσης, εκκίνησης και παραμετροποίησης της εφαρμογής, αρχεία ελέγχου πρόσβασης, καθώς και βοηθητικά αρχεία που υλοποιούν κοινές συναρτήσεις, σχήματα και μεθόδους.
- Φάκελος «js»: Περιλαμβάνει αρχεία JavaScript που χρησιμοποιούνται στο frontend της εφαρμογής.

- Φάκελος «libs»: Περιλαμβάνει αρχεία βιβλιοθηκών που χρησιμοποιούνται από την πλατφόρμα.
- Φάκελος «phpmailer»: Περιλαμβάνει τη βιβλιοθήκη ανοικτού κώδικα PHPMailer, η οποία χρησιμοποιείται για την αποστολή ηλεκτρονικού ταχυδρομείου μέσω του πρωτοκόλλου SMTP.
- Φάκελος «phpspreadsheet»: Περιλαμβάνει τη βιβλιοθήκη ανοικτού κώδικα PHPSpreadsheet, η οποία χρησιμοποιείται για την επεξεργασία αρχείων Excel (.xls, .xlsx) κατά τη μεταφόρτωση από τον χρήστη αρχείων δεδομένων για μαζική εισαγωγή.
- Φάκελος «pathtopem»: Διαδρομή προς το αρχείο πιστοποιητικού PEM, το οποίο χρησιμοποιείται για την επαλήθευση της ασφάλειας της σύνδεσης μέσω cURL.
- Φάκελος «stats»: Αποτελεί τη διαδρομή αποθήκευσης αρχείων στατιστικών χρήσης της πλατφόρμας.
- Φάκελος «logs»: Υποφάκελος αποθήκευσης αρχείων καταγραφής συμβάντων και σφαλμάτων, συμπληρωματικά της καταγραφής που πραγματοποιείται στη βάση δεδομένων.
- Φάκελος «uploads»: Στον φάκελο αυτό πραγματοποιείται η αποθήκευση των αρχείων που μεταφορτώνουν οι χρήστες.
- Φάκελος «imports»: Υποφάκελος για την προσωρινή αποθήκευση των αρχείων μεταφόρτωσης με τα δεδομένα μαζικής εισαγωγής.

4.2.2 Frontend

Η υποενότητα αυτή αφορά την υλοποίηση του frontend της πλατφόρμας, δηλαδή τη διεπαφή χρήστη και την παρουσίαση της πληροφορίας. Αναλύεται ο τρόπος με τον οποίο υλοποιείται η εμφάνιση, η πλοήγηση και η αλληλεπίδραση με τον χρήστη, με έμφαση στη σαφήνεια, τη χρηστικότητα και την προσαρμογή σε διαφορετικές συσκευές.

Αρχείο _html.php: Βασικές συναρτήσεις εξόδου HTML προς τη διεπαφή χρήστη.

```php
<?php

/**
 * _html.php
 * HTML output globaly used functions that are stand-alone
 * ### IMPORTANT: Page must contain only stand-alone functions that do not require
 * loading other files.
 *
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : All pages
 * Linked: none
 */

declare(strict_types=1);

/**
 * Echo with new line.
 *
 * @param $str  The outpout string to echo with new line
 */
function ecn(string $str): void {
  echo($str . "\n");
}

/**
 * Echo with html line break.
 *
 * @param string|int $str  The outpout string to echo with html line break
 */
function ecb(string|int $str): void {
  echo($str . '<br>');
}

/**
 * Short function for HTML (to echo html return).
 *
 * @param mixed $text   A mixed text to clear for html output
 *                      (also add custom break as <br> if "\n" given)
 * @return string       The text string to output for html
 */
function e_html($text): string {
  $str = ((is_scalar($text)) || ($text === null)) ? (string) $text : '';
  return nl2br(htmlspecialchars($str, ENT_QUOTES, 'UTF-8'));
}

/**
 * Short function for Javascript (to echo javascript).
 *
 * @param string $text  The text to handle and output for javascript
 * @return string       The text string to output
 */
```

```php
function e_js(string $text): string {
  $jsonEncode = json_encode($text, JSON_UNESCAPED_UNICODE | JSON_UNESCAPED_SLASHES);
  return ($jsonEncode !== false) ? mb_trim($jsonEncode, '"') : '';
}

/**
 * Set the maxlength and field size for the input.
 * @param $value    The max length of the field, omit if not set
 * @return string   The text string to output
 */
function e_field_size(int $value = 0): string {
  $return = '';
  if ($value !== 0) {
    $return = ' maxlength="' . e_html($value) . '" ';
  }
  return $return;
}

/**
 * Set the placeholder for the input text field.
 * @param $value   The placeholder text for the field
 * @return string  Outputs html element attributes
 */
function e_placeholder(string $value = ''): string {
  $return = '';
  if (!isNullEmpty($value)) {
    $return = ' placeholder="(' . e_html($value) . ')" ';
  }
  return $return;
}

/**
 * Add class for small fields to set field size.
 * @param $value   The max length of the field, omit if not set
 * @return string  Outputs html class name for auto width
 */
function e_short_field_class(int $value = 0): string {
  $return = '';
  if (($value > 0) && ($value <= 20)) {
    $return = ' w-auto ';
  }
  return $return;
}

/**
 * Add required attribute to form field
 * @param $attribute  The schema attribute for a specific field
 * @return string     Outputs html required attributes
 */
function e_required(string $attribute = ''): string {
  $return = '';
  if ($attribute === REQUIRED) {
    $return = ' required ';
  }
  return $return;
}
```

```php
/**
 * Output input text form element.
 * @SuppressWarnings("ExcessiveParameterList")
 * @param string $field           The field name in the schema
 * @param string $field_id        The field id to use for the element
 * @param string $value           The field value in the form
 * @param int $max_length         The max length for the input
 * @param string $placeholder     Placeholder to show in the field
 * @param string $ttl             Title/label of the input
 * @param string $data_normalize  Javascript normalization function to call
 * @param string $data_validate   Javascript validation function to call
 * @param string $required        If the field is required in the form
 * @param bool $editable          If the field is editable or disabled/read-only
 * @param string $tooltip         Tooltip text if set
 * @param string $classes         Custom class names to override default
 * @return string                 Outputs html input text
 */
function e_input_text(
  string $field,
  string $field_id,
  string $value,
  int $max_length,
  string $placeholder,
  string $ttl,
  string $data_normalize,
  string $data_validate,
  string $required,
  bool $editable,
  string $tooltip,
  string $classes = ''
): string {
  $calendar = '';
  if (($data_normalize === 'normalize_date') && ($editable)) {
    $classes .= ' d-inline-block ';
    $calendar = (
      '<input type="date" id="' . e_html($field_id) . '-custom" value=""' .
      '  class="visually-hidden js-date-custom"' .
      '  tabindex="-1"' .
      '>' .
      '<img' .
      '  src="images/calendar.png"' .
      '  id="' . e_html($field_id) . '-btn" class="cursor-pointer d-inline-block"' .
      '  data-btn-type="date" data-btn-id="' . e_html($field_id) . '-custom"' .
      '>' . "\n"
    );
  }
  $classes .= (($data_normalize === 'normalize_date') ? ' d-inline-block ' : '');
  return('<input ' .
    'type="text" ' .
    'class="form-control app-shadow-sm ' .
      e_short_field_class($max_length) . ' ' . e_html($classes) . '" ' .
    'name="' . e_html($field) . '" ' .
    'id="' . e_html($field_id) . '" ' .
    'value="' . e_html($value) . '" ' .
    e_field_size($max_length) . ' ' .
```

```
    e_placeholder($placeholder) . ' ' .
    'data-label="' . e_html($ttl) . '" ' .
    'data-normalize="' . e_html($data_normalize) . '" ' .
    'data-validate="' . e_html($data_validate) . '" ' .
    ($editable ? '' : ' readonly ') .
    e_required($required) . '>' .
    "\n" .
    $calendar .
    ($editable ? e_tooltip_str($tooltip, false) : '') .
    "\n"
  );
}

/**
 * Output input select form element.
 * The width of the select element is 100% when in mobile or when option list has
 * long text. For smaller select option lists, JavaScript on load will add the
 * class "select-w-short" to force 200px width
 * and match the input text elements with short values for better visuals.
 * To force a select element to have short width (200px), we need to add the class
 * "select-w-short" in the call below or we can force other fiexed width with proper
 * class like "select-w-300" (300px) (for desktop only).
 *
 * @param string $disabled        If field is disabled, write disable to tag, field name
 * @param string $field           The field name in the schema
 * @param string $field_id        The field id to use for the element
 * @param string $ttl             Title/label of the input
 * @param string $required        If the field is required in the form
 * @param bool $editable          If the field is editable or disabled/read-only
 * @param string $tooltip         Tooltip text if set
 * @param string $classes         Custom class names to override default
 * @return string                 Outputs html input select
 */
function e_input_select(
  string $disabled,
  string $field,
  string $field_id,
  string $ttl,
  string $required,
  bool $editable,
  string $tooltip,
  string $classes = ''
): string {
  return(((($editable) ? e_tooltip_str($tooltip, false) : '') . "\n" .
    '<select ' .
    e_html($disabled) . ' ' .
    'name="' . e_html($field . $disabled) . '" ' .
    'id="' . e_html($field_id . $disabled) . '" ' .
    'class="selectbox app-shadow-sm select-w-max ' .
      e_html($disabled) . ' ' . e_html($classes) . '" ' .
    'data-label="' . e_html($ttl) . '" ' .
    e_required($required) . '>' .
    "\n"
  );
}
```

```php
/**
 * Output tooltip button (input as text directly).
 * @param string $tooltip         The tooltip text
 * @param bool $inline            True to put directly near the element
 * @return string                 Outputs html button tooltip
 */
function e_tooltip_str(string $tooltip, bool $inline): string {
  if (!isNullEmpty($tooltip)) {
    $icon = 'd-flex align-items-center justify-content-center mt-1 me-1 help-icon';
    if ($inline) {
      $icon = 'help-icon-inline';
    }
    $tooltip =
      "\n" .
      '<!-- Help icon with tooltip -->' .
      '<button ' .
      'type="button" ' .
      'tabindex="1000" ' .
      'class="btn btn-outline-secondary ' . $icon . '" ' .
      'data-bs-toggle="tooltip" ' .
      'data-bs-placement="top" ' .
      'data-bs-trigger="hover focus" ' .
      'data-bs-custom-class="tip-light" ' .
      'data-bs-title="' . e_html(translate($tooltip)) . '" ' .
      'title="' . e_html(translate($tooltip)) . '" ' .
      'aria-label="' . e_html(translate($tooltip)) . '">' .
      '?' .
      '</button>'
    ;
  }
  return $tooltip;
}

/**
 * Output tooltip button (reference by array key tooltip).
 * @param string $field           Field to get tooltip text if set
 * @return string                 Outputs html button tooltip
 */
function e_tooltip_fld(string $field): string {
  $fld = get_field_validator($field);
  $tooltip = $fld['tooltip'] ?? '';
  return e_tooltip_str($tooltip, false);
}

/**
 * Output input hidden form element.
 *
 * @param string $name
 * @param string $fid
 * @param string $value
 * @return string                 Outputs html input text
 */
function e_input_hidden(
  string $name,
  string $fid,
  string $value,
```

```
): string {
  return('<input ' .
    'type="hidden" ' .
    (($name !== '') ? 'name="' . $name . '" ' : '') .
    (($name !== '') ? 'id="' . $fid . '" ' : '') .
    'value="' . $value . '">' .
    "\n"
  );
}

/**
 * Write script to show reset button on form change
 * @param string $form_id   The form id to add change event listener
 * @return string           Output script
 */
function e_form_reset_js(string $form_id): string {
  $output  = '<script>' . "\n";
  $output .= e_js('document.addEventListener(\'DOMContentLoaded\', () => {') . "\n";
  $output .= e_js('  let form_reset = document.getElementById(\'' . $form_id . '\');') .
    "\n";
  $output .= e_js('  form_reset.addEventListener(\'change\', function() {') . "\n";
  $output .= e_js('    if (document.getElementById(\'' . $form_id . '-reset\')) {') . "\n";
  $output .= e_js('      document.getElementById(\'' . $form_id .
    '-reset\').classList.remove(\'hidden\');') . "\n";
  $output .= e_js('      document.getElementById(\'' . $form_id .
    '-reset\').style.opacity = \'1\';') . "\n";
  $output .= e_js('    }') . "\n";
  $output .= e_js('  });') . "\n";
  $output .= e_js('});') . "\n";
  $output .= '</script>' . "\n";
  return $output;
}

/**
 * Show current selected semester text on top of card
 * @SuppressWarnings("UnusedFormalParameter")
 * @param array<string, bool> $pageSettings
 * @param string $action
 * @return string
 */
function e_show_semester(array $pageSettings, string $action): string {
  $output = '';
  if ((isset($pageSettings['show_sem'])) && ($pageSettings['show_sem'])) {
    $output .= '<div class="app-semester">';
    $output .= e_html(translate(getSessionArrStr('smsys_curr_semester', 'label')));
    $output .= ' ';
    $output .= e_html(getSessionArrStr('smsys_curr_semester', 'year'));
    $output .= '</div>';
  }
  return $output;
}
```

Πίνακας 4-1: Απόσπασμα κώδικα αρχείου "_html.php"

Αρχείο common_html_header.php: Κοινό αρχείο φόρτωσης του τμήματος κεφαλίδας κάθε ιστοσελίδας και καθορισμό προβολής βάσει ρόλου χρήστη.

```php
<?php

/**
 * common_html_header.php
 * Common header for html pages
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : {frontend}
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Enum\AdminPage as NavPage;
use SMSYS\Enum\Roles;

// Prevent direct page access and check access level
$scriptFilename = filter_input(INPUT_SERVER, 'SCRIPT_FILENAME');
$basename_filename = is_string($scriptFilename) ? $scriptFilename : '';
$samePage = (basename(__FILE__) === basename($basename_filename));
// Set roles of Access Level for the guard
$requiredRoles = [];
// Load page guard, if it fails any of the checks (page, role, initialization) it redirects
 to start page
require_once __DIR__ . '/../includes/_guard.php';

require_once __DIR__ . '/common_html_menu.php';

$curr_page = basename($basename_filename);
$menu_links_common = [
  'Αρχική'              => [
    'icon' => 'icon-home.png', 'link' => 'index.php',
  ],
  'Ωρολόγιο Πρόγραμμα'  => [
    'icon' => 'icon-timetable.png', 'link' => 'timetable.php',
  ],
  'Πρόγραμμα Εξετάσεων' => [
    'icon' => 'icon-exams.png', 'link' => 'exams.php',
  ],
  'Ανακοινώσεις'        => [
    'icon' => 'icon-announcements.png', 'link' => 'announcements.php',
  ],
];
$menu_links_admin = [
  'Κεντρική Σελίδα'       => [
    'icon' => 'icon-home.png', 'link' => 'admin_main.php'
  ],
  'Διαχείριση Χρηστών'    => [
    'icon' => 'icon-users.png',
    'link' => [
```

```
      'Προβολή Εκκρεμών Ενεργοποιήσεων Λογαριασμών',
      'Διαχείριση Λογαριασμών Διαχειριστών',
      'Διαχείριση Διδασκόντων',
      'Διαχείριση Φοιτητών',
    ]
  ],
  'Διαχείριση Παραμέτρων' => [
    'icon' => 'icon-params.png',
    'link' => [
      'Ρυθμίσεις Παραμέτρων Συστήματος',
      'Ορισμός Κατευθύνσεων',
      'Ορισμός Επίσημων Αργιών',
      'Διαχείριση Σχολών',
      'Διαχείριση Τμημάτων',
      'Διαχείριση Αιθουσών',
      'Διαχείριση Μαθημάτων',
    ]
  ],
  'Διαχείριση Προγραμμάτων' => [
    'icon' => 'icon-schedules.png',
    'link' => [
      'Διαθεσιμότητα Διδασκότων',
      'Πρόγραμμα Εξαμήνου',
      'Πρόγραμμα Εξετάσεων',
    ]
  ],
];

/** @var bool|null $active_view */
$tmp_active_view = $active_view ?? null;
$active_view = is_bool($tmp_active_view) ? $tmp_active_view : false;
/** @var string|null $header_class */
$tmp_header_class = $header_class ?? null;
$header_class = is_string($tmp_header_class) ? $tmp_header_class : '';
/** @var string|null $html_mainpage */
/** @var string|null $html_btn_id */
/** @var string|null $html_btn_name */
/** @var string $html_scroll_flag */
?>
</head>

<body class="<?= e_active_body($active_view) ?>">
<input type="hidden" id="page-lang" value="<?= getSessionStr('smsys_language') ?>">

<header class="app-header my-shadow-sm <?= e_html($header_class) ?>">
  <nav class="navbar navbar-expand-lg app-navbar container-fluid">
    <a class="navbar-brand fw-bold app-logo d-flex align-items-center" href="#">
      <span class="app-logo-mark">
        <?php if (isNullEmpty(getSessionStr('smsys_attr_university_logo'))) : ?>
        <img
          src="images/your-logo.png"
          alt="<?= e_html(translate(getSessionStr('smsys_attr_university_name'), true)) ?>"
          class="app-logo-img"
        >
        <?php else : ?>
        <img
```

```
        src="uploads/<?= e_html(getSessionStr('smsys_attr_university_logo')) ?>"
        alt="<?= e_html(translate(getSessionStr('smsys_attr_university_name'), true)) ?>"
        class="app-logo-img"
      >
      <?php endif; ?>
    </span>
    <span class="app-logo-text">
      <span class="app-logo-title">
        <?php
        if (
          (isset($_SESSION['smsys_loggedin'])) &&
          ($_SESSION['smsys_loggedin'] === true) &&
          (!isSuperAdmin())
        ) : ?>
          <?= e_html(translate(getSessionStr('smsys_user_department_name'), true)) ?>
          <?php
        else :
        ?>
          <?= e_html(translate('Πλατφόρμα Διαχείρισης Ωρολογίου Προγράμματος')) ?>
          <?php
        endif;
        ?>
      </span>
      <span
        class="app-logo-university"
      ><?= e_html(translate(getSessionStr('smsys_attr_university_name'), true)) ?></span>
    </span>
  </a>

  <button class="navbar-toggler app-navbar-toggler" type="button"
    data-bs-toggle="collapse" data-bs-target="#mainNavbar"
    aria-controls="mainNavbar" aria-expanded="false"
    aria-label="<?= e_html(translate('Εναλλαγή πλοήγησης')) ?>"
  ><span class="navbar-toggler-icon"></span></button>

  <div class="collapse navbar-collapse" id="mainNavbar">
    <ul class="navbar-nav mx-auto mb-2 mb-lg-0 app-nav-links">
    <?php
    if ((isset($_SESSION['smsys_loggedin'])) && ($_SESSION['smsys_loggedin'] === true)) :
      $menu_counter = 0;
      /** @var array<string, array{
       * icon: string|null, link: string|list<string>}> $menu_links_admin */
      foreach ($menu_links_admin as $title => $menu_item) :
        $link = $menu_item['link'];
        if (!is_array($menu_item['link'])) :
          ?>
          <li class="nav-item">
            <a class="nav-link" aria-current="page" href="<?= e_html($link) ?>">
              <?php
              if (!isNullEmpty($menu_item['icon'])) :
                ?>
                <img
                  src="images/<?= e_html($menu_item['icon']) ?>"
                  class="menu-icon invert" alt=""
                >
                <?php
```

```
                endif;
                ?>
                <?= render_menu_title(translate($title)) ?>
                <span class="menu-dd-space" aria-hidden="true"> </span>
              </a>
            </li>
            <?php
          else :
            $menu_counter++;
            $menu = $menu_item['link'];
            ?>
            <div class='simpledropdown--center simpledropdown mpointer me-1'>
              <input
                type="checkbox" class="simpledropdown-toggle js-menu-toggle"
                id="menu-checkbox-<?= e_html($menu_counter) ?>"
              >
              <label
                class='simpledropbtn'
                for="menu-checkbox-<?= e_html($menu_counter) ?>"
                aria-haspopup="true" aria-expanded="false"
              >
                <li class="nav-item">
                  <span class="nav-link" aria-current="page">
                  <?php
                    if (!isNullEmpty($menu_item['icon'])) :
                      ?>
                      <img
                        src="images/<?= e_html($menu_item['icon']) ?>"
                        class="menu-icon invert" alt=""
                      >
                      <?php
                    endif;
                    ?>
                    <?= render_menu_title(translate($title)) ?>
                    <span class="menu-dd-arrow" aria-hidden="true">&#8628;</span>
                  </span>
                </li>
              </label>
              <div class='simpledropdown-arrow-top '></div>
              <div class='simpledropdown-content txtleft'>
                <div class='simpledropdown-gap'></div>
                <?php foreach ($menu as $title_inner) : ?>
                  <?= render_menu_item($title_inner) ?>
                <?php endforeach; ?>
              </div>
            </div>
            <?php
          endif;
        endforeach;
        ?>
        </ul>
/* code continues further... */
```

Πίνακας 4-2: Απόσπασμα κώδικα αρχείου "common_html_header.php"

Αρχείο styles.css: Κύριο αρχείο που περιέχει τους κανόνες (rules) μορφοποίησης CSS της πλατφόρμας.

```
/* Colors for different modes (Night/Day or EAA) */
:root {
  color-scheme: light dark;
}
:root {
  color-scheme: light;

  --app-bg: #f5f7fb;
  --app-bg-alt: #ffffff;
  --app-bg-deep: #0b1020;

  --app-text-main: #111827;
  --app-text-muted: #6b7280;
  --app-text-invert: #f9fafb;

  --app-primary: #2563eb;
  --app-primary-soft: rgba(37, 99, 235, 0.12);
  --app-primary-border: rgba(37, 99, 235, 0.4);

  --app-outline: #4f76cc;
  --app-outline-soft: rgba(113, 144, 211, 0.12);
  --app-outline-border: rgba(126, 150, 201, 0.4);

  --app-card-border: rgba(15, 23, 42, 0.08);
  --app-shadow-soft: 0 15px 30px rgba(15, 23, 42, 0.12);

  --app-input-bg: #f9fafb;
  --app-input-border: #d1d5db;
  --app-footer-bg: #020617;
  --app-footer-text: #9ca3af;

  --hex-ffffff: #ffffff;
  --hex-000000: #000000;
  --rgb-255-155-0: rgb(255, 155, 0);
  --rgba-192-192-192--0-75: rgba(192, 192, 192, .75);
  --rgba-0-0-0--0-03: rgba(0, 0, 0, .03);
  --rgba-0-0-0--0-15: rgba(0, 0, 0, .15);
  --rgba-0-0-0--0-25: rgba(0, 0, 0, .25);
  --rgba-0-0-0--0-3: rgba(0, 0, 0, .3);
  --rgba-140-210-210-2: rgba(140, 210, 210, 0.2);
  --rgba-210-140-140-2: rgba(210, 140, 140, 0.2);
  --rgba-30-90-200-3: rgba(30, 90, 200, 0.3);

  /* Bootstrap custom variables */
  --bs-box-my-shadow-sm: 0 0.25rem 0.5rem rgba(0, 0, 0, 0.3);
  --bs-box-shadow-sm:0 0.15rem 0.30rem rgba(0, 0, 0, 0.15);
  --bs-box-shadow-md: 0 0.5rem 1rem rgba(0, 0, 0, 0.3);
  --bs-primary: #0d6efd;
  --bs-body-bg: #fff;
  --bs-body-color: #000;
}
.dark-mode {
```

```
  color-scheme: dark;

  --app-bg: #020617;
  --app-bg-alt: #020617;
  --app-bg-deep: #020617;

  --app-text-main: #e5e7eb;
  --app-text-muted: #9ca3af;
  --app-text-invert: #f9fafb;

  --app-primary: #38bdf8;
  --app-primary-soft: rgba(56, 189, 248, 0.14);
  --app-primary-border: rgba(56, 189, 248, 0.55);

  --app-outline: #38bdf8;
  --app-outline-soft: rgba(56, 189, 248, 0.14);
  --app-outline-border: rgba(56, 189, 248, 0.55);

  --app-card-border: rgba(148, 163, 184, 0.3);
  --app-shadow-soft: 0 15px 30px rgba(15, 23, 42, 0.9);

  --app-input-bg: #020617;
  --app-input-border: #1e293b;
  --app-footer-bg: #000000;
  --app-footer-text: #9ca3af;

  --hex-ffffff: #121212;
  --hex-000000: #ffffff;
  --rgb-255-155-0: rgb(255, 155, 0);
  --rgba-192-192-192--0-75: rgba(63, 63, 63, .75);
  --rgba-0-0-0--0-03: rgba(255, 255, 255, .03);
  --rgba-0-0-0--0-15: rgba(255, 255, 255, .15);
  --rgba-0-0-0--0-25: rgba(255, 255, 255, .25);
  --rgba-0-0-0--0-3: rgba(255, 255, 255, .3);
  --rgba-140-210-210-2: rgba(40, 110, 110, 0.2);
  --rgba-210-140-140-2: rgba(110, 40, 40, 0.2);
  --rgba-30-90-200-3: rgba(240, 220, 210, 0.3);

  /* Bootstrap custom variables */
  --bs-primary: #1255bb;
  --bs-body-bg: #121212;
  --bs-body-color: #eaeaea;
}

/* Adjustment for footer and containers */
html {
  height:100%;
}

/* Avoid white space in top/bottom and left/right of the page */
body {
  margin: 0 !important;
  padding: 0 !important;
  overflow-y: scroll;
  font-family: system-ui, -apple-system, BlinkMacSystemFont,
               "Segoe UI", "Roboto", "Ubuntu", "Cantarell",
```

```
               "Noto Sans", sans-serif;
  font-size: 1rem; /* 15px */
  line-height: 1.5;
  background: radial-gradient(  circle at top, rgba(37, 99, 235, 0.16), transparent 55%),
              radial-gradient(circle at bottom, rgba(56, 189, 248, 0.1), transparent 55%),
              var(--app-bg);
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: 100% 100%, 100% 100%, auto;
  color: var(--app-text-main);
  min-height: 100vh;
  display: flex;
  flex-direction: column;
}
@media (max-width: 991px) {
  body { background-attachment: scroll; }
}
body.prev-ver {
  color: var(--app-text-main);
}

/* Smooth loading page */
.trandiv {
  width: 100%;
  opacity: 0; /* 0 in production for fade load effect */
  transition: opacity 0.5s ease;
  -webkit-transition: opacity 0.5s ease;
}
.trandiv.is-loaded { opacity: 1; }
.htitle {
  display: block; margin: 0.0vh 0 2vh; width: 0%; height: 2px; border: 0;
  background-color: rgba(0,0,0,0.85); transition: all 1.0s ease; transition-delay: 0.1s;
}
.htitle.is-loaded { width: 100%; }

.app-header.h-wide { width: 100%; }

/* Page packground effects */
.mobBackStyleC {
  position: absolute;
  top: 0;
  left: 0;
  overflow: hidden;
  z-index: -1;
  opacity: 0.5;
  background: url(images/calend3d.png);
  background-position: 107% 95%;
  background-size: 40vh;
  background-repeat: no-repeat;
}

/* Basic links */
.href-blue { text-decoration: none; color: #1364b4; }
.href-blue:visited { text-decoration: none; color: #1364b4; }
.href-blue:hover { text-decoration: none; color: #1a8cff; }
.href-blue:active { text-decoration: none; color: #1a8cff; }
```

```css
body.dark-mode .href-blue { text-decoration: none; color: #ffffff; }
.href-none { text-decoration: none; }
.href-none:visited { text-decoration: none; }
.href-none:hover { text-decoration: none; color: #1a8cff; }
.href-none:active { text-decoration: none; color: #1a8cff; }
body.dark-mode .href-none { text-decoration: none; color: #ffffff; }
.btn-href { padding: 0; font-weight: 500; margin-top: -4px; }

/* Max container for main content */
.layout-container {
  max-width: 1440px; margin: 0 auto; padding: 0 1rem; box-sizing: border-box;
}
.lang-button {
  padding: 0; background-color: transparent; border: none;
}
.lang-img {
  vertical-align: middle;
  margin-bottom: 3px;
  border-radius: 3px;
  border: 1px solid rgb(180, 186, 192);
  max-height: 18px;
  box-shadow: 0 0 1px 0px white inset, 0 0 1px 0px white; /* anti-aliasing */
}
body.eaa-mode .lang-img { max-height: 30px; }
```

Πίνακας 4-3: Απόσπασμα κώδικα αρχείου "styles.css"

4.2.3 Backend

Στην παρούσα υποενότητα παρουσιάζεται η υλοποίηση του backend της πλατφόρμας, το οποίο είναι υπεύθυνο για την επιχειρησιακή λογική, τη διαχείριση δεδομένων και τον έλεγχο πρόσβασης. Περιγράφεται η οργάνωση του κώδικα, η επικοινωνία με τη Βάση Δεδομένων και οι μηχανισμοί ασφάλειας και ελέγχου που εφαρμόζονται.

Αρχείο config_smsys_const.php: Σταθερές τιμές (constants) που χρησιμοποιούνται στο σύστημα για την αποφυγή «μαγικών αριθμών» (magic numbers), για τον εύκολο καθορισμό και την αλλαγή τιμών βάσει των εκάστοτε αναγκών, καθώς και για την αποφυγή σφαλμάτων πληκτρολόγησης τιμών σε συγκρίσεις και ελέγχους.

```php
<?php

declare(strict_types=1);

const LOGIN_TRIES_CAPTCHA = 3;      // 3 login attempts from same username or from same IP
const LOGIN_TRIES_LIMIT = 6;        // 6 login attempts from same username or from same IP
const LOGIN_TIMER_RESET = 300;      // 5 minutes reset time
const LOGIN_TIMER_LOGOUT = 1800;    // 30 minutes idle time to log out
const LOGIN_PING_CHECK = 900;       // 15 minutes idle ping time to show stay in modal
```

```
const CSRF_TOKEN_POOL = 50;        // 50 random tokens on each session for form validation
const MAX_2FA = 5;                 // Maximum 2fa re-tries
const EXPIRE_CACHE_FILE_SEC = (3600 * 24);                // 1 DAY in seconds
const EXPIRE_ACCOUNT_ACTIVATION_SEC = (3600 * 24 * 30);   // 1 month in seconds
const EXPIRE_VERIFICATION_COOKIE_SEC = (3600 * 24 * 90);  // 3 months in seconds
const EXPIRE_VERIFICATION_ACCOUNT_SEC = (3600 * 30);      // 30 day in seconds
const EXPIRE_VERIFICATION_EMAIL_SEC = (60 * 30);          // 30 minutes
const EXPIRE_RESET_PASSWORD_SEC = (60 * 10);              // 10 minutes in seconds
const MAX_ACTIVE_HASH_TOKENS = 6;  // Maximum number of active hash tokens in same command
const MAX_IMPORT_BULK_ROWS = 25;   // Maximum number of data rows in bulk import
const MAX_MONTH_DAYS = 31;         // Max number of days for duration selections
const INIT_YEAR = 2025;            // Initial year for managing information
const SPRING = 'SPRING';
const FALL = 'FALL';
const SEMESTER = [
  SPRING => ['label' => 'Εαρινό Εξάμηνο',    'word' => 'SPRING'],
  FALL   => ['label' => 'Χειμερινό Εξάμηνο', 'word' => 'FALL']
];
const SEMESTER_PATTERN = '/\A(?P<year>20\d{2}|[3-9]\d{3})-(?P<semester>SPRING|FALL)\z/';
const COURSE_CATEGORY = [
  'MANDATORY'           => 'Υποχρεωτικό',
  'RESTRICTED_ELECTIVE' => 'Κατ’ επιλογήν υποχρεωτικό',
  'ELECTIVE'            => 'Επιλογής',
  'THESIS_SEMINAR'      => 'Σεμινάριο Δ.Ε.',
  'OPTIONAL'            => 'Προαιρετικό',
];
const COURSE_SEMESTER = [
  '1'  => 'Α\' Εξάμηνο',
  '2'  => 'Β\' Εξάμηνο',
  '3'  => 'Γ\' Εξάμηνο',
  '4'  => 'Δ\' Εξάμηνο',
  '5'  => 'Ε\' Εξάμηνο',
  '6'  => 'ΣΤ\' Εξάμηνο',
  '7'  => 'Ζ\' Εξάμηνο',
  '8'  => 'Η\' Εξάμηνο',
  '9'  => 'Θ\' Εξάμηνο',
  '10' => 'Ι\' Εξάμηνο',
  '11' => 'Κ\' Εξάμηνο',
  '12' => 'Λ\' Εξάμηνο',
];
const COURSE_TYPE = [
  'THEORY'    => 'Θεωρία / Διάλεξη',
  'LAB'       => 'Εργαστήριο',
  'EXERCISE'  => 'Φροντιστήριο / Άσκηση',
  'FIELDWORK' => 'Άσκηση Πεδίου',
  'CLINICAL'  => 'Κλινική Άσκηση',
  'WORKSHOP'  => 'Καλλιτεχνικό Εργαστήριο',
];
const DURATION_OF_DEGREE = [
  '1'  => '1 Έτος',
  '2'  => '2 Έτη',
  '3'  => '3 Έτη',
  '4'  => '4 Έτη',
  '5'  => '5 Έτη',
  '6'  => '6 Έτη',
];
```

```php
const CLASSROOM_CATEGORY = [
  'LECTURE'    => 'Αίθουσα Θεωρίας',
  'LABORATORY' => 'Αίθουσα Εργαστηρίων',
  'MIXED'      => 'Θεωρία + Εργαστήρια',
  'AUDITORIUM' => 'Αμφιθέατρο',
  'OTHER'      => 'Άλλη Κατηγορία',
];
const LANG_TEXT = ['en' => 'English', 'gr' => 'Ελληνικά'];
const WEEKDAYS = [
  '1'  => 'Δευτέρα',
  '2'  => 'Τρίτη',
  '3'  => 'Τετάρτη',
  '4'  => 'Πέμπτη',
  '5'  => 'Παρασκευή',
  '6'  => 'Σάββατο',
  '7'  => 'Κυριακή',
];
const AVA_TYPES = [
  '0'  => 'Περιορισμός',
  '1'  => 'Προτίμηση',
];

// Check server name to set status. false: testing, true: production
define('IS_IN_PRODUCTION',
  isset($_SERVER['HTTP_HOST']) &&
  (in_array($_SERVER['HTTP_HOST'] ?? '', ['smsys.edu.gr', 'www.smsys.edu.gr'], true))
);
define('CACHE_IS_ON',
  isset($_SERVER['HTTP_HOST']) &&
  (in_array($_SERVER['HTTP_HOST'] ?? '', ['smsys.edu.gr', 'www.smsys.edu.gr'], true))
);

define('CACHE_DIR', __DIR__ . '/cache');
define('UPLOAD_DIR', __DIR__ . '/uploads');

// Global variables for meaningfull comparisons
define('ADMIN', 'admin');
define('PROFESSOR', 'professor');
define('STUDENT', 'student');
define('USER', 'user');
define('NEWUSER', 'newuser');
define('NEWADMIN', 'newadmin');
define('NEWPROFESSOR', 'newprofessor');
define('NEWSTUDENT', 'newstudent');
define('RESET_SUCCESS', 'OK');
define('STATUS_OK', 1);
define('STATUS_FAIL', 0);
define('STATUS_ERROR', -1);
define('STATUS_IS_PENDING', 2);
define('STATUS_IS_ACTIVE', 1);
define('STATUS_IS_INACTIVE', 0);
define('FLD_NULL', 'null');
define('FLD_EMPTY', '');
define('FLD_IGNORE', 'igonre');
```

Πίνακας 4-4: Απόσπασμα κώδικα αρχείου “config_smsys_const.php”

Αρχείο _session_start.php: Αρχείο έναρξης και ελέγχου συνεδρίας (session) που καλείται κατά την έναρξη εκτέλεσης κάθε ιστοσελίδας ή API.

```php
<?php

/**
 * _sessions_start.php
 * Session start commands with proper checks, naming and security measures
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : _init.php
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Classes\Database;

// Load application constant values if not already loaded.
// phpcs:ignoreFile PSR1.Files.SideEffects.FoundWithSymbols
require_once __DIR__ . '/../config_smsys_const.php';

/**
 * Enforce secure session cookie parameters (for production).
 *
 * @return void   Starts the session
 */
function smsys_session_start(): void {
  if (session_status() !== PHP_SESSION_ACTIVE) {
    ini_set('session.cookie_httponly', 1); // no access to cookie from JS
    ini_set('session.use_strict_mode', 1); // reject uninitialized session ID
    ini_set('session.use_only_cookies', '1'); // disallow url-based ID
    // Set custom cookie parameters
    $secureHttps = isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] === 'on';
    session_name(SMSYS_SESSION_NAME);
    session_set_cookie_params([
      'lifetime' => 0,
      'path'     => '/',
      'domain'   => SMSYS_COOKIE_DOMAIN,
      'secure'   => $secureHttps,
      'httponly' => true,
      'samesite' => SMSYS_COOKIE_SAMESITE,
    ]);
    // Start new session if not started
    session_start();
```

```
  }
}

/**
 * Periodically regenerates session id if it has passed the time limit set in constants
 * Function must be called only in main fronend pages and after all initializations have be
en made
 *
 * @return void   Regenerate session id or return with no action
 */
function session_periodic_regenerate(): void {
  if (session_status() !== PHP_SESSION_ACTIVE) {
    return;
  }
  $now_time = time();
  $last_time = (int) getSessionInt('smsys_regen_at');
  $interval = (int) SMSYS_SESSION_REGEN_SEC;
  $regenerate = ($now_time - $last_time) >= $interval;
  if ($regenerate) {
    if (!isset($_SESSION['OBSOLETE']) || $_SESSION['OBSOLETE'] !== true) {
      $_SESSION['OBSOLETE'] = true;
      $_SESSION['EXPIRES']  = $now_time + 10;
      session_regenerate_id(false); // Creates new id with old id to stay valid for 10s to
avoid logout
      $newId = session_id();
      session_write_close(); // Release session
      session_id(($newId !== false) ? $newId : null);
      session_start(); // Resume session with the new id
      unset($_SESSION['OBSOLETE']);
      unset($_SESSION['EXPIRES']);
      $_SESSION['smsys_regen_at'] = $now_time;
      // Refresh the logged session id for the current user
      refresh_logged_session_id();
    }
  }
}

/**
 * Refresh the session id logged in the user table.
 * Function is called only from pages that load the initialization php to get the database
connection
 *
 * @return void
 */
function refresh_logged_session_id(): void {
  try {
    $pdo = initDatabase()->getConnection();
```

```
    $stmt = $pdo->prepare('
      UPDATE tbl_login_user
      SET user_session_id = ?
      WHERE user_id = ?
    ');
    $stmt-
>execute([encryptAES_public((string) session_id()), getSessionStr('smsys_userid')]);
  } catch (PDOException $err) {
    log_error(getSessionStr('smsys_userid'), $err-
>getMessage(), 'Log new session id error [ERR0065]', funcCallFile(2));
  }
}
```

Πίνακας 4-5: Απόσπασμα κώδικα αρχείου "_session_start.php"

Αρχείο AdminPage.php: Ορισμός τύπων απαρίθμησης Enum που υποστηρίζεται από την έκδοση PHP 8.1+, για την τυποποιημένη αναπαράσταση πεδίων, με σταθερές τιμές για έλεγχο εγκυρότητας και δυνατότητα αντιστοίχισής τους με άλλα σύνολα τιμών. Το αρχείο εξυπηρετεί την πλοήγηση και το σχήμα των σελίδων διαχειριστών (παρατίθεται ενδεικτικό τμήμα του αρχείου).

```
<?php

/**
 * AdminPage.php
 * Enum for page navigation
 * AccessLevel: Administrator
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : on-demand
 * Linked: none
 */

declare(strict_types=1);

namespace SMSYS\Enum;

// List of requested pages for admin enviroment navigation
enum AdminPage: string {
  // Frontend navigation pages
  case AdminMain                = '';
  case PendingActivation        = 'admin_account_pendingactivation.php';
  case CreateAdmin              = 'admin_account_createadmin.php';
  case ManageAdmin              = 'admin_account_manageadmin.php';
  case AdminsListing            = 'admin_account_adminlisting.php';
  case CreateProfessor          = 'admin_account_createprofessor.php';
  case ManageProfessor          = 'admin_account_manageprofessor.php';
```

```
  case ProfessorsListing         = 'admin_account_professorlisting.php';
  case UploadProfessor           = 'admin_account_uploadprofessor.php';
  case UploadProfessorVldt       = 'admin_account_uploadprofessorvldt.php';
  case SystemAttributes          = 'admin_system_attributes.php';
  case FacultiesListing          = 'admin_faculties_listing.php';
  case FacultiesSettings         = 'admin_faculties_settings.php';
  case FacultiesCreate           = 'admin_faculties_create.php';
  // Backend action pages
  case ActionCreateAdmin             = 'backend/admin_account_createadmin_action.php';
  case ActionManageAdmin             = 'backend/admin_account_manageadmin_action.php';
  case ActionDeleteAdmin             = 'backend/admin_account_deleteadmin_action.php';
  case ActionCreateProfessor         = 'backend/admin_account_createprofessor_action.php';
  case ActionManageProfessor         = 'backend/admin_account_manageprofessor_action.php';
  case ActionDeleteProfessor         = 'backend/admin_account_deleteprofessor_action.php';
  case ActionUploadProfessor         = 'backend/admin_account_uploadprofessor_action.php';
  case ActionSystemAttributes        = 'backend/admin_system_attributes_action.php';
  case ActionFacultiesSettings       = 'backend/admin_faculties_settings_action.php';
  case ActionFacultiesCreate         = 'backend/admin_faculties_create_action.php';
  case ActionFacultiesDelete         = 'backend/admin_faculties_delete_action.php';
  // Get the value of the enum name
  public static function getFromName(string $name): ?self {
    foreach (self::cases() as $case) {
      if ($case->name === $name) {
        return $case;
      }
    }
    return null;
  }

   /** @return array<string> */
  public function jsUsedFiles(): array {
    return match ($this) {
      self::AdminMain              => [],
      self::CreateAdmin            => [],
      self::ManageAdmin            => ['js/admin_account_manage.js', 'js/send_msg.js'],
      self::PendingActivation      => [],
      self::AdminsListing          => [],
      self::CreateProfessor        => [],
      self::ManageProfessor        => ['js/admin_account_manage.js', 'js/send_msg.js'],
      self::ProfessorsListing      => [],
      self::UploadProfessor        => ['js/admin_account_uploadprofessor.js'],
      self::UploadProfessorVldt    => ['js/admin_account_uploadprofessorvldt.js'],
      self::SystemAttributes       => [],
      self::FacultiesListing       => [],
      self::FacultiesSettings      => ['js/admin_faculties_settings.js'],
      self::FacultiesCreate        => [],
      // backend action pages do not use js files
    };
  }

  /** @return array<string, bool> */
  public function pageSettings(): array {
    return match ($this) {
      self::AdminMain                => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => false],
      self::CreateAdmin              => [
```

```
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::ManageAdmin               => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => false],
      self::PendingActivation         => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::AdminsListing             => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::CreateProfessor           => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::ManageProfessor           => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => false],
      self::ProfessorsListing         => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => false],
      self::UploadProfessor           => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::UploadProfessorVldt       =>
         ['show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::SystemAttributes          => [
        'show_sem' => false, 'select_semester' => false, 'only_superadmin' => true],
      self::FacultiesListing          => [
        'show_sem' => true,  'select_semester' => true,  'only_superadmin' => false],
      self::FacultiesSettings         => [
        'show_sem' => true,  'select_semester' => false, 'only_superadmin' => true],
      self::FacultiesCreate           => [
        'show_sem' => true,  'select_semester' => false, 'only_superadmin' => true],
      // backend action pages do not use settings
    };
  }

  /** @return array<string, string> */
  public function pageMeta(): array {
    return match ($this) {
      self::AdminMain               => [
        'title' => 'SMSYS', 'icon' => 'appoint_logo_alpha.png'
      ],
      self::CreateAdmin             => [
        'title' => 'Δημιουργία Λογαριασμού Διαχειριστή', 'icon' => 'icon-admins.png'
      ],
      self::ManageAdmin             => [
        'title' => 'Διαχείριση Λογαριασμού Διαχειριστή', 'icon' => 'icon-admins.png'
      ],
      self::PendingActivation       => [
        'title' => 'Εκκρεμείς Λογαριασμοί προς Ενεργοποίηση', 'icon' => 'icon-pending.png'
      ],
      self::AdminsListing           => [
        'title' => 'Λίστα Διαχειριστών', 'icon' => 'icon-admins.png'
      ],
      // backend action pages do not use settings
    };
  }
}
```

Πίνακας 4-6: Απόσπασμα κώδικα αρχείου Enum "AdminPage.php"

Αρχείο _common.php: Συχνά και κοινώς χρησιμοποιούμενες συναρτήσεις (παρουσιάζεται ενδεικτικό μέρος του κώδικα).

```php
<?php

/**
 * _common.php
 * Basic and globaly used functions and variables that are stand-alone
 * ### IMPORTANT: Page must contain only stand-alone functions that do not require
 * loading other files.
 *
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : All pages
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Enum\ValidateFieldName;
use SMSYS\Enum\Roles;

// phpcs:ignoreFile PSR1.Files.SideEffects.FoundWithSymbols
// Load html helper functions
require_once __DIR__ . '/_html.php';

/**
 * Set the current language of the page and other available languages.
 *
 * @return void
 */
function setLanguage(): void {
  $languages = ['gr', 'en'];
  $getlang = formFields()->sanitizeGet('lang', true, true, 10, null);
  if ((!isNullEmpty($getlang)) && (in_array($getlang, $languages))) {
    $_SESSION['smsys_language'] = $getlang;
  }
  if (!isset($_SESSION['smsys_language'])) {
    $_SESSION['smsys_language'] = 'gr';
  }
  if (isNullEmpty($_SESSION['smsys_language'])) {
    $_SESSION['smsys_language'] = 'gr';
  }
  // Set available languages
  $lang_key = array_search(getSessionStr('smsys_language'), $languages);
  if ($lang_key !== false) {
    unset($languages[$lang_key]);
  }
  $_SESSION['smsys_other_languages'] = $languages;
}

/**
 * Custom function for multibyte characters in utf-8 (Greek) to replace substr_replace.
```

```
 *
 * @param string $input     The input string to replace text within
 * @param string $replace   The text to replace with
 * @param int $pos          The position to start the replace
 * @param int $len          The position to end the replace
 * @return string           The input string with replaces values
 */
if (!function_exists('mb_substr_replace')) {
  function mb_substr_replace(string $input, string $replace, int $pos, int $len): string {
    $left  = mb_substr($input, 0, $pos, 'UTF-8');
    $right = mb_substr($input, ($pos + $len), mb_strlen($input), 'UTF-8');
    return $left . $replace . $right;
  }
}

/**
 * Custom function for multibyte characters in utf-8 (Greek) to trim left/right
 * white-space-like chars.
 *
 * @param string $input     The input string to trim
 * @param string $chars     The white-space-like chars to trim off
 * @return string           The input string trimmed
 */
if (!function_exists('mb_trim')) {
  function mb_trim(string $input, string $chars = " \t\n\r\0"): string {
    $input = (string) $input; // Case string for string types
    $remove = preg_quote($chars, '/'); // Escape the characters for the regex
    $output = preg_replace('/^[' . $remove . ']+|[' . $remove . ']+$/u', '', $input);
    return ($output !== null) ? $output : '';
  }
}

/**
 * Custom null/empty variable check.
 *
 * @param mixed $var  The input to check for null/empty
 * @phpstan-assert-if-false non-empty-string|int|float $var
 * @return bool       True if null or empty, false otherwise (and for numerci values)
 */
function isNullEmpty($var): bool {
  if ($var === null) {
    $res_bool = true;
  } elseif (is_string($var)) {
    $res_bool = (mb_trim($var) === '');
  } elseif ((is_int($var)) || (is_float($var))) {
    $res_bool = false;
  } else {
    $res_bool = empty($var);
  }
  return $res_bool;
}

/**
 * Base 64 encode for url usage
 *
 * @param string $str
```

```
 * @return string
 */
function base64url_encode(string $str): string {
  return rtrim(strtr(base64_encode($str), '+/', '-_'), '=');
}

/**
 * Base 64 decode for url usage
 *
 * @param string $str
 * @return string
 */
function base64url_decode(string $str): string {
  return base64_decode(strtr($str, '-_', '+/'));
}

/**
 * Logout user automatically and redirect to login page (session timeout,
 * invalid form security check).
 *
 * @SuppressWarnings("StaticAccess")
 * @SuppressWarnings("ExitExpression")
 * @SuppressWarnings("ErrorControlOperator")
 * @param string $msg   Friendly message to show after logout
 * @param string $role  If the call was made from specific user role/page
 */
function logout_user(string $msg, string $role = ''): void {
  // ### HERE WE MUST ADD A LOG TO WRITE THE LOGOUT DATE/TIME ###
  $url = 'login.php';
  if (
    in_array(Roles::tryFrom(getSessionStr('smsys_user_role')), Roles::adminRoles(), true)
  ) {
    $url = 'login.php';
  }
  if (in_array(Roles::tryFrom($role), Roles::adminRoles(), true)) {
    $url = 'login.php';
  }
  if ($role === NEWUSER) {
    $url = 'registration.php?mp=' . hash('sha256', CURRDATE . 'Registration');
  }
  $lang = getSessionStr('smsys_language');
  $_SESSION = [];
  if (session_status() === PHP_SESSION_ACTIVE) {
    session_destroy();
  }
  // Expire cookie on browser with same params (error safe)
  $params = session_get_cookie_params();
  $sess_name = session_name();
  if ($sess_name !== false) {
    /** @var array{
     *   lifetime: int<0, max>,
     *   path: non-falsy-string,
     *   domain: string,
     *   secure: bool,
     *   httponly: bool,
     *   samesite: 'Lax'|'lax'|'None'|'none'|'Strict'|'strict'
```

```
     * } $params
     */
    $params_path   = $params['path'];
    $params_domain = $params['domain'];
    $params_secure = ($params['secure'] === true);
    @setcookie(
      $sess_name,
      '',
      time() - 3600,
      $params_path,
      $params_domain,
      $params_secure,
      true
    );
  }
  // Session starter
  smsys_session_start();
  $_SESSION['smsys_message'] = $msg;
  $_SESSION['smsys_language'] = $lang;
  header('Location: ' . LOCAL_SERVER . $url);
  die('<a href="../index.php">Please click here to go to start page</a>');
}

/**
 * Redirect to another page with optional params and exit
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @SuppressWarnings("ExitExpression")
 * @param string $page    Page to redirect to
 * @param string $params  Parameters that we may need to pass
 */
function redirect(string $page, string $params = ''): void {
  $url = LOCAL_SERVER;
  if ($page === 'Refresh:0') {
    $url = '';
  }
  header('Location: ' . $url . $page);
  die('<a href="../index.php">Please click here to go to start page</a>');
}

// Session to string function, to avoid mixed types
function getSessionStr(string $sess_key, string $default = ''): string {
  $val = ((isset($_SESSION[$sess_key])) && (is_scalar($_SESSION[$sess_key])))
         ? strval($_SESSION[$sess_key])
         : $default;
  return $val;
}

/**
 * Check if message notifications session exists.
 *
 * @return bool
 */
function hasSessionMsg(): bool {
  return (!isNullEmpty(getSessionStr('smsys_message')));
}
```

```php
// Session to string function, for message notifications (keeps in session)
function getSessionMsgTmp(): string {
  $val = getSessionStr('smsys_message');
  return $val;
}
// Session to string function, for message notifications and clearing
function getSessionMsg(): string {
  $val = getSessionStr('smsys_message');
  $_SESSION['smsys_message'] = ''; // Clear display message after showing to user
  return $val;
}
// Session to int function, to avoid mixed types
function getSessionInt(string $sess_key, int $default = 0): int {
  $val = ((isset($_SESSION[$sess_key])) && (is_scalar($_SESSION[$sess_key])))
         ? intval('0' . (string) $_SESSION[$sess_key])
         : $default;
  return $val;
}
// Session to bool function, to avoid mixed types
function getSessionBool(string $sess_key): bool {
  $val = ((isset($_SESSION[$sess_key])) && (is_bool($_SESSION[$sess_key])))
         ? $_SESSION[$sess_key]
         : false;
  return $val;
}

/**
 * cURL handler function to manage commands globaly and avoid repeating code
 *
 * @param string $curl_url    URL for the curl command
 * @param string $post_data   Html query data to post with curl
 * @param string $cont_type   Content Type (if any)
 * @return array{response:bool|string|null, http_code:int, curl_error:string|null, curl_err
no:int}
 */
function curl_call(string $curl_url, string $post_data, string $cont_type = ''): array {
  $response = null;
  $http_code = 0;
  $curl_error = null;
  $curl_errno = 0;
  if (!isNullEmpty($curl_url)) {
    $curl = curl_init();
    if ($curl !== false) {
      curl_setopt($curl, CURLOPT_URL, $curl_url);
      curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, true);
      curl_setopt($curl, CURLOPT_CAINFO, __DIR__ . '/../pathtopem/cacert.pem');
      curl_setopt($curl, CURLOPT_NOPROGRESS, true);
      curl_setopt($curl, CURLOPT_TIMEOUT, 300);
      curl_setopt($curl, CURLOPT_POST, true);
      curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
      curl_setopt($curl, CURLOPT_POSTFIELDS, $post_data);
      if ($cont_type !== '') {
        curl_setopt($curl, CURLOPT_HTTPHEADER, [$cont_type]);
      }
      $response = curl_exec($curl);
      $http_code = curl_getinfo($curl, CURLINFO_HTTP_CODE);
```

```
      $curl_error = curl_error($curl);
      $curl_errno = curl_errno($curl);
      curl_close($curl);
    }
  }
  return ['response' => $response, 'http_code' => $http_code, 'curl_error' => $curl_error,
'curl_errno' => $curl_errno];
}

/**
 * Return a date/time to Greek local format
 *
 * @SuppressWarnings("StaticAccess")
 * @param string $datetime
 * @return string
 */
function formatDateTime(string $datetime): string {
  $datetime = DateTimeImmutable::createFromFormat('Y-m-d H:i:s', $datetime);
  if ($datetime !== false) {
    $datetime = $datetime->format('d/m/Y H:i:s');
  }
  return ($datetime === false) ? '' : $datetime;
}

/**
 * Fix date/time format between UI and Database
 *
 * @param scalar|null $input
 * @param string $format_from
 * @param string $format_to
 * @return string
 */
function fixDateTimeUiDb(?string $input, string $format_from, string $format_to): string {
  $result = '';
  if (!isNullEmpty($input)) {
    $date = DateTime::createFromFormat($format_from, $input);
    if ($date instanceof DateTime) {
      $result = $date->format($format_to);
    } else {
      // Return same if already correct format
      $date = DateTime::createFromFormat($format_to, $input);
      if ($date instanceof DateTime) {
        $result = $input;
      }
    }
  }
  return $result;
}
```

Πίνακας 4-7: Απόσπασμα κώδικα αρχείου "_common.php"

4.3 Σχεδίαση Βάσης Δεδομένων (Database Schema)

Η ενότητα αυτή περιγράφει τη σχεδίαση της Βάσης Δεδομένων της πλατφόρμας και το αντίστοιχο σχήμα της. Αναλύονται οι πίνακες, οι συσχετίσεις και οι βασικές σχεδιαστικές επιλογές, με στόχο την αποδοτική αποθήκευση, την ακεραιότητα των δεδομένων και την υποστήριξη των λειτουργικών απαιτήσεων του συστήματος.

4.3.1 Σχήμα Βάσης Δεδομένων

Στις παρακάτω εικόνες παρουσιάζονται ενδεικτικά αποσπάσματα του σχήματος της βάσης δεδομένων, όπως βασικοί πίνακες, ξένα κλειδιά, περιορισμοί και ευρετήρια, καθώς και ενδεικτικές οπτικές αναπαραστάσεις πινάκων και σχέσεων. Το πλήρες σχήμα παρατίθεται στο Παράρτημα Θ: «Σχήμα Βάσης Δεδομένων».

Δομή πινάκων καταγραφής (log)

tbl_account_log: Χρησιμοποιείται για την καταγραφή της τελευταίας ενημέρωσης μέσω email των νέων χρηστών, για την ενεργοποίηση του λογαριασμού τους και έλεγχο επαναποστολής ειδοποιήσεων (ατομικά ή μαζικά).

tbl_active_log: Χρησιμοποιείται για την καταγραφή διαφόρων ενεργειών των χρηστών όπως: η ενεργοποίηση λογαριασμού, η μεταφόρτωση αρχείων, η αποστολή μηνυμάτων και γενικότερα ενέργειες προς εξωτερικά συστήματα ή API.

tbl_change_log: Αποθηκεύονται οι αλλαγές σε δεδομένα της Βάσης Δεδομένων, με αναφορά στον πίνακα (όνομα πίνακα), το είδος αλλαγής (εισαγωγή, τροποποίηση, διαγραφή), την ημερομηνία και ώρα, τον χρήστη και τα δεδομένα που τροποποιήθηκαν σε μορφή JSON (παλαιά και νέα τιμή, με κρυπτογράφηση των ευαίσθητων δεδομένων.

tbl_error_log: Ο πίνακας καταγραφής όλων των σφαλμάτων, με σύντομη και πλήρη περιγραφή του σφάλματος, της ημερομηνίας και ώρα, το μοναδικό id σφάλματος για εντοπισμό στον κώδικα, τη σελίδα που εντοπίστηκε το σφάλμα, το id χρήστη (εάν υπάρχει) και το φιλικό μήνυμα που εμφανίστηκε στον χρήστη.

tbl_login_log: Πίνακας καταγραφής γεγονότων αυθεντικοποίησης χρηστών. Καταγράφει όλες τις ενέργειες που σχετίζονται με τη διαδικασία εισόδου και εξόδου, συμπεριλαμβανομένων του αναγνωριστικού χρήστη ή του ονόματος χρήστη (σε περιπτώσεις

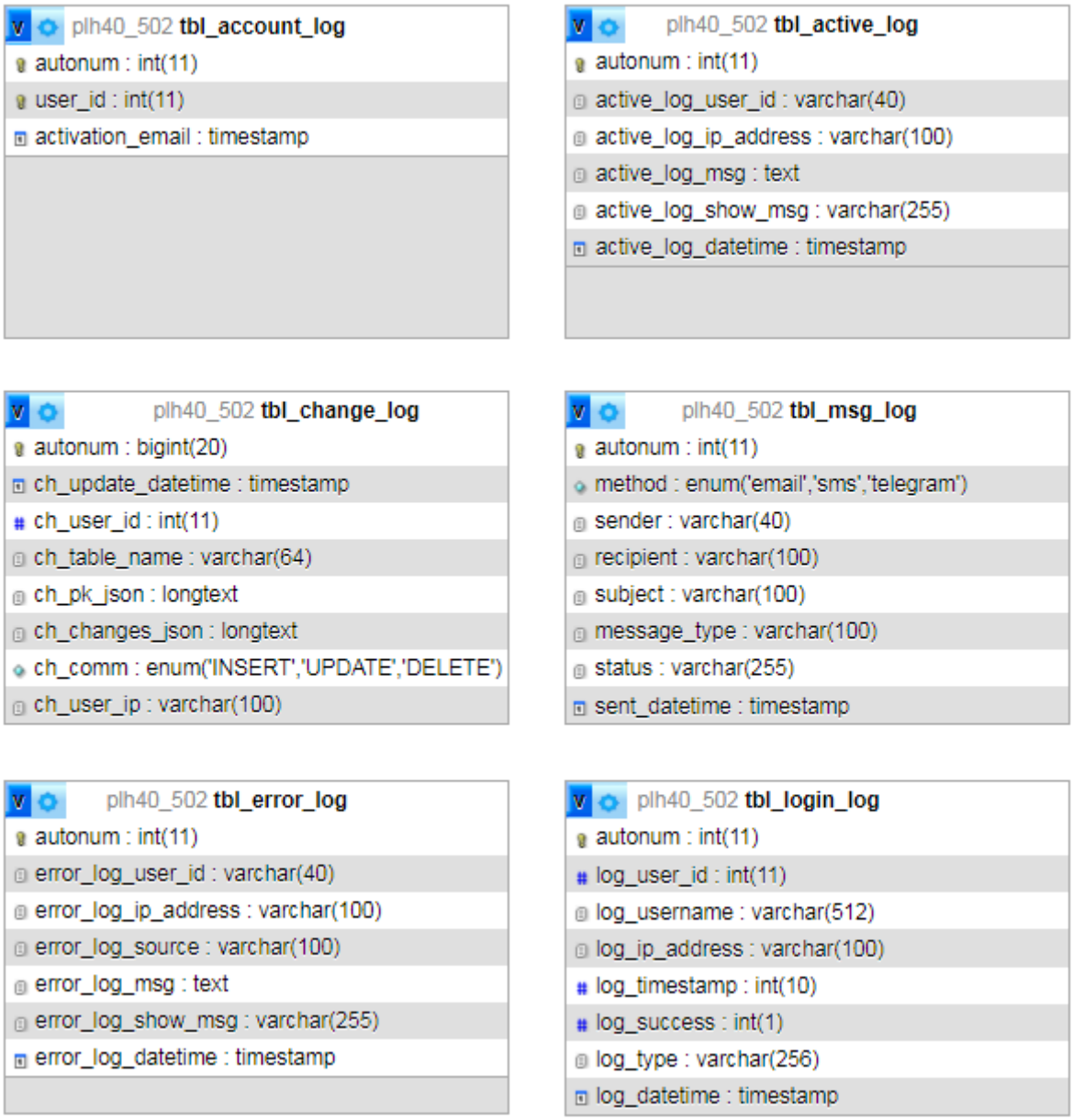
αποτυχημένης εισόδου), του τύπου συμβάντος (επιτυχής ή αποτυχημένη είσοδος, αποσύνδεση χρήστη, λήξη συνεδρίας, επιτυχής ή αποτυχημένη επαλήθευση δεύτερου παράγοντα 2FA, αίτημα επαναφοράς κωδικού), της ημερομηνίας και ώρας και της διεύθυνσης IP. Ο πίνακας αξιοποιείται επιπλέον από το σύστημα για την ανίχνευση επαναλαμβανόμενων ή χρονικά σύντομων αποτυχημένων προσπαθειών, με στόχο την πρόληψη επιθέσεων και την ενίσχυση της ασφάλειας..

tbl_msg_log: Πίνακας καταγραφής απεσταλμένων μηνυμάτων από τους διαχειριστές και το σύστημα. Περιλαμβάνει στοιχεία όπως: μέθοδος αποστολής (email, sms, Telegram), τρόπο αποστολής (αυτοματοποιημένο ή id διαχειριστή), το email ή το τηλέφωνο παραλήπτη (με μάσκα απόκρυψης), το κείμενο του μηνύματος (με μάσκα απόκρυψης ευαίσθητων δεδομένων), το είδος/λόγο αποστολής (ενεργοποίηση λογαριασμού, 2FA, κ.ά.), την κατάσταση αποστολής (επιτυχία/αποτυχία αποστολής), την ημερομηνία και ώρα αποστολής.

```
-- Table structure for table `tbl_account_log`
CREATE TABLE IF NOT EXISTS `tbl_account_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `activation_email` timestamp NULL DEFAULT NULL ON UPDATE current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_user_id` (`user_id`),
  CONSTRAINT `fk_user_id_log` FOREIGN KEY (`user_id`) REFERENCES `tbl_login_user`
(`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=148 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_active_log`
CREATE TABLE IF NOT EXISTS `tbl_active_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `active_log_user_id` varchar(40) NOT NULL,
  `active_log_ip_address` varchar(100) NOT NULL,
  `active_log_msg` text DEFAULT NULL,
  `active_log_show_msg` varchar(255) DEFAULT NULL,
  `active_log_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`)
) ENGINE=InnoDB AUTO_INCREMENT=1427 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_change_log`
CREATE TABLE IF NOT EXISTS `tbl_change_log` (
  `autonum` bigint(20) NOT NULL AUTO_INCREMENT,
  `ch_update_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `ch_user_id` int(11) NOT NULL,
  `ch_table_name` varchar(64) NOT NULL,
  `ch_pk_json` longtext CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL,
  `ch_changes_json` longtext CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL,
  `ch_comm` enum('INSERT','UPDATE','DELETE') NOT NULL DEFAULT 'UPDATE',
```

```
  `ch_user_ip` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`autonum`)
) ENGINE=InnoDB AUTO_INCREMENT=6522 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_error_log`
CREATE TABLE IF NOT EXISTS `tbl_error_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `error_log_user_id` varchar(40) NOT NULL,
  `error_log_ip_address` varchar(100) NOT NULL,
  `error_log_source` varchar(100) DEFAULT NULL,
  `error_log_msg` text DEFAULT NULL,
  `error_log_show_msg` varchar(255) DEFAULT NULL,
  `error_log_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`)
) ENGINE=InnoDB AUTO_INCREMENT=1777 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_login_log`
CREATE TABLE IF NOT EXISTS `tbl_login_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `log_user_id` int(11) NOT NULL DEFAULT 0,
  `log_username` varchar(512) DEFAULT NULL,
  `log_username_hash` varchar(255) DEFAULT NULL,
  `log_ip_address` varchar(100) DEFAULT NULL,
  `log_ip_address_hash` varchar(255) DEFAULT NULL,
  `log_timestamp` int(10) NOT NULL DEFAULT 0,
  `log_success` int(1) NOT NULL DEFAULT 0,
  `log_type` varchar(256) DEFAULT NULL,
  `log_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`),
  KEY `idx_log_success` (`log_success`),
  KEY `idx_log_timestamp` (`log_timestamp`),
  KEY `idx_log_username` (`log_username`) USING HASH,
  KEY `idx_log_ip_address` (`log_ip_address`) USING HASH,
  KEY `idx_username_hash` (`log_username_hash`) USING BTREE,
  KEY `idx_ip_address_hash` (`log_ip_address_hash`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=5803 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_msg_log`
CREATE TABLE IF NOT EXISTS `tbl_msg_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `method` enum('email','sms','telegram') NOT NULL,
  `sender` varchar(40) NOT NULL,
  `recipient` varchar(100) NOT NULL,
  `subject` varchar(100) DEFAULT NULL,
  `message_type` varchar(100) DEFAULT NULL,
  `status` varchar(255) DEFAULT NULL,
  `sent_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`)
) ENGINE=InnoDB AUTO_INCREMENT=155 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

Πίνακας 4-8: Απόσπασμα SQL ορισμού πινάκων καταγραφής (log)



Εικόνα 1 – Οπτική αναπαράσταση πινάκων καταγραφής (log)

Δομή βοηθητικών πινάκων

tbl_unique_id: Καταγράφεται η επόμενη αρίθμηση για τη δημιουργία των πεδίων-κλειδιών (ids), στου πίνακες που δεν χρησιμοποιείται αυτόματη αρίθμηση. Εκτός του ελέγχου που επιτυγχάνεται εσωτερικά σε συσχετίσεις και ξένα κλειδιά, αποφεύγονται κενά στις αριθμήσεις σε αποτυχημένες προσπάθειες δημιουργίας νέων εγγραφών.

tbl_attributes: Περιλαμβάνει τα παραμετροποιήσιμη στοιχεία των βασικών ρυθμίσεων του συστήματος, όπως τίτλος, ονόματα λογότυπων, ημερομηνίες προβολής και επεξεργασίας

προγραμμάτων, κ.ά. Τα δεδομένα αποθηκεύονται και χρησιμοποιούνται με το σύστημα cache όπως αναλύθηκε στην υποενότητα 4.1.6, για αποφυγή πολλαπλών κλήσεων προς τη ΒΔ.

tbl_file: Στον πίνακα αποθηκεύονται τα δεδομένα της μεταφόρτωσης από τους χρήστες, αρχείων προς αποθήκευση στο σύστημα (εικόνες, pdf, κ.ά.). Περιλαμβάνει στοιχεία όπως: το hashed όνομα του αρχείου όπως αποθηκεύτηκε στο σύστημα, για απόκρυψη και ασφάλεια, σύμφωνα με την υποενότητα 4.1.4, το αρχικό όνομα αρχείου, τον τύπου αρχείου, την ημερομηνία και ώρα μεταφόρτωσης, το αναγνωριστικό χρήστη που πραγματοποίησε τη μεταφόρτωση και την κατάσταση (προσωρινή ή οριστική), με αυτόματη διαγραφή παλαιών προσωρινών μεταφορτώσεων.

tbl_login_trusted_device: Χρησιμοποιείται για τον έλεγχο και την καταγραφή εισόδου χρηστών από νέα συσκευή και την απαίτηση έλεγχου ταυτότητας δύο παραγόντων. Περιέχει μοναδικό hash token αναγνώρισης της συσκευής.

tbl_temp_hash_log: Στον πίνακα αποθηκεύονται προσωρινά τα μοναδικά κλειδιά (hash tokens) για ενέργειες όπως επαναφοράς κωδικού πρόσβασης, ενεργοποίησης λογαριασμού, κ.ο.κ.), νε ημερομηνία λήξης και αυτοματοποιημένη διαγραφή παλαιών εγγραφών.

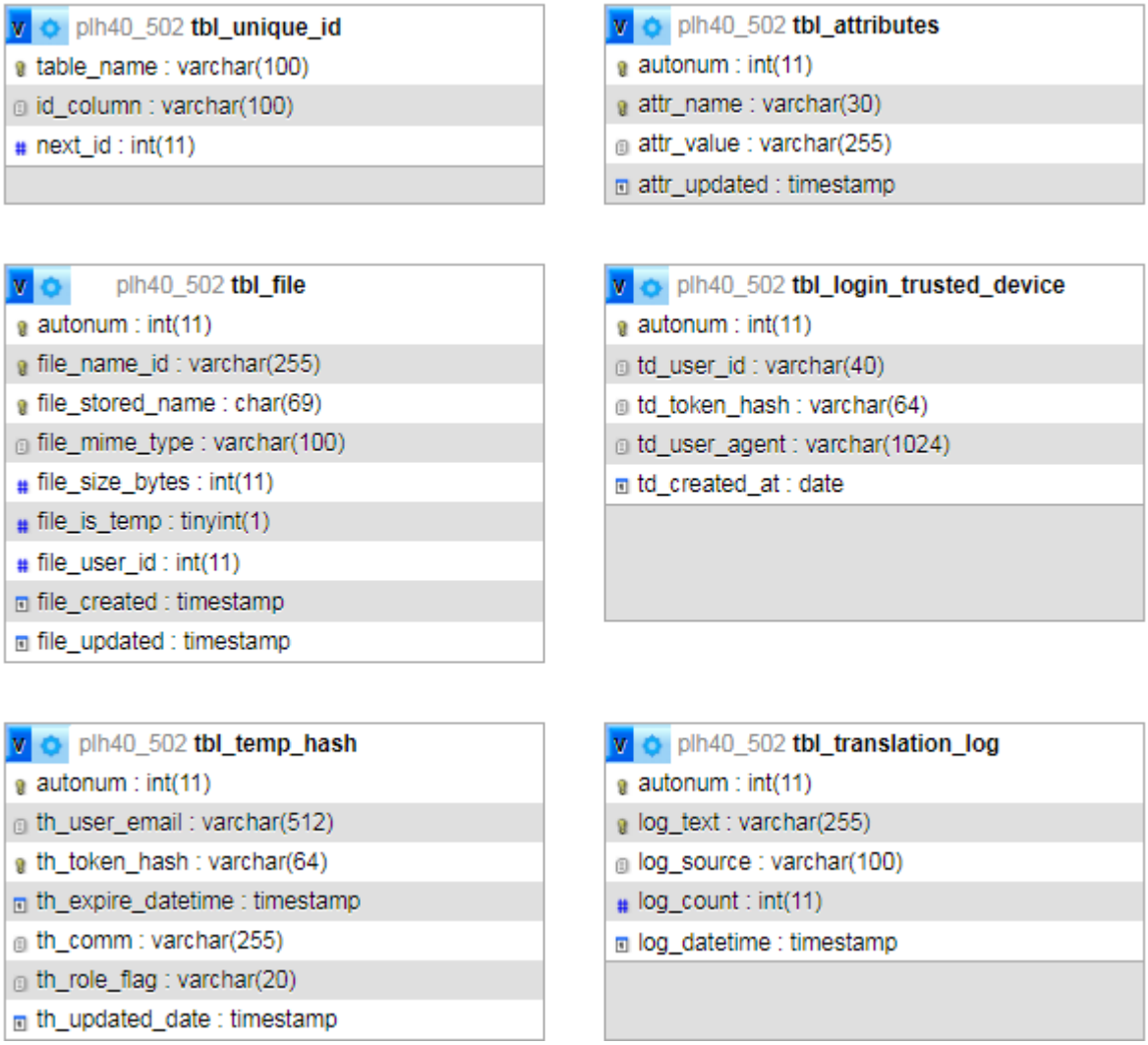
tbl_translation_log: Πίνακας καταγραφής λέξεων και φράσεων για τις οποίες δεν βρέθηκε κείμενο μετάφρασης από ελληνικά σε αγγλικά, με ένδειξη σελίδας εντοπισμού, για τον έλεγχο της πολυγλωσσικής λειτουργίας. Οι λέξεις ή φράσεις αποθηκεύονται μόνο μία φορά όταν εντοπίζονται για αποφυγή άσκοπου φόρτου εργασίας και μεγέθους του πίνακα.

```
-- Table structure for table `tbl_unique_id`
CREATE TABLE IF NOT EXISTS `tbl_unique_id` (
  `table_name` varchar(100) NOT NULL,
  `id_column` varchar(100) NOT NULL,
  `next_id` int(11) NOT NULL,
  PRIMARY KEY (`table_name`) USING BTREE,
  KEY `idx_next_id` (`next_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_attributes`
CREATE TABLE IF NOT EXISTS `tbl_attributes` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `attr_name` varchar(30) NOT NULL,
  `attr_value` varchar(255) DEFAULT NULL,
  `attr_updated` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_attr_name` (`attr_name`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

```
-- Table structure for table `tbl_file`
CREATE TABLE IF NOT EXISTS `tbl_file` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `file_name_id` varchar(255) NOT NULL,
  `file_stored_name` char(69) NOT NULL,
  `file_mime_type` varchar(100) NOT NULL,
  `file_size_bytes` int(11) NOT NULL,
  `file_is_temp` tinyint(1) NOT NULL DEFAULT 0,
  `file_user_id` int(11) NOT NULL,
  `file_created` timestamp NOT NULL DEFAULT current_timestamp(),
  `file_updated` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_file_stored_name` (`file_stored_name`),
  UNIQUE KEY `unique_file_name_id` (`file_name_id`)
) ENGINE=InnoDB AUTO_INCREMENT=32 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_login_trusted_device`
CREATE TABLE IF NOT EXISTS `tbl_login_trusted_device` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `td_user_id` varchar(40) NOT NULL,
  `td_token_hash` varchar(64) NOT NULL,
  `td_user_agent` varchar(1024) DEFAULT NULL,
  `td_created_at` date NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`),
  KEY `idx_td_user_id` (`td_user_id`),
  KEY `idx_td_token_hash` (`td_token_hash`) USING HASH
) ENGINE=InnoDB AUTO_INCREMENT=67 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_temp_hash`
CREATE TABLE IF NOT EXISTS `tbl_temp_hash` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `th_user_email` varchar(512) NOT NULL,
  `th_token_hash` varchar(64) NOT NULL,
  `th_expire_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `th_comm` varchar(255) NOT NULL,
  `th_role_flag` varchar(20) DEFAULT NULL,
  `th_updated_date` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `idx_th_token_hash` (`th_token_hash`) USING BTREE,
  KEY `idx_th_user_id` (`th_user_email`)
) ENGINE=InnoDB AUTO_INCREMENT=87 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_translation_log`
CREATE TABLE IF NOT EXISTS `tbl_translation_log` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `log_text` varchar(255) NOT NULL,
  `log_source` varchar(100) DEFAULT NULL,
  `log_count` int(11) NOT NULL DEFAULT 1,
  `log_datetime` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
```

```
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_log_text` (`log_text`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=1179 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

Πίνακας 4-9: Απόσπασμα SQL ορισμού βοηθητικών πινάκων



Εικόνα 2 – Οπτική αναπαράσταση βοηθητικών πινάκων

Δομή πινάκων κύριων οντοτήτων χρηστών

tbl_login_user: Ο κεντρικός πίνακας διαχείρισης χρηστών του συστήματος. Περιλαμβάνει το μοναδικό αναγνωριστικό χρήστη (user_id), τα διαπιστευτήρια αυθεντικοποίησης (username και password), τα οποία αποθηκεύονται σε κρυπτογραφημένη μορφή σύμφωνα με την ανάλυση ασφάλειας της υποενότητας 4.1.4, τη διεύθυνση ηλεκτρονικού ταχυδρομείου

του χρήστη για την υποστήριξη διαδικασιών επαναφοράς κωδικού, καθώς και την ημερομηνία και ώρα τελευταίας εισόδου. Επιπλέον, περιέχει τον ρόλο του χρήστη (ως πεδίο τύπου enum για την αποτροπή μη έγκυρων τιμών και τη μελλοντική επεκτασιμότητα), το αναγνωριστικό συνεδρίας (session_id) για τον έλεγχο ταυτόχρονων συνδέσεων και την κατάσταση του λογαριασμού (ενεργός, ανενεργός, σε εκκρεμότητα), βάσει της οποίας επιτρέπεται ή απορρίπτεται η πρόσβαση στο σύστημα.

tbl_admin / tbl_professor / tbl_student: Περιλαμβάνουν τα αναλυτικά στοιχεία των χρηστών του συστήματος, σε αντιστοίχηση με τον πίνακα «*tbl_login_user*». Για κάθε χρήστη αποθηκεύονται: το μοναδικό αναγνωριστικό χρήστη (user_id) ως ξένο κλειδί του πίνακα στον πίνακα «*tbl_login_user*», τα στοιχεία ταυτοποίησης (όνομα, επώνυμο, πατρώνυμο), τα στοιχεία επικοινωνίας (σταθερό και κινητό τηλέφωνο), καθώς και ο μοναδικός πανεπιστημιακός κωδικός (uni_id) που χρησιμοποιείται από το Ίδρυμα. Επιπλέον, καταγράφεται το Τμήμα στο οποίο υπάγεται, μέσω ξένου κλειδιού προς τον αντίστοιχο πίνακα. Οι πίνακες περιλαμβάνουν επίσης χρονικές σφραγίδες δημιουργίας και τελευταίας ενημέρωσης της εγγραφής, οι οποίες επιτρέπουν την ιχνηλάτηση μεταβολών στα δεδομένα.

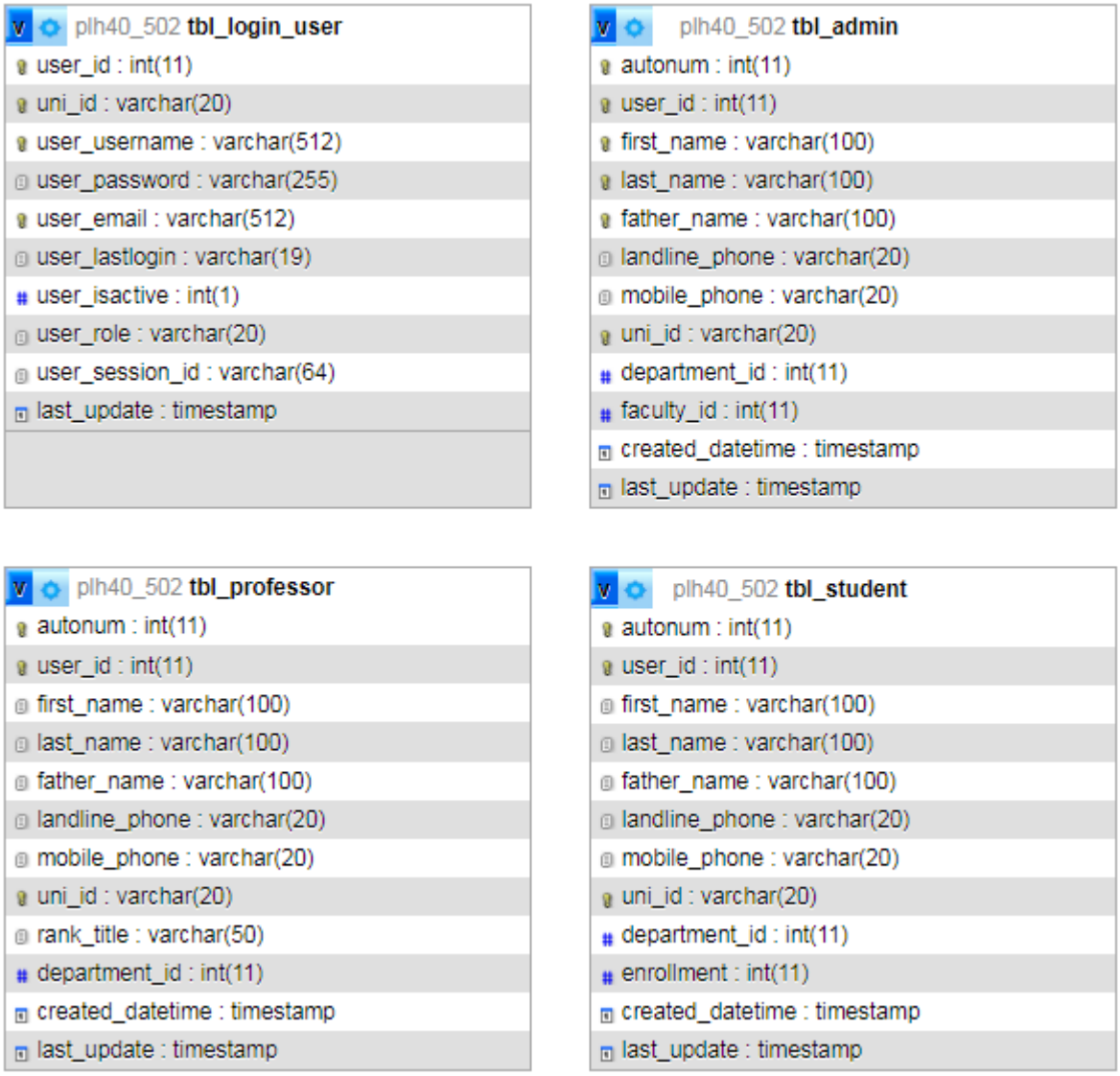
tbl_professor_rank: Πίνακας αναφοράς των ακαδημαϊκών βαθμίδων των διδασκόντων. Περιλαμβάνει τον τίτλο βαθμίδας (rank_title) ως πρωτεύον κλειδί και τον τύπο ή κατηγορία της βαθμίδας (rank_type). Ο πίνακας χρησιμοποιείται για την τυποποιημένη και ελεγχόμενη καταχώρηση των βαθμίδων στον πίνακα «*tbl_professor*», εξασφαλίζοντας ομοιομορφία και επεκτασιμότητα των σχετικών δεδομένων.

```
-- Table structure for table `tbl_login_user`
CREATE TABLE IF NOT EXISTS `tbl_login_user` (
  `user_id` int(11) NOT NULL AUTO_INCREMENT,
  `uni_id` varchar(20) NOT NULL,
  `user_username` varchar(512) NOT NULL,
  `user_password` varchar(255) NOT NULL,
  `user_email` varchar(512) NOT NULL,
  `user_lastlogin` datetime DEFAULT NULL,
  `user_isactive` int(1) NOT NULL DEFAULT 1,
  `user_role` varchar(20) NOT NULL COMMENT 'student, professor, admin, superadmin',
  `user_session_id` varchar(64) DEFAULT NULL,
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  `password_set` tinyint(1) NOT NULL DEFAULT 0,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `unique_login_user__uni_id` (`uni_id`),
  UNIQUE KEY `unique_login_user__user_username` (`user_username`),
  UNIQUE KEY `unique_login_user__user_email` (`user_email`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=264 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_admin`
CREATE TABLE IF NOT EXISTS `tbl_admin` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `first_name` varchar(100) DEFAULT NULL,
  `last_name` varchar(100) DEFAULT NULL,
  `father_name` varchar(100) DEFAULT NULL,
  `landline_phone` varchar(20) DEFAULT NULL,
  `mobile_phone` varchar(20) DEFAULT NULL,
  `uni_id` varchar(20) NOT NULL,
  `department_id` int(11) DEFAULT NULL,
  `faculty_id` int(11) DEFAULT NULL,
  `created_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_admin__user_id` (`user_id`),
  UNIQUE KEY `unique_admin__uni_id` (`uni_id`),
  UNIQUE KEY `unique_admin__full_name` (`first_name`,`last_name`,`father_name`),
  KEY `fk_admin__department_id` (`department_id`),
  KEY `fk_admin__faculty_id` (`faculty_id`),
  CONSTRAINT `fk_admin_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`) ON DELETE SET NULL,
  CONSTRAINT `fk_admin_user_id` FOREIGN KEY (`user_id`) REFERENCES `tbl_login_user`
(`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=72 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_professor`
CREATE TABLE IF NOT EXISTS `tbl_professor` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `first_name` varchar(100) DEFAULT NULL,
  `last_name` varchar(100) DEFAULT NULL,
  `father_name` varchar(100) DEFAULT NULL,
  `landline_phone` varchar(20) DEFAULT NULL,
  `mobile_phone` varchar(20) DEFAULT NULL,
  `uni_id` varchar(20) NOT NULL,
  `rank_title` varchar(50) DEFAULT NULL,
  `department_id` int(11) NOT NULL,
  `created_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_professor__user_id` (`user_id`),
  UNIQUE KEY `unique_professor__uni_id` (`uni_id`),
  KEY `unique_professor__full_name` (`first_name`,`last_name`,`father_name`),
  KEY `fk_professor_department_id` (`department_id`),
  KEY `fk_professor_rank_title` (`rank_title`),
  CONSTRAINT `fk_professor_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`),
```

```
  CONSTRAINT `fk_professor_rank_title` FOREIGN KEY (`rank_title`) REFERENCES
`tbl_professor_rank` (`rank_title`),
  CONSTRAINT `fk_professor_user_id` FOREIGN KEY (`user_id`) REFERENCES
`tbl_login_user` (`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=159 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_student`
CREATE TABLE IF NOT EXISTS `tbl_student` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `first_name` varchar(100) DEFAULT NULL,
  `last_name` varchar(100) DEFAULT NULL,
  `father_name` varchar(100) DEFAULT NULL,
  `landline_phone` varchar(20) DEFAULT NULL,
  `mobile_phone` varchar(20) DEFAULT NULL,
  `uni_id` varchar(20) NOT NULL,
  `birth_date` date DEFAULT NULL,
  `post_address` varchar(255) DEFAULT NULL,
  `post_city` varchar(50) DEFAULT NULL,
  `post_zip` varchar(5) DEFAULT NULL,
  `enrollment` int(11) NOT NULL,
  `reserved_field` varchar(1) DEFAULT NULL,
  `department_id` int(11) DEFAULT NULL,
  `curr_semester` int(11) DEFAULT NULL,
  `user_special` int(11) DEFAULT NULL,
  `registration_id` int(11) DEFAULT NULL,
  `registration_date` date DEFAULT NULL,
  `created_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_student__user_id` (`user_id`),
  UNIQUE KEY `unique_student__uni_id` (`uni_id`),
  UNIQUE KEY `unique_student__full_name_dep`
(`first_name`,`last_name`,`father_name`,`department_id`),
  KEY `fk_student__department_id` (`department_id`),
  KEY `fk_students_registration_id` (`registration_id`),
  KEY `fk_student_specializaion` (`user_special`) USING BTREE,
  CONSTRAINT `fk_student__department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`),
  CONSTRAINT `fk_student_registration_id` FOREIGN KEY (`registration_id`)
REFERENCES `tbl_registration` (`reg_id`),
  CONSTRAINT `fk_student_specializaion` FOREIGN KEY (`user_special`) REFERENCES
`tbl_specialization` (`spe_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_student_user_id` FOREIGN KEY (`user_id`) REFERENCES
`tbl_login_user` (`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

Πίνακας 4-10: Απόσπασμα SQL ορισμού πινάκων κύριων οντοτήτων χρηστών



Εικόνα 3 – Οπτική αναπαράσταση πινάκων κύριων οντοτήτων χρηστών

Δομή πινάκων κύριων οντοτήτων ακαδημαϊκής οργάνωσης

tbl_faculty: Αποτελεί τον βασικό (σταθερό) πίνακα αναφοράς των Σχολών του Ιδρύματος. Περιλαμβάνει το εσωτερικό αναγνωριστικό της σχολής (fac_id) ως πρωτεύον κλειδί και τον μοναδικό κωδικό σχολής που χρησιμοποιείται από το Πανεπιστήμιο (fac_uni_id), ο οποίος επιβάλλεται ως μοναδικός μέσω αντίστοιχου ευρετηρίου.

tbl_faculty_version: Πίνακας εκδόσεων (versioning) των στοιχείων Σχολής, ο οποίος επιτρέπει τη διατήρηση ιστορικότητας και την ισχύ δεδομένων ανά ακαδημαϊκή περίοδο. Για κάθε σχολή (fac_id) αποθηκεύονται οι περιγραφικές πληροφορίες (faculty_name, faculty_url), ένα πεδίο μοναδικής ταυτοποίησης/κανονικοποίησης (faculty_unique) για

αποφυγή διπλοεγγραφών, καθώς και ένδειξη κατάστασης (is_active). Η εγκυρότητα κάθε έκδοσης ορίζεται από το έτος και το εξάμηνο έναρξης ισχύος (valid_from_year, valid_from_semester), ενώ οι χρονικές σφραγίδες version_datetime και last_update υποστηρίζουν την ιχνηλάτηση δημιουργίας και μεταβολών. Τα σύνθετα μοναδικά ευρετήρια διασφαλίζουν ότι: α) δεν μπορεί να υπάρξει περισσότερη από μία έκδοση ανά σχολή και ακαδημαϊκή περίοδο και β) η ονομασία σχολής (μέσω του faculty_unique) παραμένει μοναδική ανά περίοδο. Τέλος, το ξένο κλειδί προς τον πίνακα «*tbl_faculty*» διασφαλίζει την ακεραιότητα των αναφορών μεταξύ σταθερών και δεδομένων εκδόσεων.

tbl_department: Βασικός πίνακας αναφοράς των Τμημάτων του Ιδρύματος. Περιλαμβάνει το εσωτερικό αναγνωριστικό (dep_id) και τον μοναδικό πανεπιστημιακό κωδικό τμήματος (dep_uni_id), καθώς και τη χρονική σήμανση δημιουργίας.

tbl_ department _version: Πίνακας εκδόσεων των στοιχείων Τμήματος, που επιτρέπει τη διαχείριση ιστορικότητας και ισχύος ανά ακαδημαϊκή περίοδο. Για κάθε τμήμα αποθηκεύονται το όνομα, το μοναδικό αναγνωριστικό (department_unique), η διεύθυνση ιστοσελίδας, η Σχολή στην οποία ανήκει, η διάρκεια σπουδών, καθώς και η κατάσταση. Η εγκυρότητα κάθε έκδοσης ορίζεται από το έτος και το εξάμηνο (valid_from_year, valid_from_semester), ενώ τα σύνθετα μοναδικά ευρετήρια και τα ξένα κλειδιά διασφαλίζουν τη μοναδικότητα και την ακεραιότητα των δεδομένων.

tbl_course: Βασικός πίνακας αναφοράς των μαθημάτων. Περιλαμβάνει το εσωτερικό αναγνωριστικό μαθήματος (cou_id), τον πανεπιστημιακό κωδικό (cou_uni_id) και το τμήμα στο οποίο ανήκει το μάθημα, διασφαλίζοντας μοναδικότητα ανά τμήμα και κωδικό.

tbl_course_version: Πίνακας εκδόσεων μαθημάτων, που επιτρέπει τη διαχείριση ιστορικότητας και ισχύος ανά ακαδημαϊκή περίοδο. Για κάθε μάθημα αποθηκεύονται το όνομα, το μοναδικό αναγνωριστικό (course_unique), η κατηγορία, οι πιστωτικές μονάδες (ECTS), ο εκτιμώμενος αριθμός φοιτητών, η δυνατότητα επικάλυψης, η ειδίκευση, καθώς και η κατάσταση ενεργότητας. Η εγκυρότητα ορίζεται από το έτος και το εξάμηνο, ενώ τα σύνθετα μοναδικά ευρετήρια και τα ξένα κλειδιά διασφαλίζουν τη συνέπεια των δεδομένων.

tbl_course_classroom: Πίνακας αντιστοίχισης μαθημάτων με αίθουσες ανά τύπο διδασκαλίας (θεωρία, εργαστήριο κ.λπ.), επιτρέποντας την καταγραφή των επιτρεπόμενων ή διαθέσιμων αιθουσών για κάθε έκδοση μαθήματος.

tbl_course_professor: Πίνακας συσχέτισης πολλών-προς-πολλά μεταξύ μαθημάτων και διδασκόντων, ο οποίος καταγράφει ποιοι καθηγητές διδάσκουν κάθε έκδοση μαθήματος.

tbl_course_semester: Πίνακας κατανομής μαθημάτων ανά εξάμηνο σπουδών, επιτρέποντας την αντιστοίχιση κάθε έκδοσης μαθήματος σε ένα ή περισσότερα εξάμηνα.

tbl_course_type: Πίνακας ορισμού των τύπων διδασκαλίας ενός μαθήματος (π.χ. θεωρία, εργαστήριο) και των αντίστοιχων ωρών διδασκαλίας, ανά έκδοση μαθήματος.

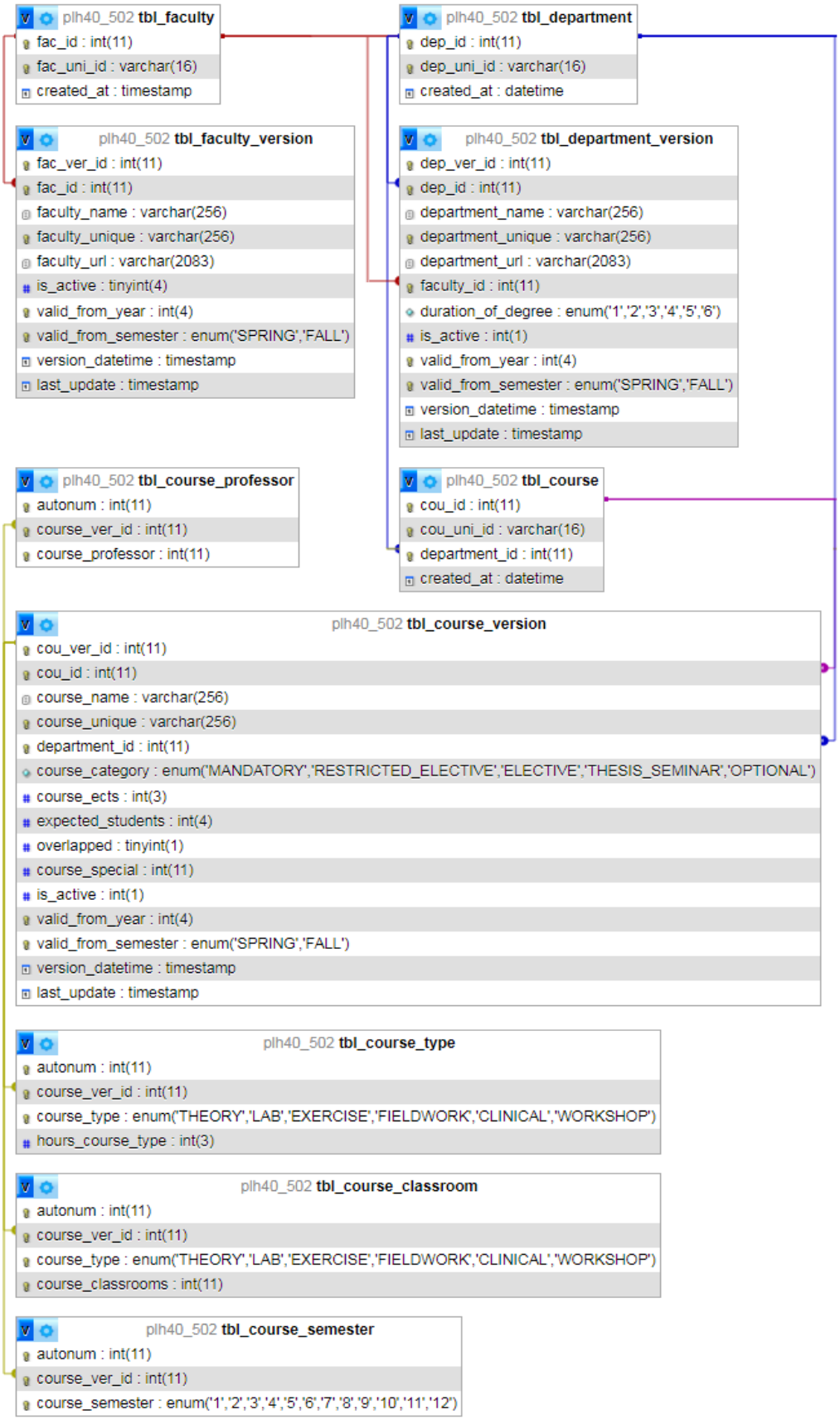
```
-- Table structure for table `tbl_faculty`
CREATE TABLE IF NOT EXISTS `tbl_faculty` (
  `fac_id` int(11) NOT NULL,
  `fac_uni_id` varchar(16) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`fac_id`),
  UNIQUE KEY `unique_faculty__fac_uni_id` (`fac_uni_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_faculty_version`
CREATE TABLE IF NOT EXISTS `tbl_faculty_version` (
  `fac_ver_id` int(11) NOT NULL,
  `fac_id` int(11) NOT NULL,
  `faculty_name` varchar(256) NOT NULL,
  `faculty_unique` varchar(256) NOT NULL,
  `faculty_url` varchar(2083) DEFAULT NULL,
  `is_active` tinyint(4) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL DEFAULT 'SPRING',
  `version_datetime` timestamp NOT NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`fac_ver_id`),
  UNIQUE KEY `unique_faculty_version__fac_id_year_semester`
(`fac_id`,`valid_from_year`,`valid_from_semester`),
  UNIQUE KEY `unique_faculty_version__unique_year_semester`
(`faculty_unique`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_faculty_name` (`faculty_name`),
  CONSTRAINT `fk_fac_id` FOREIGN KEY (`fac_id`) REFERENCES `tbl_faculty` (`fac_id`)
ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_department`
CREATE TABLE IF NOT EXISTS `tbl_department` (
  `dep_id` int(11) NOT NULL,
  `dep_uni_id` varchar(16) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`dep_id`),
  UNIQUE KEY `unique_department__dep_uni_id` (`dep_uni_id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_department_version`
CREATE TABLE IF NOT EXISTS `tbl_department_version` (
  `dep_ver_id` int(11) NOT NULL,
  `dep_id` int(11) NOT NULL,
  `department_name` varchar(256) NOT NULL,
  `department_unique` varchar(256) NOT NULL,
  `department_url` varchar(2083) DEFAULT NULL,
  `faculty_id` int(11) NOT NULL,
  `duration_of_degree` enum('1','2','3','4','5','6') NOT NULL,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`dep_ver_id`),
  UNIQUE KEY `unique_department_version__dep_id_faculty_year_semester`
(`dep_id`,`faculty_id`,`valid_from_year`,`valid_from_semester`),
  UNIQUE KEY `unique_department_version__unique_faculty_year_semester`
(`department_unique`,`faculty_id`,`valid_from_year`,`valid_from_semester`) USING
BTREE,
  KEY `idx_faculty_id` (`faculty_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_department_name` (`department_name`),
  CONSTRAINT `fk_dep_id` FOREIGN KEY (`dep_id`) REFERENCES `tbl_department`
(`dep_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_faculty_id` FOREIGN KEY (`faculty_id`) REFERENCES `tbl_faculty`
(`fac_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course`
CREATE TABLE IF NOT EXISTS `tbl_course` (
  `cou_id` int(11) NOT NULL,
  `cou_uni_id` varchar(16) NOT NULL,
  `department_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`cou_id`),
  UNIQUE KEY `unique_course__cou_uni_id_department_id`
(`cou_uni_id`,`department_id`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  CONSTRAINT `fk_course_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_version`
CREATE TABLE IF NOT EXISTS `tbl_course_version` (
  `cou_ver_id` int(11) NOT NULL,
  `cou_id` int(11) NOT NULL,
  `course_name` varchar(256) NOT NULL,
  `course_unique` varchar(256) NOT NULL,
```

```
  `department_id` int(11) NOT NULL,
  `course_category`
enum('MANDATORY','RESTRICTED_ELECTIVE','ELECTIVE','THESIS_SEMINAR','OPTIONAL') NOT
NULL,
  `course_mode` enum('IN_PERSON','HYBRID','ONLINE','') NOT NULL DEFAULT
'IN_PERSON',
  `course_ects` int(3) NOT NULL DEFAULT 0,
  `expected_students` int(4) NOT NULL DEFAULT 0,
  `overlapped` tinyint(1) NOT NULL DEFAULT 0 COMMENT '0: No overlap, 1: Can be
overlapped',
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`cou_ver_id`),
  UNIQUE KEY `unique_course_version__unique_department_year_semester`
(`course_unique`,`department_id`,`valid_from_year`,`valid_from_semester`) USING
BTREE,
  UNIQUE KEY `unique_course_version__cou_id_department_year_semester`
(`cou_id`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_course_name` (`course_name`),
  CONSTRAINT `fk_cou_id` FOREIGN KEY (`cou_id`) REFERENCES `tbl_course` (`cou_id`)
ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_course_version_deparment_id` FOREIGN KEY (`department_id`)
REFERENCES `tbl_department` (`dep_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_classroom`
CREATE TABLE IF NOT EXISTS `tbl_course_classroom` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `course_ver_id` int(11) NOT NULL,
  `course_type` enum('THEORY','LAB','EXERCISE','FIELDWORK','CLINICAL','WORKSHOP')
NOT NULL,
  `course_classrooms` int(11) DEFAULT NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_course_type_classroom`
(`course_ver_id`,`course_type`,`course_classrooms`) USING BTREE,
  KEY `idx_course_ver_id` (`course_ver_id`) USING BTREE,
  KEY `fk_course_classrooms` (`course_classrooms`),
  KEY `idx_course_type` (`course_type`),
  CONSTRAINT `fk_course_classroom_ver_id` FOREIGN KEY (`course_ver_id`) REFERENCES
`tbl_course_version` (`cou_ver_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_course_classrooms` FOREIGN KEY (`course_classrooms`) REFERENCES
`tbl_classroom` (`cla_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=193 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_professor`
CREATE TABLE IF NOT EXISTS `tbl_course_professor` (
```

```
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `course_ver_id` int(11) NOT NULL,
  `course_professor` int(11) NOT NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_course_professor` (`course_ver_id`,`course_professor`) USING
BTREE,
  KEY `idx_course_ver_id` (`course_ver_id`) USING BTREE,
  KEY `fk_course_professor` (`course_professor`),
  CONSTRAINT `fk_course_professor` FOREIGN KEY (`course_professor`) REFERENCES
`tbl_professor` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_course_professor_ver_id` FOREIGN KEY (`course_ver_id`) REFERENCES
`tbl_course_version` (`cou_ver_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=257 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_semester`
CREATE TABLE IF NOT EXISTS `tbl_course_semester` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `course_ver_id` int(11) NOT NULL,
  `course_semester` enum('1','2','3','4','5','6','7','8','9','10','11','12') NOT
NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_course_semester` (`course_ver_id`,`course_semester`),
  KEY `idx_course_ver_id` (`course_ver_id`) USING BTREE,
  KEY `idx_course_semester` (`course_semester`),
  CONSTRAINT `fk_course_semester_ver_id` FOREIGN KEY (`course_ver_id`) REFERENCES
`tbl_course_version` (`cou_ver_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=505 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_type`
CREATE TABLE IF NOT EXISTS `tbl_course_type` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `course_ver_id` int(11) NOT NULL,
  `course_type` enum('THEORY','LAB','EXERCISE','FIELDWORK','CLINICAL','WORKSHOP')
NOT NULL,
  `hours_course_type` int(3) NOT NULL DEFAULT 0,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_course_type` (`course_ver_id`,`course_type`) USING BTREE,
  KEY `idx_course_ver_id` (`course_ver_id`) USING BTREE,
  KEY `idx_course_type` (`course_type`),
  KEY `idx_hours_course_type` (`hours_course_type`),
  CONSTRAINT `fk_course_type_ver_id` FOREIGN KEY (`course_ver_id`) REFERENCES
`tbl_course_version` (`cou_ver_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=929 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

Πίνακας 4-11: Απόσπασμα SQL ορισμού πινάκων οντοτήτων ακαδημαϊκής οργάνωσης



Εικόνα 4 – Οπτική αναπαράσταση πινάκων οντοτήτων ακαδημαϊκής οργάνωσης

4.4 Μηχανισμός Αυτόματης Δημιουργίας Ωρολογίου Προγράμματος

Στο πλαίσιο της υλοποίησης της πλατφόρμας, αναπτύχθηκε υποστηρικτικός μηχανισμός αυτόματης παραγωγής προτεινόμενων ωρολογίων προγραμμάτων. Η ανάπτυξη του μηχανισμού αυτού δεν αποτέλεσε βασική απαίτηση του θέματος της παρούσας εργασίας, ωστόσο ενσωματώθηκε με στόχο την ενίσχυση της λειτουργικότητας του συστήματος και τη μείωση του χρόνου δημιουργίας προγράμματος.

Ο μηχανισμός βασίζεται σε γενετικό αλγόριθμο και λειτουργεί ως εργαλείο υποστήριξης της διαδικασίας χρονοπρογραμματισμού, παρέχοντας προτάσεις κατανομής μαθημάτων σε χρονικές θέσεις (slots), αίθουσες και διδάσκοντες. Έχει σχεδιαστεί με τρόπο ανεξάρτητο από την υπόλοιπη πλατφόρμα και μπορεί να λειτουργήσει είτε για τη δημιουργία προτάσεων προγράμματος από την αρχή, είτε υποστηρικτικά σε ήδη μερικώς συμπληρωμένο πρόγραμμα.

Τα δεδομένα εισόδου εξάγονται αρχικά σε δομημένη μορφή JSON μέσω REST API, το οποίο μπορεί να χρησιμοποιηθεί και από εξωτερικά συστήματα ή χρήστες για τη λήψη των δεδομένων και των παραμέτρων που έχουν καταχωρηθεί στην πλατφόρμα. Μέσω των δεδομένων αυτών, ο μηχανισμός αυτόματης δημιουργίας εκτελείται χωρίς άμεση πρόσβαση στη βάση δεδομένων κατά την εκτέλεση του αλγορίθμου και χωρίς επιβάρυνση του συστήματος. Η προσέγγιση αυτή επιτρέπει την αξιοποίηση των ίδιων δεδομένων από διαφορετικές μεθόδους επίλυσης στο μέλλον.

Κατά την εκτέλεση του solver, δημιουργείται αρχικός πληθυσμός υποψήφιων λύσεων, οι οποίες αξιολογούνται βάσει συνάρτησης καταλληλότητας. Στη συνέχεια εφαρμόζονται επαναληπτικά γενετικοί τελεστές επιλογής, διασταύρωσης και μετάλλαξης, καθώς και διαδικασία επιδιόρθωσης μη έγκυρων λύσεων, με στόχο τη σταδιακή βελτίωση του πληθυσμού.

Το αποτέλεσμα της διαδικασίας δεν αποθηκεύεται απευθείας ως τελικό ωρολόγιο πρόγραμμα, αλλά ως προσωρινή πρόταση σε ειδικούς πίνακες της Βάσης Δεδομένων. Ο διαχειριστής έχει τη δυνατότητα να προβάλει το προτεινόμενο πρόγραμμα, να το τροποποιήσει και να το εγκρίνει πριν την οριστική αποδοχή και αποθήκευση.

Αναλυτική περιγραφή της λειτουργίας του γενετικού αλγορίθμου παρουσιάζεται στο Παράρτημα ΣΤ: «Ανάλυση Επιλυτή Γενετικού Αλγορίθμου» της παρούσας εργασίας.

4.5 Σύνοψη Κεφαλαίου

Στο παρόν κεφάλαιο παρουσιάστηκε αναλυτικά η σχεδίαση και η υλοποίηση της πλατφόρμας, καλύπτοντας το επίπεδο του πηγαίου κώδικα και το επίπεδο της Βάσης Δεδομένων. Αρχικά αναλύθηκαν οι αρχές ορθού προγραμματισμού που υιοθετήθηκαν, με έμφαση στη σαφή ονοματοδοσία, την αναγνωσιμότητα, τη συμμόρφωση με πρότυπα, τον διαχωρισμό ευθυνών και τη συστηματική χρήση εργαλείων ελέγχου ποιότητας. Στη συνέχεια παρουσιάστηκε η αρχιτεκτονική της εφαρμογής και η δομή των αρχείων, με διακριτό διαχωρισμό μεταξύ frontend και backend, καθώς και οι μηχανισμοί ασφάλειας, διαχείρισης σφαλμάτων, απόδοσης και προσωρινής αποθήκευσης δεδομένων. Κατόπιν, περιγράφηκε η σχεδίαση της Βάσης Δεδομένων και το σχήμα της, με έμφαση στην κανονικοποίηση, τις συσχετίσεις, την ακεραιότητα και την υποστήριξη των λειτουργικών απαιτήσεων του συστήματος. Τέλος, παρουσιάστηκε ο μηχανισμός επίλυσης και αυτόματης δημιουργίας ωρολογίου προγράμματος με χρήση γενετικού αλγορίθμου. Συνολικά, το κεφάλαιο τεκμηριώνει ότι η πλατφόρμα αναπτύχθηκε με δομημένη, επεκτάσιμη και ασφαλή προσέγγιση, θέτοντας σταθερά θεμέλια για τη λειτουργία, τη συντήρηση και τη μελλοντική εξέλιξή της.

Στο επόμενο κεφάλαιο ακολουθεί η αξιολόγηση της πλατφόρμας, όπου παρουσιάζονται οι διαδικασίες ελέγχου, καθώς και τα αποτελέσματα που προέκυψαν από τη λειτουργική και ποιοτική εξέταση του συστήματος.

5. Αξιολόγηση και Αποτελέσματα

Στο κεφάλαιο αυτό παρουσιάζεται η αξιολόγηση της λειτουργίας και της απόδοσης της πλατφόρμας που αναπτύχθηκε, οι μετρικές του κώδικα καθώς και στιγμιότυπα λειτουργίας (screenshots). Αρχικά παρουσιάζονται ενδεικτικά παραδείγματα χρήσης και βασικές λειτουργίες του συστήματος, συνοδευόμενα από στιγμιότυπα της διεπαφής χρήστη. Στη συνέχεια περιγράφονται δοκιμές απόδοσης που πραγματοποιήθηκαν με προσομοίωση πολλαπλών χρηστών, καθώς και μετρικές ποιότητας του πηγαίου κώδικα που προέκυψαν από εργαλεία στατικής ανάλυσης. Τέλος, συζητούνται οι περιορισμοί του συστήματος και προτείνονται πιθανές μελλοντικές επεκτάσεις της πλατφόρμας.

5.1 Περιβάλλον Δοκιμών

Για την αξιολόγηση της λειτουργίας της πλατφόρμας πραγματοποιήθηκαν δοκιμές σε περιβάλλον τοπικής εγκατάστασης. Η εφαρμογή εκτελέστηκε σε web server IIS της Microsoft (βλ. υποενότητα 2.8.6), καθώς και σε Apache server μέσω του XAMPP (βλ. υποενότητα 2.8.7). Ως σύστημα διαχείρισης βάσης δεδομένων χρησιμοποιήθηκε το MariaDB μέσω του HeidiSQL τοπικά, καθώς και το phpMyAdmin σε online παραγωγικό περιβάλλον (βλ. υποενότητα 2.8.4).

Οι δοκιμές πραγματοποιήθηκαν μέσω σύγχρονων web browsers (Chrome, Firefox, Opera) καθώς και προσομοίωση φορητών συσκευών με χρήση των εργαλείων που παρέχουν οι web browsers, όσο και σε πραγματικές φορητές συσκευές (κινητό, tablet). Ελέγχθηκαν τόσο η λειτουργικότητα των επιμέρους υποσυστημάτων όσο και η απόδοση της εφαρμογής σε σενάρια πολλαπλών αιτημάτων. Παράλληλα εκτελούνταν τακτικοί έλεγχοι στατικής ανάλυσης για τον εντοπισμό σφαλμάτων, την ορθή συγγραφή κώδικα και τον έλεγχο της πολυπλοκότητας μεθόδων, συναρτήσεων και κλάσεων.

Το βασικό περιβάλλον δημιουργίας και ελέγχου κώδικα περιλαμβάνει τα ακόλουθα πακέτα λογισμικού και εφαρμογές:

- MariaDB 10.11
- PHP 8.2.29
- IIS Express Server
- Chrome 139+ / Firefox 141+

5.2 Δοκιμές Απόδοσης με Apache JMeter

Πραγματοποιήθηκαν δειγματοληπτικοί έλεγχοι εκτέλεσης με προσομοίωση πολλαπλών χρηστών μέσω του εργαλείου Apache JMeter (βλ. υποενότητα 2.8.5), τόσο σε δοκιμαστική τοπική εκτέλεση (localhost) όσο και σε περιβάλλον παραγωγής σε shared hosting.

Για μικρό αριθμό χρηστών (έως περίπου 100 ταυτόχρονους χρήστες, συνθετικό φορτίο), η απόδοση της εφαρμογής ήταν ιδιαίτερα υψηλή, με πολύ μικρούς χρόνους απόκρισης. Ακόμη και σε σενάρια με 200 ταυτόχρονους χρήστες (συνθετικό φορτίο), η συνολική απόδοση του συστήματος παρέμεινε σε πολύ ικανοποιητικό επίπεδο.

Οι διαφορές μεταξύ εκτέλεσης με χρήση cache και εκτέλεσης αποκλειστικά μέσω ερωτημάτων προς τη βάση δεδομένων εμφανίζονται σχετικά μικρές, γεγονός που υποδηλώνει ότι τα ερωτήματα της εφαρμογής είναι ήδη επαρκώς βελτιστοποιημένα. Ωστόσο, σε πιο σύνθετες λειτουργίες, όπως η προβολή ωρολογίων προγραμμάτων όπου απαιτούνται συνενώσεις (joins) πολλών πινάκων, η χρήση μηχανισμών προσωρινής αποθήκευσης (cache) παρουσιάζει βελτιωμένη απόδοση.

Κατά τη διάρκεια των δοκιμών δεν παρατηρήθηκαν σφάλματα εκτέλεσης (error rate 0%), γεγονός που υποδηλώνει σταθερή λειτουργία του συστήματος ακόμη και υπό αυξημένο φόρτο. Οι μέγιστοι χρόνοι απόκρισης εμφανίζουν φυσιολογικές διακυμάνσεις λόγω ουράς εξυπηρέτησης αιτημάτων (queuing) όταν αυξάνεται η ταυτόχρονη ζήτηση σελίδων. Ο ρυθμός εξυπηρέτησης αιτημάτων (throughput) διατηρήθηκε σταθερός καθ' όλη τη διάρκεια της δοκιμής, με περίπου 29 αιτήματα ανά δευτερόλεπτο.

Στους παρακάτω πίνακες αναλύεται δοκιμαστική εκτέλεση με 200 ταυτόχρονους χρήστες (συνθετικό φορτίο), 4 εκτελέσεις ανά σελίδα και ανά χρήστη, σταδιακή είσοδο χρηστών ανά 20 δευτερόλεπτα και τυχαίο χρόνο παραμονής σε σελίδα από 1 έως 4 δευτερόλεπτα για προσομοίωση πραγματικών χρηστών. Κατά τη δοκιμή χρήσης, προσομοιώθηκε η πρόσβαση στην αρχική σελίδα και σε σελίδα διαχειριστή για προβολή λίστας όλων των Τμημάτων του Πανεπιστημίου με κλήσεις προς την ΒΔ για ερωτήματα ή χρήση cache αρχείων.

Τεχνικά χαρακτηριστικά server: Windows 64-bit, 4GB RAM, 2-cores @ 2.93GHz, SSD.

Ρυθμίσεις IIS Express / FastCGI: Προεπιλεγμένες ρυθμίσεις: μέγιστο μήκος ουράς αιτημάτων 1000, χρόνος αδράνειας 300sec, λήξη χρονικού ορίου αιτήματος 90sec.

Σενάριο Δοκιμής	Σελίδα (Page)	Requests	Avg Response Time (ms)	Min (ms)	Max (ms)
Cache ON (Limited queries)	Homepage	800	2333	43	4672
	List Departments	800	2563	88	4631
	TOTAL	**1600**	**2448**	**43**	**4672**
Cache OFF (DB queries)	Homepage	800	2148	42	4443
	List Departments	800	2273	81	4497
	TOTAL	**1600**	**2210**	**42**	**4497**

Πίνακας 5-1: Αποτελέσματα εκτέλεσης με πολλαπλούς χρήστες: χρόνοι απόκρισης

Σενάριο Δοκιμής	Σελίδα (Page)	Std Dev	Throughput (req/sec)	Error %
Cache ON (Limited queries)	Homepage	1312	15.01	0%
	List Departments	1324	14.97	0%
	TOTAL	**1323**	**28.81**	**0%**
Cache OFF (DB queries)	Homepage	1401	15.32	0%
	List Departments	1377	15.04	0%
	TOTAL	**1390**	**29.39**	**0%**

Πίνακας 5-2: Αποτελέσματα ελέγχου εκτέλεσης με πολλαπλούς χρήστες

5.3 Αποτελέσματα Ελέγχου Ποιότητας Κώδικα

Παράλληλα με τις δοκιμές λειτουργίας πραγματοποιήθηκε αξιολόγηση της ποιότητας του πηγαίου κώδικα με εργαλεία στατικής ανάλυσης (static analysis) καθ' όλη τη διάρκεια ανάπτυξης της πλατφόρμας.

Συγκεκριμένα χρησιμοποιήθηκαν τα εργαλεία PHPStan και Psalm (βλ. υποενότητα 2.6.2) για τον εντοπισμό και τη διόρθωση πιθανών σφαλμάτων τύπων και λογικών ασυνεπειών στον κώδικα. Τα αποτελέσματα των ελέγχων έδειξαν ελάχιστο αριθμό προειδοποιήσεων, οι οποίες σχετίζονται κυρίως με περιπτώσεις ειδικής υλοποίησης ή με τμήματα κώδικα που βρίσκονται υπό σταδιακή βελτίωση.

Για την τήρηση προτύπων μορφοποίησης και συγγραφής κώδικα χρησιμοποιήθηκε το εργαλείο PHP_CodeSniffer (βλ. υποενότητα 2.6.2) σε συνδυασμό με το πρότυπο PSR-12. Ο κώδικας παρουσιάζει υψηλό βαθμό συμμόρφωσης με το πρότυπο, ενώ σε ορισμένες περιπτώσεις χρησιμοποιήθηκαν στοχευμένες εξαιρέσεις, όταν η λειτουργικότητα της εφαρμογής απαιτούσε αποκλίσεις από τα αυστηρά πρότυπα μορφοποίησης, όπως για παράδειγμα στον κώδικα του Πίνακα 4-7.

Επιπλέον χρησιμοποιήθηκε το εργαλείο PHPMD (βλ. υποενότητα 2.6.2) για τον εντοπισμό πιθανών βελτιώσεων στη δομή του κώδικα, όπως περιπτώσεις αυξημένης κυκλωματικής πολυπλοκότητας (βλ. υποενότητα 2.6.2). Τα αποτελέσματα δείχνουν ότι η συνολική δομή του κώδικα παραμένει σε αποδεκτά επίπεδα συντηρησιμότητας.

5.4 Μετρικές Συστήματος

Για την αξιολόγηση της δομής και της συντηρησιμότητας του πηγαίου κώδικα χρησιμοποιήθηκε το εργαλείο PHP Metrics (βλ. υποενότητα 2.6.2), το οποίο παρέχει μετρικές σχετικά με την πολυπλοκότητα, τη δομή και την ποιότητα του κώδικα.

Η ανάλυση πραγματοποιήθηκε σε συνολικά 474 μονάδες κώδικα (συναρτήσεις και κλάσεις). Οι μετρικές που υπολογίστηκαν περιλαμβάνουν την κυκλωματική πολυπλοκότητα (Cyclomatic Complexity), τον δείκτη συντηρησιμότητας (Maintainability Index), καθώς και μετρικές μεγέθους κώδικα (Lines of Code). Τα αποτελέσματα που παρουσιάζονται στον Πίνακα 5-3, δείχνουν ότι η μέση κυκλωματική πολυπλοκότητα των μονάδων κώδικα παραμένει σε αποδεκτά επίπεδα, ενώ ο μέσος δείκτης συντηρησιμότητας βρίσκεται σε υψηλό επίπεδο, γεγονός που υποδηλώνει ότι ο κώδικας είναι κατανοητός και συντηρήσιμος.

Η αρχιτεκτονική της εφαρμογής, η οποία βασίζεται σε σαφή διαχωρισμό επιπέδων (frontend, backend), καθώς και στη χρήση ανεξάρτητων κλάσεων για τη διαχείριση δεδομένων και λειτουργιών, συμβάλλει στη βελτίωση της αναγνωσιμότητας και της επεκτασιμότητας του συστήματος.

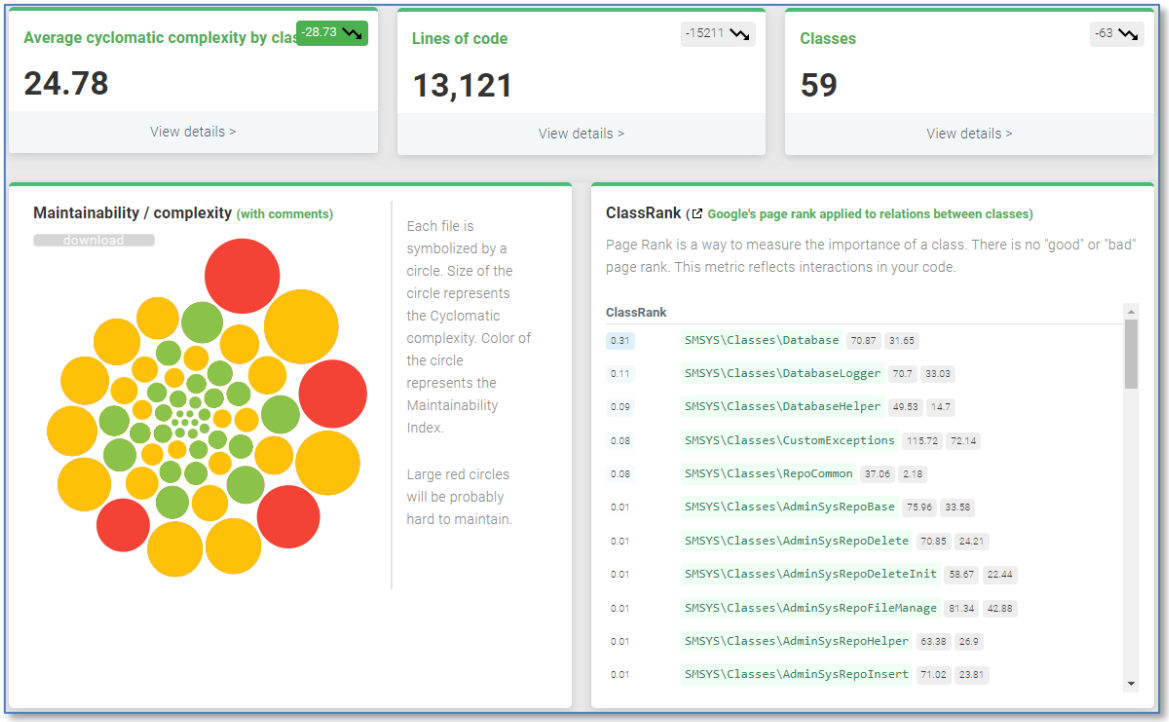
Σε ορισμένες περιπτώσεις παρατηρούνται αυξημένες τιμές πολυπλοκότητας σε κεντρικές κλάσεις της εφαρμογής, οι οποίες υλοποιούν πολλαπλές λειτουργίες πρόσβασης δεδομένων και διαχείρισης επιχειρησιακής λογικής. Οι τιμές αυτές θεωρούνται αναμενόμενες για κλάσεις με κεντρικό ρόλο στη λειτουργία του συστήματος, καθώς συγκεντρώνουν σημαντικό μέρος της βασικής λειτουργικότητας. Η διατήρηση της συγκεκριμένης σχεδιαστικής προσέγγισης διευκολύνει τη συντήρηση και την κατανόηση του κώδικα, αποφεύγοντας τον διασκορπισμό της σχετικής λογικής σε μεγάλο αριθμό αρχείων και επιτρέποντας πιο συνεκτική οργάνωση των λειτουργιών του συστήματος.

Συνολικά, τα αποτελέσματα της ανάλυσης υποδεικνύουν ότι η εφαρμογή παρουσιάζει καλή ποιότητα κώδικα και ικανοποιητικό επίπεδο συντηρησιμότητας για σύστημα αντίστοιχου μεγέθους και πολυπλοκότητας.

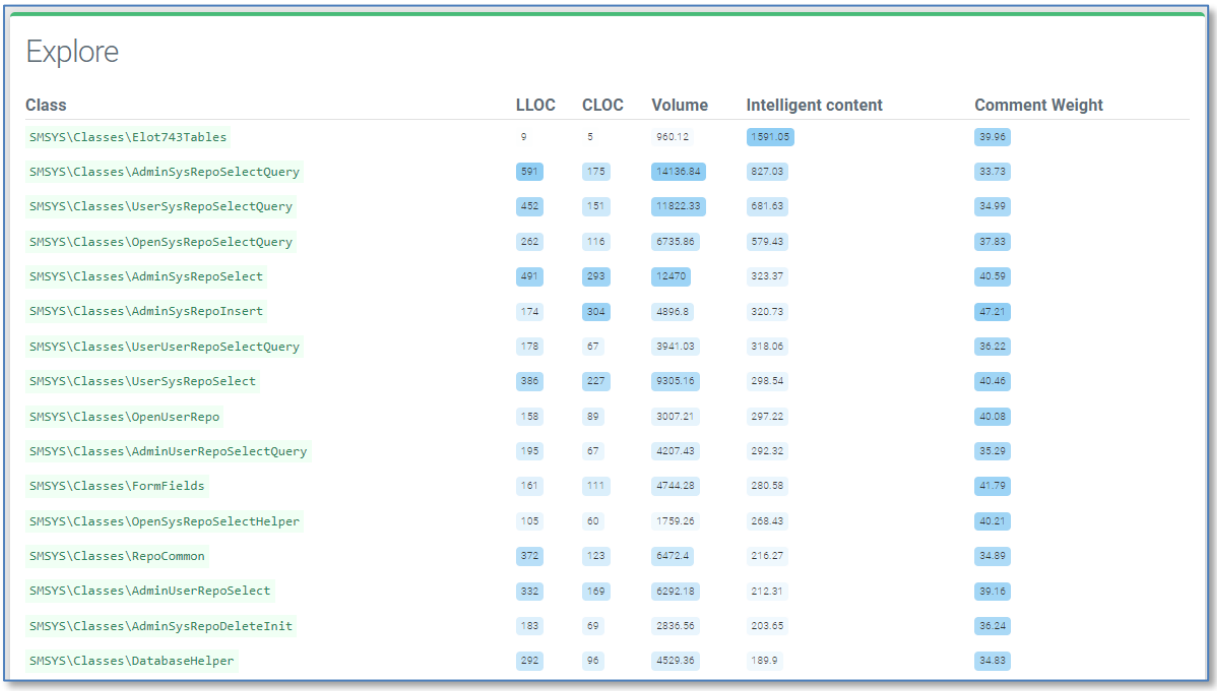
Μετρική	Τιμή	Ερμηνεία
Αριθμός μονάδων ανάλυσης κώδικα (Classes / Functions)	474	Συνολικές μονάδες που αναλύθηκαν από το PHPMetrics
Μέση κυκλωματική πολυπλοκότητα (Average Cyclomatic Complexity)	9–17	Μικρή προς μέτρια πολυπλοκότητα, αναμενόμενη σε backend λογική
Μέγιστη κυκλωματική πολυπλοκότητα	35	Εμφανίζεται σε κεντρικές κλάσεις διαχείρισης δεδομένων
Μέσος δείκτης συντηρησιμότητας (Maintainability Index)	~79	Καλό επίπεδο συντηρησιμότητας
Ελάχιστος δείκτης συντηρησιμότητας	~55	Παρατηρείται σε κλάσεις με αυξημένη λειτουργικότητα
Μέσο μέγεθος μονάδας κώδικα	~40–50 γραμμές	Κανονικό μέγεθος για συναρτήσεις εφαρμογής
Μέγιστο μέγεθος μονάδας κώδικα	~600 γραμμές	Κεντρικές κλάσεις διαχείρισης δεδομένων

Πίνακας 5-3: Μετρικές Πλατφόρμας με χρήση του εργαλείου PHP Metrics

Στις παρακάτω εικόνες παρατίθενται στιγμιότυπα οθόνης των μετρικών αποτελεσμάτων που παρήχθησαν με το εργαλείο PHP Metrics.



Εικόνα 5 – Στιγμιότυπο οθόνης μετρικών αποτελεσμάτων PHP Metrics



Εικόνα 6 – Στιγμιότυπο οθόνης ανάλυσης ανά κλάση του PHP Metrics

Επιπλέον των παραπάνω μετρήσεων, πραγματοποιήθηκε στατιστική ανάλυση της πλατφόρμας με χρήση του εργαλείου phploc[30], με στόχο τον υπολογισμό βασικών μεγεθών του έργου. Οι μετρήσεις εκτελέστηκαν τόσο για τον κώδικα που αναπτύχθηκε στο πλαίσιο της παρούσας εργασίας, όσο και για το σύνολο του κώδικα, συμπεριλαμβανομένων των χρησιμοποιούμενων βιβλιοθηκών τρίτων.

Μετρική (βασικά στατιστικά)	**Τιμή (χωρίς βιβλιοθήκες)**	**Τιμή (με βιβλιοθήκες)**
Φάκελοι	9	92
Αρχεία	471	1389
Συνολικές Γραμμές Κώδικα (LOC)	102.314	305.498
Γραμμές Σχολίων (CLOC)	21.645 (21,16%)	71.570 (23,43%)
Καθαρός Κώδικας (NCLOC)	80.669 (78,84%)	233.928 (76,57%)
Λογικές Γραμμές (LLOC)	23.830	75.619

Πίνακας 5-4: Βασικά στατιστικά πλατφόρμας με χρήση του εργαλείου phploc

Μετρική (δομικά στατιστικά)	**Τιμή (χωρίς βιβλιοθήκες)**	**Τιμή (με βιβλιοθήκες)**
Κλάσεις (Classes)	70	629
Μέθοδοι (Methods)	571	5.962
Συναρτήσεις (Functions)	748	1.214
Σταθερές (Constants)	420	2.079
Namespaces	5	81

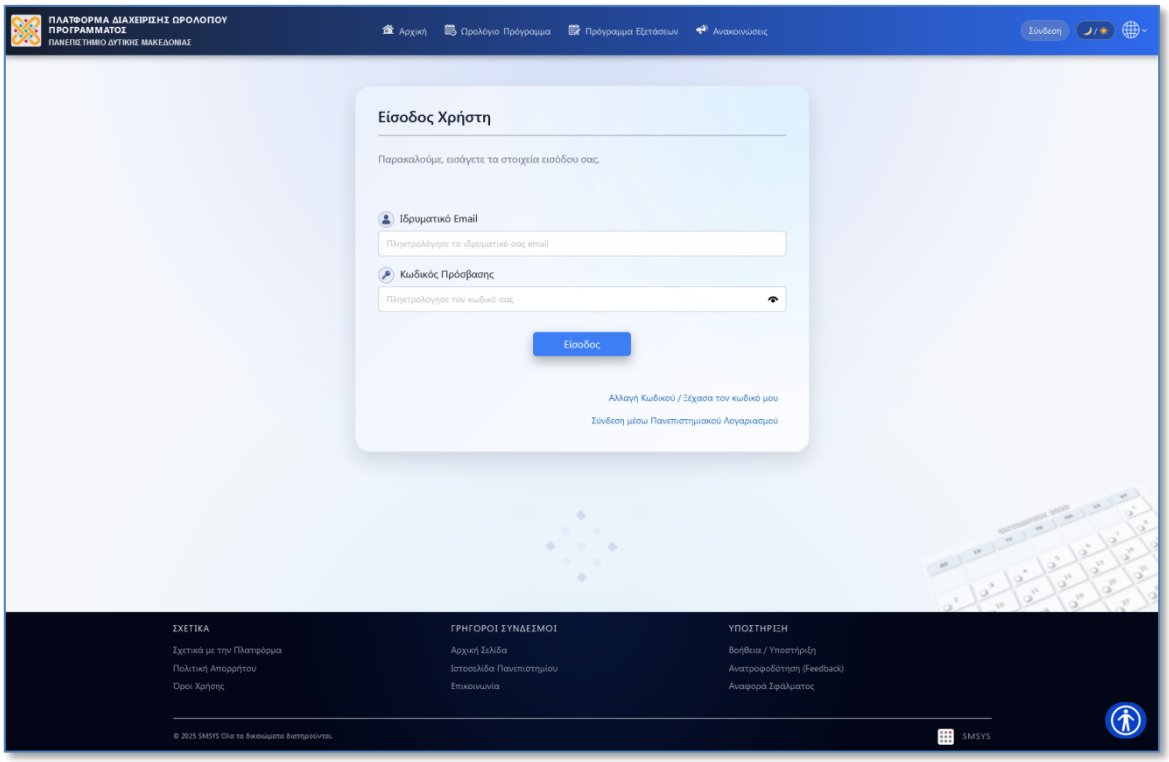
Πίνακας 5-5: Δομικά στατιστικά πλατφόρμας με χρήση του εργαλείου phploc

[30] Εργαλείο ανάλυσης στατιστικών μεγέθους για την PHP https://github.com/sebastianbergmann/phploc

5.5 Παραδείγματα Βασικών Λειτουργιών

Σελίδα εισόδου χρηστών:

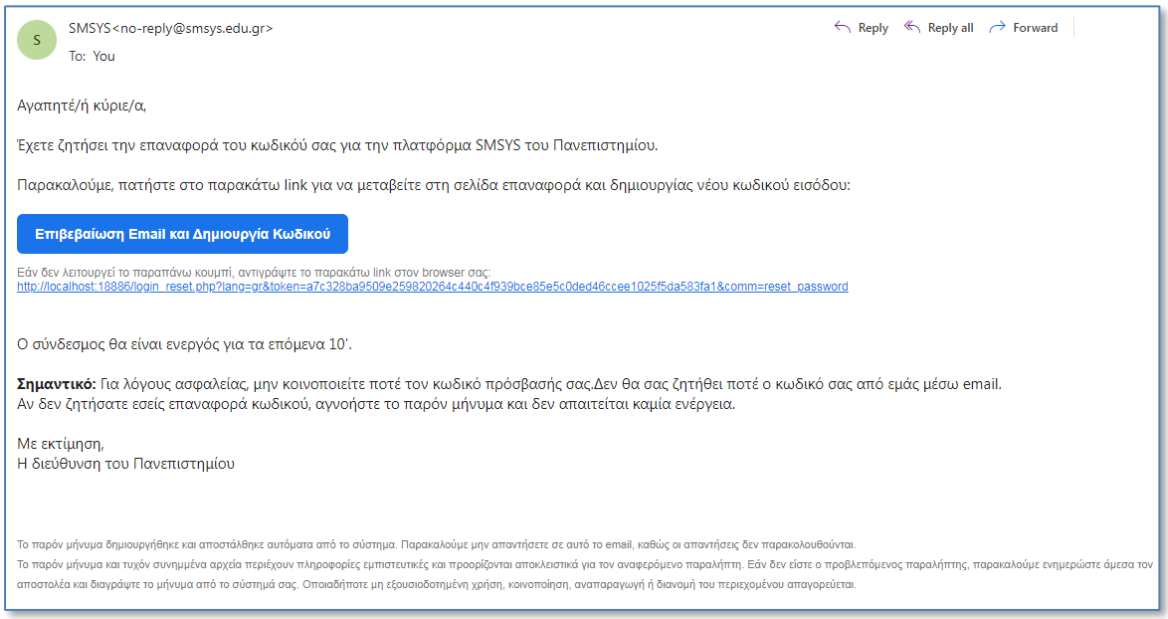
Η σελίδα παρέχει τη δυνατότητα εισόδου των χρηστών στην πλατφόρμα με τη χρήση των προσωπικών τους διαπιστευτηρίων που έχουν καταχωρηθεί στο σύστημα. Παρέχεται επίσης η δυνατότητα εισόδου μέσω μηχανισμού SSO (Single Sign-On) που προσφέρεται από το Πανεπιστήμιο για Φοιτητές και Διδάσκοντες. Ο κωδικός πρόσβασης εμφανίζεται αρχικά αποκρυμμένος για λόγους ασφάλειας, ενώ παρέχεται κουμπί προσωρινής προβολής του ώστε ο χρήστης να μπορεί να ελέγξει την ορθότητα της εισαγωγής.



Εικόνα 7 – Σελίδα Εισόδου Χρηστών με τοπικά διαπιστευτήρια

Σε περίπτωση πολλαπλών αποτυχημένων προσπαθειών εισόδου, εμφανίζεται επιπλέον πεδίο CAPTCHA, με στόχο την αποτροπή αυτοματοποιημένων επιθέσεων (bots) και άλλων μη εξουσιοδοτημένων ενεργειών.

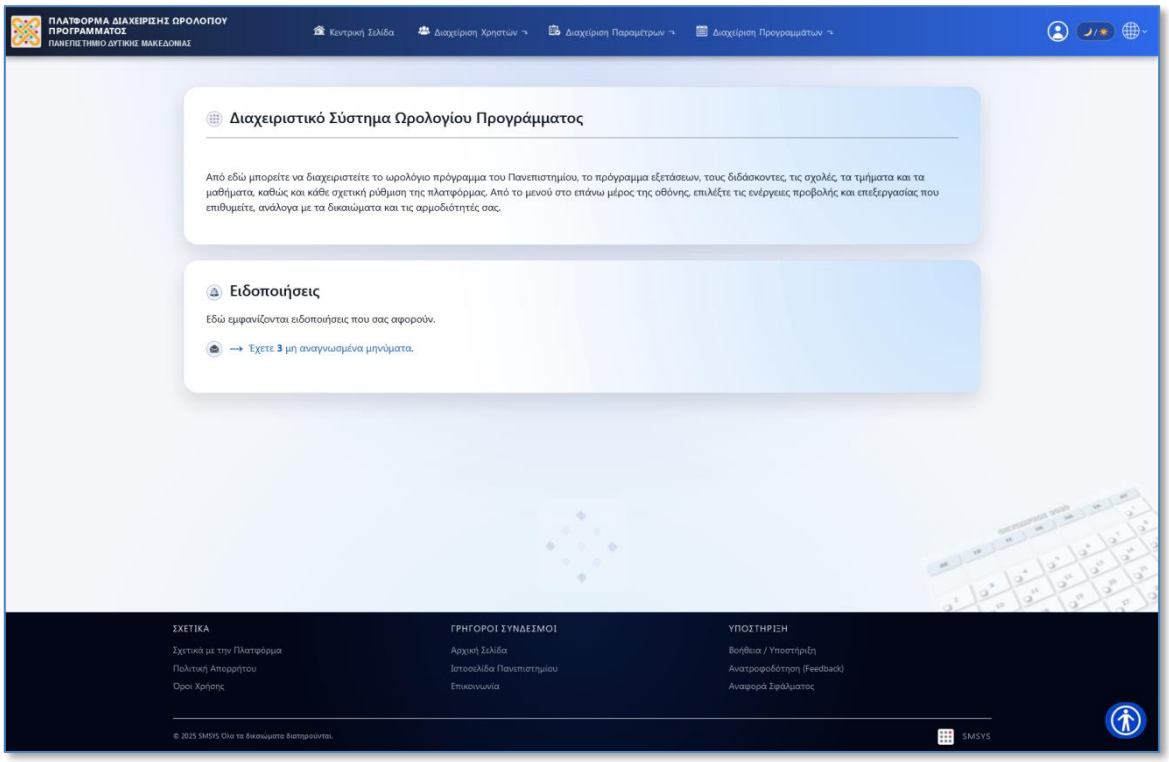
Επιπλέον, παρέχεται δυνατότητα αλλαγής ή επαναφοράς κωδικού πρόσβασης σε περίπτωση απώλειας. Ο χρήστης καλείται να εισαγάγει στο σχετικό πεδίο το ιδρυματικό του email. Εφόσον επιβεβαιωθεί η ύπαρξη του χρήστη στο σύστημα, αποστέλλεται αυτόματα μήνυμα ηλεκτρονικού ταχυδρομείου με προσωρινό σύνδεσμο επαναδημιουργίας κωδικού καθώς και σχετικές οδηγίες.



Εικόνα 8 – Μήνυμα email αλλαγής / επαναφοράς κωδικού πρόσβασης

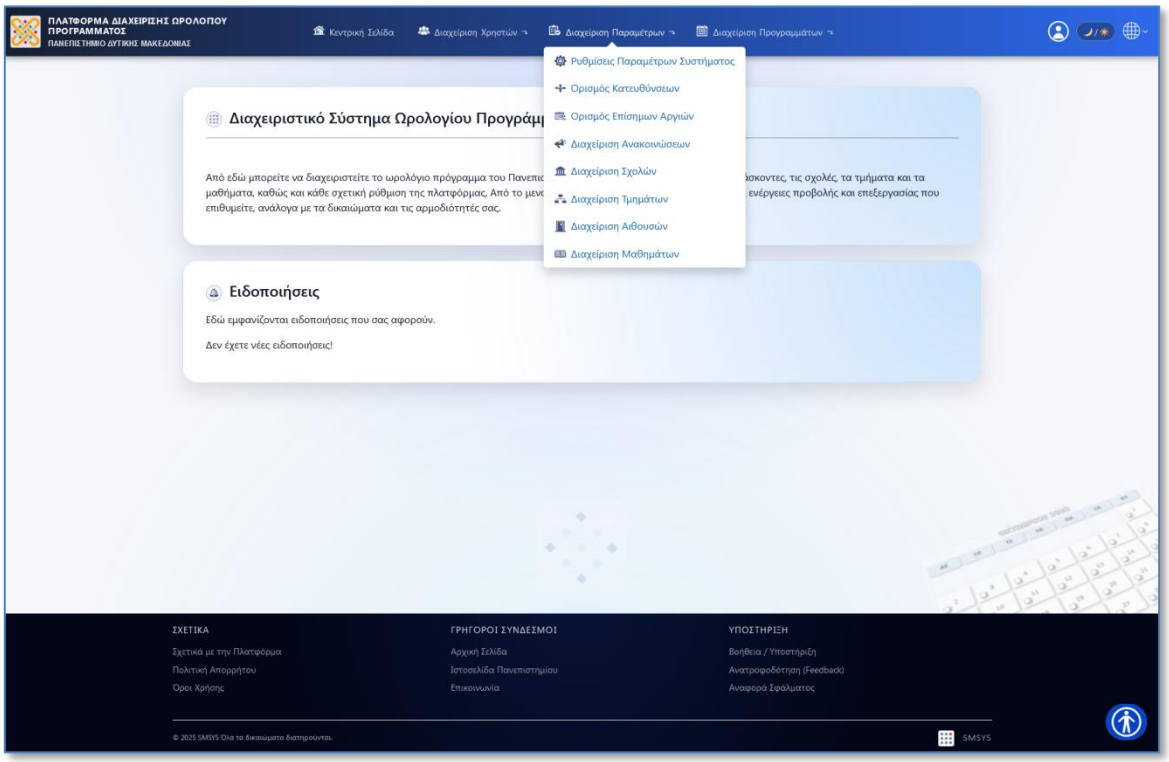
Κεντρική Σελίδα Διαχειριστή:

Η σελίδα περιγράφει τις βασικές λειτουργίες που παρέχονται στον χρήστη, ενώ περιλαμβάνει τμήμα ειδοποιήσεων για άμεση ενημέρωση όπως, νέα μηνύματα, νέες εγγραφές, έκτακτες αλλαγές προγράμματος από διδάσκοντα, κ.ά.



Εικόνα 9 – Κεντρική Σελίδα Διαχειριστή

Σε κάθε σελίδα περιλαμβάνεται μενού επιλογής στο πάνω μέρος από όπου ο χρήστης μπορεί να επιλέξει τις ενέργειες και λειτουργίες που επιθυμεί. Στα δεξιά υπάρχουν επιπλέον οι επιλογές προβολής και επεξεργασίας προφίλ, αποσύνδεσης, αλλαγής φωτεινού/σκοτεινού θέματος και αλλαγής γλώσσας (Ελληνικά / Αγγλικά). Το μενού του Κύριου Διαχειριστή περιλαμβάνει όλες τις δυνατότητες και λειτουργίες που παρέχονται από το σύστημα.



Εικόνα 10 – Βασικό μενού επιλογών διαχειριστή

Ανάλογο μενού εμφανίζεται στους Διαχειριστές Τμημάτων, στους Διδάσκοντες και στους Φοιτητές, με περιορισμό των επιλογών σύμφωνα με το ρόλο κάθε χρήστη.

Μενού Κύριου Διαχειριστή:

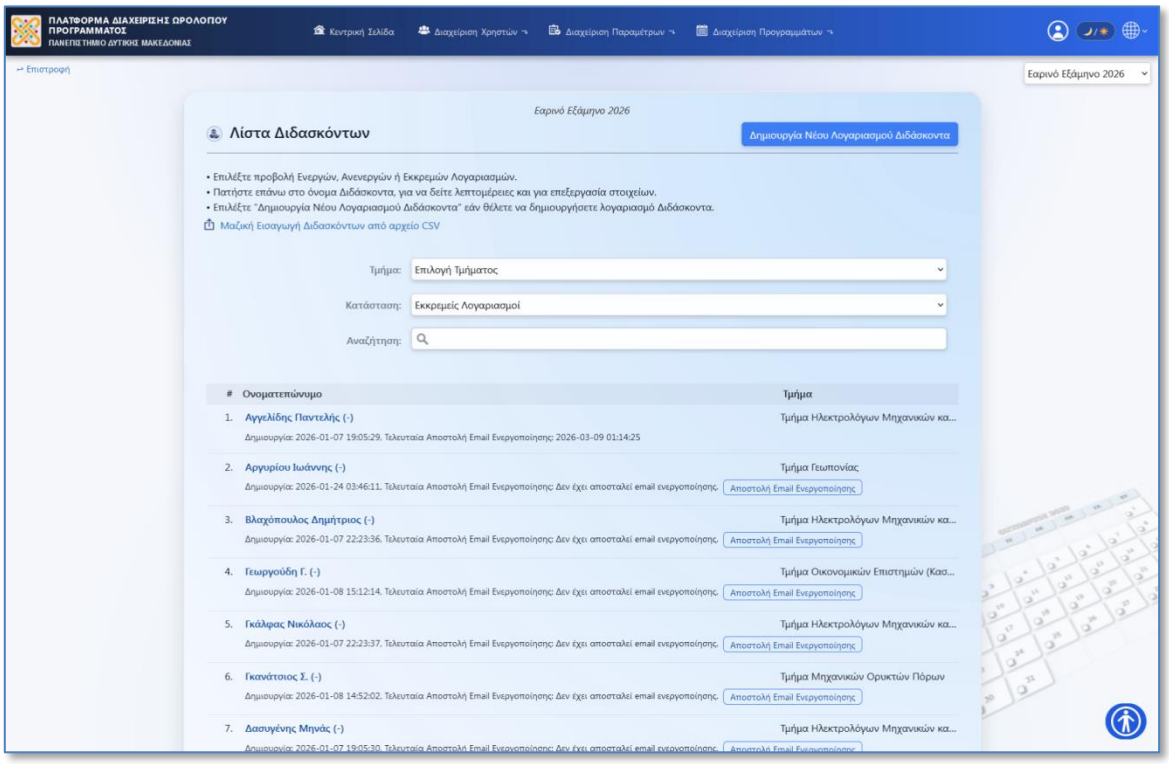
- Διαχείριση Χρηστών
    - Προβολή Εκκρεμών Ενεργοποιήσεων Λογαριασμών
    - Διαχείριση Λογαριασμών Διαχειριστών
    - Διαχείριση Διδασκόντων
    - Διαχείριση Φοιτητών
    - Κέντρο Μηνυμάτων και Υποστήριξης
- Διαχείριση Παραμέτρων
    - Ρυθμίσεις Παραμέτρων Συστήματος

- Ορισμός Κατευθύνσεων
- Ορισμός Επίσημων Αργιών
- Διαχείριση Ανακοινώσεων
- Διαχείριση Σχολών
- Διαχείριση Τμημάτων
- Διαχείριση Αιθουσών
- Διαχείριση Μαθημάτων

- Διαχείριση Προγραμμάτων
  - Διαθεσιμότητα Διδασκόντων
  - Ημερολόγιο Μαθημάτων
  - Πρόγραμμα Εξαμήνου
  - Πρόγραμμα Εξετάσεων

Σελίδα Προβολής Λίστας Διδασκόντων:

Η σελίδα περιλαμβάνει λίστα Διδασκόντων με επιλογή προβολής ανά Τμήμα, ανά κατάσταση λογαριασμού (ενεργή, ανενεργή, εκκρεμής ενεργοποίηση λογαριασμού) και δυνατότητα αναζήτησης.



Εικόνα 11 – Λίστα Λογαριασμών Διδασκόντων

Σελιδοποίηση αποτελεσμάτων (pagination):

Σε σελίδες προβολής λίστας δεδομένων, όπως οι Διδάσκοντες, οι Φοιτητές, τα Μηνύματα κ.ά., υλοποιείται λειτουργία σελιδοποίησης (pagination). Η λειτουργία αυτή παρέχει δυνατότητες μετάβασης στην προηγούμενη ή επόμενη σελίδα, επιλογής μετάβασης σε συγκεκριμένο αριθμό σελίδας, καθώς και προβολής του συνολικού αριθμού εγγραφών.

Πέρα από τη βελτίωση της εμπειρίας χρήστη, η σελιδοποίηση συμβάλλει και στη βελτιστοποίηση της απόδοσης του συστήματος, καθώς αποτρέπει την εκτέλεση ερωτημάτων που θα επέστρεφαν μεγάλο όγκο δεδομένων ταυτόχρονα, περιορίζοντας έτσι την άσκοπη χρήση υπολογιστικών πόρων.

Υποστήριξη εκδόσεων δεδομένων (versioning):

Βασικό στοιχείο της δομής και της αρχιτεκτονικής του συστήματος αποτελεί η υποστήριξη εκδόσεων δεδομένων (versioning). Η προσέγγιση αυτή επιτρέπει την προβολή των δεδομένων (Σχολές, Τμήματα, Μαθήματα, Αίθουσες) ανά ακαδημαϊκό εξάμηνο, διατηρώντας παράλληλα το ιστορικό των μεταβολών και αποφεύγοντας την άσκοπη επανάληψη δεδομένων κατά την αποθήκευση.

Η λειτουργία του μηχανισμού βασίζεται στην αποθήκευση των δεδομένων ως ξεχωριστές «εκδόσεις» που συνδέονται με συγκεκριμένο εξάμηνο και ακαδημαϊκό έτος, για παράδειγμα Εαρινό Εξάμηνο 2025. Κάθε μεταβολή που πραγματοποιείται σε ένα στοιχείο επηρεάζει μόνο την εγγραφή που αντιστοιχεί στο συγκεκριμένο εξάμηνο, διατηρώντας τόσο τις σχέσεις του με τα υπόλοιπα δεδομένα του συστήματος όσο και τις προηγούμενες εκδόσεις για σκοπούς προβολής και παρακολούθησης ιστορικότητας.

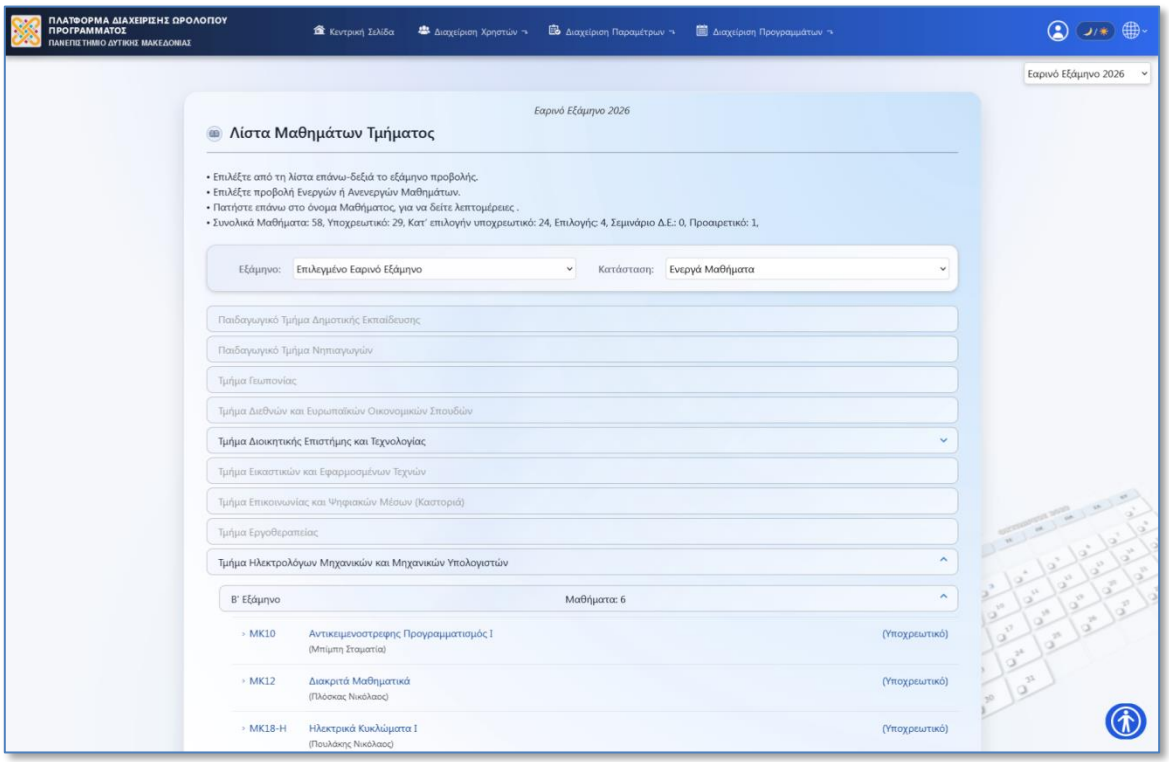
Στο επάνω δεξιά τμήμα των σελίδων στις οποίες εφαρμόζεται ο μηχανισμός εκδόσεων, παρέχεται λίστα επιλογής για την προβολή συγκεκριμένου εξαμήνου και έτους. Ως προεπιλογή εμφανίζεται το τρέχον ακαδημαϊκό εξάμηνο, ενώ υπάρχει δυνατότητα επιλογής και του επόμενου εξαμήνου, ώστε να είναι δυνατή η προετοιμασία και επεξεργασία των δεδομένων από τους Διαχειριστές.

Για λόγους σαφήνειας, το εξάμηνο και το ακαδημαϊκό έτος των δεδομένων που προβάλλονται εμφανίζονται εμφανώς στο επάνω κεντρικό μέρος κάθε φόρμας του συστήματος.

Λίστα Μαθημάτων ανά Τμήμα:

Η παρακάτω απεικόνιση παρουσιάζει την προβολή που διαθέτει ο Κύριος Διαχειριστής, ενώ στους Διαχειριστές Τμημάτων εμφανίζονται μόνο τα Μαθήματα που ανήκουν στο Τμήμα ευθύνης τους. Παρέχεται η δυνατότητα επιλογής προβολής δεδομένων για συγκεκριμένο ακαδημαϊκό εξάμηνο.

Επιπλέον, εμφανίζονται συνοπτικά στατιστικά στοιχεία, όπως ο αριθμός μαθημάτων ανά κατηγορία. Για κάθε μάθημα παρουσιάζονται πληροφορίες όπως ο κωδικός, ο πλήρης τίτλος, η κατεύθυνση (εφόσον υπάρχει), η κατηγορία του μαθήματος (π.χ. υποχρεωτικό, επιλογής κ.λπ.), οι διδάσκοντες που το υποστηρίζουν, καθώς και ένδειξη εάν πρόκειται για μάθημα που προσφέρεται από άλλο Τμήμα.

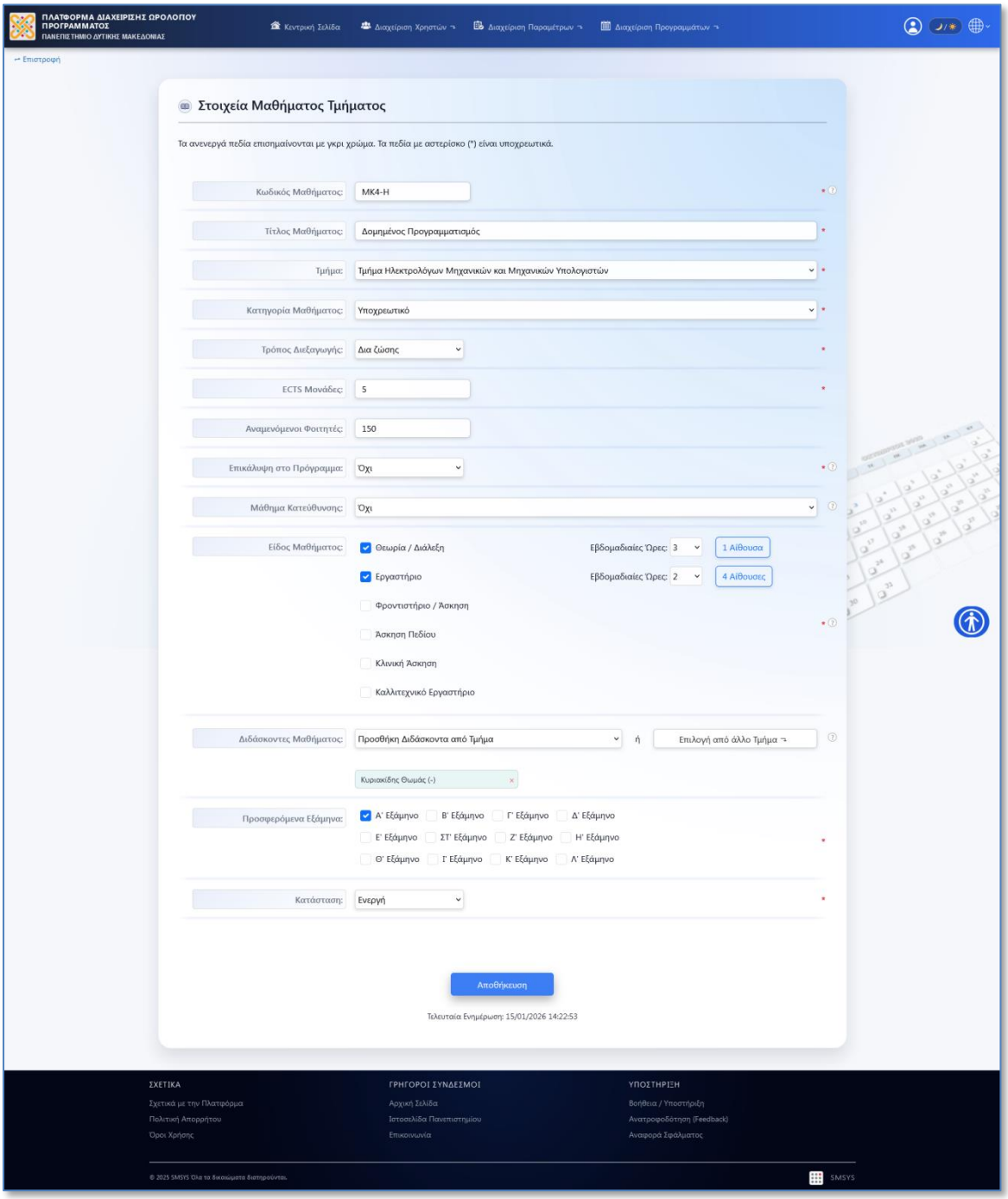


Εικόνα 12 – Λίστα Μαθημάτων ανά Τμήμα και Εξάμηνο

Από τη συγκεκριμένη λίστα, ο χρήστης μπορεί να προβάλει ή να επεξεργαστεί τα στοιχεία κάθε μαθήματος, επιλέγοντάς το με κλικ στον τίτλο του μαθήματος.

Προβολή και Επεξεργασία Μαθήματος:

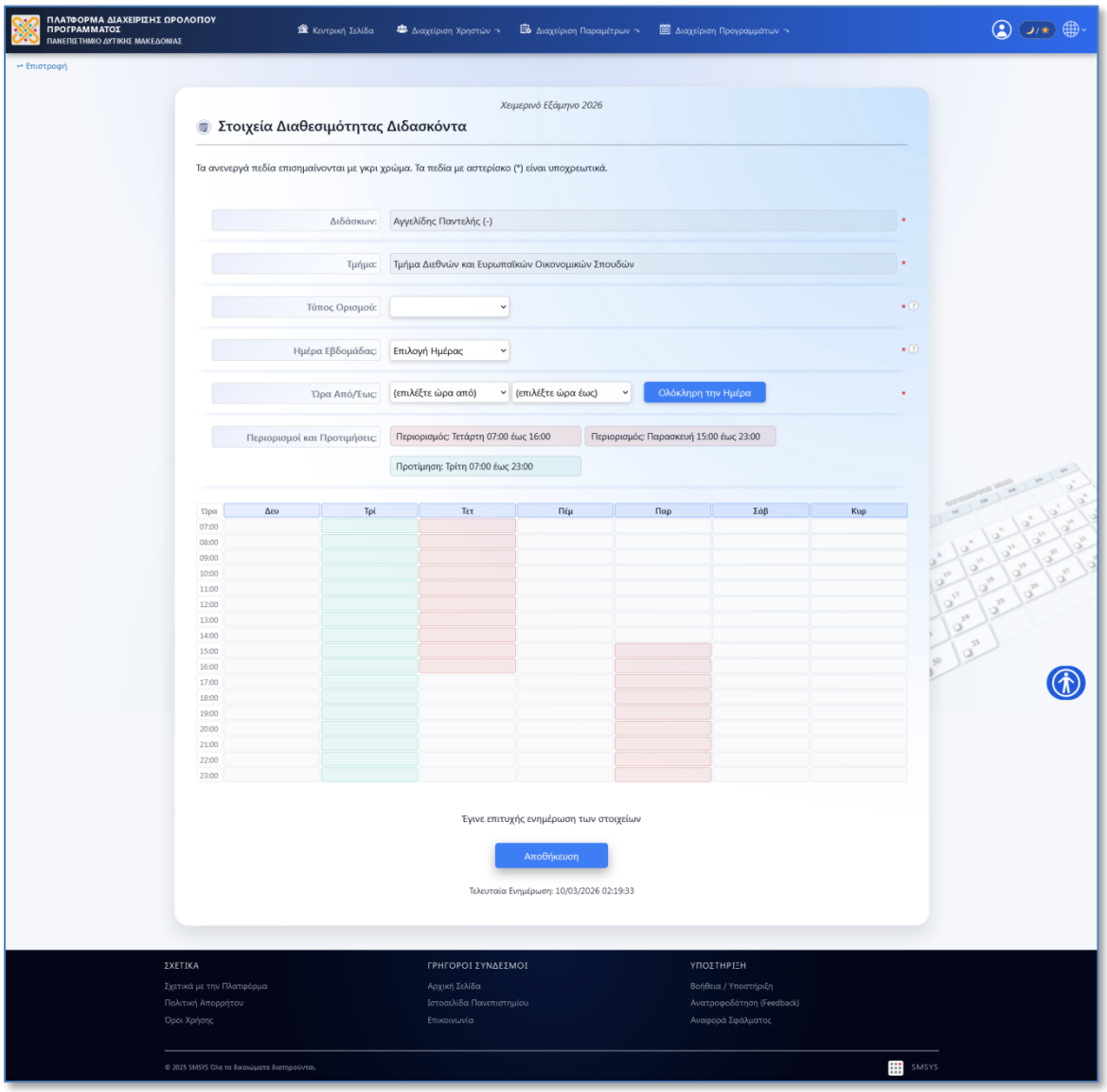
Η σελίδα παρουσιάζει τη φόρμα προβολής και επεξεργασίας των στοιχείων ενός μαθήματος. Περιλαμβάνονται όλα τα απαραίτητα πεδία τόσο για την ορθή προβολή του μαθήματος στο ωρολόγιο πρόγραμμα όσο και για την τοποθέτησή του, με τήρηση και έλεγχο των σχετικών περιορισμών κατά την εκπόνηση του προγράμματος.



Εικόνα 13 – Προβολή και Επεξεργασία Στοιχείων Μαθήματος

Προβολή και Επεξεργασία Διαθεσιμότητας Διδάσκοντα:

Μέσω της συγκεκριμένης σελίδας, ο Διαχειριστής μπορεί να ορίσει για κάθε Διδάσκοντα περιορισμούς και προτιμήσεις σχετικά με τις ημέρες και τις ώρες διδασκαλίας. Ο ορισμός των στοιχείων διαθεσιμότητας μπορεί να πραγματοποιηθεί είτε μέσω λίστας επιλογής (ημέρα και χρονικό διάστημα από/έως) είτε μέσω διαδραστικής επιλογής στο εβδομαδιαίο ημερολόγιο, επιλέγοντας τις επιθυμητές ώρες με τη χρήση του ποντικιού. Οι δηλώσεις διαθεσιμότητας αφορούν τον εκάστοτε Διδάσκοντα σε σχέση με συγκεκριμένο Τμήμα, το οποίο επιλέγεται στη φόρμα. Με τον τρόπο αυτό είναι δυνατός ο ορισμός διαφορετικών περιορισμών ή προτιμήσεων για τον ίδιο Διδάσκοντα σε διαφορετικά Τμήματα στα οποία ενδεχομένως διδάσκει.



Εικόνα 14 – Ορισμός Στοιχείων Διαθεσιμότητας Διδάσκοντα

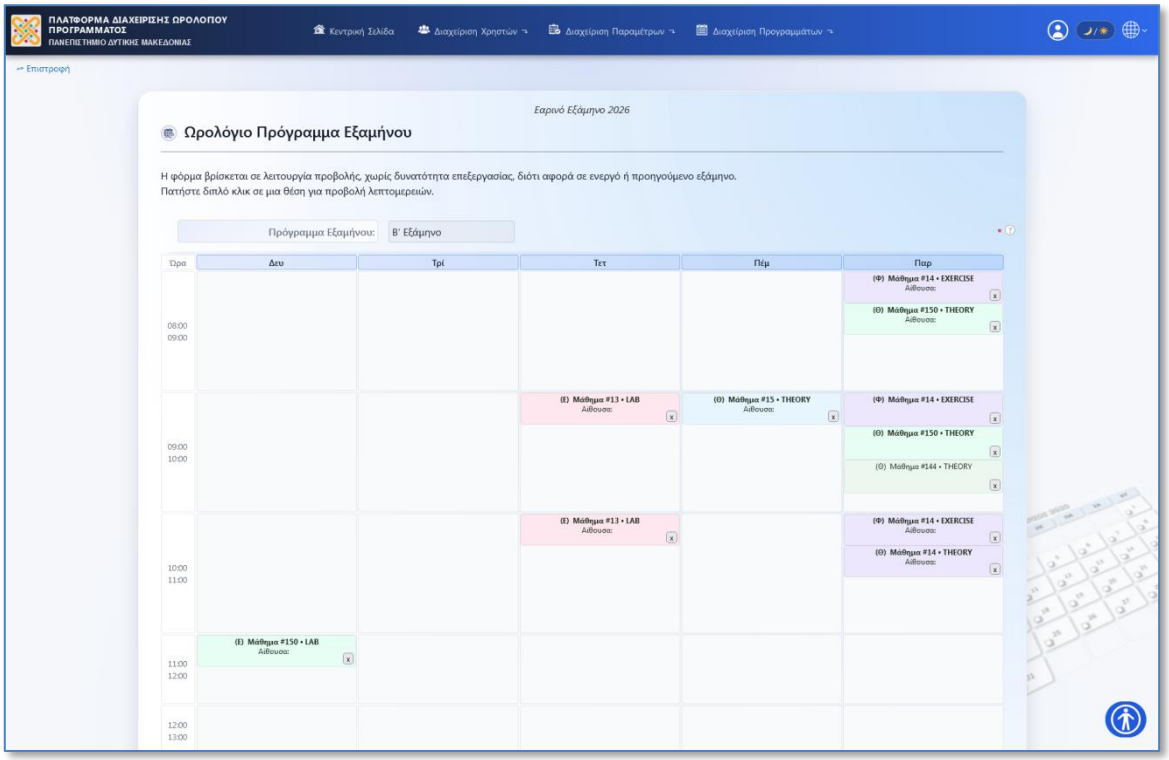
Δημιουργία και Προβολή Ωρολογίου Προγράμματος Εξαμήνου:

Παρακάτω παρουσιάζεται η απεικόνιση του εβδομαδιαίου ημερολογίου κατά τη διαδικασία δημιουργίας του ωρολογίου προγράμματος εξαμήνου ενός Τμήματος. Το ημερολόγιο είναι χωρισμένο σε ωριαίες θέσεις (slots), στις οποίες γίνεται ο ορισμός των μαθημάτων.

Με διπλό κλικ σε μία θέση ανοίγει παράθυρο επιλογής μαθήματος, διδάσκοντα και αίθουσας, ενώ εμφανίζονται σχετικά ενημερωτικά μηνύματα σε περίπτωση επιλογής που δημιουργεί διενέξεις ή παραβιάζει καθορισμένους περιορισμούς.

Το ημερολόγιο είναι διαδραστικό και υποστηρίζει λειτουργίες drag and drop για την αντιγραφή ενός μαθήματος σε νέα χρονική θέση, καθώς και εύκολη διαγραφή με σχετική ερώτηση επιβεβαίωσης.

Η ενημέρωση των δεδομένων πραγματοποιείται δυναμικά σε κάθε ενέργεια μέσω κλήσεων AJAX, αποφεύγοντας τις συνεχείς επαναφορτώσεις της σελίδας και βελτιώνοντας την εμπειρία χρήσης.



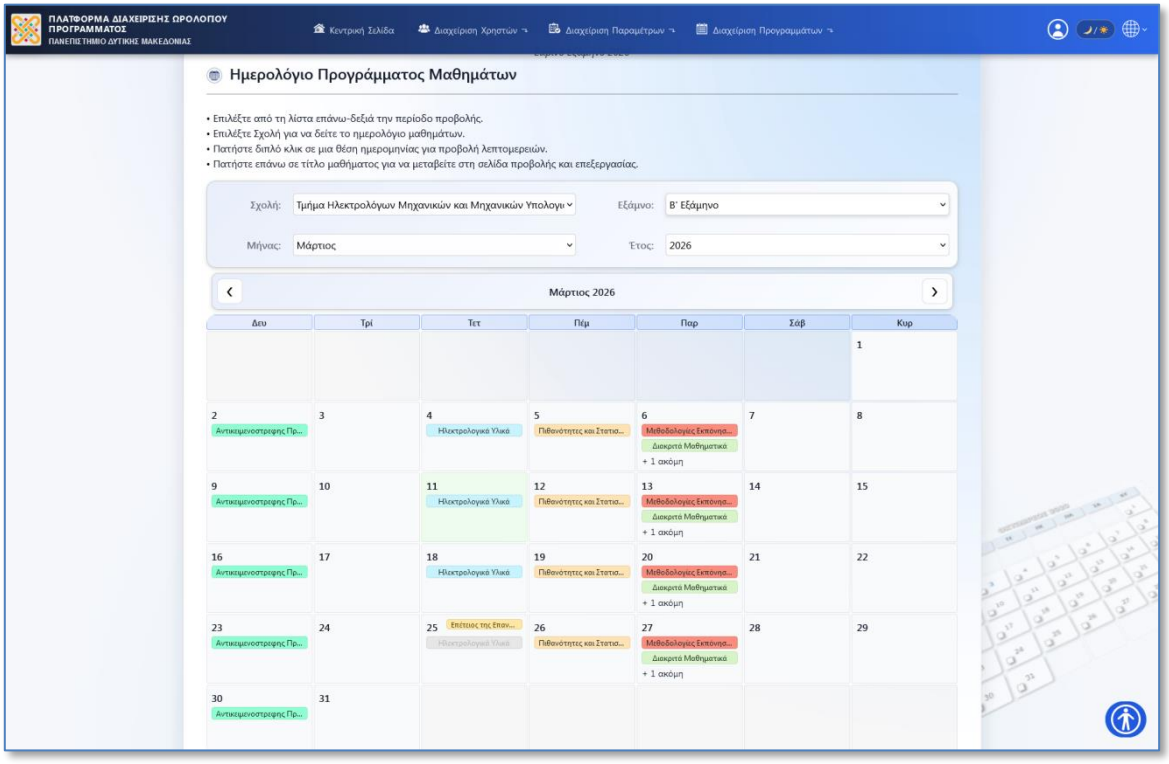
Εικόνα 15 – Δημιουργία Ωρολογίου Προγράμματος Εξαμήνου

Ημερολόγιο Προγράμματος Μαθημάτων:

Η σελίδα περιλαμβάνει μηνιαίο ημερολόγιο προβολής των μαθημάτων ανά Τμήμα και εξάμηνο. Ο χρήστης μπορεί να επιλέξει μήνα και έτος προβολής, ενώ παρέχονται κουμπιά μετακίνησης στον προηγούμενο και στον επόμενο μήνα.

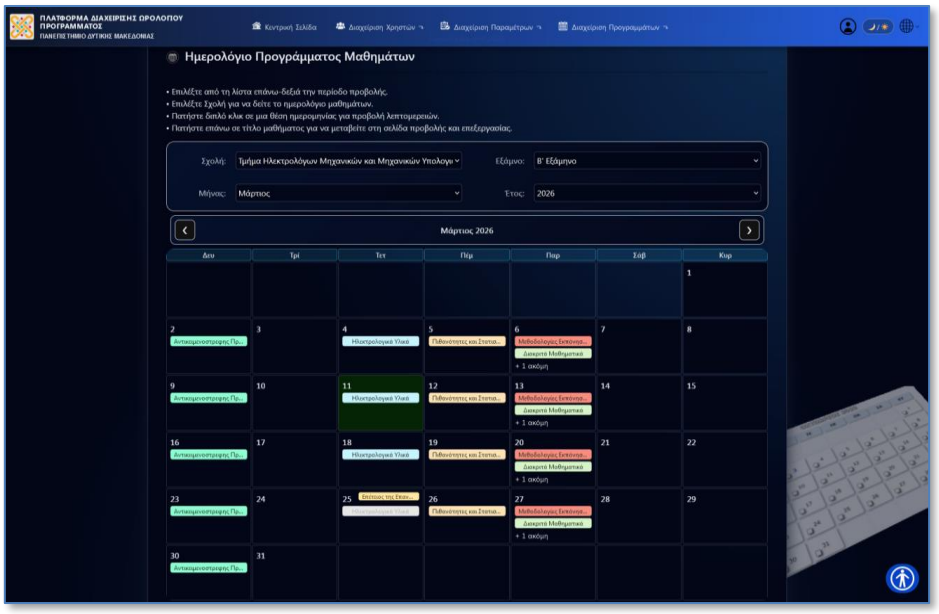
Με διπλό κλικ σε συγκεκριμένη ημερομηνία εμφανίζονται οι λεπτομέρειες των μαθημάτων της ημέρας, ενώ επιλέγοντας κάποιο μάθημα προβάλλεται το ωρολόγιο πρόγραμμα που αντιστοιχεί στη συγκεκριμένη ημέρα της εβδομάδας. Εφόσον έχουν οριστεί επίσημες αργίες από τον Διαχειριστή Τμήματος, αυτές εμφανίζονται στο ημερολόγιο και τα μαθήματα στις αντίστοιχες ημερομηνίες παρουσιάζονται με γκρι χρώμα.

Παρέχεται επίσης δυνατότητα εξαγωγής των δεδομένων σε μορφή PDF και Excel. Μέσω του ημερολογίου, οι Διδάσκοντες και οι Διαχειριστές μπορούν να δηλώνουν έκτακτες αλλαγές, όπως αναβολή ή αναπλήρωση μαθήματος με ορισμό αίθουσας. Με την καταχώριση της αλλαγής ενημερώνεται αυτόματα η δημόσια προβολή του προγράμματος και αποστέλλεται σχετική ειδοποίηση στους Φοιτητές. Επιπλέον, παρέχεται δυνατότητα αναζήτησης με φίλτρα για την προβολή κενών ημερομηνιών και διαθέσιμων αιθουσών.



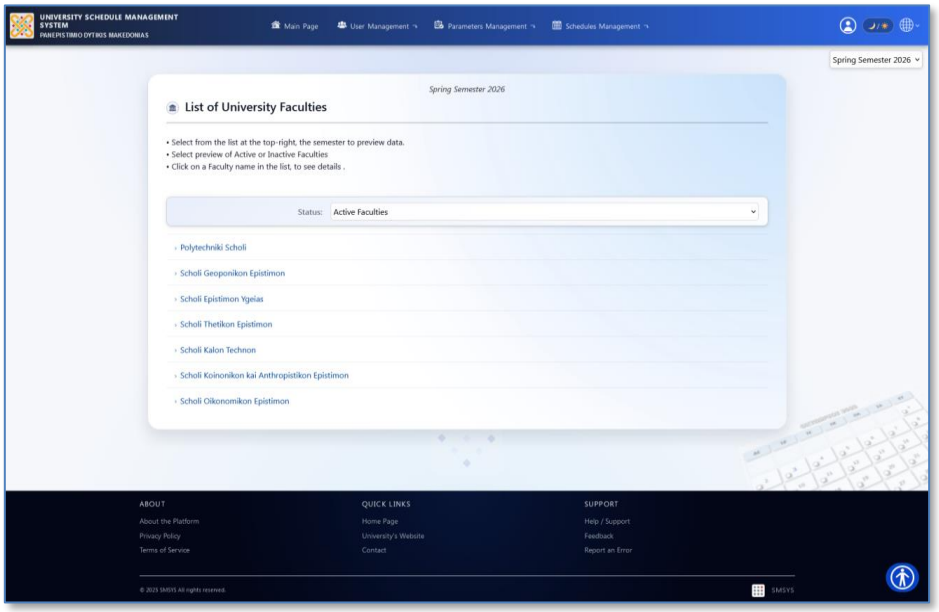
Εικόνα 16 – Ημερολόγιο Προγράμματος Μαθημάτων

Λειτουργία σκοτεινού θέματος (Dark Theme): Η πλατφόρμα παρέχει τη δυνατότητα εναλλαγής μεταξύ φωτεινού και σκοτεινού θέματος, με στόχο τη βελτίωση της εμπειρίας χρήσης και τη διευκόλυνση της πρόσβασης.



Εικόνα 17 – Λειτουργία Σκοτεινού Θέματος (Dark Theme)

Αλλαγή Γλώσσας: Η πλατφόρμα περιλαμβάνει πλήρη μετάφραση στα Αγγλικά, με αυτοματοποιημένο τρόπο (χρήση λεξικού) και παράλληλη μετατροπή μη μεταφρασμένων λέξεων σε λατινικούς χαρακτήρες σύμφωνα με το πρότυπο ΕΛΟΤ 743.



Εικόνα 18 – Αλλαγή Γλώσσας σε Αγγλικά

5.6 Περιορισμοί του Συστήματος

Παρά τη λειτουργική πληρότητα της πλατφόρμας, υπάρχουν ορισμένοι περιορισμοί οι οποίοι σχετίζονται κυρίως με το στάδιο ανάπτυξης και το περιβάλλον δοκιμών του συστήματος.

Αρχικά, οι δοκιμές απόδοσης πραγματοποιήθηκαν κυρίως σε περιβάλλον τοπικής εκτέλεσης, γεγονός που δεν επιτρέπει την πλήρη προσομοίωση συνθηκών μεγάλης κλίμακας, όπως αυτές που θα εμφανίζονταν σε εγκατάσταση με πολύ μεγάλο αριθμό χρηστών.

Επιπλέον, ορισμένες λειτουργίες της εφαρμογής βασίζονται σε σύνθετα ερωτήματα προς τη βάση δεδομένων, τα οποία σε συνθήκες ιδιαίτερα υψηλού φόρτου ενδεχομένως να απαιτήσουν περαιτέρω βελτιστοποίηση ή χρήση πρόσθετων μηχανισμών προσωρινής αποθήκευσης δεδομένων (caching).

Οι περιορισμοί που επιβάλλονται από τις παρεχόμενες υπηρεσίες των περιβαλλόντων shared hosting, ως προς τη δυνατότητα εγκατάστασης και χρήσης επεκτάσεων caching, όπως το Memcache[31] ή το APCu[32], οδήγησαν στην ανάπτυξη προσαρμοσμένου μηχανισμού προσωρινής αποθήκευσης δεδομένων εντός της εφαρμογής. Ο μηχανισμός αυτός χρησιμοποιείται επιλεκτικά, καθώς παρουσιάζει περιορισμούς σε περιπτώσεις πολύπλοκων ερωτημάτων που περιλαμβάνουν δεδομένα από πολλαπλούς πίνακες ή που ενημερώνονται συχνά.

Τέλος, η ανάπτυξη της πλατφόρμας επικεντρώθηκε κυρίως στη λειτουργική πληρότητα και στην αρχιτεκτονική οργάνωση του συστήματος, ενώ ορισμένες δευτερεύουσες λειτουργίες της διεπαφής χρήστη θα μπορούσαν να βελτιωθούν περαιτέρω σε μελλοντικές εκδόσεις.

[31] https://www.php.net/manual/en/book.memcache.php
[32] https://www.php.net/manual/en/book.apcu.php

5.7 Σύνοψη Κεφαλαίου

Συνολικά, τα αποτελέσματα των δοκιμών δείχνουν ότι η πλατφόρμα παρουσιάζει σταθερή λειτουργία, ικανοποιητική απόδοση και καλή ποιότητα κώδικα. Η αρχιτεκτονική του συστήματος επιτρέπει την επεκτασιμότητα και τη συντήρηση της εφαρμογής, ενώ οι δοκιμές απόδοσης και οι μετρικές του κώδικα επιβεβαιώνουν την ορθότητα της σχεδίασης και την αποτελεσματικότητα της υλοποίησης για το προτεινόμενο έργο.

Στο επόμενο κεφάλαιο παρουσιάζονται οι προτεινόμενες μελλοντικές επεκτάσεις, η ανάλυση S.W.O.T., τα τελικά συμπεράσματα της εργασίας, καθώς και συνοπτική αποτίμηση της συμβολής της προτεινόμενης πλατφόρμας.

6. Συμπεράσματα και Προτάσεις για Μελλοντικές Επεκτάσεις

6.1 S.W.O.T Ανάλυση

Η ανάλυση SWOT χρησιμοποιείται για την αποτύπωση των βασικών χαρακτηριστικών ενός πληροφοριακού συστήματος, εξετάζοντας τόσο εσωτερικούς όσο και εξωτερικούς παράγοντες που επηρεάζουν τη λειτουργία και την εξέλιξή του. Στην παρούσα υποενότητα αξιολογείται το σύστημα διαχείρισης ωρολογίου προγράμματος που αναπτύχθηκε, λαμβάνοντας υπόψη τις υφιστάμενες δυνατότητες, τους περιορισμούς, καθώς και τις προοπτικές περαιτέρω ανάπτυξης.

6.1.1 Strengths (Δυνατά Σημεία)

Το σύστημα είναι πλήρως προσαρμοσμένο στις ανάγκες διαχείρισης ωρολογίου προγράμματος σε ακαδημαϊκό περιβάλλον, υποστηρίζοντας εξάμηνα, κατευθύνσεις και διαφορετικούς τύπους μαθημάτων (θεωρία, εργαστήριο κ.λπ.).

Υλοποιείται σαφής διαχωρισμός ρόλων χρηστών (διαχειριστής, καθηγητής, φοιτητής), με κατάλληλο περιορισμό ενεργειών ανά ρόλο και ασφαλή διαχείριση πρόσβασης. Παράλληλα, περιλαμβάνονται ενσωματωμένοι έλεγχοι εγκυρότητας που αποτρέπουν συγκρούσεις, όπως επικαλύψεις ωρών, διπλές αναθέσεις διδασκόντων και ταυτόχρονη χρήση αιθουσών.

Παρέχεται δυναμική απεικόνιση του προγράμματος τόσο σε μορφή εβδομαδιαίου πίνακα όσο και σε μορφή ημερολογίου, καλύπτοντας διαφορετικές ανάγκες προβολής. Υποστηρίζονται πολλαπλά φίλτρα (ανά εξάμηνο, κατεύθυνση, καθηγητή, αίθουσα), επιτρέποντας στο χρήστη στοχευμένη αναζήτηση και προβολή δεδομένων. Υπάρχει δυνατότητα επιλογής «Τα Μαθήματά μου» (watchlist), προσφέροντας εξατομικευμένη εμπειρία χρήσης, ιδιαίτερα για φοιτητές.

Υποστηρίζεται εξαγωγή δεδομένων σε διάφορες μορφές όπως PDF, Excel και iCalendar, διευκολύνοντας την εκτύπωση και την περαιτέρω επεξεργασία του προγράμματος. Η διεπαφή χρήστη είναι διαδραστική, με χρήση δυναμικών στοιχείων (modals, tags, δυναμική ενημέρωση σε πραγματικό χρόνο), επιτρέποντας άμεση πρόσβαση σε πληροφορίες χωρίς επαναφόρτωση της σελίδας.

Παρέχεται στους χρήστες άμεση ενημέρωση για τροποποιήσεις του ημερολογιακού προγράμματος, επιτρέποντας την έγκαιρη προσαρμογή τους σε αλλαγές όπως μεταθέσεις, αναβολές ή αναπληρώσεις μαθημάτων.

Για τους διδάσκοντες, το σύστημα προσφέρει αυξημένη αμεσότητα στη διαχείριση των μαθημάτων τους, παρέχοντας εύκολο εντοπισμό των προγραμματισμένων διαλέξεων και δυνατότητα γρήγορης καταχώρισης αλλαγών (όπως αναβολές, μεταθέσεις και αναπληρώσεις), με ταυτόχρονη άμεση αποτύπωση των αλλαγών στο δημοσιοποιημένο πρόγραμμα.

Επιπλέον, το σύστημα περιλαμβάνει υποστηρικτικό μηχανισμό αυτόματης παραγωγής προτεινόμενων ωρολογίων προγραμμάτων, βασισμένο σε γενετικό αλγόριθμο. Ο μηχανισμός αυτός λειτουργεί συμπληρωματικά προς τη χειροκίνητη διαχείριση, παρέχοντας αρχικές προτάσεις κατανομής μαθημάτων και μειώνοντας τον χρόνο δημιουργίας προγράμματος.

6.1.2 Weaknesses (Αδυναμίες)

Η δημιουργία και τροποποίηση του ωρολογίου προγράμματος βασίζεται κυρίως στη διαδραστική παρέμβαση του χρήστη, με υποστηρικτική αξιοποίηση αυτοματοποιημένων προτάσεων, οι οποίες λειτουργούν συμπληρωματικά και όχι ως πλήρως αυτόνομος μηχανισμός σχεδιασμού.

Η διαχείριση πολύπλοκων περιπτώσεων, όπως ο διαχωρισμός εργαστηριακών μαθημάτων σε πολλαπλά τμήματα, βρίσκεται υπό περαιτέρω επέκταση σε επίπεδο μοντελοποίησης και απαιτεί επιπλέον χειρισμούς.

Η ενημέρωση των χρηστών υποστηρίζεται μέσω εσωτερικών μηχανισμών ειδοποιήσεων και επιλεκτικά μέσω ηλεκτρονικού ταχυδρομείου για συγκεκριμένες λειτουργίες (όπως επαναφορά κωδικού πρόσβασης, ενεργοποίηση λογαριασμού και διαδικασίες ταυτοποίησης). Ωστόσο, δεν έχει ακόμη επεκταθεί πλήρως σε αυτοματοποιημένες ενημερώσεις μέσω email για αλλαγές προγράμματος, ανακοινώσεις ή μαζικές ειδοποιήσεις.

Η άμεση διασύνδεση με άλλα συστήματα Πανεπιστημιακού Ιδρύματος (π.χ. δηλώσεις μαθημάτων, e-classes πλατφόρμες) δεν έχει ακόμη ολοκληρωθεί πλήρως.

Η πλατφόρμα περιλαμβάνει υποστηρικτική αρχιτεκτονική για Single Sign-On (SSO) και έχει υλοποιηθεί δοκιμαστικός μηχανισμός προσομοίωσης, ωστόσο η πλήρης διασύνδεση με τα

κεντρικά συστήματα ταυτοποίησης του εκάστοτε Πανεπιστημιακού Ιδρύματος δεν έχει ολοκληρωθεί, καθώς απαιτείται πρόσβαση και αντιστοίχιση με τα πραγματικά δεδομένα και πεδία των αντίστοιχων υπηρεσιών αυθεντικοποίησης.

6.1.3 Opportunities (Ευκαιρίες)

Η περαιτέρω εξέλιξη του ενσωματωμένου μηχανισμού αυτόματης δημιουργίας και βελτιστοποίησης ωρολογίου προγράμματος μπορεί να μειώσει σημαντικά τον χειροκίνητο φόρτο εργασίας και να βελτιώσει την ποιότητα των παραγόμενων λύσεων.

Η πλήρης υλοποίηση SSO (βλ. 3.2.7) και η σύνδεση με την κεντρική βάση χρηστών του ιδρύματος θα απλοποιήσει τη διαδικασία ταυτοποίησης και θα ενισχύσει την ασφάλεια.

Η διασύνδεση με άλλα πληροφοριακά συστήματα (π.χ. πλατφόρμες μαθημάτων, δηλώσεις φοιτητών) μπορεί να οδηγήσει σε ενοποιημένη διαχείριση ακαδημαϊκών δεδομένων.

Η επέκταση της λειτουργικότητας των ειδοποιήσεων (π.χ. email alerts για αλλαγές προγράμματος) θα βελτιώσει την άμεση ενημέρωση των χρηστών.

Η ανάπτυξη mobile εφαρμογής ή περαιτέρω mobile βελτιστοποίηση μπορεί να ενισχύσει την προσβασιμότητα και τη χρηστικότητα του συστήματος.

Η δυνατότητα υιοθέτησης του συστήματος από άλλα ιδρύματα μπορεί να συμβάλει στην περαιτέρω εξέλιξη και βελτίωσή του.

6.1.4 Threats (Απειλές)

Απαιτείται συνεχής συντήρηση και επικαιροποίηση ώστε το σύστημα να προσαρμόζεται στις μεταβαλλόμενες ανάγκες του ακαδημαϊκού περιβάλλοντος.

Η πιθανή αντίσταση χρηστών (π.χ. διοικητικό προσωπικό ή διδάσκοντες) σε νέα ψηφιακά εργαλεία μπορεί να καθυστερήσει την πλήρη υιοθέτηση.

Η ύπαρξη ολοκληρωμένων πληροφοριακών συστημάτων σε πανεπιστημιακό επίπεδο, τα οποία ενσωματώνουν πολλαπλές λειτουργίες (όπως διαχείριση μητρώου φοιτητών, βαθμολογίες και πλατφόρμες e-classes), ενδέχεται να περιορίσει την ανάγκη υιοθέτησης ενός αυτόνομου συστήματος διαχείρισης ωρολογίου προγράμματος. Επιπλέον, σε περιπτώσεις όπου το ίδρυμα διαθέτει ήδη ενιαία πλατφόρμα διαχείρισης ακαδημαϊκών διαδικασιών, η

ενσωμάτωση νέου ανεξάρτητου συστήματος ενδέχεται να παρουσιάσει δυσκολίες ως προς τη διαλειτουργικότητα και την αποδοχή από τους χρήστες.

Ζητήματα ασφάλειας, όπως η προστασία προσωπικών δεδομένων και η διαχείριση πρόσβασης, αποτελούν κρίσιμο παράγοντα για την αξιοπιστία της εφαρμογής.

6.2 Τελικά Συμπεράσματα

Η παρούσα εργασία οδήγησε στην ανάπτυξη μιας ολοκληρωμένης πλατφόρμας διαχείρισης ωρολογίου προγράμματος Πανεπιστημίου, η οποία καλύπτει τις βασικές ανάγκες ενός ακαδημαϊκού περιβάλλοντος, εκσυγχρονίζοντας και αυτοματοποιώντας επιμέρους λειτουργίες. Το σύστημα συνδυάζει ευελιξία στη διαχείριση, διαδραστική διεπαφή χρήστη και δυνατότητες εξαγωγής και διασύνδεσης δεδομένων.

Ιδιαίτερη έμφαση δόθηκε στην ευελιξία διαχείρισης και στην υποστήριξη διαδικασιών δημιουργίας και τροποποίησης του ωρολογίου προγράμματος. Η πλατφόρμα παρέχει αυτοματοποιημένους ελέγχους συγκρούσεων και προειδοποιήσεων κατά τη διαδικασία δημιουργίας προγράμματος, συμβάλλοντας στον εντοπισμό προβλημάτων που σχετίζονται με μαθήματα, αίθουσες, διδάσκοντες και χρονικές επικαλύψεις. Παράλληλα, υποστηρίζονται λειτουργίες αναζήτησης και φιλτραρίσματος, όπως εύρεση κενών ωρών και διαθέσιμων αιθουσών, διευκολύνοντας τη διαχείριση και την πραγματοποίηση έκτακτων αλλαγών.

Επιπλέον, υποστηρίζεται μηχανισμός αυτόματης δημιουργίας ωρολογίου προγράμματος με χρήση γενετικού αλγορίθμου, ο οποίος λειτουργεί υποστηρικτικά προς τον χρήστη και συμβάλλει στη μείωση του χρόνου σχεδιασμού.

Παράλληλα, δόθηκε ιδιαίτερη προσοχή σε ζητήματα ασφάλειας, προσβασιμότητας και ιστορικότητας δεδομένων, μέσω της εφαρμογής μηχανισμών ελέγχου πρόσβασης, διαχείρισης εκδόσεων δεδομένων (versioning) και υποστήριξης προσαρμοζόμενης και εύχρηστης διεπαφής χρήστη. Η δυνατότητα εξατομικευμένης ενημέρωσης των χρηστών και η άμεση δημοσιοποίηση αλλαγών ενισχύουν τη χρηστικότητα και τη λειτουργική αξία της πλατφόρμας για το ακαδημαϊκό περιβάλλον.

6.3 Αποτίμηση της Συμβολής της Προτεινόμενης Πλατφόρμας

Η συμβολή της προτεινόμενης πλατφόρμας έγκειται στην παροχή ενός ευέλικτου και επεκτάσιμου εργαλείου διαχείρισης ωρολογίου προγράμματος, το οποίο συνδυάζει λειτουργικότητα, ευχρηστία, επεκτασιμότητα και δυνατότητες αυτοματοποίησης. Σε αντίθεση με απλές λύσεις παρουσίασης προγράμματος, το σύστημα υποστηρίζει ενεργή διαχείριση, τροποποίηση, έλεγχο συγκρούσεων και παρακολούθηση αλλαγών, καθώς και ενσωμάτωση υποστηρικτικών μηχανισμών βελτιστοποίησης και αυτόματης δημιουργίας προγράμματος.

Η υιοθέτηση δομημένων δεδομένων και διεπαφών επικοινωνίας (API), όπως παρουσιάστηκε στην υποενότητα 3.2.7, επιτρέπει τη διασύνδεση με άλλα πληροφοριακά συστήματα και την αξιοποίηση των δεδομένων της πλατφόρμας σε διαφορετικά περιβάλλοντα χρήσης, ενισχύοντας τη διαλειτουργικότητα και τη βιωσιμότητα του συστήματος.

Παράλληλα, η έμφαση στη διαδραστική εμπειρία χρήστη, στη δυνατότητα παρακολούθησης αλλαγών και στην υποστήριξη αυτοματοποίησης συμβάλλει στη δημιουργία ενός ολοκληρωμένου και πρακτικά αξιοποιήσιμου πληροφοριακού συστήματος για ακαδημαϊκή χρήση.

6.4 Προτεινόμενες μελλοντικές επεκτάσεις

Η αρχιτεκτονική της πλατφόρμας σχεδιάστηκε με στόχο την επεκτασιμότητα και την προσαρμογή σε διαφορετικά ακαδημαϊκά περιβάλλοντα. Στο πλαίσιο αυτό μπορούν να προταθούν ορισμένες πιθανές μελλοντικές επεκτάσεις του συστήματος.

Μία σημαντική επέκταση αφορά την άμεση διασύνδεση της πλατφόρμας με υφιστάμενα πληροφοριακά συστήματα των Πανεπιστημίων, όπως συστήματα διαχείρισης μητρώου φοιτητών και μητρώα μαθημάτων. Η διασύνδεση αυτή μπορεί να υλοποιηθεί μέσω εξωτερικών διεπαφών προγραμματισμού εφαρμογών (RESTful APIs[33]) , οι οποίες αποτελούν έναν τυποποιημένο τρόπο επικοινωνίας μεταξύ διαφορετικών συστημάτων μέσω διαδικτύου. Μέσω των διεπαφών αυτών καθίσταται δυνατή η αυτοματοποιημένη ανταλλαγή δεδομένων σε πραγματικό χρόνο, μειώνοντας σημαντικά την ανάγκη χειροκίνητης καταχώρησης και ελαχιστοποιώντας τα σφάλματα συγχρονισμού.

Η πλατφόρμα ήδη διαθέτει μηχανισμούς εισαγωγής δεδομένων από αρχεία CSV ή Excel, καθώς και μια βασική εσωτερική διεπαφή (internal API) για την αποστολή δεδομένων προς άλλα συστήματα. Η υπάρχουσα λειτουργικότητα θα μπορούσε να επεκταθεί περαιτέρω ώστε να υποστηρίζει πλήρη αμφίδρομη επικοινωνία, επιτρέποντας όχι μόνο την αποστολή αλλά και τη λήψη δεδομένων, καθώς και τον συνεχή συγχρονισμό μεταξύ των εμπλεκόμενων πληροφοριακών συστημάτων.

Μια ακόμη προτεινόμενη επέκταση συνιστά η βελτίωση και η ενίσχυση της προσβασιμότητας, μέσω της ενσωμάτωσης επιπρόσθετων λειτουργιών που θα διευκολύνουν τη χρήση της πλατφόρμας από ευρύτερο φάσμα χρηστών. Η βελτίωση αυτή θα μπορούσε να επιτευχθεί είτε με τη διασύνδεση και χρήση εξωτερικών εργαλείων υποστήριξης προσβασιμότητας είτε με την περαιτέρω επέκταση και βελτίωση του ήδη υπάρχοντος βασικού πλαισίου.

Η πλατφόρμα ήδη αξιοποιεί βασικές τεχνικές βελτιστοποίησης και προσωρινής αποθήκευσης (caching, βλ. υποενότητα 4.1.6) για τη βελτίωση της απόδοσης. Παρ’ όλα αυτά, η περαιτέρω επέκταση των μηχανισμών αυτών σε μεγαλύτερη κλίμακα (π.χ. σε επίπεδο δεδομένων, queries ή και κατανεμημένα συστήματα caching) θα μπορούσε να ενισχύσει σημαντικά την

[33] Οι RESTful APIs (Representational State Transfer) αποτελούν έναν ευρέως χρησιμοποιούμενο τρόπο επικοινωνίας μεταξύ πληροφοριακών συστημάτων μέσω του διαδικτύου, βασισμένο σε HTTP αιτήματα (π.χ. GET, POST), επιτρέποντας την ανταλλαγή δεδομένων σε μορφές όπως JSON ή XML.

αποδοτικότητα της πλατφόρμας, ιδιαίτερα σε περιβάλλοντα με μεγάλο αριθμό ταυτόχρονων χρηστών και αυξημένες απαιτήσεις προσπέλασης δεδομένων.

Παρότι η πλατφόρμα έχει ήδη σχεδιαστεί με προσαρμοστική διεπαφή (responsive design) και υποστήριξη για φορητές συσκευές (mobile-friendly), μία σημαντική μελλοντική επέκταση είναι η ανάπτυξη εγγενών εφαρμογών (native applications) για Android και iOS. Μία τέτοια υλοποίηση θα μπορούσε να προσφέρει βελτιωμένη εμπειρία χρήσης, ταχύτερη απόκριση και δυνατότητες όπως ειδοποιήσεις σε πραγματικό χρόνο για αλλαγές στο πρόγραμμα. Ωστόσο, η ανάπτυξη native εφαρμογών συνεπάγεται αυξημένες απαιτήσεις συντήρησης και την ανάγκη υλοποίησης πιο ολοκληρωμένων APIs για την αποτελεσματική επικοινωνία με το κεντρικό σύστημα.

Τέλος, η πλατφόρμα περιλαμβάνει ήδη υποστηρικτικό μηχανισμό επίλυσης (solver) για την αυτόματη δημιουργία και βελτιστοποίηση του ωρολογίου προγράμματος, βασισμένο σε γενετικό αλγόριθμο. Ο μηχανισμός αυτός λαμβάνει υπόψη πολλαπλούς περιορισμούς, όπως διαθεσιμότητα αιθουσών, ωράρια διδασκόντων, τύπους μαθημάτων και επικαλύψεις, και παράγει προτεινόμενες λύσεις που μπορεί να επεξεργαστεί περαιτέρω ο χρήστης. Μελλοντική επέκταση του μηχανισμού αυτού μπορεί να αποτελέσει η βελτίωση της απόδοσης και της ποιότητας των παραγόμενων λύσεων, καθώς και η ενσωμάτωση εναλλακτικών ή υβριδικών μεθόδων βελτιστοποίησης.

Βιβλιογραφία

[1] OECD. Country Digital Education Ecosystems and Governance: A Companion to Digital Education Outlook 2023. OECD Publishing; 2023. https://doi.org/10.1787/906134d4-en.

[2] Υπουργείο Παιδείας και Θρησκευμάτων. Διακήρυξη Ηλεκτρονικό Πανεπιστήμιο: Ψηφιακές Υπηρεσίες Ακαδημαϊκών Ιδρυμάτων - Παράρτημα Ι 2022. https://www.minedu.gov.gr/publications/docs2020/2886_14.09.2022_τεύχος_e-univer_τελικο_με_ΑΔΑΜ.pdf (accessed October 19, 2025).

[3] Εθνικό Μετσόβιο Πολυτεχνείο. Ωρολόγιο πρόγραμμα – Σχολή πολιτικών μηχανικών 2025. https://www.civil.ntua.gr/wrologio-programma/ (accessed October 6, 2025).

[4] Πανεπιστήμιο Πατρών. Τμήμα Μαθηματικών – Ανακοινώσεις και ωρολόγιο πρόγραμμα 2025. https://www.math.upatras.gr/el/ (accessed October 6, 2025).

[5] Πανεπιστήμιο Δυτικής Αττικής. Τμήμα επιστήμης και τεχνολογίας τροφίμων – Ωρολόγιο πρόγραμμα εξεταστικής 2025. https://fst.uniwa.gr/category/orologio-programma-examinoy/ (accessed October 6, 2025).

[6] Πανεπιστήμιο Θεσσαλίας. Τμήμα Πληροφορικής και Τηλεπικοινωνιών – Ωρολόγιο πρόγραμμα μαθημάτων ΧΕ 2025-2026 2025. https://dit.uth.gr/?p=41753 (accessed October 15, 2025).

[7] Gaebel, M., Morrisroe, A. The future of digitally enhanced learning and teaching in european higher education institutions. European University Association absl.; 2023.

[8] Βεσκούκης, Β. Αρχές τεχνολογίας λογισμικού, Τόμος Α: Τεχνολογία λογισμικού Ι. Ελληνικό Ανοικτό Πανεπιστήμιο; 2000.

[9] Boehm, B. A spiral model of software development and enhancement. ACM SIGSOFT Softw Eng Notes 1986;11:14–24. https://doi.org/10.1145/12944.12948.

[10] Pressman, Roger S. Software engineering: a practitioner's approach. 7th ed. Dubuque, IA: McGraw-Hill; 2010.

[11] Sommerville, I. Software engineering. Tenth edition. Boston Columbus Indianapolis New York San Francisco Hoboken Amsterdam Cape Town Dubai London: Pearson; 2016.

[12] PHP Documentation Group. PHP manual [Online documentation] 2025. https://www.php.net/manual/en/ (accessed October 19, 2025).

[13] World Wide Web Consortium (W3C). HTML5: A vocabulary and associated APIs for HTML and XHTML 2017. https://www.w3.org/TR/html5/ (accessed September 7, 2025).

[14] World Wide Web Consortium (W3C). CSS Cascading Style Sheets Level 3 (W3C Recommendation) 2023. https://www.w3.org/Style/CSS/ (accessed September 19, 2025).

[15] ECMAScript® Language Specification. Ecma International; 2025.

[16] Bray, T. The JavaScript Object Notation (JSON) Data Interchange Format. IETF (Internet Engineering Task Force); 2017.

[17] Bootstrap Authors. Bootstrap 5 Documentation 2021. https://getbootstrap.com/docs/5.0/getting-started/introduction/ (accessed September 19, 2025).

[18] MySQL reference manual [Online documentation]. Oracle Corporation; 2025.

[19] MariaDB Foundation. MariaDB Knowledge Base. MariaDB Doc 2025. https://mariadb.com/kb/en/ (accessed October 22, 2025).

[20] WHATWG. HTML Living Standard. WHATWG n.d. https://html.spec.whatwg.org/ (accessed September 1, 2025).

[21] Microsoft Corporation. Visual Studio Code Documentation. Vis Studio Code n.d. https://code.visualstudio.com/docs (accessed September 10, 2025).

[22] Don Ho. Notepad++ Official Website n.d. https://notepad-plus-plus.org/ (accessed September 2, 2025).

[23] phpMyAdmin Team. phpMyAdmin Documentation. phpMyAdmin n.d. https://www.phpmyadmin.net/docs/ (accessed September 2, 2025).

[24] Microsoft Corporation. Microsoft OneDrive. Microsoft n.d. https://onedrive.live.com (accessed September 1, 2025).

[25] GitHub, Inc. GitHub. GitHub n.d. https://github.com/ (accessed September 4, 2025).

[26] Microsoft Corporation. IIS Express. Microsoft Docs n.d. https://learn.microsoft.com/en-us/iis/extensions/introduction-to-iis-express/iis-express-overview (accessed September 15, 2025).

[27] Apache Friends. XAMPP. Apache Friends n.d. https://www.apachefriends.org/ (accessed September 15, 2025).

[28] Apache Software Foundation. Apache HTTP Server Project. Apache n.d. https://httpd.apache.org/ (accessed September 15, 2025).

[29] Visual Paradigm International. Visual Paradigm. Vis Paradigm n.d. https://www.visual-paradigm.com/ (accessed October 22, 2025).

[30] Λεβέντη, Ε. Μελέτη και Κατασκευή Πληροφοριακού Συστήματος για την Εξαγωγή Προγράμματος Διδασκαλίας του Τμήματος Μηχανικών Πληροφορικής και Τηλεπικοινωνιών του Πανεπιστημίου Δυτικής Μακεδονίας. Διπλωματική Εργασία. Πανεπιστήμιο Δυτικής Μακεδονίας, 2012.

[31] Ορφανίδου, Ε. Αυτόματη Δημιουργία Ωρολογίων Προγραμμάτων με Χρήση Μεθόδων Υπολογιστικής Νοημοσύνης. Πτυχιακή Εργασία. Πανεπιστήμιο Στερεάς Ελλάδας, 2012.

[32] Σιδέρης, Θ. Διαδικτυακή εφαρμογή για συμμετοχικό ορισμό ωρολόγιου προγράμματος με περιορισμούς. Πτυχιακή Εργασία. Χαροκόπειο Πανεπιστήμιο, 2021.

[33] Στάμος, Κ. Σχεδίαση και Υλοποίηση ιστοσελίδας για διαχείριση προγράμματος Πανεπιστημίου. Πτυχιακή Εργασία. Πανεπιστήμιο Δυτικής Μακεδονίας, 2025.

[34] Antinoos Software Applications. ΩΡΟΛΟΓΙΟ : Σύστημα Σύνταξης Ωρολογίου Προγράμματος 2017.

[35] Müller, T. ITC2007 Solver Description: A Hybrid Approach, 2007.

[36] De Causmaecker, P., Özcan, E., Vanden Berghe, G. Proceedings of the 13th International Conference on the Practice and Theory of Automated Timetabling. vol. Vol. 3. Leuven, Belgium: KU Leuven / PATAT; 2022.

[37] Apereo Foundation. UniTime. UniTime n.d. https://www.unitime.org/ (accessed September 14, 2025).

[38] Liviu Lalescu. FET – Free Timetabling Software. FET Softw n.d. https://lalescu.ro/liviu/fet/ (accessed September 14, 2025).

[39] Celcat. CELCAT Timetabler n.d. https://celcat.com/ (accessed September 14, 2025).

[40] Wise Technologies. Wise Timetable n.d. https://wtimetable.com/ (accessed October 15, 2025).

[41] Colorni, A., Dorigo, M., Maniezzo, V. A genetic algorithm to solve the timetable problem. Comput Optim Appl 1994.

[42] Glover, F. Tabu Search: A Tutorial. Interfaces 1990;20:74–94. https://doi.org/10.1287/inte.20.4.74.

[43] Hansen, P., Mladenović, N., Moreno-Pérez, J. Variable neighbourhood search: methods and applications. Ann Oper Res 2010;175:367–407. https://doi.org/10.1007/s10479-009-0657-6.

[44] Rappos, E., Thiémard, E., Robert, S., Hêche, J.F. A mixed-integer programming approach for solving university course timetabling problems. J Sched 2022;25:391–404. https://doi.org/10.1007/s10951-021-00715-5.

[45] Kembuan O, Rorimpandey GC, Rompas PTD, Paulina J, Runtuwene A. Development of Web based Timetabling System: Proc. 7th Eng. Int. Conf. Educ. Concept Appl. Green Technol., Semarang, Indonesia: SCITEPRESS - Science and Technology Publications; 2018, p. 317–22. https://doi.org/10.5220/0009010403170322.

[46] Romaguera D, Plender-Nabas J, Matias J, Austero L. Development of a Web-based Course Timetabling System based on an Enhanced Genetic Algorithm. Procedia Comput Sci 2024;234:1714–21. https://doi.org/10.1016/j.procs.2024.03.177.

[47] W3C Technical Architecture Group. W3C – Architecture of the World Wide Web, Volume One. W3Org 2004. https://www.w3.org/TR/webarch/ (accessed December 21, 2025).

[48] Fielding R., Ed., Nottingham M., Ed., Reschke J., Ed. IETF – RFC 9110 (HTTP Semantics) 2022. https://www.rfc-editor.org/rfc/rfc9110.html (accessed December 21, 2025).

[49] IETF. IETF – RFC 8446 (TLS 1.3) - The Transport Layer Security (TLS) Protocol Version 1.3. RFC Ed 2018. https://www.rfc-editor.org/rfc/rfc8446.html (accessed December 21, 2025).

[50] IETF. IETF – RFC 5321 – Simple Mail Transfer Protocol. n.d. https://datatracker.ietf.org/doc/html/rfc5321 (accessed December 21, 2025).

[51] MDN Web Docs. Front-end web development n.d. https://developer.mozilla.org/en-US/curriculum/ (accessed December 21, 2025).

[52] MDN Web Docs. MDM – Responsive web design n.d. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout/Responsive_Design (accessed December 21, 2025).

[53] MDN Web Docs. MDM – JavaScript Guide n.d. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide (accessed December 21, 2025).

[54] MDN Web Docs. MDN – Ajax n.d. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/Network_requests (accessed December 21, 2025).

[55] MDN Web Docs. MDN – JSON n.d. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON (accessed December 21, 2025).

[56] MDN Web Docs. MDN – Server-side programming n.d. https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side (accessed December 21, 2025).

[57] MDN Web Docs. MDN – Cookies and Sessions n.d. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Cookies (accessed December 21, 2025).

[58] MDN Web Docs. MDN – Databases n.d. https://developer.mozilla.org/en-US/docs/Glossary/Database (accessed December 21, 2025).

[59] W3C. W3C – Web SQL Database. W3Org n.d. https://www.w3.org/TR/webdatabase/ (accessed December 21, 2025).

[60] OWASP. OWASP – Top Ten Web Application Security Risks. OwaspOrg n.d. https://owasp.org/www-project-top-ten/ (accessed December 21, 2025).

[61] MDN Web Docs. MDM – Security on the web n.d. https://developer.mozilla.org/en-US/docs/Web/Security (accessed December 21, 2025).

[62] Wikipedia. Static program analysis n.d. https://en.wikipedia.org/wiki/Static_program_analysis (accessed December 21, 2025).

[63] MDN Web Docs. MDN – Client-side and server-side testing n.d. https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Testing (accessed December 21, 2025).

[64] PHP Framework Interop Group. PSR-12: Extended Coding Style n.d. https://www.php-fig.org/psr/psr-12/ (accessed December 21, 2025).

[65] Ebert C, Cain J, Antoniol G, Counsell S, Laplante P. Cyclomatic Complexity. IEEE Softw 2016;33:27–9. https://doi.org/10.1109/MS.2016.147.

[66] MDN Web Docs. MDM – Version control n.d. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Version_control (accessed December 21, 2025).

[67] Λυκοθανάσης, Σ. Τεχνητή νοημοσύνη – εφαρμογές, Τόμος Γ': Γενετικοί αλγόριθμοι και εφαρμογές. vol. Γ. Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο; 2001.

[68] Φιτσιλής, Π. Ευέλικτες μέθοδοι διοίκησης και διαχείρισης έργων 2023. https://doi.org/10.57713/KALLIPOS-196.

[69] European Accessibility Act. Προσβασιμότητα Ιστότοπων Και Εφαρμογών 2016. https://www.consilium.europa.eu/el/press/press-releases/2016/07/18/accessible-websites-apps-wide-rules/ (accessed September 18, 2025).

[70] Cohn, M. User stories applied: for agile software development. 18. print. Boston, Mass.: Addison-Wesley; 2013.

[71] Βεσκούκης, Β. Σχεδιασμός λογισμικού, Τόμος Β: Τεχνολογία λογισμικού ΙΙ. Ελληνικό Ανοικτό Πανεπιστήμιο; 2004.

[72] Φιτσιλής, Π. Μελέτη περίπτωσης: Ηλεκτρονικό κατάστημα. Ελληνικό Ανοικτό Πανεπιστήμιο; 2004.

[73] Ξένος, Μ., Χριστοδουλάκης, Δ. Αρχές τεχνολογίας λογισμικού, Τόμος Γ': Βάσεις δεδομένων. Ελληνικό Ανοικτό Πανεπιστήμιο; 2000.

[74] Oracle. MySQL Limits on Table Column Count and Row Size n.d. https://dev.mysql.com/doc/refman/9.4/en/column-count-limit.html#:~:text=The%20internal%20representation%20of%20a,capable%20of%20supporting%20larger%20rows. (accessed December 21, 2025).

Παράρτημα Α: «Ποιοτική αποτύπωση διερεύνησης κατάστασης»

Στόχος & ορισμοί

Στόχος της παρούσας διερεύνησης ήταν η λήψη ενδεικτικών ποιοτικών δεδομένων σχετικά με τις μεθόδους που χρησιμοποιούν τα Τμήματα των Ελληνικών Δημόσιων Πανεπιστημιακών Ιδρυμάτων για την εκπόνηση και παρουσίαση των ωρολογίων προγραμμάτων και εξετάσεων.

Ο όρος «συμβατικά μέσα» χρησιμοποιείται για να περιγράψει εφαρμογές γενικής χρήσης, όπως το Microsoft Excel, το Apache OpenOffice Calc και το Adobe PDF, τα οποία δεν αποτελούν εξειδικευμένα εργαλεία αυτόματης δημιουργίας και διαχείρισης ωρολογίων προγραμμάτων.

Δείγμα

Απεστάλησαν διερευνητικά email σε 35 γραμματείες Τμημάτων που ανήκουν σε 11 διαφορετικά Πανεπιστήμια, από τα οποία και παρελήφθησαν 6 απαντήσεις. Η περίοδος αποστολής και λήψης των απαντήσεων ήταν από 24/07/2025 έως 23/09/2025.

Παράλληλα, πραγματοποιήθηκε αναζήτηση μέσω διαδικτύου σε επίσημους ιστοχώρους 29 Τμημάτων από 12 Πανεπιστήμια, με εξέταση του τρόπο παρουσίασης των ωρολογίων προγραμμάτων τους. Η αναζήτηση πραγματοποιήθηκε από 18/09/2025 έως 06/11/2025.

Η επιλογή των Τμημάτων και για τις δύο μεθόδους έγιναν με τυχαίο τρόπο, με δειγματοληπτική στρατηγική την συμπερίληψη Πανεπιστημίων και Τμημάτων σε αστικά κέντρα αλλά και στην περιφέρεια.

Μέθοδος

Η διερεύνηση βασίστηκε σε:

1. Επικοινωνία μέσω email, με ποσοστό ανταπόκρισης 17%. Το email περιλάμβανε ερωτήσεις σχετικά με τον τρόπο εκπόνησης του ωρολογίου προγράμματος και του προγράμματος εξετάσεων (εάν γίνεται χρήση κάποιου ειδικευμένου ψηφιακού εργαλείου ή μόνο με συμβατικά μέσα), τα πρόσωπα (γραμματεία, διδάσκοντες) που διαμόρφωναν τα προγράμματα, τον τρόπο επικοινωνίας με τους διδάσκοντες και τον τρόπο και τα μέσα παρουσίασης προς τους φοιτητές.

2. Διαδικτυακή έρευνα, με αναζήτηση και προβολή από ιστοχώρους Τμημάτων μέσω των επίσημων διευθύνσεών τους και λήψη των πιο πρόσφατων ωρολογίων προγραμμάτων, με στόχο την καταγραφή του τρόπου δημοσίευσής τους.

Εύρημα

Από τις απαντήσεις των γραμματειών σε email, προέκυψε ότι η πλειονότητα των Τμημάτων εκπονεί τα ωρολόγια προγράμματα χειρωνακτικά, με κύριο εργαλείο το Microsoft Excel ή άλλα παρόμοια υπολογιστικά φύλλα. Η διαδικασία στηρίζεται σε μεγάλο βαθμό στην προσωπική εμπειρία των διοικητικών υπαλλήλων ή/και των διδασκόντων που συμμετέχουν στη διαμόρφωση του προγράμματος.

Όσον αφορά τη δημοσίευση των ωρολογίων προγραμμάτων, η συνηθέστερη πρακτική είναι η ανάρτηση στατικών αρχείων PDF ή Excel στον επίσημο ιστοχώρο των Τμημάτων. Η έρευνα επίσης κατέδειξε ότι σε ελάχιστες περιπτώσεις διατίθεται διαδραστικό ή δυναμικό περιβάλλον παρουσίασης, κάτι που επιβεβαιώνει τη γενικευμένη χρήση μη αυτοματοποιημένων μεθόδων.

Ενδεικτικά παρακάτω κάποιες δημόσιες επίσημες αναρτήσεις προγραμμάτων:

- Σχολή Πολιτικών Μηχανικών στο Εθνικό Μετσόβιο Πολυτεχνείο [3]
- Τμήμα Επιστήμης και Τεχνολογίας Τροφίμων στο Πανεπιστήμιο Δυτικής Αττικής [5]
- Τμήμα Μαθηματικών στο Πανεπιστήμιο Πατρών [4]
- Τμήμα Πληροφορικής και Τηλεπικοινωνιών στο Πανεπιστήμιο Θεσσαλίας [6]

Περιορισμοί

Η παρούσα διερεύνηση χαρακτηρίζεται ως ανεπίσημη και μη στατιστικά αντιπροσωπευτική, καθώς δεν ακολουθεί τυποποιημένη ερευνητική μεθοδολογία, βασίζεται σε τυχαία επιλογή δείγματος και δεν καλύπτει το σύνολο των Ελληνικών ΑΕΙ. Σκοπός της ήταν η ενδεικτική αποτύπωση της κατάστασης σχετικά με τα ψηφιακά μέσα και τις μεθόδους που χρησιμοποιούνται σήμερα από τα δημόσια Πανεπιστημιακά Ιδρύματα για τη διαμόρφωση και παρουσίαση των ωρολογίων προγραμμάτων, και όχι η παραγωγή γενικεύσιμων ποσοτικών συμπερασμάτων.

Παράρτημα Β: «Οδηγίες εγκατάστασης»

1. Κωδικοποίηση Βάσης Δεδομένων

Το σύστημα χρησιμοποιεί κωδικοποίηση χαρακτήρων utf8mb4_general_ci. Για την ορθή λειτουργία της εφαρμογής και την αποφυγή αλλοιώσεων χαρακτήρων κατά την αποθήκευση και ανάκτηση δεδομένων, η Βάση Δεδομένων θα πρέπει να δημιουργηθεί χρησιμοποιώντας την συγκεκριμένη κωδικοποίηση.

Η ρύθμιση αυτή θα πρέπει να ισχύει:

- σε επίπεδο βάσης δεδομένων (database)
- σε επίπεδο πινάκων (tables)
- σε επίπεδο προβολών (views)

Η χρήση διαφορετικής κωδικοποίησης ενδέχεται να προκαλέσει προβλήματα στην αποθήκευση ή εμφάνιση ελληνικών και ειδικών χαρακτήρων.

2. Απαιτήσεις Βάσης Δεδομένων (RDBMS)

Ελάχιστες απαιτήσεις για το σύστημα διαχείρισης βάσης δεδομένων:

MySQL 8 ή νεότερη έκδοση

MariaDB 10.5 ή νεότερη έκδοση

Η εφαρμογή περιλαμβάνει ενσωματωμένους μηχανισμούς ελέγχου συμβατότητας που επιτρέπουν τη λειτουργία και σε ορισμένες παλαιότερες εκδόσεις των παραπάνω συστημάτων. Ωστόσο, η λειτουργία σε παλαιότερες εκδόσεις δεν υποστηρίζεται επίσημα και δεν παρέχεται εγγύηση ορθής λειτουργίας.

3. Απαιτήσεις Περιβάλλοντος Εκτέλεσης (PHP)

Ελάχιστη απαιτούμενη έκδοση γλώσσας εκτέλεσης:

PHP 8.1 ή νεότερη έκδοση

Όπως και στην περίπτωση της βάσης δεδομένων, η εφαρμογή διαθέτει μηχανισμούς ελέγχου συμβατότητας για πιθανή λειτουργία σε ορισμένες παλαιότερες εκδόσεις της PHP, χωρίς όμως να παρέχεται εγγύηση πλήρους λειτουργικότητας.

4. Απαιτήσεις Server

Η εγκατάσταση του συστήματος προτείνεται να πραγματοποιηθεί σε σύγχρονο server υποδομής τελευταίας τριετίας με τα ελάχιστα χαρακτηριστικά που απαιτούνται για την ασφαλή και αξιόπιστη λειτουργία της εφαρμογής.

Η εφαρμογή έχει σχεδιαστεί και δοκιμαστεί σε περιβάλλον Windows Server και Linux Server, με υποστήριξη ασφαλών συνδέσεων TLS/SSL.

5. Ρυθμίσεις PHP (php.ini)

Για την ορθή και ασφαλή λειτουργία του συστήματος, ο εξυπηρετητής PHP θα πρέπει να έχει ρυθμιστεί κατάλληλα σε περιβάλλον παραγωγής.

Ιδιαίτερη προσοχή απαιτείται στις παρακάτω ρυθμίσεις:

- Ορισμός σωστής ζώνης ώρας:

  date.timezone = Europe/Athens

- Απενεργοποίηση εμφάνισης σφαλμάτων προς τον τελικό χρήστη:

  display_errors = Off

- Ενεργοποίηση καταγραφής σφαλμάτων σε αρχείο:

  log_errors = On

- Ορισμός UTF-8 ως προεπιλεγμένου character set:

  default_charset = "UTF-8"

- Ενεργοποίηση μόνο των απαραίτητων PHP extensions που απαιτούνται για τη λειτουργία της εφαρμογής.

Ρύθμιση ασφαλούς διαχείρισης συνεδριών (sessions), με τις ακόλουθες βασικές πρακτικές:

- χρήση cookies αποκλειστικά για sessions

- ενεργοποίηση HttpOnly cookies
- ενεργοποίηση Secure cookies
- ενεργοποίηση strict session mode

Ορισμός κατάλληλων φακέλων για:

- προσωρινά αρχεία
- αποθήκευση sessions
- καταγραφή σφαλμάτων (error logs)

Οι φάκελοι αυτοί θα πρέπει να διαθέτουν δικαιώματα εγγραφής από τον web server.

Απενεργοποίηση εργαλείων αποσφαλμάτωσης ή προφίλ σε περιβάλλον παραγωγής (π.χ. Xdebug). Συνιστάται η ενεργοποίηση του OPcache για βελτίωση της απόδοσης του συστήματος.

6. Μεταφόρτωση Αρχείων Εφαρμογής

Το πακέτο αρχείων της εφαρμογής (αρχεία .php, .js, .css κ.λπ.) περιλαμβάνει την πλήρη δομή φακέλων όπως αυτή θα πρέπει να εγκατασταθεί στον root κατάλογο της ιστοσελίδας στον server. Μετά την μεταφόρτωση των αρχείων θα πρέπει να οριστούν τα κατάλληλα δικαιώματα πρόσβασης στους φακέλους:

Όλοι οι βασικοί φάκελοι της εφαρμογής θα πρέπει να διαθέτουν δικαιώματα READ. Οι φάκελοι `/stats/` και `/uploads/` θα πρέπει να διαθέτουν δικαιώματα READ / WRITE, καθώς χρησιμοποιούνται για αποθήκευση αρχείων που μεταφορτώνονται από το σύστημα και δημιουργία προσωρινών αρχείων, αναφορών και στατιστικών δεδομένων.

Παράλληλα, για την ορθή λειτουργία της εφαρμογής, οι συγκεκριμένοι φάκελοι θα πρέπει να έχουν οριστεί και στις επιτρεπόμενες διαδρομές πρόσβασης μέσω της ρύθμισης open_basedir της PHP.

7. Αρχείο Ρυθμίσεων Συστήματος

Το αρχείο config_smsys.php περιλαμβάνει κρίσιμες ρυθμίσεις της εφαρμογής και θα πρέπει να τοποθετηθεί εκτός του root καταλόγου της ιστοσελίδας, ώστε να μην είναι προσβάσιμο από εξωτερικούς χρήστες μέσω web.

Στο αρχείο αυτό περιλαμβάνονται:

- κλειδιά κρυπτογράφησης του συστήματος
- στοιχεία αποστολής email μέσω του mail server
- στοιχεία σύνδεσης με την τοπική βάση δεδομένων

Κατά την εγκατάσταση θα πρέπει να συμπληρωθούν οι τιμές των αντίστοιχων σταθερών σύμφωνα με τις ρυθμίσεις του server στον οποίο πραγματοποιείται η εγκατάσταση.

Για τα κλειδιά κρυπτογράφησης μπορούν να χρησιμοποιηθούν τιμές που θα οριστούν από τον διαχειριστή του συστήματος, σύμφωνα με το πρότυπο μορφής που περιγράφεται στις αντίστοιχες σταθερές μέσα στο αρχείο.

8. Εγκατάσταση Βάσης Δεδομένων

Η εγκατάσταση της Βάσης Δεδομένων πραγματοποιείται μέσω του συνοδευόμενου αρχείου .sql, το οποίο περιέχει το πλήρες σχήμα της βάσης δεδομένων της εφαρμογής.

Κατά την εκτέλεση του αρχείου δημιουργούνται:

- οι πίνακες του συστήματος (tables)
- οι προβολές (views)
- οι απαραίτητοι δείκτες (indexes)
- τα ξένα κλειδιά (foreign keys) και οι μοναδικά περιορισμοί (unique constraints)
- οι βασικές τιμές και δεδομένα αρχικής λειτουργία της εφαρμογής

9. Δημιουργία Κύριου Διαχειριστή

Κατά την αρχική εγκατάσταση της βάσης δεδομένων δημιουργείται αυτόματα ένας Κύριος Διαχειριστής (Super Administrator) του συστήματος.

Τα στοιχεία πρόσβασης ορίζονται ως εξής:

- Username: το email που έχει δηλωθεί κατά την εγκατάσταση
- Password: αποστέλλεται στον διαχειριστή σε ξεχωριστό μήνυμα

Κατά την πρώτη είσοδο στο σύστημα, ο χρήστης θα κληθεί να αλλάξει τον αρχικό κωδικό πρόσβασης για λόγους ασφαλείας.

10. Δημιουργία Αντιγράφων Ασφαλείας (Backup)

Για τη διασφάλιση της ακεραιότητας των δεδομένων και τη δυνατότητα αποκατάστασης του συστήματος σε περίπτωση σφάλματος ή απώλειας δεδομένων, συνιστάται η τακτική δημιουργία αντιγράφων ασφαλείας της βάσης δεδομένων και των αρχείων της εφαρμογής.

Τα αντίγραφα ασφαλείας θα πρέπει να περιλαμβάνουν:

- τη Βάση Δεδομένων της εφαρμογής
- τον φάκελο `/uploads/`, ο οποίος ενδέχεται να περιλαμβάνει αρχεία που έχουν μεταφορτωθεί από χρήστες
- τον φάκελο `/stats/`, στον οποίο αποθηκεύονται αναφορές και στατιστικά δεδομένα
- τυχόν αρχεία ρυθμίσεων που βρίσκονται εκτός του root της ιστοσελίδας (π.χ. config_smsys.php)

Η δημιουργία αντιγράφων ασφαλείας μπορεί να πραγματοποιείται μέσω προγραμματισμένων εργασιών (schedule tasks), μέσω εργαλείων διαχείρισης βάσεων δεδομένων ή μέσω μηχανισμών backup που παρέχει το hosting περιβάλλον. Συνιστάται η διατήρηση πολλαπλών εκδόσεων αντιγράφων ασφαλείας καθώς και η αποθήκευσή τους σε ξεχωριστό αποθηκευτικό χώρο ή σε διαφορετικό server.

11. Βασικές Ρυθμίσεις Ασφάλειας Server

Για την ασφαλή λειτουργία της εφαρμογής προτείνεται η υιοθέτηση των ακόλουθων βασικών πρακτικών ασφάλειας στον server:

- Υποχρεωτική χρήση HTTPS για όλες τις συνδέσεις προς το σύστημα.

- Ενεργοποίηση σύγχρονων εκδόσεων πρωτοκόλλου TLS και απενεργοποίηση παλαιότερων και μη ασφαλών πρωτοκόλλων.
- Περιορισμός πρόσβασης σε κρίσιμους φακέλους και αρχεία του συστήματος μέσω κατάλληλων ρυθμίσεων του web server.
- Περιορισμός πρόσβασης στη βάση δεδομένων μόνο από τον ίδιο τον server ή από εξουσιοδοτημένες διευθύνσεις IP.
- Χρήση ισχυρών κωδικών πρόσβασης για όλους τους λογαριασμούς διαχειριστών.
- Τακτική ενημέρωση του λειτουργικού συστήματος, του web server, της PHP και της βάσης δεδομένων με τις τελευταίες διαθέσιμες ενημερώσεις ασφαλείας.

Επιπλέον, συνιστάται η χρήση firewall σε επίπεδο server ή δικτύου για τον περιορισμό μη εξουσιοδοτημένης πρόσβασης.

12. Απαιτούμενες PHP Extensions

Για την ορθή λειτουργία του συστήματος απαιτείται η ενεργοποίηση ορισμένων βασικών επεκτάσεων της PHP. Οι επεκτάσεις αυτές παρέχουν λειτουργίες πρόσβασης στη βάση δεδομένων, διαχείρισης κειμένου, κρυπτογράφησης, επικοινωνίας με εξωτερικές υπηρεσίες και επεξεργασίας αρχείων.

Ενδεικτικά απαιτούνται οι ακόλουθες επεκτάσεις:

- PDO και pdo_mysql για πρόσβαση στη βάση δεδομένων
- mbstring για σωστή διαχείριση πολυγλωσσικών χαρακτήρων
- openssl για λειτουργίες κρυπτογράφησης και ασφαλών συνδέσεων
- curl για επικοινωνία με εξωτερικές υπηρεσίες
- json για επεξεργασία δεδομένων σε μορφή JSON
- fileinfo για αναγνώριση τύπων αρχείων
- zip για δημιουργία ή αποσυμπίεση αρχείων
- gd ή αντίστοιχη βιβλιοθήκη για βασική επεξεργασία εικόνων

13. Δομή Φακέλων Εφαρμογής

Το πακέτο εγκατάστασης περιλαμβάνει προκαθορισμένη δομή φακέλων, η οποία θα πρέπει να διατηρηθεί κατά την εγκατάσταση στον server. Η δομή των φακέλων δεν θα πρέπει να τροποποιείται, καθώς η εφαρμογή χρησιμοποιεί προκαθορισμένες διαδρομές πρόσβασης για την ανάγνωση και αποθήκευση αρχείων.

14. Έλεγχος Ορθής Εγκατάστασης

Μετά την ολοκλήρωση της εγκατάστασης συνιστάται η εκτέλεση βασικών ελέγχων για την επιβεβαίωση της σωστής λειτουργίας του συστήματος.

Οι έλεγχοι αυτοί περιλαμβάνουν ενδεικτικά:

- επιβεβαίωση ότι η εφαρμογή είναι προσβάσιμη μέσω του web browser,
- έλεγχο επιτυχούς σύνδεσης με τη βάση δεδομένων,
- έλεγχο ότι οι φάκελοι /uploads και /stats διαθέτουν τα κατάλληλα δικαιώματα εγγραφής (δοκιμαστικό upload αρχείου και έλεγχος καταγραφής log),
- έλεγχο σωστής λειτουργίας της διαδικασίας σύνδεσης χρηστών,
- επιβεβαίωση αποστολής δοκιμαστικού μηνύματος email μέσω του mail server,
- έλεγχο καταγραφής σφαλμάτων (error logs) σε περίπτωση αποτυχίας λειτουργίας

Σε περίπτωση εμφάνισης σφαλμάτων κατά την εκκίνηση της εφαρμογής, τα αρχεία καταγραφής σφαλμάτων της PHP και του web server μπορούν να χρησιμοποιηθούν για την διάγνωση του προβλήματος.

15. Έλεγχος Ασφαλείας Μετά την Εγκατάσταση (Security Checklist)

Μετά την ολοκλήρωση της εγκατάστασης της εφαρμογής συνιστάται η εκτέλεση βασικών ελέγχων ασφαλείας, ώστε να διασφαλιστεί η σωστή και ασφαλής λειτουργία του συστήματος στο περιβάλλον παραγωγής.

Ενδεικτικά προτείνεται να επιβεβαιωθούν τα ακόλουθα:

- Η πρόσβαση στην εφαρμογή πραγματοποιείται αποκλειστικά μέσω HTTPS με ενεργό πιστοποιητικό TLS/SSL.
- Το αρχείο ρυθμίσεων config_smsys.php βρίσκεται εκτός του root της ιστοσελίδας και δεν είναι προσβάσιμο μέσω web.
- Οι φάκελοι του συστήματος έχουν τα κατάλληλα δικαιώματα πρόσβασης.
- Η εμφάνιση σφαλμάτων προς τους τελικούς χρήστες είναι απενεργοποιημένη (display_errors = Off) και τα σφάλματα καταγράφονται σε αρχεία καταγραφής.
- Η βάση δεδομένων είναι προσβάσιμη μόνο από εξουσιοδοτημένες διευθύνσεις ή από τον ίδιο τον server.
- Οι λογαριασμοί διαχειριστών χρησιμοποιούν ισχυρούς κωδικούς πρόσβασης και έχουν πραγματοποιήσει αλλαγή του αρχικού κωδικού κατά την πρώτη είσοδο.
- Το λειτουργικό σύστημα, ο web server, η PHP και η βάση δεδομένων διαθέτουν τις τελευταίες ενημερώσεις ασφαλείας.
- Υπάρχει ενεργή διαδικασία τακτικής δημιουργίας αντιγράφων ασφαλείας (backup) της βάσης δεδομένων και των αρχείων της εφαρμογής.

Η τήρηση των παραπάνω ελέγχων συμβάλλει σημαντικά στη διατήρηση της ασφάλειας, της αξιοπιστίας και της διαθεσιμότητας του συστήματος.

16. Πηγές και Οδηγίες Λογισμικού Εγκατάστασης

Τα βασικά λογισμικά που απαιτούνται για την εγκατάσταση και λειτουργία του συστήματος μπορούν να ληφθούν από τις επίσημες ιστοσελίδες των αντίστοιχων έργων.

MySQL Community Server

Επίσημη σελίδα λήψης: https://www.mysql.com/downloads/

MariaDB Community Server

Επίσημη σελίδα λήψης: https://mariadb.com/downloads/

PHP

Επίσημη σελίδα λήψης: https://www.php.net/downloads.php

Παράρτημα Γ: «Ιστορίες Χρηστών & Κριτήρια Αποδοχής»

Στο παρόν παράρτημα παρατίθενται συγκεντρωτικά όλες οι ιστορίες χρηστών και κριτήρια αποδοχής, σε μορφή πίνακα για λόγους ευκολίας ανάγνωσης, χωρισμένοι σε τμήματα ανά ρόλο/χρήστη.

Μη Πιστοποιημένος Χρήστης / Επισκέπτης:

Τίτλος: Προβολή γενικού ωρολογίου προγράμματος ανά Τμήμα
Ως επισκέπτης **Θέλω να** δω το ωρολόγιο πρόγραμμα κάποιου Τμήματος για συγκεκριμένο εξάμηνο **Ώστε** να ενημερωθώ για τις ώρες διδασκαλίας κάθε μαθήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Ωρολόγιο Πρόγραμμα» από το κεντρικό μενού. • Επιλέγει από αναπτυσσόμενη λίστα τη Σχολή και το Τμήμα που τον ενδιαφέρει. • Επιλέγει από αναπτυσσόμενη λίστα το εξάμηνο που τον ενδιαφέρει (προεπιλεγμένα εμφανίζεται το τρέχον εξάμηνο. • Το σύστημα εμφανίζει το εβδομαδιαίο ωρολόγιο πρόγραμμα του επιλεγμένου Τμήματος, με τις ημέρες, ώρες και αίθουσες διδασκαλίας για κάθε μάθημα.

Πίνακας Γ-US-1: Προβολή γενικού ωρολογίου προγράμματος ανά Τμήμα

Τίτλος: Αναζήτηση μαθήματος στο δημόσιο ωρολόγιο πρόγραμμα
Ως επισκέπτης **Θέλω να** αναζητήσω ένα συγκεκριμένο μάθημα στο δημόσιο ωρολόγιο πρόγραμμα **Ώστε** να ενημερωθώ για τις ημέρες και ώρες διδασκαλίας του.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Ωρολόγιο Πρόγραμμα» από το κεντρικό μενού. • Εισάγει στο πεδίο «Αναζήτηση Μαθήματος» τον τίτλο ή τον κωδικό του μαθήματος που τον ενδιαφέρει. **Σενάριο 1: Επιτυχής εύρεση αποτελεσμάτων** • Εμφανίζεται λίστα μαθημάτων που αντιστοιχούν στο κριτήριο αναζήτησης, με στοιχεία όπως

τίτλος μαθήματος, κωδικός, εξάμηνο, Τμήμα. • Ο χρήστης επιλέγει το επιθυμητό μάθημα από τη λίστα, πατώντας στον τίτλο του. • Εμφανίζεται η σελίδα του ωρολογίου προγράμματος που περιλαμβάνει το επιλεγμένο μάθημα, το οποίο επισημαίνεται οπτικά (π.χ. με διαφορετικό χρώμα ή σκίαση). **Σενάριο 2: Αποτυχία εύρεσης αποτελεσμάτων** • Εμφανίζεται μήνυμα ενημέρωσης ότι δεν βρέθηκαν μαθήματα που να αντιστοιχούν στο κριτήριο αναζήτησης. • Ο χρήστης μπορεί να τροποποιήσει το κριτήριο ή να πραγματοποιήσει νέα αναζήτηση.

Πίνακας Γ-US-2: Αναζήτηση μαθήματος στο δημόσιο ωρολόγιο πρόγραμμα

Τίτλος: Λήψη δημόσιου ωρολογίου προγράμματος σε αρχείο PDF/Excel
Ως επισκέπτης **Θέλω να** αναζητήσω ένα συγκεκριμένο μάθημα στο δημόσιο ωρολόγιο πρόγραμμα **Ώστε** να ενημερωθώ για τις ημέρες και ώρες διδασκαλίας του.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο Ο χρήστης ακολουθεί τις ενέργειες όπως περιγράφονται στην Ιστορία Χρήστη «Προβολή γενικού ωρολογίου προγράμματος ανά Τμήμα», προβάλλοντας το πρόγραμμα που τον ενδιαφέρει. • Επιλέγει το κουμπί «Λήψη σε PDF» ή «Λήψη σε Excel» ανάλογα με τη μορφή που επιθυμεί. • Το σύστημα εξάγει το ωρολόγιο πρόγραμμα της τρέχουσας προβολής σε αρχείο της αντίστοιχης μορφής. • Κατά την επιτυχή δημιουργία, το σύστημα παραπέμπει μέσω του περιηγητή στη λήψη του αρχείου. • Το παραγόμενο αρχείο περιλαμβάνει: επικεφαλίδα με τη Σχολή, το Τμήμα και το εξάμηνο, πίνακα ωρολογίου προγράμματος (ημέρες, ώρες, αίθουσες, μαθήματα, διδάσκοντες), ημερομηνία εξαγωγής. • Σε περίπτωση σφάλματος κατά τη δημιουργία του αρχείου (π.χ. διακοπή σύνδεσης ή μεγάλο μέγεθος), εμφανίζεται κατάλληλο μήνυμα και ο χρήστης μπορεί να επαναλάβει τη λήψη.

Πίνακας Γ-US-3: Λήψη δημόσιου ωρολογίου προγράμματος σε αρχείο PDF/Excel

Τίτλος: Προβολή δημόσιων ανακοινώσεων αλλαγών ωρολογίου προγράμματος
Ως επισκέπτης **Θέλω να** προβάλλω τις δημόσιες ανακοινώσεις που αφορούν αλλαγές στο ωρολόγιο πρόγραμμα **Ώστε** να ενημερωθώ έγκαιρα για πιθανές τροποποιήσεις στις ώρες ή ημέρες διδασκαλίας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Ανακοινώσεις» από το κεντρικό μενού. • Το σύστημα εμφανίζει λίστα δημόσιων ανακοινώσεων σχετικά με αλλαγές στα ωρολόγια προγράμματα των Τμημάτων. • Οι ανακοινώσεις εμφανίζονται ταξινομημένες κατά ημερομηνία ανάρτησης (νεότερες πρώτες). • Για κάθε ανακοίνωση εμφανίζονται: τίτλος, ημερομηνία ανάρτησης, Τμήμα, και σύντομη περιγραφή. • Ο χρήστης μπορεί να επιλέξει από αναπτυσσόμενη λίστα συγκεκριμένο Τμήμα, ώστε να προβάλλονται μόνο οι σχετικές ανακοινώσεις. • Ο χρήστης μπορεί να επιλέξει μια ανακοίνωση για να δει αναλυτικές πληροφορίες (π.χ. ημερομηνία αλλαγής, επηρεαζόμενα μαθήματα, διδάσκων). • Το σύστημα διασφαλίζει ότι μόνο δημόσιες/εγκεκριμένες ανακοινώσεις είναι ορατές στους επισκέπτες.

Πίνακας Γ-US-4: Προβολή δημόσιων ανακοινώσεων αλλαγών ωρολογίου προγράμματος

Κοινές Λειτουργίες Πιστοποιημένων Χρηστών:

Τίτλος: Είσοδος στην πλατφόρμα με χρήση SSO (Single-Sign-On)
Ως πιστοποιημένος χρήστης **Θέλω να** συνδεθώ στην πλατφόρμα με χρήση του ιδρυματικού μου λογαριασμού **Ώστε** να έχω πρόσβαση σε λειτουργίες πιστοποιημένων χρηστών της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος με SSO» στη σελίδα. • Ο χρήστης μεταφέρεται στην κεντρική υπηρεσία πιστοποίησης χρηστών του Ιδρύματος. • Εισάγει έγκυρα διαπιστευτήρια (ιδρυματικό email και κωδικό πρόσβασης). **Σενάριο 1: Επιβεβαίωση διαπιστευτηρίων** • Η σύνδεση ολοκληρώνεται και ο χρήστης μεταφέρεται στην πλατφόρμα. **Σενάριο 1α: Επιτυχής λήψη επιβεβαίωσης και δεδομένων μέσω SSO** • Ο χρήστης μεταφέρεται στην κεντρική σελίδα πιστοποιημένων χρηστών της πλατφόρμας. • Ο χρήστης ενημερώνεται ότι μπορεί να δημιουργήσει τοπικό κωδικό εισόδου εάν δεν το έχει ήδη πράξει, για είσοδο απευθείας από την πλατφόρμα. **Σενάριο 1β: Αποτυχία λήψη επιβεβαίωσης ή δεδομένων μέσω SSO ή διακοπή σύνδεσης** • Ο χρήστης μεταφέρεται στην αρχική σελίδα εισόδου, χωρίς να έχει πραγματοποιηθεί σύνδεση/είσοδος και εμφανίζεται μήνυμα αποτυχίας εισόδου και παρότρυνση επαναπροσπάθειας. **Σενάριο 2: Απόρριψη διαπιστευτηρίων από το Ίδρυμα** • Εμφανίζεται στον χρήστη μήνυμα αποτυχίας όπως αυτό έχει οριστεί στη σελίδα SSO εισόδου. • Ο χρήστης μπορεί να επιλέξει την εκ νέου προσπάθεια εισόδου.

Πίνακας Γ-US-5: Είσοδος στην πλατφόρμα με χρήση SSO (Single-Sign-On)

Τίτλος: Είσοδος στην πλατφόρμα με κωδικό και ταυτοποίηση 2FA
Ως πιστοποιημένος χρήστης (Φοιτητής ή Διδάσκων) **Θέλω να** πραγματοποιήσω είσοδο στην πλατφόρμα με κωδικό **Ώστε** να έχω πρόσβαση σε λειτουργίες πιστοποιημένων χρηστών της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. • Εισάγει έγκυρα διαπιστευτήρια (ιδρυματικό email και κωδικό πρόσβασης). • Τα πεδία της φόρμας περιέχουν έγκυρες τιμές. • Εμφανίζονται κατάλληλα μηνύματα για μη έγκυρες ή ελλιπείς τιμές πεδίων. **Σενάριο 1: Επιτυχής ταυτοποίηση** • Ο χρήστης ανακατευθύνεται σε κατάλληλη σελίδα. **Σενάριο 1α: Επιβεβαιωμένη συσκευή (χωρίς 2FA)** • Μετά την επιτυχή ταυτοποίηση ο χρήστης μεταφέρεται στον λογαριασμό του. **Σενάριο 1β: Νέα συσκευή (απαιτείται 2FA)** • Μετά την επιτυχή ταυτοποίηση, ζητείται από τον χρήστη επαλήθευση μέσω δευτερεύοντος παράγοντα (2FA). • Ο χρήστης επιλέγει τρόπο λήψης κωδικού επαλήθευσης (email, sms, Telegram). • Λαμβάνει έναν μοναδικό προσωρινό αριθμό στο μέσο που επέλεξε. • Εισάγει τον κωδικό στο σχετικό πεδίο της σελίδας επαλήθευσης δευτερεύοντος παράγοντα (2FA). **Σενάριο 1β-1: Επιτυχής εισαγωγή 2FA** • Ο κωδικός 2FA επαληθεύεται και ο χρήστης μεταφέρεται στον λογαριασμό του. **Σενάριο 1β-2: Λανθασμένος κωδικός 2FA ή λήξη ισχύος** • Εμφανίζεται κατάλληλο μήνυμα σφάλματος και επιλογή επαναποστολής κωδικού. **Σενάριο 2: Αποτυχία ταυτοποίησης** • Εμφανίζεται κατάλληλο μήνυμα αποτυχίας σύνδεσης και ενημερώνει για έλεγχο των διαπιστευτηρίων που εισήγαγε και επαναπροσπάθεια, χωρίς να αποκαλύπτει λεπτομέρειες όπως το πεδίο αποτυχίας, και ταυτόχρονα προτείνει τη χρήση της υπηρεσίας εισόδου SSO σε

περίπτωση εκ νέου αδυναμίας εισόδου με κωδικό. • Η περίπτωση αποτυχίας μπορεί να οφείλεται τόσο σε λανθασμένα στοιχεία εισόδου, όσο και σε μη ενεργοποιημένο λογαριασμό, αφού οι λογαριασμοί χρηστών κατά την πρώτη χρήση της πλατφόρμας απαιτείται η πιστοποίηση μέσω του Ιδρύματος και την παρεχόμενη υπηρεσία εισόδου SSO. Στον χρήστη εμφανίζεται ενημερωτικό κείμενο κατά την επίσκεψή του στη σελίδα εισόδου χρηστών. **Σενάριο 3: Πολλαπλές συνεχόμενες αποτυχίες ταυτοποίησης** • Εμφανίζεται στον χρήστη μήνυμα προσωρινής διακοπής των προσπαθειών εισόδου και επαναπροσπάθειας μετά από ορισμένο χρόνο.

Πίνακας Γ-US-6: Είσοδος στην πλατφόρμα με κωδικό και ταυτοποίηση 2FA

Τίτλος: Ανάκτηση / επαναφορά κωδικού πρόσβασης
Ως πιστοποιημένος χρήστης (Φοιτητής ή Διδάσκων) **Θέλω να** επαναφέρω τον κωδικό πρόσβασής μου σε περίπτωση απώλειας **Ώστε** να μπορέσω να συνδεθώ ξανά στον λογαριασμό μου.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. • Επιλέγει τον σύνδεσμο «Ξέχασα τον κωδικό μου». • Στη σελίδα επαναφοράς κωδικού πρόσβασης που εμφανίζεται, εισάγει το ιδρυματικό του email, με το οποίο είχε πραγματοποιήσει την αρχική του εγγραφή στην πλατφόρμα. • Το πεδίο εισαγωγής πρέπει να περιέχει έγκυρη τιμή (σωστή μορφή email). • Εμφανίζονται κατάλληλα μηνύματα για μη έγκυρη ή ελλιπή τιμή πεδίου. • Ο χρήστης επιλέγει το κουμπί «Υποβολή». • Για λόγους ασφαλείας, το σύστημα δεν αποκαλύπτει εάν ένα email αντιστοιχεί σε λογαριασμό. **Σενάριο 1: Έγκυρη και καταχωρημένη διεύθυνση email** • Εμφανίζεται ενημερωτικό μήνυμα, ότι εφόσον η διεύθυνση email είναι έγκυρη, θα λάβει στη διεύθυνση αυτή σύνδεσμο περιορισμένης χρονικής ισχύος, για την επαναφορά του κωδικού. • Εφόσον η διεύθυνση email είναι έγκυρη και αντιστοιχεί σε ενεργό λογαριασμό χρήστη, το σύστημα αποστέλλει μήνυμα με προσωρινό μοναδικό σύνδεσμο περιορισμένου χρόνου, για την

επαναφορά του κωδικού.

- Ο χρήστης λαμβάνει το email και ακολουθεί τον σύνδεσμο.
- Εφόσον ο σύνδεσμος είναι ενεργός και εντός χρονικού ορίου, εμφανίζεται στον περιηγητή η σελίδα επαναδημιουργίας κωδικού.
- Αν ο σύνδεσμος έχει λήξει ή είναι άκυρος, εμφανίζεται μήνυμα σφάλματος και οδηγία επανάληψης της διαδικασίας, ενώ η διαδικασία τερματίζεται.
- Ο χρήστης ορίζει νέο κωδικό πρόσβασης και επαληθεύει τον κωδικό σε ξεχωριστό πεδίο.
- Η φόρμα δεν μπορεί να υποβληθεί αν οι δύο τιμές του κωδικού δεν ταυτίζονται.
- Ο νέος κωδικός πρέπει να πληροί τα κριτήρια ελάχιστης πολυπλοκότητας.
- Ο χρήστης επιλέγει «Υποβολή» για τη δημιουργία του νέου του κωδικού.

Σενάριο 1α: Επιτυχής ενημέρωση κωδικού εισόδου

- Εμφανίζεται μήνυμα επιβεβαίωσης ότι η αλλαγή κωδικού ολοκληρώθηκε επιτυχώς.
- Ο χρήστης ανακατευθύνεται στη σελίδα εισόδου.
- Λαμβάνει μήνυμα μέσω email που επιβεβαιώνει την επιτυχή ενημέρωση του κωδικού του.

Σενάριο 1β: Ανεπιτυχής ενημέρωση κωδικού εισόδου

- Προβάλλεται μήνυμα ενημέρωσης για την αποτυχία, ώστε ο χρήστης να διορθώσει τα πεδία (π.χ. μη ταύτιση ή μη επαρκής πολυπλοκότητα).
- Ο χρήστης συμπληρώνει εκ νέου τα πεδία κωδικού.
- Μετά από επαναλαμβανόμενες αποτυχημένες προσπάθειες, η διαδικασία τερματίζεται και ο χρήστης λαμβάνει ειδοποίηση για να επαναλάβει τη διαδικασία από την αρχή.

Σενάριο 2: Μη έγκυρη ή μη καταχωρημένη διεύθυνση email ή μη ενεργός λογαριασμός

- Εμφανίζεται ενημερωτικό μήνυμα ότι εφόσον η διεύθυνση email είναι έγκυρη, θα λάβει στη διεύθυνση αυτή ένα σύνδεσμο περιορισμένης χρονικής ισχύος, για την επαναφορά του κωδικού.
- Αν η διεύθυνση δεν είναι καταχωρημένη ή δεν αντιστοιχεί σε ενεργό λογαριασμό, δεν αποστέλλεται email, αλλά εμφανίζεται γενικό μήνυμα ενημέρωσης (χωρίς να αποκαλύπτεται αν ο λογαριασμός υπάρχει).
- Η διαδικασίας τερματίζει.

Πίνακας Γ-US-7: Ανάκτηση / επαναφορά κωδικού πρόσβασης

Τίτλος: Διαχείριση προσωπικού προφίλ
Ως πιστοποιημένος χρήστης **Θέλω να** διαχειριστώ το προσωπικό μου προφίλ **Ώστε** να ενημερώσω, διορθώσω ή προσθέσω στοιχεία επικοινωνίας και λογαριασμού.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Προφίλ Χρήστη» από το κεντρικό μενού. • Εμφανίζεται η σελίδα με τα στοιχεία λογαριασμού του προφίλ του χρήστη. • Τα σταθερά στοιχεία που διαχειρίζεται αποκλειστικά η γραμματεία της Σχολής (π.χ. Αριθμός Μητρώου, Ιδρυματικό Email, κ.ά.) εμφανίζονται ως μη επεξεργάσιμα (κλειδωμένα) πεδία. • Ο χρήστης μπορεί να τροποποιήσει ή να προσθέτει στοιχεία σε συγκεκριμένα πεδία που είναι ανοιχτά για επεξεργασία όπως κινητό τηλέφωνο, σταθερό τηλέφωνο ή δευτερεύουσα διεύθυνση επικοινωνίας. • Τα πεδία επεξεργασίας πρέπει να περιέχουν έγκυρες τιμές (π.χ. σωστή μορφή τηλεφώνου ή διεύθυνση email). • Εμφανίζονται κατάλληλα μηνύματα για μη έγκυρες ή ελλιπείς τιμές πεδίων. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». • Εμφανίζεται μήνυμα επιτυχούς ενημέρωσης και αποθήκευσης των στοιχείων. • Ο χρήστης βλέπει εκ νέου τη σελίδα του προφίλ του με τα ενημερωμένα δεδομένα. • Το σύστημα καταγράφει την ημερομηνία και ώρα τελευταίας ενημέρωσης των στοιχείων του προφίλ, η οποία εμφανίζεται στη σελίδα. • Σε αλλαγή δευτερεύοντος email επικοινωνίας, αποστέλλεται μήνυμα επιβεβαίωσης στο νέο email και η αλλαγή οριστικοποιείται μόνο μετά την επιβεβαίωση του συνδέσμου (link).

Πίνακας Γ-US-8: Διαχείριση προσωπικού προφίλ

Διαχειριστής:

Τίτλος: Ενεργοποίηση λογαριασμού Κύριου Διαχειριστή
Ως πιστοποιημένος χρήστης (Κύριος Διαχειριστής) **Θέλω να** ενεργοποιήσω τον λογαριασμό μου μετά την αρχική εγκατάσταση της πλατφόρμας **Ώστε** να έχω πρόσβαση στις διαθέσιμες λειτουργίες που αντιστοιχούν στο ρόλο μου.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. • Εισάγει έγκυρα διαπιστευτήρια (ιδρυματικό email και προσωρινό κωδικό εισόδου). • Τα πεδία της φόρμας περιέχουν έγκυρες τιμές. • Εμφανίζονται κατάλληλα μηνύματα για μη έγκυρες ή ελλιπείς τιμές πεδίων. **Σενάριο 1: Επιτυχής ταυτοποίηση** • Ο χρήστης ανακατευθύνεται στη σελίδα δημιουργίας νέου κωδικού εισόδου. • Ο χρήστης ορίζει νέο κωδικό πρόσβασης και επαληθεύει τον κωδικό σε ξεχωριστό πεδίο. • Η φόρμα δεν μπορεί να υποβληθεί αν οι δύο τιμές του κωδικού δεν ταυτίζονται. • Ο νέος κωδικός πρέπει να πληροί τα κριτήρια ελάχιστης πολυπλοκότητας. • Ο χρήστης επιλέγει «Υποβολή» για τη δημιουργία του νέου του κωδικού. **Σενάριο 1α: Επιτυχής ενημέρωση κωδικού εισόδου** • Εμφανίζεται μήνυμα επιβεβαίωσης ότι η δημιουργία νέου κωδικού ολοκληρώθηκε επιτυχώς και έχει ενεργοποιηθεί ο λογαριασμός του. • Λαμβάνει μήνυμα μέσω email που επιβεβαιώνει την επιτυχή δημιουργία νέου κωδικού και ενεργοποίησης λογαριασμού. • Ο χρήστης ανακατευθύνεται στη σελίδα εισόδου για να πραγματοποιήσει είσοδο με τον νέο κωδικό. **Σενάριο 1β: Ανεπιτυχής ενημέρωση κωδικού εισόδου** • Προβάλλεται μήνυμα ενημέρωσης για την αποτυχία, ώστε ο χρήστης να διορθώσει τα πεδία (π.χ. μη ταύτιση ή μη επαρκής πολυπλοκότητα). • Ο χρήστης συμπληρώνει εκ νέου τα πεδία κωδικού. • Μετά από επαναλαμβανόμενες αποτυχημένες προσπάθειες, η διαδικασία τερματίζεται και ο

χρήστης λαμβάνει ειδοποίηση για να επαναλάβει τη διαδικασία από την αρχή. **Σενάριο 2: Αποτυχία ταυτοποίησης** • Εμφανίζεται κατάλληλο μήνυμα αποτυχίας σύνδεσης και ενημερώνει για έλεγχο των διαπιστευτηρίων που εισήγαγε και επαναπροσπάθεια, χωρίς να αποκαλύπτει λεπτομέρειες όπως το πεδίο αποτυχίας. **Σενάριο 3: Πολλαπλές συνεχόμενες αποτυχίες ταυτοποίησης** • Εμφανίζεται στον χρήστη μήνυμα προσωρινής διακοπής των προσπαθειών εισόδου και επαναπροσπάθειας μετά από ορισμένο χρόνο.

Πίνακας Γ-US-9: Ενεργοποίηση λογαριασμού Κύριου Διαχειριστή

Τίτλος: Είσοδος στην πλατφόρμα ως διαχειριστής
Ως πιστοποιημένος χρήστης (Διαχειριστής) **Θέλω να** πραγματοποιήσω είσοδο στην πλατφόρμα ως διαχειριστής **Ώστε** να έχω πρόσβαση στις διαθέσιμες λειτουργίες που αντιστοιχούν στο ρόλο μου.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού. ή Ο χρήστης ανοίγει τη σελίδα εισόδου διαχειριστών που του έχει παρασχεθεί. • Εισάγει έγκυρα διαπιστευτήρια (ιδρυματικό email και κωδικό εισόδου). • Τα πεδία της φόρμας περιέχουν έγκυρες τιμές. • Εμφανίζονται κατάλληλα μηνύματα για μη έγκυρες ή ελλιπείς τιμές πεδίων. **Σενάριο 1: Επιτυχής ταυτοποίηση** • Ο χρήστης ανακατευθύνεται σε κατάλληλη σελίδα. **Σενάριο 1α: Επιβεβαιωμένη συσκευή (χωρίς 2FA)** • Μετά την επιτυχή ταυτοποίηση ο χρήστης μεταφέρεται στον λογαριασμό του. **Σενάριο 1β: Νέα συσκευή (απαιτείται 2FA)** • Μετά την επιτυχή ταυτοποίηση, ζητείται από τον χρήστη επαλήθευση μέσω δευτερεύοντος παράγοντα (2FA). • Ο χρήστης επιλέγει τρόπο λήψης κωδικού επαλήθευσης (email, sms, Telegram).

• Λαμβάνει έναν μοναδικό προσωρινό αριθμό στο μέσο που επέλεξε. • Εισάγει τον κωδικό στο σχετικό πεδίο της σελίδας επαλήθευσης δευτερεύοντος παράγοντα (2FA). **Σενάριο 1β-1: Επιτυχής εισαγωγή 2FA** • Ο κωδικός 2FA επαληθεύεται και ο χρήστης μεταφέρεται στον λογαριασμό του. **Σενάριο 1β-2: Λανθασμένος κωδικός 2FA ή λήξη ισχύος** • Εμφανίζεται κατάλληλο μήνυμα σφάλματος και επιλογή επαναποστολής κωδικού. **Σενάριο 2: Αποτυχία ταυτοποίησης** • Εμφανίζεται κατάλληλο μήνυμα αποτυχίας σύνδεσης και ενημερώνει για έλεγχο των διαπιστευτηρίων που εισήγαγε και επαναπροσπάθεια, χωρίς να αποκαλύπτει λεπτομέρειες όπως το πεδίο αποτυχίας. **Σενάριο 3: Πολλαπλές συνεχόμενες αποτυχίες ταυτοποίησης** • Εμφανίζεται στον χρήστη μήνυμα προσωρινής διακοπής των προσπαθειών εισόδου και επαναπροσπάθειας μετά από ορισμένο χρόνο.

Πίνακας Γ-US-10: Είσοδος στην πλατφόρμα ως διαχειριστής

Τίτλος: Δημιουργία λογαριασμού Διαχειριστή Τμήματος
Ως Κύριος Διαχειριστής **Θέλω να** δημιουργήσω νέο λογαριασμό Διαχειριστή Τμήματος **Ώστε** να του παραχωρηθεί πρόσβασης στις λειτουργίες που αντιστοιχούν στο ρόλο του.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού. • Επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Διαχειριστών» από το δευτερεύον μενού. • Επιλέγει τον σύνδεσμο «Δημιουργία Νέου Λογαριασμού Διαχειριστή» από το δευτερεύον μενού. • Ο χρήστης συμπληρώνει όλα τα υποχρεωτικά πεδία της φόρμας. • Τα πεδία της φόρμας περιέχουν έγκυρες τιμές. • Ορίζει το ιδρυματικό email του Διαχειριστή Τμήματος.

- Το πεδίο εισαγωγής πρέπει να περιέχει έγκυρη τιμή (σωστή μορφή email).
- Ορίζει το Τμήμα που ανήκει ο Διαχειριστής Τμήματος.
- Εμφανίζονται κατάλληλα μηνύματα σφάλματος για μη έγκυρες ή ελλιπείς τιμές πεδίων.
- Αν κάποιο μοναδικό πεδίο είναι ήδη καταχωρημένο σε άλλο λογαριασμό (π.χ. ιδρυματικό email, ονοματεπώνυμο και πατρώνυμο, κινητό τηλέφωνο), εμφανίζεται μήνυμα προειδοποίησης.
- Ο χρήστης επιλέγει «Αποθήκευση και Δημιουργία Λογαριασμού».
- Εμφανίζεται μήνυμα επιτυχούς δημιουργίας και γίνεται ανακατεύθυνση στη σελίδα διαχείρισης του νέου λογαριασμού.
- Εμφανίζεται η επιλογή «Αποστολή Ειδοποίησης Ενεργοποίησης Λογαριασμού» και ο χρήστης μπορεί την επιλέξει, πατώντας στο σχετικό κουμπί.

Σενάριο 1: Επιτυχής αποστολή ειδοποίησης

- Εφόσον το σύστημα απέστειλε επιτυχώς μήνυμα και σύνδεσμο ενεργοποίησης του λογαριασμού μέσω email προς τον Διαχειριστή Τμήματος, εμφανίζεται μήνυμα επιτυχίας με την ημερομηνία και ώρα αποστολής, καθώς και λήξης ισχύος του συνδέσμου.

Σενάριο 2: Αποτυχία αποστολής ειδοποίησης

- Αν προκύψει σφάλμα κατά την αποστολή email, εμφανίζεται μήνυμα αποτυχίας με δυνατότητα επανάληψης της διαδικασίας αποστολής ειδοποίησης.

Πίνακας Γ-US-11: Δημιουργία λογαριασμού Διαχειριστή Τμήματος

Τίτλος: Ενεργοποίηση λογαριασμού Διαχειριστή Τμήματος από πρόσκληση
Ως προσκεκλημένος Διαχειριστής Τμήματος **Θέλω να** ενεργοποιήσω τον λογαριασμό μου στην πλατφόρμα **Ώστε** να έχω πρόσβαση στις παρεχόμενες λειτουργίες που αντιστοιχούν στο ρόλο μου.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης λαμβάνει μήνυμα ενεργοποίησης στο ιδρυματικό του email, το οποίο περιλαμβάνει μοναδικό σύνδεσμο (link) για τη δημιουργία κωδικού εισόδου. • Ο χρήστης ακολουθεί τον σύνδεσμο ενεργοποίησης λογαριασμού και μεταφέρεται στη σελίδα δημιουργίας κωδικού πρόσβασης. • Αν ο σύνδεσμος έχει λήξει ή είναι άκυρος, εμφανίζεται μήνυμα σφάλματος και οδηγίες για επανάληψη της διαδικασίας και παραπομπή επικοινωνίας με τη Γραμματεία. Η διαδικασία σε

αυτή την περίπτωση τερματίζεται. • Εφόσον ο σύνδεσμος είναι ενεργός και εντός χρονικού ορίου ισχύος, εμφανίζεται η φόρμα δημιουργίας κωδικού. • Ο χρήστης ορίζει νέο κωδικό πρόσβασης και τον επαληθεύει σε ξεχωριστό πεδίο. • Η φόρμα δεν μπορεί να υποβληθεί αν οι δύο τιμές του κωδικού δεν ταυτίζονται. • Ο νέος κωδικός πρέπει να πληροί τα κριτήρια ελάχιστης πολυπλοκότητας (π.χ ελάχιστο μήκος, χρήση κεφαλαίων, πεζών, αριθμών και ειδικών χαρακτήρες κ.λπ.). • Ο χρήστης επιλέγει την αποδοχή όρων χρήσης και πολιτικής απορρήτου (GDPR) για την υποβολή. • Ο χρήστης επιλέγει «Υποβολή και Ενεργοποίηση Λογαριασμού» ώστε να ολοκληρωθεί η δημιουργία του κωδικού και η ενεργοποίηση του λογαριασμού. • Εμφανίζεται μήνυμα επιβεβαίωσης ότι η διαδικασία ολοκληρώθηκε επιτυχώς. • Ο χρήστης ανακατευθύνεται στη σελίδα εισόδου (login) της πλατφόρμας. • Ο χρήστης λαμβάνει επιβεβαιωτικό μήνυμα μέσω email για την επιτυχή ενεργοποίηση του λογαριασμού του.

Πίνακας Γ-US-12: Ενεργοποίηση λογαριασμού Διαχειριστή Τμήματος από πρόσκληση

Τίτλος: Διαχείριση εκκρεμών λογαριασμών Διαχειριστών προς ενεργοποίηση
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω λίστα εκκρεμών λογαριασμών Διαχειριστών Τμημάτων **Ώστε** να επεξεργαστώ στοιχεία, να αποστείλω εκ νέου ειδοποίηση ή να διαγράψω λογαριασμούς που δεν έχουν ενεργοποιηθεί.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης πλοηγείται μέσω του μενού στην επιλογή «Προβολή Λογαριασμών Διαχειριστών». • Εμφανίζεται λίστα με τους λογαριασμούς Διαχειριστών Τμημάτων που εκκρεμούν προς ενεργοποίηση. • Η λίστα περιλαμβάνει το ονοματεπώνυμο του Διαχειριστή Τμήματος, το Τμήμα που έχει οριστεί, την ημερομηνία δημιουργίας λογαριασμού, την ημερομηνία τελευταίας αποστολής ειδοποίησης ενεργοποίησης (εάν έχει αποσταλεί), καθώς και μήνυμα εάν έχει παρέλθει ο χρόνος ισχύος του συνδέσμου ενεργοποίησης.

- Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά.
- Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Αποστολή Ειδοποίησης»: Ναι / Όχι / Λήξαν Σύνδεσμος ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα.
- Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, ημερομηνία, Τμήμα, τηλέφωνο, email), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν.
- Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα.
- Ο χρήστης μπορεί να επιλέξει την επαναποστολή ειδοποίησης σε επιλεγμένο εκκρεμή λογαριασμό, πατώντας το αντίστοιχο κουμπί δίπλα από τα στοιχεία του χρήστη.
- Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διαχειριστή Τμήματος.
- Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διαχειριστή Τμήματος με όλα τα σχετικά πεδία.
- Ο χρήστης μπορεί να τροποποιήσει τα στοιχεία του λογαριασμού.
- Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Ο χρήστης μπορεί να επιλέξει την επαναποστολή ειδοποίησης ενεργοποίησης λογαριασμού, οπότε ενημερώνονται τα πεδία ημερομηνίας/ώρας τελευταίας αποστολής ειδοποίησης και χρόνος λήξης συνδέσμου ενεργοποίησης εφόσον η αποστολή ήταν επιτυχής, διαφορετικά ο χρήστης βλέπει σχετικό μήνυμα αποτυχίας αποστολής και παραπομπή για νέα προσπάθεια.
- Ο χρήστης μπορεί να διαγράψει κάποιον εκκρεμή λογαριασμό, εφόσον αυτός δεν έχει ενεργοποιηθεί, επιλέγοντας «Διαγραφή Εκκρεμούς Λογαριασμού» και κατόπιν ελέγχουν από το σύστημα ότι ο λογαριασμός δεν έχει ενεργοποιηθεί, ολοκληρώνεται η διαγραφή και ο χρήστης επιστρέφει στη λίστα εκκρεμών λογαριασμών.
- Ο χρήστης επιστρέφει στη λίστα εκκρεμών λογαριασμών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-13: Διαχείριση εκκρεμών λογαριασμών Διαχειριστών προς ενεργοποίηση

Τίτλος: Διαχείριση ενεργών λογαριασμών Διαχειριστών Τμημάτων
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω λίστα ενεργών λογαριασμών Διαχειριστών Τμημάτων **Ώστε** να επεξεργαστώ στοιχεία, να προβάλλω ιστορικό ενεργειών ή να απενεργοποιήσω λογαριασμούς που δεν είναι πλέον σε ισχύ.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης πλοηγείται μέσω του μενού στην επιλογή «Προβολή Λογαριασμών Διαχειριστών». • Εμφανίζεται λίστα με τους ενεργούς λογαριασμούς Διαχειριστών Τμημάτων. • Η λίστα περιλαμβάνει το ονοματεπώνυμο του Διαχειριστή Τμήματος, το Τμήμα που διαχειρίζεται και την ημερομηνία/ώρα τελευταίας εισόδου στο σύστημα, καθώς και τρόπο εισόδου (SSO, τοπικός λογαριασμός). • Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά. • Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Κατάσταση Λογαριασμού»: ενεργός / ανενεργός / όλοι ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα. • Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, ημερομηνία, Τμήμα, τηλέφωνο, email), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν. • Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα. • Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διαχειριστή Τμήματος. • Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διαχειριστή Τμήματος με όλα τα σχετικά πεδία. • Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, Τμήμα που διαχειρίζεται, κατάσταση λογαριασμού (ενεργός/ανενεργός). • Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.

• Ο χρήστης μπορεί να επιλέξει την απενεργοποίηση του λογαριασμού, οπότε ο συγκεκριμένος Διαχειριστής Τμήματος δεν έχει πλέον πρόσβαση στην πλατφόρμα, ενώ γίνεται αυτόματα και αποσύνδεσή του εάν είναι συνδεδεμένος τη στιγμή εκείνη. • Ο χρήστης μπορεί να επιλέξει την προβολή ιστορικού κινήσεων εισόδου/εξόδου και ενεργειών που έχει πραγματοποιήσει ο συγκεκριμένος Διαχειριστής Τμήματος, επιλέγοντας το κουμπί «Προβολή Ιστορικού Χρήστη». • Μπορεί να επιλέξει τη λήψη του ιστορικού χρήστη σε αρχείο CSV και τη διαγραφή του ιστορικού παλαιότερων μηνών με ορισμό χρονικού ορίου διαγραφής (π.χ. 1, 2, 3 μήνες) από λίστα (το ιστορικό διατηρείται κατόπιν σε εξωτερικό αρχείο στον διακομιστή). • Ο χρήστης επιστρέφει στη λίστα ενεργών λογαριασμών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-14: Διαχείριση ενεργών λογαριασμών Διαχειριστών Τμημάτων

Τίτλος: Δημιουργία λογαριασμού Διδάσκοντα
Ως Κύριος Διαχειριστής **Θέλω να** δημιουργήσω νέο λογαριασμό Διδάσκοντα **Ώστε** να του παραχωρηθεί πρόσβαση στις λειτουργίες που αντιστοιχούν στο ρόλο του και να ενημερωθεί η λίστα διδασκόντων της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού. • Ο χρήστης πλοηγείται μέσω του μενού στην επιλογή «Εισαγωγή Νέου Διδάσκοντα». • Ο χρήστης συμπληρώνει όλα τα υποχρεωτικά πεδία της φόρμας. • Τα πεδία της φόρμας περιέχουν έγκυρες τιμές. • Ορίζει το ιδρυματικό email του Διδάσκοντα. • Το πεδίο εισαγωγής πρέπει να περιέχει έγκυρη τιμή (σωστή μορφή email). • Ορίζει το Τμήμα που ανήκει ο Διδάσκων (οργανική θέση). • Ορίζει τα Μαθήματα ανά Σχολή και Τμήμα που διδάσκει. • Εμφανίζονται κατάλληλα μηνύματα σφάλματος για μη έγκυρες ή ελλιπείς τιμές πεδίων. • Αν κάποιο μοναδικό πεδίο είναι ήδη καταχωρημένο σε άλλο λογαριασμό (π.χ. ιδρυματικό email, ονοματεπώνυμο και πατρώνυμο, κινητό τηλέφωνο), εμφανίζεται μήνυμα προειδοποίησης.

• Ο χρήστης επιλέγει «Αποθήκευση και Δημιουργία Διδάσκοντα». • Εμφανίζεται μήνυμα επιτυχούς δημιουργίας με μοναδικό αναγνωριστικό (user ID) και γίνεται ανακατεύθυνση στη σελίδα διαχείρισης του νέου λογαριασμού. • Ο χρήστης ενημερώνεται για την αυτόματη αποστολή email ειδοποίησης του Διδάσκοντα σχετικά με τη δημιουργία του νέου του λογαριασμού, καθώς και με οδηγίες ενεργοποίησης και εισόδου στην πλατφόρμα. • Εμφανίζεται η επιλογή «Αποστολή Υπενθύμισης Ενεργοποίησης Λογαριασμού» και ο χρήστης μπορεί να την επιλέξει, πατώντας στο σχετικό κουμπί. • Ο Διδάσκων εμφανίζεται στο σύστημα της πλατφόρμας για ανάθεση μαθημάτων, δημιουργία ωρολογίου προγράμματος κ.ο.κ., ενώ ο λογαριασμός του είναι σε κατάσταση εκκρεμούς ενεργοποίησης μέχρι την πρώτη του είσοδο στην πλατφόρμα, είτε μέσω του παρεχόμενου από το Ίδρυμα συστήματος πιστοποίησης SSO είτε μέσω της πλατφόρμας και με τη δημιουργία ενός κωδικού εισόδου. **Σενάριο 1: Επιτυχής αποστολή ειδοποίησης** • Εφόσον το σύστημα απέστειλε επιτυχώς μήνυμα με οδηγίες και σύνδεσμο ενεργοποίησης του λογαριασμού μέσω email προς τον Διδάσκοντα, εμφανίζεται μήνυμα επιτυχίας με την ημερομηνία και ώρα αποστολής, καθώς και λήξης ισχύος του συνδέσμου. **Σενάριο 2: Αποτυχία αποστολής ειδοποίησης** • Αν προκύψει σφάλμα κατά την αποστολή email, εμφανίζεται μήνυμα αποτυχίας και κωδικό σφάλματος (αν είναι διαθέσιμος) και δυνατότητα επανάληψης της διαδικασίας αποστολής ειδοποίησης.

Πίνακας Γ-US-15: Δημιουργία λογαριασμού Διδάσκοντα

Τίτλος: Διαχείριση εκκρεμών λογαριασμών Διδασκόντων προς ενεργοποίηση
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω λίστα εκκρεμών λογαριασμών Διδασκόντων προς ενεργοποίηση **Ώστε** να επεξεργαστώ στοιχεία, να αποστείλω εκ νέου ειδοποίηση ή να διαγράψω εκκρεμείς λογαριασμούς που δεν ισχύουν.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού. • Ο χρήστης πλοηγείται μέσω του μενού στην επιλογή «Προβολή Εκκρεμών Λογαριασμών

Διδασκόντων».

- Εμφανίζεται λίστα με τους εκκρεμείς λογαριασμούς Διδασκόντων προς ενεργοποίηση.
- Η λίστα περιλαμβάνει το ονοματεπώνυμο του Διδάσκοντα, το Τμήμα που έχει οριστεί, την ημερομηνία δημιουργίας λογαριασμού, την ημερομηνία τελευταίας αποστολής ειδοποίησης ενεργοποίησης, καθώς και μήνυμα εάν έχει παρέλθει ο χρόνος ισχύος του συνδέσμου ενεργοποίησης.
- Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά.
- Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Αποστολή Ειδοποίησης»: Ναι / Όχι / Λήξαν Σύνδεσμος ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα.
- Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, ημερομηνία, Τμήμα, τηλέφωνο, email, αριθμό μητρώου), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν.
- Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα.
- Ο χρήστης μπορεί να επιλέξει την επαναποστολή ειδοποίησης σε επιλεγμένο εκκρεμή λογαριασμό, πατώντας το αντίστοιχο κουμπί δίπλα από τα στοιχεία του χρήστη.
- Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διδάσκοντα.
- Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διδάσκοντα με όλα τα σχετικά πεδία.
- Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, οργανική θέση / Τμήμα που ανήκει, μαθήματα που διδάσκει, κατάσταση λογαριασμού (ενεργός / ανενεργός).
- Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Ο χρήστης μπορεί να επιλέξει την επαναποστολή ειδοποίησης ενεργοποίησης λογαριασμού, οπότε ενημερώνονται τα πεδία ημερομηνίας/ώρας τελευταίας αποστολής ειδοποίησης και χρόνος λήξης συνδέσμου ενεργοποίησης εφόσον η αποστολή ήταν επιτυχής, διαφορετικά ο χρήστης βλέπει σχετικό μήνυμα αποτυχίας αποστολής και παραπομπή για νέα προσπάθεια.

• Ο χρήστης μπορεί να διαγράψει κάποιον εκκρεμή λογαριασμό, εφόσον αυτός δεν έχει ενεργοποιηθεί, επιλέγοντας «Διαγραφή Εκκρεμούς Λογαριασμού» και κατόπιν ελέγχουν από το σύστημα ότι ο λογαριασμός δεν έχει ενεργοποιηθεί και δεν έχει οριστεί / αντιστοιχιστεί σε κάποιο μάθημα ή ωρολόγιο πρόγραμμα, ολοκληρώνεται η διαγραφή και ο χρήστης επιστρέφει στη λίστα εκκρεμών λογαριασμών. • Ο χρήστης επιστρέφει στη λίστα εκκρεμών λογαριασμών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-16: Διαχείριση εκκρεμών λογαριασμών Διδασκόντων προς ενεργοποίηση

Τίτλος: Επεξεργασία νέου λογαριασμού Διδάσκοντα (αυτόματη δημιουργία)
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τα στοιχεία νέου λογαριασμού Διδάσκοντα που δημιουργήθηκε αυτόματα μετά την είσοδό του με χρήση SSO **Ώστε** να συμπληρώσω τα απαραίτητα πεδία για την ορθή προβολή και χρήση στις λειτουργίες της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού. • Ο χρήστης πλοηγείται μέσω του μενού στην επιλογή «Προβολή Νέων Λογαριασμών Διδασκόντων». • Εμφανίζεται λίστα με τους νέους λογαριασμούς Διδασκόντων που δημιουργήθηκαν αυτόματα μετά από πιστοποίηση και είσοδο μέσω SSO, για τους οποίους απαιτείται έλεγχος και επιβεβαίωση από την χρήστη. • Η λίστα περιλαμβάνει τα στοιχεία που έχουν ληφθεί από το σύστημα SSO του Ιδρύματος όπως το ονοματεπώνυμο του Διδάσκοντα, το Τμήμα που έχει οριστεί, την ημερομηνία δημιουργίας λογαριασμού. • Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά. • Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα. • Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό

πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, ημερομηνία, Τμήμα, αριθμό μητρώου), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν. • Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα. • Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διδάσκοντα. • Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διδάσκοντα με όλα τα σχετικά πεδία. • Ο χρήστης μπορεί να τροποποιήσει τα στοιχεία του λογαριασμού. • Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, οργανική θέση / Τμήμα που ανήκει, μαθήματα που διδάσκει, κατάσταση λογαριασμού (ενεργός / ανενεργός). • Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εφόσον ο χρήστης έχει συμπληρώσει το υποχρεωτικό πεδίο για ενεργούς Διδάσκοντες «μαθήματα που διδάσκει», στη φόρμα εμφανίζεται επιλογή οριστικής επιβεβαίωσης λογαριασμού. • Ο χρήστης μπορεί να επιλέξει την οριστική επιβεβαίωση λογαριασμού οπότε και ο Διδάσκων καταχωρείται ως επιβεβαιωμένος λογαριασμός με δυνατότητα πλέον ορισμού του σε μαθήματα και ωρολόγια προγράμματα, ενώ αφαιρείται από τη λίστα νέων λογαριασμών διδασκόντων. • Ο χρήστης μπορεί να ορίσει την κατάσταση λογαριασμού ως ανενεργή, οπότε και απενεργοποιείται ο λογαριασμός του Διδάσκοντα, διακόπτοντας τη δυνατότητα εισόδου στην πλατφόρμα είτε τοπικά είτε μέσω SSO. • Ο χρήστης επιστρέφει στη λίστα εκκρεμών λογαριασμών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-17: Επεξεργασία νέου λογαριασμού Διδάσκοντα (αυτόματη δημιουργία)

Τίτλος: Προβολή και διαχείριση ενεργών και επιβεβαιωμένων Διδασκόντων
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω λίστα Διδασκόντων και να διαχειριστώ τον λογαριασμό Διδάσκοντα **Ώστε** να επεξεργαστώ τα στοιχεία τους, τα μαθήματα που διδάσκουν, να αλλάξω την κατάσταση λογαριασμού και να ελέγξω ιστορικό ενεργειών τους.
Κριτήρια Αποδοχής (Acceptance criteria):

- Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Διδασκόντων» από το κεντρικό μενού.
- Εμφανίζεται λίστα με τους ενεργούς και επιβεβαιωμένους λογαριασμούς Διδασκόντων.
- Η λίστα περιλαμβάνει το ονοματεπώνυμο του Διδάσκοντα, το Τμήμα που έχει οριστεί, τα μαθήματα που διδάσκει και την κατάσταση λογαριασμού.
- Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά.
- Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Κατάσταση Λογαριασμού»: ενεργός / ανενεργός / όλοι ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων ή «Μάθημα»: επιλογή από λίστα διαθέσιμων μαθημάτων) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα.
- Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, Τμήμα, μάθημα, τηλέφωνο, email, αριθμό μητρώου), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν.
- Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα.
- Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Διδάσκοντα.
- Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Διδάσκοντα με όλα τα σχετικά πεδία.
- Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, οργανική θέση / Τμήμα που ανήκει, μαθήματα που διδάσκει, κατάσταση λογαριασμού (ενεργός / ανενεργός).
- Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Ο χρήστης μπορεί να επιλέξει την προβολή ιστορικού κινήσεων εισόδου/εξόδου και ενεργειών που έχει πραγματοποιήσει ο συγκεκριμένος Διδάσκων, επιλέγοντας το κουμπί «Προβολή Ιστορικού Χρήστη».
- Μπορεί να επιλέξει τη λήψη του ιστορικού χρήστη σε αρχείο CSV και τη διαγραφή του ιστορικού παλαιότερων μηνών με ορισμό χρονικού ορίου διαγραφής (π.χ. 1, 2, 3 μήνες) από λίστα (το ιστορικό διατηρείται κατόπιν σε εξωτερικό αρχείο στον διακομιστή).
- Ο χρήστης επιστρέφει στη λίστα ενεργών λογαριασμών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-18: Προβολή και διαχείριση ενεργών και επιβεβαιωμένων Διδασκόντων

Τίτλος: Μαζική εισαγωγή Διδασκόντων από αρχείο CSV/Excel
Ως Κύριος Διαχειριστής **Θέλω να** εισαγάγω μαζικά στοιχεία Διδασκόντων από λίστα σε αρχείο CSV ή Excel **Ώστε** να καταχωρηθούν τα απαιτούμενα στοιχεία για να δημιουργηθούν οι λογαριασμοί τους και να ενημερωθούν αυτόματα μέσω email.

Κριτήρια Αποδοχής (Acceptance criteria):

- Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού.
- Ο χρήστης επιλέγει τον σύνδεσμο «Μαζική Εισαγωγή Διδασκόντων».
- Εμφανίζεται παράθυρο επιλογής τοπικού αρχείου CSV ή Excel.
- Ο χρήστης επιλέγει το αρχείο προς εισαγωγή.
- Το σύστημα εμφανίζει μήνυμα επιτυχούς λήψης ή αποτυχίας λήψης για έναν από τους παρακάτω λόγους: λανθασμένης μορφής αρχείου ή μη συμβατή κωδικοποίηση ή κενό αρχείο ή υπέρβαση ορίου μεγέθους αρχείου.

Σενάριο 1: Επιτυχής λήψη και έλεγχος δεδομένων αρχείου

- Το σύστημα εμφανίζει μήνυμα επιτυχούς λήψης του αρχείου.
- Το αρχείο περιλαμβάνει όλα τα υποχρεωτικά πεδία για τη δημιουργία λογαριασμού Διδάσκοντα (ιδρυματικό email, ονοματεπώνυμο, αριθμό μητρώου).
- Το αρχείο περιλαμβάνει ορθά δεδομένα (σωστή μορφή email και domain, ονοματεπώνυμο, αριθμό μητρώου, τηλέφωνο, κωδικούς μαθημάτων).
- Το αρχείο δεν περιέχει διπλότυπες εγγραφές ή δεδομένα που αντιστοιχούν σε ήδη υπάρχοντες λογαριασμούς (ίδιο ιδρυματικό email ή ονοματεπώνυμο ή αριθμό μητρώου).
- Το σύστημα εμφανίζει μήνυμα επιτυχούς ολοκλήρωσης ελέγχου και κουμπί για την οριστική εισαγωγή των δεδομένων στο σύστημα.

Σενάριο 1α: Ακύρωση διαδικασίας

- Ο χρήστης επιλέγει το κουμπί «Ακύρωση Εισαγωγής Λίστας Διδασκόντων», οπότε η διαδικασία τερματίζεται χωρίς άλλη ενέργεια και εμφανίζεται μήνυμα ακύρωσης.

Σενάριο 1β: Συνέχεια διαδικασίας

- Ο χρήστης επιλέγει το κουμπί «Οριστική Εισαγωγή Λίστας Διδασκόντων».
- Το σύστημα εμφανίζει μήνυμα επιβεβαίωσης της οριστικής εισαγωγής και ενημέρωση μη

αντιστρεψιμότητας της ενέργειας.

- Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Οριστικής Εισαγωγής Λίστας Διδασκόντων».

Σενάριο 1β-1: Επιτυχής εισαγωγή δεδομένων αρχείου

- Το σύστημα εμφανίζει μήνυμα επιτυχούς ολοκλήρωσης της εισαγωγής δεδομένων από αρχείο.
- Το σύστημα εμφανίζει λίστα με τους Διδάσκοντες για τους οποίους δημιουργήθηκαν λογαριασμοί, με επισήμανση επιτυχούς μαζικής και αυτόματης αποστολής ειδοποίησης μέσω email.
- Ο χρήστης μπορεί να εξάγει τη λίστα σε αρχείο PDF για αποθήκευση ή εκτύπωση.
- Σε περίπτωση αποτυχίας αυτόματης αποστολής ορισμένων μηνυμάτων email, εμφανίζεται μήνυμα ενημέρωσης προς τον χρήστη με αναφορά στον αριθμό ή τη λίστα των παραληπτών που δεν έλαβαν την ειδοποίηση. Ο χρήστης μπορεί να επαναλάβει τη διαδικασία μαζικής αποστολής, επιλέγοντας το κουμπί «Επαναποστολή ειδοποιήσεων σε μη απεσταλμένα email».

Σενάριο 1β-2: Αποτυχία εισαγωγή δεδομένων αρχείου

- Εμφανίζεται μήνυμα αποτυχίας και κωδικός σφάλματος (εφόσον υπάρχει) και ο χρήστης ενημερώνεται για την επανάληψη της διαδικασίας εισαγωγής.

Σενάριο 2: Αποτυχία λήψης ή ελέγχου δεδομένων αρχείου

- Διακόπτεται η διαδικασία εισαγωγής και επεξεργασίας του αρχείου.

Σενάριο 2α: Αποτυχία λήψης αρχείου

- Το σύστημα εμφανίζει μήνυμα αποτυχίας λήψης για έναν από τους παρακάτω λόγους: λανθασμένης μορφής αρχείου ή μη συμβατή κωδικοποίηση ή κενό αρχείο ή υπέρβαση ορίου μεγέθους αρχείου.

Σενάριο 2β: Αποτυχία ελέγχου δεδομένων αρχείου

- Το σύστημα εμφανίζει μήνυμα αποτυχίας ελέγχου των δεδομένων της εισαγωγής από αρχείο.
- Εμφανίζεται λίστα των λογαριασμών Διδασκόντων που εντοπίστηκε σφάλμα, με επισήμανση των δεδομένων για τα οποία ο έλεγχος απέτυχε.
- Ο χρήστης παροτρύνεται να προβεί στις απαραίτητες διορθώσεις στα δεδομένα του αρχείου που εισήγαγε, σύμφωνα με τις επισημάνσεις της λίστας ελέγχου και κατόπιν να επαναλάβει τη διαδικασία εισαγωγής.

Πίνακας Γ-US-19: Μαζική εισαγωγή Διδασκόντων από αρχείο CSV/Excel

Τίτλος: Προβολή και διαχείριση εγγεγραμμένων Φοιτητών
Ως Διαχειριστής Τμήματος **Θέλω να** προβάλλω λίστα και να διαχειριστώ τους εγγεγραμμένους Φοιτητές **Ώστε** να επεξεργαστώ τα στοιχεία τους, τα μαθήματα που παρακολουθούν, να αλλάξω την κατάσταση φοίτησης και να ελέγξω ιστορικό ενεργειών τους.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Φοιτητών» από το κεντρικό μενού. • Εμφανίζεται λίστα με τους εγγεγραμμένους Φοιτητές. • Η λίστα περιλαμβάνει το ονοματεπώνυμο του Φοιτητή, το Τμήμα που είναι εγγεγραμμένος, το έτος εισαγωγής και το εξάμηνο που παρακολουθεί και την κατάσταση φοίτησης. • Ο χρήστης μπορεί να επιλέξει την ταξινόμηση της λίστας βάσει συγκεκριμένου πεδίου της λίστας, πατώντας στον τίτλο του πεδίου αυτού, οπότε η λίστα εμφανίζεται σε αύξουσα ή φθίνουσα σειρά. • Ο χρήστης μπορεί να κάνει χρήση φίλτρου προβολής συγκεκριμένων αποτελεσμάτων, επιλέγοντας το σχετικό φίλτρο («Κατάσταση Φοίτησης»: ενεργός, σε αναστολή φοίτησης, απόφοιτος, μετεγγραφή, διαγραμμένος, αποχωρήσας, όλοι ή «Τμήμα»: επιλογή από λίστα διαθέσιμων Τμημάτων ή «Έτος Εισαγωγής»: επιλογή από λίστα ετών) οπότε η λίστα ενημερώνεται με τα συγκεκριμένα αποτελέσματα. • Ο χρήστης μπορεί να πραγματοποιήσει αναζήτηση στη λίστα, συμπληρώνοντας στο σχετικό πεδίο λέξη-κλειδί ή τμήμα αυτής (π.χ. όνομα, επώνυμο, Τμήμα, μάθημα, τηλέφωνο, email, αριθμό μητρώου), οπότε η λίστα εμφανίζει μόνο τις εγγραφές που ταιριάζουν. • Εάν η λίστα είναι κενή, εμφανίζεται κατάλληλο μήνυμα. • Ο χρήστης μπορεί να επιλέξει έναν λογαριασμό από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στο ονοματεπώνυμο του Φοιτητή. • Επιλέγοντας έναν λογαριασμό από τη λίστα, εμφανίζεται η φόρμα προφίλ του συγκεκριμένου λογαριασμού Φοιτητή με όλα τα σχετικά πεδία. • Ο χρήστης μπορεί να τροποποιήσει στοιχεία του λογαριασμού όπως ονοματεπώνυμο, αριθμό κινητού ή σταθερού τηλεφώνου, μαθήματα που παρακολουθεί, κατάσταση φοίτησης. • Τα πεδία που δεν μπορούν να τροποποιηθούν (κλειδωμένα πεδία), εμφανίζονται με γκρι χρώμα ή σχετική σήμανση.

• Κατά την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Ο χρήστης μπορεί να επιλέξει την προβολή ιστορικού κινήσεων εισόδου/εξόδου και ενεργειών που έχει πραγματοποιήσει ο συγκεκριμένος Φοιτητής, επιλέγοντας το κουμπί «Προβολή Ιστορικού Χρήστη». • Μπορεί να επιλέξει τη λήψη του ιστορικού χρήστη σε αρχείο CSV και τη διαγραφή του ιστορικού παλαιότερων μηνών με ορισμό χρονικού ορίου διαγραφής (π.χ. 1, 2, 3 μήνες) από λίστα (το ιστορικό διατηρείται κατόπιν σε εξωτερικό αρχείο στον διακομιστή). • Ο χρήστης επιστρέφει στη λίστα εγγεγραμμένων Φοιτητών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-20: Προβολή και διαχείριση εγγεγραμμένων Φοιτητών

Τίτλος: Μαζική εισαγωγή Φοιτητών από αρχείο CSV/Excel
Ως Διαχειριστής Τμήματος **Θέλω να** εισαγάγω μαζικά στοιχεία Φοιτητών από λίστα σε αρχείο CSV ή Excel **Ώστε** να καταχωρηθούν ή ενημερωθούν τα στοιχεία λογαριασμών και η κατάσταση φοίτησης.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Λογαριασμών Χρηστών» από το κεντρικό μενού. • Ο χρήστης επιλέγει τον σύνδεσμο «Μαζική Εισαγωγή και Ενημέρωση Φοιτητών». • Εμφανίζεται παράθυρο επιλογής τοπικού αρχείου CSV ή Excel. • Ο χρήστης επιλέγει το αρχείο προς εισαγωγή. • Το σύστημα εμφανίζει μήνυμα επιτυχούς λήψης ή αποτυχίας λήψης για έναν από τους παρακάτω λόγους: λανθασμένης μορφής αρχείου ή μη συμβατή κωδικοποίηση ή κενό αρχείο ή υπέρβαση ορίου μεγέθους αρχείου. **Σενάριο 1: Επιτυχής λήψη και έλεγχος δεδομένων αρχείου** • Το σύστημα εμφανίζει μήνυμα επιτυχούς λήψης του αρχείου. • Το αρχείο περιλαμβάνει όλα τα υποχρεωτικά πεδία για τη δημιουργία λογαριασμού Φοιτητή (ιδρυματικό email, ονοματεπώνυμο, αριθμό μητρώου) ή για την αντιστοίχιση προς ενημέρωση υφιστάμενου λογαριασμού (ιδρυματικό email). • Το αρχείο περιλαμβάνει ορθά δεδομένα (σωστή μορφή email και domain, ονοματεπώνυμο, αριθμό μητρώου, τηλέφωνο, κατάσταση φοίτησης).

- Το αρχείο δεν περιέχει διπλότυπες εγγραφές.
- Εφόσον το αρχείο περιλαμβάνει υφιστάμενους λογαριασμούς προς ενημέρωση, γίνεται έλεγχος για πεδία που καταχωρούνται αυτόματα κατά την πρώτη είσοδο του Φοιτητή μέσω της υπηρεσίας SSO στα οποία δεν επιτρέπεται τροποποίηση και εμφανίζει σχετικό ενημερωτικό μήνυμα.
- Το σύστημα εμφανίζει μήνυμα επιτυχούς ολοκλήρωσης ελέγχου και κουμπί για την οριστική εισαγωγή των δεδομένων στο σύστημα.

Σενάριο 1α: Ακύρωση διαδικασίας

- Ο χρήστης επιλέγει το κουμπί «Ακύρωση Εισαγωγής και Ενημέρωσης Φοιτητών», οπότε η διαδικασία τερματίζεται χωρίς άλλη ενέργεια και εμφανίζεται μήνυμα ακύρωσης.

Σενάριο 1β: Συνέχεια διαδικασίας

- Ο χρήστης επιλέγει το κουμπί «Οριστική Εισαγωγής και Ενημέρωσης Φοιτητών».
- Το σύστημα εμφανίζει μήνυμα επιβεβαίωσης της οριστικής εισαγωγής και ενημέρωση μη αντιστρεψιμότητας της ενέργειας.
- Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Οριστικής Εισαγωγής και Ενημέρωσης Φοιτητών».

Σενάριο 1β-1: Επιτυχής εισαγωγή δεδομένων αρχείου

- Το σύστημα εμφανίζει μήνυμα επιτυχούς ολοκλήρωσης της εισαγωγής δεδομένων από αρχείο.
- Το σύστημα εμφανίζει λίστα με τους Φοιτητές για τους οποίους δημιουργήθηκαν λογαριασμοί, καθώς και των υφιστάμενων λογαριασμών που ενημερώθηκαν τα στοιχεία τους.
- Ο χρήστης μπορεί να εξάγει τη λίστα σε αρχείο PDF για αποθήκευση ή εκτύπωση.

Σενάριο 1β-2: Αποτυχία εισαγωγή δεδομένων αρχείου

- Εμφανίζεται μήνυμα αποτυχίας και κωδικός σφάλματος (εφόσον υπάρχει) και ο χρήστης ενημερώνεται για την επανάληψη της διαδικασίας εισαγωγής.

Σενάριο 2: Αποτυχία λήψης ή ελέγχου δεδομένων αρχείου

- Διακόπτεται η διαδικασία εισαγωγής και επεξεργασίας του αρχείου.

Σενάριο 2α: Αποτυχία λήψης αρχείου

- Το σύστημα εμφανίζει μήνυμα αποτυχίας λήψης για έναν από τους παρακάτω λόγους: λανθασμένης μορφής αρχείου ή μη συμβατή κωδικοποίηση ή κενό αρχείο ή υπέρβαση ορίου

μεγέθους αρχείου.

Σενάριο 2β: Αποτυχία ελέγχου δεδομένων αρχείου

- Το σύστημα εμφανίζει μήνυμα αποτυχίας ελέγχου των δεδομένων της εισαγωγής από αρχείο.
- Εμφανίζεται λίστα των λογαριασμών Φοιτητών που εντοπίστηκε σφάλμα, με επισήμανση των δεδομένων για τα οποία ο έλεγχος απέτυχε.
- Ο χρήστης παροτρύνεται να προβεί στις απαραίτητες διορθώσεις στα δεδομένα του αρχείου που εισήγαγε, σύμφωνα με τις επισημάνσεις της λίστας ελέγχου και κατόπιν να επαναλάβει τη διαδικασία εισαγωγής.

Πίνακας Γ-US-21: Μαζική εισαγωγή Φοιτητών από αρχείο CSV/Excel

Τίτλος: Διαχείριση Σχολών: Προβολή και Τροποποίηση στοιχείων
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω τις Σχολές του Πανεπιστημίου **Ώστε** να ελέγξω ή τροποποιήσω τα στοιχεία κάποιας Σχολής.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Σχολών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τις Σχολές (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Σχολών του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Σχολών, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργές Σχολές του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει μία Σχολή από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο της. • Επιλέγοντας μία Σχολή από τη λίστα, εμφανίζεται η φόρμα στοιχείων της συγκεκριμένης Σχολής με όλα τα σχετικά πεδία (κωδικός, τίτλος, διεύθυνση ιστοτόπου, κατάσταση: ενεργή / ανενεργή). • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Το πεδίο «Κατάσταση» εμφανίζεται κλειδωμένο, εφόσον η Σχολή χρησιμοποιείται σε ωρολόγιο πρόγραμμα (μάθημα Τμήματος της Σχολής είναι σε χρήση) ή έχει ενεργά Τμήματα.

• Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία στοιχείων της Σχολής. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων** • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Σχολής). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Ο χρήστης επιστρέφει στη λίστα Σχολών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-22: Διαχείριση Σχολών: Προβολή και Τροποποίηση στοιχείων

Τίτλος: Διαχείριση Σχολών: Προβολή στοιχείων
Ως Κύριος Διαχειριστής ή Διαχειριστής Τμήματος **Θέλω να** προβάλλω τις Σχολές του Πανεπιστημίου **Ώστε** να ελέγξω τα στοιχεία κάποιας Σχολής.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Σχολών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να προβάλλει τις Σχολές (προηγούμενο, τρέχον ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Σχολών του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Σχολών, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργές Σχολές του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει μία Σχολή από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο της.

• Επιλέγοντας μία Σχολή από τη λίστα, εμφανίζεται η φόρμα στοιχείων της συγκεκριμένης Σχολής με όλα τα σχετικά πεδία (κωδικός, τίτλος, διεύθυνση ιστοτόπου, κατάσταση: ενεργή / ανενεργή). • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Εφόσον η προβολή αφορά προηγούμενο εξάμηνο ή η ενέργεια πραγματοποιείται από Διαχειριστή Τμήματος, όλα τα πεδία είναι κλειδωμένα, ενώ δεν μπορεί να γίνει τροποποίηση και αποθήκευση. • Ο χρήστης επιστρέφει στη λίστα Σχολών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-23: Διαχείριση Σχολών: Προβολή στοιχείων

Τίτλος: Διαχείριση Σχολών: Προσθήκη Σχολής
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τις Σχολές του Πανεπιστημίου **Ώστε** να προσθέσω μια νέα Σχολή.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Σχολών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το επερχόμενο εξάμηνο για το οποίο επιθυμεί να εισάγει νέα Σχολή (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται κουμπί «Εισαγωγή Νέας Σχολής». • Ο χρήστης επιλέγει το κουμπί «Εισαγωγή Νέας Σχολής». • Εμφανίζεται η φόρμα στοιχείων προσθήκης νέας Σχολής με όλα τα σχετικά πεδία (κωδικός, τίτλος, διεύθυνση ιστοτόπου, κατάσταση: ενεργή / ανενεργή). • Ο χρήστης συμπληρώνει τα υποχρεωτικά πεδία «Κωδικός» και «Τίτλος Σχολής», καθώς και τα λοιπά πεδία. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων**

• Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Σχολής ή ίδιος Κωδικός). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Εμφανίζεται στον χρήση η φόρμα με τα στοιχεία της νέας Σχολής. • Ο χρήστης επιστρέφει στη λίστα Σχολών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-24: Διαχείριση Σχολών: Προσθήκη Σχολής

Τίτλος: Διαχείριση Σχολών: Διαγραφή Σχολής
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τις Σχολές του Πανεπιστημίου **Ώστε** να διαγράψω μία Σχολή που καταχωρήθηκε εκ παραδρομής.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Σχολών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το επερχόμενο εξάμηνο για το οποίο επιθυμεί να διαγράψει κάποια Σχολή (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Σχολών του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Σχολών, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργές Σχολές του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει μία Σχολή από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο της. • Επιλέγοντας μία Σχολή από τη λίστα, εμφανίζεται η φόρμα στοιχείων της Σχολής με όλα τα σχετικά πεδία (κωδικός, τίτλος, διεύθυνση ιστοτόπου, κατάσταση: ενεργή / ανενεργή). • Εφόσον η Σχολή δεν έχει οριστεί σε καμία σχέση (δεν περιέχει Τμήματα και δεν έχει ιστορικό χρήσης σε ωρολόγια προγράμματα) ή η εγγραφή αφορά πρόσφατη τροποποίηση στοιχείων Σχολής από προηγούμενο εξάμηνο, εμφανίζεται στον χρήστη κουμπί «Διαγραφή Σχολής». • Ο χρήστης επιλέγει το κουμπί «Διαγραφή Σχολής». • Εμφανίζεται μήνυμα επιβεβαίωσης της πρόθεσης διαγραφής.

• Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Διαγραφής Σχολής». **Σενάριο 1: Αποτυχία διαγραφής** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας διαγραφής με το λόγο της αποτυχίας (π.χ. η Σχολή έχει δηλωμένα Τμήματα ή η Σχολή περιλαμβάνεται σε ωρολόγιο πρόγραμμα) και κωδικό σφάλματος (εάν υπάρχει). **Σενάριο 2: Επιτυχής διαγραφή** • Ο χρήστης επιστρέφει στη λίστα Σχολών. • Εμφανίζεται μήνυμα επιτυχούς διαγραφής.

Πίνακας Γ-US-25: Διαχείριση Σχολών: Διαγραφή Σχολής

Τίτλος: Διαχείριση Τμημάτων: Προβολή και Τροποποίηση στοιχείων
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω τα Τμήματα των Σχολών του Πανεπιστημίου **Ώστε** να ελέγξω ή τροποποιήσω τα στοιχεία κάποιου Τμήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Τμημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τα Τμήματα (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Τμημάτων του εξαμήνου που επέλεξε ανά Σχολή. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Τμημάτων, οπότε ενημερώνεται η λίστα προβολής με τα ανενεργά Τμήματα του εξαμήνου που επέλεξε ανά Σχολή. • Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Τμημάτων της, πατώντας επάνω στον τίτλο της Σχολής. • Ο χρήστης μπορεί να επιλέξει ένα Τμήμα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του. • Επιλέγοντας ένα Τμήμα από τη λίστα, εμφανίζεται η φόρμα στοιχείων του συγκεκριμένου Τμήματος με όλα τα σχετικά πεδία (κωδικός, Σχολή, τίτλος, διεύθυνση ιστοτόπου, διάρκεια φοίτησης, κατάσταση: ενεργό / ανενεργό).

• Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Το πεδίο «Κατάσταση» εμφανίζεται κλειδωμένο, εφόσον η Σχολή χρησιμοποιείται σε ωρολόγιο πρόγραμμα (μάθημα Τμήματος της Σχολής είναι σε χρήση). • Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία στοιχείων του Τμήματος. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων** • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Τμήματος). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Ο χρήστης επιστρέφει στη λίστα Τμημάτων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-26: Διαχείριση Τμημάτων: Προβολή και Τροποποίηση στοιχείων

Τίτλος: Διαχείριση Τμημάτων: Προβολή στοιχείων
Ως Κύριος Διαχειριστής ή Διαχειριστής Τμήματος **Θέλω να** προβάλλω τα Τμήματα των Σχολών του Πανεπιστημίου **Ώστε** να ελέγξω τα στοιχεία κάποιου Τμήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Τμημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να προβάλλει τα Τμήματα (προηγούμενο, τρέχον ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Τμημάτων του εξαμήνου που επέλεξε.

• Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Τμημάτων, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργά Τμήματα του εξαμήνου που επέλεξε. • Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Τμημάτων της, πατώντας επάνω στον τίτλο της Σχολής. • Ο χρήστης μπορεί να επιλέξει ένα Τμήμα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του. • Επιλέγοντας ένα Τμήμα από τη λίστα, εμφανίζεται η φόρμα στοιχείων του συγκεκριμένου Τμήματος με όλα τα σχετικά πεδία (κωδικός, Σχολή, τίτλος, διεύθυνση ιστοτόπου, διάρκεια φοίτησης, κατάσταση: ενεργό / ανενεργό). • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Εφόσον η προβολή αφορά προηγούμενο εξάμηνο ή η ενέργεια πραγματοποιείται από Διαχειριστή Τμήματος, όλα τα πεδία είναι κλειδωμένα, ενώ δεν μπορεί να γίνει τροποποίηση και αποθήκευση. • Ο χρήστης επιστρέφει στη λίστα Τμημάτων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-27: Διαχείριση Τμημάτων: Προβολή στοιχείων

Τίτλος: Διαχείριση Τμημάτων: Προσθήκη Τμήματος σε Σχολή
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τα Τμήματα των Σχολών του Πανεπιστημίου **Ώστε** να προσθέσω ένα νέο Τμήμα σε Σχολή.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Τμημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το επερχόμενο εξάμηνο για το οποίο επιθυμεί να προσθέσει νέο Τμήμα (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται κουμπί «Εισαγωγή Νέου Τμήματος». • Ο χρήστης επιλέγει το κουμπί «Εισαγωγή Νέου Τμήματος». • Εμφανίζεται η φόρμα στοιχείων προσθήκης νέου Τμήματος με όλα τα σχετικά πεδία (κωδικός, Σχολή, τίτλος, διεύθυνση ιστοτόπου, διάρκεια φοίτησης, κατάσταση: ενεργό / ανενεργό). • Ο χρήστης συμπληρώνει τα υποχρεωτικά πεδία «κωδικός» και «τίτλος», «Σχολή», καθώς και τα λοιπά πεδία.

• Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων** • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Τμήματος ή ίδιος Κωδικός). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Εμφανίζεται στον χρήση η φόρμα με τα στοιχεία του νέου Τμήματος. • Ο χρήστης επιστρέφει στη λίστα Τμημάτων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-28: Διαχείριση Τμημάτων: Προσθήκη Τμήματος σε Σχολή

Τίτλος: Διαχείριση Τμημάτων: Διαγραφή Τμήματος
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τα Τμήματα του Πανεπιστημίου **Ώστε** να διαγράψω ένα Τμήμα που καταχωρήθηκε εκ παραδρομής.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Τμημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τις Σχολές (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των ενεργών Τμημάτων του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Τμημάτων, οπότε ενημερώνεται η λίστα προβολής με τα ανενεργά Τμήματα του εξαμήνου που επέλεξε. • Ο χρήστης μπορεί να επιλέξει ένα Τμήμα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο της.

• Επιλέγοντας ένα Τμήμα από τη λίστα, εμφανίζεται η φόρμα στοιχείων του συγκεκριμένου Τμήματος με όλα τα σχετικά πεδία (κωδικός, Σχολή, τίτλος, διεύθυνση ιστοτόπου, λίστα μαθημάτων, διάρκεια φοίτησης, κατάσταση: ενεργό / ανενεργό). • Εφόσον το Τμήμα δεν έχει οριστεί σε καμία σχέση (δεν έχει ιστορικό χρήσης σε ωρολόγια προγράμματα), εμφανίζεται στον χρήστη κουμπί «Διαγραφή Τμήματος». • Ο χρήστης επιλέγει το κουμπί «Διαγραφή Τμήματος». • Εμφανίζεται μήνυμα επιβεβαίωσης της πρόθεσης διαγραφής. • Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Διαγραφής Τμήματος». **Σενάριο 1: Αποτυχία διαγραφής** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας διαγραφής με το λόγο της αποτυχίας (π.χ. το Τμήμα έχει δηλωμένα μαθήματα σε ωρολόγιο πρόγραμμα) και κωδικό σφάλματος (εάν υπάρχει). **Σενάριο 2: Επιτυχής διαγραφή** • Ο χρήστης επιστρέφει στη λίστα Τμημάτων. • Εμφανίζεται μήνυμα επιτυχούς διαγραφής.

Πίνακας Γ-US-29: Διαχείριση Τμημάτων: Διαγραφή Τμήματος

Τίτλος: Διαχείριση Μαθημάτων: Προβολή και Τροποποίηση στοιχείων
Ως Διαχειριστής Τμήματος **Θέλω να** προβάλλω τα Μαθήματα ενός Τμήματος **Ώστε** να ελέγξω ή τροποποιήσω τα στοιχεία κάποιου Μαθήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Μαθημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τα Μαθήματα (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των Σχολών, των Τμημάτων τους και των ενεργών Μαθημάτων κάτω από κάθε Τμήμα, για το επιλεγμένο εξάμηνο. • Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Τμημάτων της, πατώντας επάνω στον τίτλο της Σχολής και κατόπιν πατώντας επάνω στον τίτλο Τμήματος εμφανίζεται λίστα των προσφερόμενων Μαθημάτων.

- Ο χρήστης μπορεί να επιλέξει διαφορετική εμφάνιση της λίστας, σε μία ενιαία αλφαβητική σειρά όλων των Μαθημάτων, οπότε εμφανίζεται το όνομα Μαθήματος και σε παρένθεση το Τμήμα και η Σχολή.
- Ο χρήστης μπορεί να αναζητήσει το Μάθημα που θέλει να προβάλει ή επεξεργαστεί, εισάγοντας στο κατάλληλο πεδίο λέξεις-κλειδιά, οπότε και εμφανίζεται λίστα με τα αποτελέσματα της αναζήτησης ή μήνυμα κενής λίστας εάν δεν βρεθεί αποτέλεσμα.
- Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Μαθημάτων, οπότε ενημερώνεται η λίστα προβολής με τα ανενεργά Μαθήματα του εξαμήνου που επέλεξε.
- Ο χρήστης μπορεί να επιλέξει ένα Μάθημα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του.
- Επιλέγοντας ένα Μάθημα από τη λίστα, εμφανίζεται η φόρμα στοιχείων του συγκεκριμένου Μαθήματος με όλα τα σχετικά πεδία: κωδικός, Σχολή, Τμήμα, τίτλος, διεύθυνση ιστοτόπου, λίστα Διδασκόντων, κατηγορία (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας), προσφερόμενα εξάμηνα, είδος (θεωρητικό, εργαστηριακό, μεικτό), εβδομαδιαίες διδακτικές ώρες ανά είδος (ώρες θεωρίας, εργαστηρίου, άσκησης), μονάδες ECTS, διαθέσιμες αίθουσες (μεμονωμένες ή συνδυασμός αιθουσών), κατάσταση: ενεργό / ανενεργό.
- Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης.
- Τα πεδία «Σχολή», «Τμήμα», «Διδάσκοντες», «κατηγορία», «προσφερόμενα εξάμηνα», «είδος», «εβδομαδιαίες διδακτικές ώρες» και «κατάσταση» μπορούν να τροποποιηθούν εφόσον δεν έχει οριστεί το Μάθημα σε ωρολόγιο πρόγραμμα.
- Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία στοιχείων του Μαθήματος.
- Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας.
- Ο χρήστης επιλέγει το κουμπί «Αποθήκευση».

Σενάριο 1: Αποτυχία αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.

Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων

- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής

• Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Μαθήματος). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Ο χρήστης επιστρέφει στη λίστα Μαθημάτων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-30: Διαχείριση Μαθημάτων: Προβολή και Τροποποίηση στοιχείων

Τίτλος: Διαχείριση Μαθημάτων Τμήματος: Προσθήκη Μαθήματος σε Τμήμα
Ως Διαχειριστής Τμήματος **Θέλω να** επεξεργαστώ τα Μαθήματα ενός Τμήματος **Ώστε** να προσθέσω ένα νέο Μάθημα στο Τμήμα.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Μαθημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τα Μαθήματα (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται κουμπί «Εισαγωγή Νέου Μαθήματος». • Ο χρήστης επιλέγει το κουμπί «Εισαγωγή Νέου Μαθήματος». • Εμφανίζεται η φόρμα στοιχείων προσθήκης νέου Μαθήματος με όλα τα σχετικά πεδία: κωδικός, Σχολή, Τμήμα, τίτλος, διεύθυνση ιστοτόπου, λίστα Διδασκόντων, κατηγορία (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας), προσφερόμενα εξάμηνα, είδος (θεωρητικό, εργαστηριακό, μεικτό), εβδομαδιαίες διδακτικές ώρες ανά είδος (ώρες θεωρίας, εργαστηρίου, άσκησης), μονάδες ECTS, κατάσταση: ενεργό / ανενεργό. • Ο χρήστης συμπληρώνει τα υποχρεωτικά πεδία «κωδικός» και «τίτλος», «Σχολή», «Τμήμα», «κατηγορία», «προσφερόμενα εξάμηνα», «είδος», καθώς και τα λοιπά πεδία. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων**

• Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος Μαθήματος ή ίδιος Κωδικός). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Εμφανίζεται στον χρήση η φόρμα με τα στοιχεία του νέου Μαθήματος. • Ο χρήστης επιστρέφει στη λίστα Μαθημάτων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-31: Διαχείριση Μαθημάτων Τμήματος: Προσθήκη Μαθήματος σε Τμήμα

Τίτλος: Διαχείριση Μαθημάτων: Διαγραφή ή Απενεργοποίηση Μαθήματος
Ως Διαχειριστής Τμήματος **Θέλω να** επεξεργαστώ τα Τμήματα του Πανεπιστημίου **Ώστε** να διαγράψω οριστικά ένα Μάθημα που καταχωρήθηκε εκ παραδρομής ή να απενεργοποιήσω ένα Μάθημα από το επιλεγμένο εξάμηνο.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Μαθημάτων» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τα Μαθήματα (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των Σχολών, των Τμημάτων τους και των ενεργών Μαθημάτων κάτω από κάθε Τμήμα, για το επιλεγμένο εξάμηνο. • Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Τμημάτων της, πατώντας επάνω στον τίτλο της Σχολής και κατόπιν πατώντας επάνω στον τίτλο Τμήματος εμφανίζεται λίστα των προσφερόμενων Μαθημάτων. • Ο χρήστης μπορεί να επιλέξει διαφορετική εμφάνιση της λίστας, σε μία ενιαία αλφαβητική σειρά όλων των Μαθημάτων, οπότε εμφανίζεται το όνομα Μαθήματος και σε παρένθεση το Τμήμα και η Σχολή. • Ο χρήστης μπορεί να αναζητήσει το Μάθημα που θέλει να προβάλει ή επεξεργαστεί, εισάγοντας στο κατάλληλο πεδίο λέξεις-κλειδιά, οπότε και εμφανίζεται λίστα με τα αποτελέσματα της αναζήτησης ή μήνυμα κενής λίστας εάν δεν βρεθεί αποτέλεσμα. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Μαθημάτων,

οπότε ενημερώνεται η λίστα προβολής με τα ανενεργά Μαθήματα του εξαμήνου που επέλεξε.

- Ο χρήστης μπορεί να επιλέξει ένα Μάθημα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του.
- Επιλέγοντας ένα Μάθημα από τη λίστα, εμφανίζεται η φόρμα στοιχείων του συγκεκριμένου Μαθήματος με όλα τα σχετικά πεδία: κωδικός, Σχολή, Τμήμα, τίτλος, διεύθυνση ιστοτόπου, λίστα Διδασκόντων, κατηγορία (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας), προσφερόμενα εξάμηνα, είδος (θεωρητικό, εργαστηριακό, μεικτό), εβδομαδιαίες διδακτικές ώρες ανά είδος (ώρες θεωρίας, εργαστηρίου, άσκησης), μονάδες ECTS, κατάσταση: ενεργό / ανενεργό.

Σενάριο 1: Απενεργοποίηση Μαθήματος

- Εφόσον το Μάθημα δεν έχει οριστεί στο τρέχον ωρολόγιο πρόγραμμα, εμφανίζεται στον χρήστη η λίστα επιλογών «Κατάσταση» ως επεξεργάσιμο πεδίο και μπορεί να επιλέξει «ανενεργό».
- Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας.
- Ο χρήστης επιλέγει το κουμπί «Αποθήκευση».

Σενάριο 1α: Αποτυχία αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.

Σενάριο 1β: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων

- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 1γ: Επιτυχής αποθήκευση

- Ο χρήστης επιστρέφει στη λίστα Μαθημάτων.
- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης και απενεργοποίησης του Μαθήματος.

Σενάριο 2: Οριστική Διαγραφή Μαθήματος

- Εφόσον το Μάθημα δεν έχει οριστεί σε κανένα ωρολόγιο πρόγραμμα, εμφανίζεται στον χρήστη το κουμπί «Διαγραφή Μαθήματος».
- Ο χρήστης επιλέγει το κουμπί «Διαγραφή Μαθήματος».
- Εμφανίζεται μήνυμα επιβεβαίωσης της πρόθεσης διαγραφής.
- Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Διαγραφής Μαθήματος».

Σενάριο 2α: Αποτυχία διαγραφής • Ο χρήστης λαμβάνει μήνυμα αποτυχίας διαγραφής με το λόγο της αποτυχίας (π.χ. το Μάθημα έχει δηλωθεί σε ωρολόγιο πρόγραμμα) και κωδικό σφάλματος (εάν υπάρχει). **Σενάριο 2β: Επιτυχής διαγραφή** • Ο χρήστης επιστρέφει στη λίστα Μαθημάτων. • Εμφανίζεται μήνυμα επιτυχούς διαγραφής.

Πίνακας Γ-US-32: Διαχείριση Μαθημάτων: Διαγραφή ή Απενεργοποίηση Μαθήματος

Τίτλος: Διαχείριση Αιθουσών: Προβολή ή Τροποποίηση στοιχείων Αίθουσας
Ως Κύριος Διαχειριστής **Θέλω να** προβάλλω τις διαθέσιμες Αίθουσες ανά Σχολή **Ώστε** να ελέγξω ή τροποποιήσω τα στοιχεία τους.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Αιθουσών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τις Αίθουσες (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή). • Εμφανίζεται λίστα με τα ονόματα των Σχολών και των ενεργών Αιθουσών κάτω από κάθε Σχολή, για επιλεγμένο εξάμηνο. • Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Αιθουσών της, πατώντας επάνω στον τίτλο της Σχολής. • Ο χρήστης μπορεί να αναζητήσει την Αίθουσα που θέλει να προβάλει ή επεξεργαστεί, εισάγοντας στο κατάλληλο πεδίο λέξεις-κλειδιά, οπότε και εμφανίζεται λίστα με τα αποτελέσματα της αναζήτησης ή μήνυμα κενής λίστας εάν δεν βρεθεί αποτέλεσμα. • Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Αιθουσών, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργές Αίθουσες του επιλεγμένου εξαμήνου. • Ο χρήστης μπορεί να επιλέξει μία Αίθουσα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του. • Επιλέγοντας μία Αίθουσα από τη λίστα, εμφανίζεται η φόρμα στοιχείων της συγκεκριμένης Αίθουσας με όλα τα σχετικά πεδία: Σχολή, Τμήματα που προσφέρεται, κατηγορίες Μαθημάτων που προσφέρεται (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας),

είδος Μαθημάτων που προσφέρεται (θεωρητικά, εργαστηριακά, μεικτά), τίτλος, κτίριο, χάρτης (εικόνα png/jpg), κατάσταση: ενεργή / ανενεργή. • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Τα πεδία «Σχολή», «Τμήματα», «κατηγορίες Μαθημάτων», «είδος Μαθημάτων» και «κατάσταση» μπορούν να τροποποιηθούν εφόσον δεν έχει οριστεί η Αίθουσα σε ωρολόγιο πρόγραμμα. • Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία στοιχείων της Αίθουσας. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. **Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων** • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής** • Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος και κτίριο). **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Ο χρήστης επιστρέφει στη λίστα Αιθουσών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-33: Διαχείριση Αιθουσών: Προβολή ή Τροποποίηση στοιχείων Αίθουσας

Τίτλος: Διαχείριση Αιθουσών: Προσθήκη Αίθουσας
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τις Αίθουσες του Πανεπιστημίου **Ώστε** να προσθέσω μία νέα Αίθουσα σε Σχολή.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Αιθουσών» από το κεντρικό μενού. • Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τις

Αίθουσες (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή).

- Εμφανίζεται κουμπί «Εισαγωγή Νέας Αίθουσας».
- Ο χρήστης επιλέγει το κουμπί «Εισαγωγή Νέας Αίθουσας».
- Εμφανίζεται η φόρμα στοιχείων προσθήκης νέας Αίθουσας με όλα τα σχετικά πεδία: Σχολή, Τμήματα που προσφέρεται, κατηγορίες Μαθημάτων που προσφέρεται (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας), είδος Μαθημάτων που προσφέρεται (θεωρητικά, εργαστηριακά, μεικτά), τίτλος, κτίριο, χάρτης (εικόνα png/jpg), κατάσταση: ενεργή / ανενεργή.
- Ο χρήστης συμπληρώνει τα υποχρεωτικά πεδία «Σχολή», «Τμήματα», «τίτλο», «κτίριο», καθώς και τα λοιπά πεδία.
- Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας.
- Ο χρήστης επιλέγει το κουμπί «Αποθήκευση».

Σενάριο 1: Αποτυχία αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.

Σενάριο 1α: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων

- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 1β: Αποτυχία αποθήκευσης λόγω διπλότυπης εγγραφής

- Εμφανίζεται μήνυμα για διπλότυπη εγγραφή (ίδιος τίτλος και κτίριο).

Σενάριο 2: Επιτυχής αποθήκευση

- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης.
- Εμφανίζεται στον χρήση η φόρμα με τα στοιχεία της νέας Αίθουσας.
- Ο χρήστης επιστρέφει στη λίστα Αιθουσών επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-34: Διαχείριση Αιθουσών: Προσθήκη Αίθουσας

Τίτλος: Διαχείριση Αιθουσών: Διαγραφή ή Απενεργοποίηση Αίθουσας
Ως Κύριος Διαχειριστής **Θέλω να** επεξεργαστώ τις Αίθουσες του Πανεπιστημίου **Ώστε** να διαγράψω οριστικά μία Αίθουσα που καταχωρήθηκε εκ παραδρομής ή να απενεργοποιήσω μία Αίθουσα από το επιλεγμένο εξάμηνο.

Κριτήρια Αποδοχής (Acceptance criteria):

- Ο χρήστης επιλέγει τον σύνδεσμο «Διαχείριση Αιθουσών» από το κεντρικό μενού.
- Ο χρήστης επιλέγει από λίστα επιλογής το εξάμηνο για το οποίο επιθυμεί να διαχειριστεί τις Αίθουσες (τρέχον εξάμηνο ή επόμενο εξάμηνο) (το τρέχον εξάμηνο είναι η προεπιλογή).
- Εμφανίζεται λίστα με τα ονόματα των Σχολών και των ενεργών Αιθουσών κάτω από κάθε Σχολή, για επιλεγμένο εξάμηνο.
- Ο χρήστης επιλέγει μία Σχολή για εμφάνιση των Αιθουσών της, πατώντας επάνω στον τίτλο της Σχολής.
- Ο χρήστης μπορεί να αναζητήσει την Αίθουσα που θέλει να προβάλει ή επεξεργαστεί, εισάγοντας στο κατάλληλο πεδίο λέξεις-κλειδιά, οπότε και εμφανίζεται λίστα με τα αποτελέσματα της αναζήτησης ή μήνυμα κενής λίστας εάν δεν βρεθεί αποτέλεσμα.
- Ο χρήστης μπορεί να επιλέξει από λίστα επιλογής την προβολή των ανενεργών Αιθουσών, οπότε ενημερώνεται η λίστα προβολής με τις ανενεργές Αίθουσες του επιλεγμένου εξαμήνου.
- Ο χρήστης μπορεί να επιλέξει μία Αίθουσα από τη λίστα για προβολή λεπτομερειών και διαχείριση, πατώντας στον τίτλο του.
- Επιλέγοντας μία Αίθουσα από τη λίστα, εμφανίζεται η φόρμα στοιχείων της συγκεκριμένης Αίθουσας με όλα τα σχετικά πεδία: Σχολή, Τμήματα που προσφέρεται, κατηγορίες Μαθημάτων που προσφέρεται (κορμού, κατ' επιλογή υποχρεωτικά και σεμινάρια Διπλωματικής Εργασίας), είδος Μαθημάτων που προσφέρεται (θεωρητικά, εργαστηριακά, μεικτά), τίτλος, κτίριο, χάρτης (εικόνα png/jpg), κατάσταση: ενεργή / ανενεργή.

Σενάριο 1: Απενεργοποίηση Αίθουσας

- Εφόσον η Αίθουσα δεν έχει οριστεί στο τρέχον ωρολόγιο πρόγραμμα, εμφανίζεται στον χρήστη η λίστα επιλογών «Κατάσταση» ως επεξεργάσιμο πεδίο και μπορεί να επιλέξει «ανενεργή».
- Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές.
- Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας.

- Ο χρήστης επιλέγει το κουμπί «Αποθήκευση».

Σενάριο 1α: Αποτυχία αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.

Σενάριο 1β: Αποτυχία αποθήκευσης λόγω μη έγκυρων τιμών πεδίων

- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 1γ: Επιτυχής αποθήκευση

- Ο χρήστης επιστρέφει στη λίστα Αιθουσών.
- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης και απενεργοποίησης της Αίθουσας.

Σενάριο 2: Οριστική Διαγραφή Αίθουσας

- Εφόσον η Αίθουσα δεν έχει οριστεί σε κανένα ωρολόγιο πρόγραμμα, εμφανίζεται στον χρήστη το κουμπί «Διαγραφή Αίθουσας».
- Ο χρήστης επιλέγει το κουμπί «Διαγραφή Αίθουσας».
- Εμφανίζεται μήνυμα επιβεβαίωσης της πρόθεσης διαγραφής.
- Ο χρήστης επιλέγει το κουμπί «Επιβεβαίωση Διαγραφής Αίθουσας».

Σενάριο 2α: Αποτυχία διαγραφής

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας διαγραφής με το λόγο της αποτυχίας (π.χ. η Αίθουσα έχει δηλωθεί σε ωρολόγιο πρόγραμμα) και κωδικό σφάλματος (εάν υπάρχει).

Σενάριο 2β: Επιτυχής διαγραφή

- Ο χρήστης επιστρέφει στη λίστα Αιθουσών.
- Εμφανίζεται μήνυμα επιτυχούς διαγραφής.

Πίνακας Γ-US-35: Διαχείριση Αιθουσών: Διαγραφή ή Απενεργοποίηση Αίθουσας

Τίτλος: Καθορισμός παραμέτρων πλατφόρμας
Ως Κύριος Διαχειριστής **Θέλω να** ορίσω και επεξεργαστώ τις παραμέτρους συστήματος **Ώστε** να εμφανίζονται οι ορθές και τρέχουσες πληροφορίες της πλατφόρμας.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης επιλέγει τον σύνδεσμο «Ρύθμιση Παραμέτρων Συστήματος» από το κεντρικό μενού. • Εμφανίζεται η φόρμα παραμέτρων συστήματος της πλατφόρμας με όλα τα σχετικά πεδία: Όνομα Πανεπιστημίου, Λογότυπο Πανεπιστημίου, Email Domain, Έναρξη Διαχείρισης Χειμερινού Εξαμήνου, Έναρξη Διαχείρισης Εαρινού Εξαμήνου, Ενεργό Χειμερινό Εξάμηνο Από, Ενεργό Εαρινό Εξάμηνο Από. • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης μπορεί να ανεβάσει μια νέα εικόνα ως λογότυπο Πανεπιστημίου. • Η εικόνα αποθηκεύεται προσωρινά και εμφανίζεται μήνυμα στον χρήστη ότι απαιτείται η αποθήκευση της φόρμας των παραμέτρων για την οριστική αποθήκευση της εικόνας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης.

Πίνακας Γ-US-36: Καθορισμός παραμέτρων πλατφόρμας

Τίτλος: Καθορισμός παραμέτρων επερχόμενου εξαμήνου
Ως Διαχειριστής Τμήματος **Θέλω να** ορίσω και επεξεργαστώ τις παραμέτρους του επερχόμενου εξαμήνου **Ώστε** να είναι διαθέσιμο για την εκπόνηση του ωρολογίου προγράμματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον η τρέχουσα ημερομηνία είναι εντός της ημερομηνία έναρξης του επερχόμενου εξαμήνου, όπως αυτή έχει οριστεί στις παραμέτρους του συστήματος, εμφανίζεται στον χρήστη η επιλογή «Ρύθμιση Επόμενου Εξαμήνου». • Ο χρήστης επιλέγει τον σύνδεσμο «Ρύθμιση Επόμενου Εξαμήνου» από το κεντρικό μενού. • Εμφανίζεται η φόρμα ρύθμισης επερχόμενου εξαμήνου με όλα τα σχετικά πεδία: έτος, εξάμηνο, ημερομηνίες έναρξης και λήξης μαθημάτων, ημερομηνίες έναρξης και λήξης εξεταστικής περιόδου, επίσημες αργίες ώρες έναρξης και λήξης μαθημάτων, ώρες έναρξης και λήξης εξετάσεων, ορισμός περιόδου δήλωσης προτιμήσεων από Διδάσκοντες, ορισμός έναρξης και λήξης εβδομάδας (για προπτυχιακά ή μεταπτυχιακά), ορισμός επιθυμητών ωρών διακοπής (μεσημεριανή σίτιση ανά εξάμηνο), μέγιστο προτιμώμενο όριο ωρών διδασκαλίας ανά ημέρα ανά διδάσκοντα, μέγιστος αριθμός μαθημάτων επιλογής σε επικάλυψη. • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης. • Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας. • Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης.

Πίνακας Γ-US-37: Καθορισμός παραμέτρων επερχόμενου εξαμήνου

Τίτλος: Ορισμός διαθεσιμότητας και προτιμήσεων Διδασκόντων
Ως Διαχειριστής Τμήματος **Θέλω να** ορίσω και επεξεργαστώ τη διαθεσιμότητα και τις προτιμήσεις των Διδασκόντων **Ώστε** να καταχωρηθούν οι απαιτούμενοι περιορισμοί για τη δημιουργία του ωρολογίου προγράμματος .
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον έχουν καθοριστεί οι παράμετροι του επερχόμενου εξαμήνου, εμφανίζεται στον χρήστη η επιλογή «Ρυθμίσεις Ωρολογίου Προγράμματος». • Ο χρήστης επιλέγει τον σύνδεσμο «Ρυθμίσεις Ωρολογίου Προγράμματος» από το κεντρικό μενού. • Ο χρήστης επιλέγει τον σύνδεσμο «Διαθεσιμότητα Διδασκόντων» από το μενού. • Εμφανίζεται λίστα με όλους τους ενεργούς Διδάσκοντες του Τμήματος σε αλφαβητική σειρά. • Ο χρήστης μπορεί να αναζητήσει τον Διδάσκων που θέλει να προβάλει ή επεξεργαστεί τη διαθεσιμότητα και προτιμήσεις, εισάγοντας στο κατάλληλο πεδίο λέξεις-κλειδιά, οπότε και εμφανίζεται λίστα με τα αποτελέσματα της αναζήτησης ή μήνυμα κενής λίστας εάν δεν βρεθεί αποτέλεσμα. • Ο χρήστης μπορεί να επιλέξει έναν Διδάσκοντα από τη λίστα για προβολή και επεξεργασία της διαθεσιμότητας και προτιμήσεων, πατώντας στο όνομα του Διδάσκοντα. • Εμφανίζεται η φόρμα δήλωσης διαθεσιμότητας, περιορισμών και προτιμήσεων για τον επιλεγμένο Διδάσκοντα, για το επερχόμενο εξάμηνο με όλα τα σχετικά πεδία: ώρες σε ημέρα εβδομάδας με αδυναμία διδασκαλίας, ώρες σε ημέρα εβδομάδας με προτίμηση διδασκαλίας. • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης και τον χρήστη που πραγματοποίησε την τροποποίηση (Διαχειριστής Τμήματος ή Διδάσκων). • Ο χρήστης μπορεί να τροποποιήσει τα ανοικτά προς επεξεργασία πεδία. • Για την αποθήκευση, τα πεδία της φόρμας πρέπει να περιέχουν έγκυρες τιμές. • Εμφανίζεται προειδοποιητικό μήνυμα εάν εισαχθεί μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας. • Ο χρήστης επιλέγει το κουμπί «Αποθήκευση». **Σενάριο 1: Αποτυχία αποθήκευσης** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.

• Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης. **Σενάριο 2: Επιτυχής αποθήκευση** • Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης. • Εμφανίζεται μήνυμα αποστολής ειδοποίησης προς τον Διδάσκων. • Ο χρήστης επιστρέφει στη λίστα Διδασκόντων επιλέγοντας το κουμπί «Επιστροφή».

Πίνακας Γ-US-38: Ορισμός διαθεσιμότητας και προτιμήσεων Διδασκόντων

Τίτλος: Διαμόρφωση ωρολογίου προγράμματος
Ως Διαχειριστής Τμήματος **Θέλω να** ορίσω σε εβδομαδιαία βάση τα Μαθήματα, της Αίθουσες και τους Διδάσκοντες **Ώστε** να διαμορφώσω το ωρολόγιο πρόγραμμα για το επερχόμενο εξάμηνο.
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον έχουν καθοριστεί οι παράμετροι του επερχόμενου εξαμήνου, εμφανίζεται στον χρήστη η επιλογή «Διαμόρφωση Ωρολογίου Προγράμματος». • Ο χρήστης επιλέγει τον σύνδεσμο «Διαμόρφωση Ωρολογίου Προγράμματος» από το κεντρικό μενού. • Εμφανίζεται ημερολόγιο εβδομάδας σε μορφή πίνακα με ημέρες (Δευτέρα έως Κυριακή, ανάλογα με την ορισμένη εβδομάδα στις παραμέτρους εξαμήνου) και ώρες με θέσεις (slots) ανά μία ώρα. • Εφόσον υπάρχουν προσωρινά αποθηκευμένα δεδομένα από προηγούμενη χρήση, αυτά εμφανίζονται στον πίνακα του ημερολογίου εβδομάδας. • Ο χρήστης επιλέγει μία θέση στον πίνακα. • Εμφανίζεται αναδυόμενο παράθυρο με πεδία: Μάθημα (επιλογή από λίστα), διάρκεια μαθήματος σε ώρες (επιλογή από λίστα 1, 2, 3 κ.ο.κ.), Διδάσκων (επιλογή από λίστα), Αίθουσα (επιλογή από λίστα). • Οι επιλογές από λίστα έχουν τιμές οι οποίες υπολογίζονται αυτόματα σύμφωνα με τα δεδομένα που έχουν καταχωρηθεί στο σύστημα, καθώς και με τα δηλωθέντα στο τρέχον ωρολόγιο πρόγραμμα δεδομένα, για αποφυγή παραβίασης περιορισμών. • Σε περίπτωση επιλογής που παραβιάζει κάποιον περιορισμό ο οποίος δεν υπολογίζεται αυτόματα ή υπολογίζεται κατά τον ορισμό τιμής στο πεδίο, εμφανίζεται σχετικό μήνυμα.

- Σε παραβίαση αυστηρού περιορισμού όπως π.χ. επικάλυψη μαθημάτων κορμού, εμφανίζεται προειδοποιητικό μήνυμα και δεν επιτρέπεται οριστική αποθήκευση.
- Σε παραβίαση προαιρετικού περιορισμού όπως π.χ. δήλωση ωρών επιθυμητής διακοπής σίτισης, εμφανίζεται ενημερωτικό μήνυμα, ενώ επιτρέπεται οριστική αποθήκευση.
- Στον χρήστη εμφανίζονται γενικά στοιχεία που αφορούν τα δηλωθέντα δεδομένα στο ωρολόγιο πρόγραμμα για την διευκόλυνση της διαμόρφωσης όπως: εκκρεμείς ώρες συμπλήρωσης ελάχιστου ορίου ανά Μάθημα, συνολικές ώρες ανά Μάθημα, συνολικές ώρες ανά Διδάσκοντα, χρησιμοποιούμενες Αίθουσες.
- Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης και τον χρήστη που πραγματοποίησε την τροποποίηση (ονοματεπώνυμο Διαχειριστή Τμήματος).
- Ο χρήστης μπορεί να επιλέξει το κουμπί «Προσωρινή Αποθήκευση».

Σενάριο 1: Αποτυχία προσωρινής αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.
- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 2: Επιτυχής αποθήκευση

- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης και ο χρήστης μπορεί να συνεχίσει με τη διαμόρφωση του προγράμματος.
- Εφόσον έχουν συμπεριληφθεί όλα τα Μαθήματα εξαμήνου του Τμήματος και έχουν συμπληρωθεί οι ελάχιστες απαιτούμενες ώρες ανά Μάθημα, εμφανίζεται το κουμπί «Οριστική Αποθήκευση Ωρολογίου Προγράμματος».
- Ο χρήστης επιλέγει το κουμπί «Οριστική Αποθήκευση Ωρολογίου Προγράμματος».

Σενάριο 3: Αποτυχία οριστικής αποθήκευσης

- Ο χρήστης λαμβάνει μήνυμα αποτυχίας αποθήκευσης με επεξήγηση του λόγου αποτυχίας.
- Εμφανίζεται μήνυμα για μη έγκυρη τιμή σε κάποιο πεδίο της φόρμας και οδηγίες διόρθωσης.

Σενάριο 4: Επιτυχής οριστική αποθήκευση

- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης.
- Προβάλλεται στον χρήστη το οριστικό ωρολόγιο πρόγραμμα, χωρίς δυνατότητα περαιτέρω επεξεργασίας.

Πίνακας Γ-US-39: Διαμόρφωση ωρολογίου προγράμματος

Τίτλος: Διόρθωση ωρολογίου προγράμματος (με αίτημα two-person rule)
Ως Διαχειριστής Τμήματος **Θέλω να** επεξεργαστώ το οριστικά αποθηκευμένο ωρολόγιο πρόγραμμα **Ώστε** να διορθώσω τα δηλωθέντα δεδομένα.
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον έχει πραγματοποιηθεί οριστική αποθήκευση του ωρολογίου προγράμματος, εμφανίζεται το πρόγραμμα στον χρήστη, χωρίς δυνατότητα περαιτέρω επεξεργασίας. • Ο χρήστης επιλέγει το κουμπί «Αίτημα Επεξεργασίας Ωρολογίου Προγράμματος». • Αποστέλλεται εσωτερική ειδοποίηση του αιτήματος προς τον Κύριο Διαχειριστή και εμφανίζεται σχετικό μήνυμα στο χρήστη. **Σενάριο 1: Αποτυχία αποστολής αιτήματος** • Εμφανίζεται στον χρήστη μήνυμα αποτυχίας αποστολής αιτήματος με τον λόγο αποτυχίας της ενέργειας και κωδικό σφάλματος (εάν υπάρχει). **Σενάριο 2: Επιτυχής αποστολή αιτήματος** • Ο χρήστης ενημερώνεται με μήνυμα στην οθόνη ότι η αποστολή του αιτήματος ολοκληρώθηκε και αναμένεται η έγκριση από τον Κύριο Διαχειριστή. • Ο χρήστης μπορεί να αποχωρήσει από τη σελίδα και να επιστρέψει σε μεταγενέστερο χρόνο ή να αναμείνει στη σελίδα μέχρι τη λήψη απάντησης του αιτήματος. **Σενάριο 3: Απόρριψη αιτήματος επεξεργασίας ωρολογίου προγράμματος** • Ο Κύριος Διαχειριστής έχει απορρίψει το αίτημα και εμφανίζεται μήνυμα απόρριψης στον χρήστη. • Ο χρήστης μπορεί να επαναλάβει τη διαδικασία, εφόσον δεν έχει υπερβεί το ημερήσιο όριο αιτημάτων επεξεργασίας του ωρολογίου προγράμματος. **Σενάριο 4: Έγκριση αιτήματος επεξεργασίας ωρολογίου προγράμματος** • Ο Κύριος Διαχειριστής έχει εγκρίνει το αίτημα και εμφανίζεται μήνυμα έγκρισης στον χρήστη. • Ο χρήστης μπορεί να επεξεργαστεί το ωρολόγιο πρόγραμμα και να επιλέξει το κουμπί «Οριστική Αποθήκευση Ωρολογίου Προγράμματος».

Πίνακας Γ-US-40: Διόρθωση ωρολογίου προγράμματος (με αίτημα two-person rule)

Τίτλος: Αυτόματη δημιουργία και διαμόρφωση ωρολογίου προγράμματος
Ως Διαχειριστής Τμήματος **Θέλω να** δημιουργήσω αυτοματοποιημένες προτάσεις για το ωρολόγιο πρόγραμμα **Ώστε** να το διαμορφώσω και να προβώ σε αλλαγές και διορθώσεις.
Κριτήρια Αποδοχής (Acceptance criteria): • Εφόσον έχουν καθοριστεί οι παράμετροι του επερχόμενου εξαμήνου, εμφανίζεται στον χρήστη η επιλογή «Αυτόματη Δημιουργία Προτάσεων Ωρολογίου Προγράμματος». • Ο χρήστης επιλέγει τον σύνδεσμο «Αυτόματη Δημιουργία Προτάσεων Ωρολογίου Προγράμματος» από το κεντρικό μενού. • Το σύστημα, υπολογίζει πιθανό ωρολόγιο πρόγραμμα σύμφωνα με τις παραμέτρους και τους περιορισμούς που έχουν οριστεί στο σύστημα και εμφανίζεται συμπληρωμένο ημερολόγιο εβδομάδας σε μορφή πίνακα με ημέρες (Δευτέρα έως Κυριακή, ανάλογα με την ορισμένη εβδομάδα στις παραμέτρους εξαμήνου) και ώρες με θέσεις (slots) ανά μία ώρα. • Ο χρήστης μπορεί να επεξεργαστεί το ωρολόγιο πρόγραμμα σύμφωνα με τα Κριτήρια Αποδοχής της Ιστορίας Χρήστη «Διαμόρφωση ωρολογίου προγράμματος». • Σε παραβίαση αυστηρού περιορισμού όπως π.χ. επικάλυψη μαθημάτων κορμού, εμφανίζεται προειδοποιητικό μήνυμα και δεν επιτρέπεται οριστική αποθήκευση, ενώ σε παραβίαση προαιρετικού περιορισμού όπως π.χ. δήλωση ωρών επιθυμητής διακοπής σίτισης, εμφανίζεται ενημερωτικό μήνυμα, ενώ επιτρέπεται οριστική αποθήκευση. • Στον χρήστη εμφανίζονται γενικά στοιχεία που αφορούν τα δηλωθέντα δεδομένα στο ωρολόγιο πρόγραμμα για την διευκόλυνση της διαμόρφωσης όπως: εκκρεμείς ώρες συμπλήρωσης ελάχιστου ορίου ανά Μάθημα, συνολικές ώρες ανά Μάθημα, συνολικές ώρες ανά Διδάσκοντα, χρησιμοποιούμενες Αίθουσες. • Εμφανίζεται η ημερομηνία τελευταίας τροποποίησης και τον χρήστη που πραγματοποίησε την τροποποίηση (ονοματεπώνυμο Διαχειριστή Τμήματος). • Ο χρήστης μπορεί να επιλέξει το κουμπί «Προσωρινή Αποθήκευση». **Σενάριο 1: Αποτυχία αυτόματου υπολογισμού προγράμματος** • Ο χρήστης λαμβάνει μήνυμα αποτυχίας δημιουργίας ωρολογίου προγράμματος με αυτόματο υπολογισμό και πιθανούς λόγους αποτυχίας (περιορισμοί που δημιουργούν μη επιλύσιμες συγκρούσεις).

Πίνακας Γ-US-41: Αυτόματη δημιουργία και διαμόρφωση ωρολογίου προγράμματος

Διδάσκων:

Τίτλος: Ενεργοποίηση λογαριασμού Διδάσκοντα μετά από πρόσκληση
Ως προσκεκλημένος Διδάσκων **Θέλω να** ενεργοποιήσω τον λογαριασμό μου στην πλατφόρμα **Ώστε** να έχω πρόσβαση στις παρεχόμενες λειτουργίες που αντιστοιχούν στο ρόλο μου.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης λαμβάνει μήνυμα ενεργοποίησης στο ιδρυματικό του email, το οποίο περιλαμβάνει μοναδικό σύνδεσμο (link) για τη δημιουργία κωδικού εισόδου που αντιστοιχεί στην πλατφόρμα, καθώς και οδηγίες εισόδου και ενεργοποίησης μέσω του παρεχόμενου από το Ίδρυμα συστήματος πιστοποίησης SSO. **Σενάριο 1: Επιλογή συνδέσμου και δημιουργία κωδικού** • Ο χρήστης ακολουθεί τον σύνδεσμο ενεργοποίησης λογαριασμού και μεταφέρεται στη σελίδα δημιουργίας κωδικού πρόσβασης. • Αν ο σύνδεσμος έχει λήξει ή είναι άκυρος, εμφανίζεται μήνυμα σφάλματος και οδηγίες για επανάληψη της διαδικασίας και παραπομπή επικοινωνίας με τη Γραμματεία ή σύνδεση με χρήση SSO. Η τρέχουσα διαδικασία τερματίζεται. • Εφόσον ο σύνδεσμος είναι ενεργός και εντός χρονικού ορίου ισχύος, εμφανίζεται η φόρμα δημιουργίας κωδικού. • Ο χρήστης ορίζει νέο κωδικό πρόσβασης και τον επαληθεύει σε ξεχωριστό πεδίο. • Η φόρμα δεν μπορεί να υποβληθεί αν οι δύο τιμές του κωδικού δεν ταυτίζονται. • Ο νέος κωδικός πρέπει να πληροί τα κριτήρια ελάχιστης πολυπλοκότητας (π.χ. ελάχιστο μήκος, χρήση κεφαλαίων, πεζών, αριθμών και ειδικών χαρακτήρες κ.λπ.). • Ο χρήστης επιλέγει την αποδοχή όρων χρήσης και πολιτικής απορρήτου (GDPR) για την υποβολή. • Ο χρήστης επιλέγει «Υποβολή και Ενεργοποίηση Λογαριασμού» ώστε να ολοκληρωθεί η δημιουργία του κωδικού και η ενεργοποίηση του λογαριασμού. • Εμφανίζεται μήνυμα επιβεβαίωσης ότι η διαδικασία ολοκληρώθηκε επιτυχώς. • Ο χρήστης ανακατευθύνεται στη σελίδα εισόδου (login) της πλατφόρμας. • Ο χρήστης λαμβάνει επιβεβαιωτικό μήνυμα μέσω email για την επιτυχή ενεργοποίηση του

λογαριασμού του. **Σενάριο 2: Επιλογή εισόδου με χρήση SSO** • Ο χρήστης ακολουθεί τον σύνδεσμο προς τη σελίδα εισόδου μέσω SSO της πλατφόρμας. • Ο χρήστης επιλέγει την είσοδο με SSO του Ιδρύματος. • Ο χρήστης ανακατευθύνεται στη σελίδα πιστοποίησης χρήστη και εισόδου SSO του Πανεπιστημίου. **Σενάριο 2α: Επιτυχής πιστοποίηση χρήστη** • Ο χρήστης ανακατευθύνεται στην πλατφόρμα και στην κεντρική σελίδα του λογαριασμού του. • Προβάλλεται μήνυμα στην οθόνη του χρήστη για την επιτυχή ενεργοποίηση του λογαριασμού. • Εφόσον δεν έχει δημιουργήσει τοπικό κωδικό εισόδου, ενημερώνεται για τη δυνατότητα και προβάλλονται οι σχετικές οδηγίες. • Ο χρήστης μπορεί να κάνει χρήση των παρεχόμενων λειτουργιών. **Σενάριο 2β: Αποτυχία πιστοποίησης χρήστη** • Εμφανίζεται μήνυμα αποτυχίας εισόδου και παρότρυνση επαναπροσπάθειας.

Πίνακας Γ-US-42: Ενεργοποίηση λογαριασμού Διδάσκοντα μετά από πρόσκληση

Τίτλος: Δήλωση προτιμήσεων και διαθεσιμότητας ωρών/ημερών διδασκαλίας
Ως Διδάσκων **Θέλω να** δηλώσω τις ημέρες και ώρες της εβδομάδας στις οποίες προτιμώ ή δεν μπορώ να διδάξω **Ώστε** να ληφθούν υπόψη οι αντίστοιχες πληροφορίες κατά τη δημιουργία του ωρολογίου προγράμματος από τη Γραμματεία του Τμήματος.
Κριτήρια Αποδοχής (Acceptance criteria): • Όταν βρίσκεται σε ισχύ η περίοδος καθορισμού του ωρολογίου προγράμματος, εμφανίζεται στον χρήστη ενεργό το κουμπί «Δήλωση Ωρών Διδασκαλίας». • Ο χρήστης επιλέγει την ενότητα «Δήλωση Ωρών Διδασκαλίας» μέσω του κεντρικού μενού. • Εμφανίζεται εβδομαδιαίο ημερολόγιο με ωριαίες χρονικές θέσεις (slots). • Ο χρήστης επιλέγει μία συγκεκριμένη χρονική θέση και εμφανίζεται λίστα επιλογών: «Προτιμώμενη Ώρα» και «Μη Διαθέσιμη Ώρα».

- Ο χρήστης επιλέγει μία τιμή από τη λίστα επιλογών.
- Η επιλεγμένη ωριαία θέση εμφανίζεται με κατάλληλο χρώμα και το κείμενο της επιλογής.
- Ο χρήστης έχει τη δυνατότητα να επεξεργαστεί μία ήδη καταχωρημένη ωριαία θέση και να αναιρεί τη δήλωσή του.
- Ο χρήστης μπορεί να ορίσει προτίμηση ή μη διαθεσιμότητα για ολόκληρη ημέρα, χρησιμοποιώντας τη σχετική επιλογή που εμφανίζεται πάνω από τη συγκεκριμένη στήλη του ημερολογίου.
- Ο χρήστης πατά το κουμπί «Αποθήκευση Επιλογών» για να καταχωρήσει τις προτιμήσεις και τις μη διαθέσιμες ώρες.
- Εμφανίζεται μήνυμα επιτυχούς αποθήκευσης της δήλωσης.
- Η δήλωση μπορεί να τροποποιείται καθ' όλη τη διάρκεια της περιόδου που έχει οριστεί από τη Διεύθυνση του Τμήματος, ακολουθώντας την ίδια διαδικασία.

Σενάριο 1: Αποτυχία αποθήκευσης

- Εμφανίζεται μήνυμα που ενημερώνει τον χρήστη ότι η καταχώριση δεν ολοκληρώθηκε και του ζητά να επαναλάβει την προσπάθεια.

Πίνακας Γ-US-43: Δήλωση προτιμήσεων και διαθεσιμότητας ωρών/ημερών διδασκαλίας

Φοιτητής:

Τίτλος: Προσθήκη μαθήματος σε λίστα ενδιαφέροντος (watchlist)
Ως Φοιτητής **Θέλω να** προσθέσω ένα μάθημα προηγούμενου εξαμήνου στη λίστα ενδιαφέροντος **Ώστε** να εμφανίζεται στο εξατομικευμένο μου ωρολόγιο πρόγραμμα και να ενημερώνομαι για τυχόν έκτατες αλλαγές.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης ανοίγει από το μενού την επιλογή «Προβολή Ωρολογίου Προγράμματος». • Προβάλλεται το εξατομικευμένο πρόγραμμα του τρέχοντος εξαμήνου, μαζί με τυχόν μαθήματα που έχουν ήδη προστεθεί στη λίστα ενδιαφέροντος. • Ο χρήστης επιλέγει διαφορετικό εξάμηνο από το τρέχον μέσω σχετικής λίστας επιλογών. • Προβάλλεται το εβδομαδιαίο ωρολόγιο πρόγραμμα για το επιλεγμένο εξάμηνο. • Ο χρήστης επιλέγει μάθημα πατώντας πάνω στη θέση του στο ημερολόγιο. • Εμφανίζεται παράθυρο με αναλυτικές πληροφορίες (τίτλος, εξάμηνο, διδάσκων, αίθουσα) και κουμπί «Προσθήκη στη λίστα Ενδιαφέροντος». • Με την επιλογή «Προσθήκη στη λίστα Ενδιαφέροντος», το μάθημα καταχωρείται στη λίστα του χρήστη και το κουμπί αλλάζει σε «Αφαίρεση από τη λίστα Ενδιαφέροντος». • Ο χρήστης επιστρέφει στο εξατομικευμένο πρόγραμμα για το εξάμηνο που παρακολουθεί, στο οποίο εμφανίζεται με ξεχωριστό χρωματισμό το μάθημα που πρόσθεσε στη λίστα ενδιαφέροντος.

Πίνακας Γ-US-44: Προσθήκη μαθήματος σε λίστα ενδιαφέροντος (watchlist)

Τίτλος: Αφαίρεση μαθήματος από λίστα ενδιαφέροντος (watchlist)
Ως Φοιτητής **Θέλω να** αφαιρέσω ένα μάθημα προηγούμενου εξαμήνου από τη λίστα ενδιαφέροντος **Ώστε** να μην εμφανίζεται στο εξατομικευμένο μου ωρολόγιο πρόγραμμα.
Κριτήρια Αποδοχής (Acceptance criteria): • Ο χρήστης ανοίγει από το μενού την επιλογή «Προβολή Ωρολογίου Προγράμματος». • Προβάλλεται το εξατομικευμένο πρόγραμμα του τρέχοντος εξαμήνου, μαζί με τυχόν μαθήματα

που έχουν ήδη προστεθεί στη λίστα ενδιαφέροντος.

- Ο χρήστης επιλέγει μάθημα πατώντας πάνω στη θέση του στο ημερολόγιο.
- Εμφανίζεται παράθυρο με αναλυτικές πληροφορίες (τίτλος, εξάμηνο, διδάσκων, αίθουσα) και κουμπί «Προσθήκη στη λίστα Ενδιαφέροντος».
- Ο χρήστης επιλέγει το κουμπί «Αφαίρεση από τη λίστα Ενδιαφέροντος» και εμφανίζεται μήνυμα επιβεβαίωσης επιλογής.
- Εφόσον ο χρήστης επιβεβαιώσει, το μάθημα αφαιρείται από τη λίστα μαθημάτων προβολής τους χρήστη και το παράθυρο κλείνει.
- Ο χρήστης επιστρέφει στο εξατομικευμένο πρόγραμμα για το εξάμηνο που παρακολουθεί, στο οποίο δεν εμφανίζεται πλέον το μάθημα που αφαίρεσε από τη λίστα ενδιαφέροντος.

Σενάριο 1: Ακύρωση ενέργειας αφαίρεσης

- Ο χρήστης επιλέγει την ακύρωση της ενέργειας αφαίρεσης του μαθήματος από τη λίστα ενδιαφέροντος.
- Ο χρήστης επιστρέφει στο εξατομικευμένο πρόγραμμα για το εξάμηνο που παρακολουθεί, χωρίς να έχει γίνει κάποια τροποποίηση.

Πίνακας Γ-US-45: Αφαίρεση μαθήματος από λίστα ενδιαφέροντος (watchlist)

Παράρτημα Δ: «Περιπτώσεις Χρήσης»

Στο παρόν παράρτημα παρατίθενται συγκεντρωτικά όλες οι περιγραφές Περιπτώσεων Χρήσης (Use Case Scenarios), σε τυποποιημένη φόρμα περιγραφής, καθώς και τα διαγράμματα UML σε επίπεδα εμβάθυνσης ανάλυσης.

Τίτλος: Αρχική ενεργοποίηση λογαριασμού Κύριου Διαχειριστή
ID: UC9

Σύντομη περιγραφή:
Η περίπτωση χρήσης περιγράφει τη διαδικασία ενεργοποίησης του λογαριασμού Κύριου Διαχειριστή κατά την αρχική εγκατάσταση της πλατφόρμας. Ο χρήστης εισέρχεται με προσωρινά διαπιστευτήρια, τα οποία χρησιμοποιεί για να ορίσει νέο κωδικό πρόσβασης και να ενεργοποιήσει τον λογαριασμό του.

Χειριστές:
Κύριος Διαχειριστής

Κατάσταση Εισόδου (Προϋποθέσεις):
Ο λογαριασμός Κύριου Διαχειριστή βρίσκεται σε κατάσταση «ανενεργός».
Η πλατφόρμα έχει εγκατασταθεί επιτυχώς στον server.
Έχουν παρασχεθεί στον Κύριο Διαχειριστή τα προσωρινά διαπιστευτήρια εισόδου.

Βασική ροή:
1. Ο χρήστης επιλέγει τον σύνδεσμο «Είσοδος Χρήστη» από το κεντρικό μενού.
2. Το σύστημα εμφανίζει τη φόρμα εισόδου (ιδρυματικό email και κωδικός).
3. Ο χρήστης εισάγει έγκυρα διαπιστευτήρια.
4. Το σύστημα επαληθεύει τα διαπιστευτήρια στη Βάση Δεδομένων.
5. Το σύστημα ανακατευθύνει τον χρήστη στη σελίδα ορισμού νέου κωδικού πρόσβασης.
6. Ο χρήστης εισάγει νέο κωδικό και επαναλαμβάνει την τιμή στο πεδίο επιβεβαίωσης.
7. Το σύστημα ελέγχει ότι οι δύο τιμές του κωδικού ταυτίζονται και ο κωδικός πληροί τα κριτήρια ελάχιστης πολυπλοκότητας (περιέχει πεζά και κεφαλαία, αριθμούς, σύμβολα, έχει το ελάχιστο μήκος και δεν περιέχει πολλούς επαναλαμβανόμενους χαρακτήρες).
8. Ο χρήστης επιλέγει «Υποβολή» για τη δημιουργία του νέου του κωδικού.
9. Το σύστημα αποθηκεύει τον νέο κωδικό και ενεργοποιεί τον λογαριασμό χρήστη.
10. Το σύστημα αποστέλλει αυτοματοποιημένο email ενημέρωσης για την επιτυχή δημιουργία νέου κωδικού και την ενεργοποίηση του λογαριασμού χρήστη.
11. Το σύστημα ανακατευθύνει τον χρήστη στη σελίδα εισόδου και εμφανίζει μήνυμα επιτυχούς

δημιουργίας νέου κωδικού και ενεργοποίησης λογαριασμού. **Εναλλακτική ροή 1 – «Μη έγκυρα πεδία»:** 3γ. Τα πεδία περιέχουν μη αποδεκτές τιμές (λάθος μορφή, κενά πεδία). 3β. Το σύστημα εμφανίζει μήνυμα σφάλματος και εμποδίζει την υποβολή της φόρμα. 3γ. Η ροή μεταφέρεται στο βήμα 2. **Εναλλακτική ροή 2 – «Λανθασμένα στοιχεία (επαναπροσπάθεια)»:** 4α. Τα διαπιστευτήρια δεν αντιστοιχούν σε καταχώρηση στη ΒΔ. 4β. Το σύστημα εμφανίζει γενικό μήνυμα αποτυχίας, χωρίς περαιτέρω πληροφορίες. 4γ. Η ροή μεταφέρεται στο βήμα 2. **Εναλλακτική ροή 3 – «Λανθασμένα στοιχεία (προσωρινό κλείδωμα)»:** 4α. Τα διαπιστευτήρια είναι λανθασμένα και συμπληρωθεί το επιτρεπόμενο όριο αποτυχημένων προσπαθειών. 4β. Το σύστημα εμφανίζει μήνυμα προσωρινού αποκλεισμού. 4γ. Το σύστημα κλειδώνει τη λειτουργία εισόδου για προκαθορισμένο χρονικό διάστημα (ανά email και διεύθυνση IP). 4δ. Η περίπτωση χρήσης τερματίζεται. **Εναλλακτική ροή 4 – «Μη αποδεκτός νέος κωδικός εισόδου»:** 7α. Οι τιμές των πεδίων κωδικού δεν ταιριάζουν ή ο κωδικός δεν πληροί τα κριτήρια ελάχιστης πολυπλοκότητας. 7β. Το σύστημα εμφανίζει σχετικό μήνυμα και ζητά διόρθωση. 7γ. Η ροή μεταφέρεται στο βήμα 6.
Μη λειτουργικές απαιτήσεις: Όλα τα διαπιστευτήρια μεταδίδονται μέσω κρυπτογραφημένου καναλιού (SSL/TLS). Μετά από N αποτυχημένες προσπάθειες, το σύστημα κλειδώνει τον λογαριασμό και τη διεύθυνση IP για προκαθορισμένο διάστημα S (οι τιμές N και S έχουν οριστεί εσωτερικά στο σύστημα κατά την εγκατάσταση). Τα μηνύματα σφάλματος δεν αποκαλύπτουν πληροφορίες σχετικά με τα διαπιστευτήρια. Κάθε ενέργεια καταγράφεται σύμφωνα με την πολιτική καταγραφής (logging).
Κατάσταση εξόδου: Ο λογαριασμός Κύριου Διαχειριστή έχει ενεργοποιηθεί και ο χρήστης μπορεί να εισέλθει στην πλατφόρμα με τα νέα διαπιστευτήρια.

Πίνακας Δ-1-UC9: Αρχική ενεργοποίηση λογαριασμού Κύριου Διαχειριστή

Τίτλος: Καθορισμός παραμέτρων πλατφόρμας
ID: UC10

Σύντομη περιγραφή:
Η περίπτωση χρήσης περιγράφει τη διαδικασία καθορισμού των παραμέτρων της πλατφόρμας, από τις οποίες εξαρτώνται οι κύριες λειτουργίες του συστήματος και η βασική εμφάνιση των σελίδων. Περιλαμβάνει την ενημέρωση στοιχείων ταυτότητας του Ιδρύματος και βασικών ρυθμίσεων ακαδημαϊκών εξαμήνων.

Χειριστές:
Κύριος Διαχειριστής

Κατάσταση Εισόδου (Προϋποθέσεις):
Ο χρήστης έχει συνδεθεί στο σύστημα.

Βασική ροή:
1. Ο χρήστης επιλέγει τον σύνδεσμο «Ρύθμιση Παραμέτρων Συστήματος» από το κεντρικό μενού.
2. Το σύστημα εμφανίζει τη φόρμα παραμέτρων συστήματος της πλατφόρμας με όλα τα σχετικά πεδία: Όνομα Ιδρύματος (Πανεπιστημίου), Λογότυπο, Email Domain, Έναρξη Διαχείρισης Χειμερινού Εξαμήνου, Έναρξη Διαχείρισης Εαρινού Εξαμήνου, Ενεργό Χειμερινό Εξάμηνο Από, Ενεργό Εαρινό Εξάμηνο Από.
3. Ο χρήστης τροποποιεί τα ανοικτά προς επεξεργασία πεδία.
4. Ο χρήστης ανεβάζει μια νέα εικόνα ως λογότυπο Πανεπιστημίου σε μορφή αρχείου JPG ή PNG.
5. Το σύστημα αποθηκεύει την εικόνα προσωρινά μέχρι ο χρήστης να επιλέξει την αποθήκευση της φόρμας, οπότε πραγματοποιείται η οριστική καταχώρηση της νέας εικόνας.
6. Ο χρήστης επιλέγει το κουμπί «Αποθήκευση».
7. Το σύστημα αποθηκεύει τα τροποποιημένα πεδία ή/και την νέα εικόνα και καταγράφει την τελευταία ημερομηνία και ώρα τροποποίησης.

Εναλλακτική ροή 1 – «Μη αποδεκτή τιμή πεδίου»:
3α. Το πεδίο περιέχει μη αποδεκτή τιμή (λάθος μορφή, κενό υποχρεωτικό πεδίο κ.ο.κ).
3β. Το σύστημα εμφανίζει μήνυμα σφάλματος με επεξήγηση, ενώ εμποδίζει την υποβολή της φόρμα.
3γ. Η ροή μεταφέρεται στο βήμα 3.

Εναλλακτική ροή 2 – «Λάθος τύπος αρχείου ή υπέρβαση ορίου μεγέθους αρχείου»:
4α. Ο τύπος αρχείου δεν είναι μορφής JPG/PNG ή το μέγεθος αρχείου υπερβαίνει το όριο που έχει οριστεί στο σύστημα.
4β. Το σύστημα εμφανίζει μήνυμα σφάλματος με επεξήγηση και η διαδικασία ανεβάσματος

διακόπτεται. 4γ. Η ροή μεταφέρεται στο βήμα 4. **Εναλλακτική ροή 3 – «Έξοδος χωρίς αποθήκευση»:** 6α. Τα τροποποιημένα πεδία καθώς και η προσωρινά αποθηκευμένη εικόνα δεν αποθηκεύονται στο σύστημα. 6β. Η προσωρινά αποθηκευμένη εικόνα διαγράφεται οριστικά από το σύστημα σε καθορισμένο χρόνο κατά την ημερήσια εκτέλεση προγραμματισμένων εργασιών. 6γ. Η περίπτωση χρήσης τερματίζεται.
Μη λειτουργικές απαιτήσεις: Κάθε ενέργεια καταγράφεται σύμφωνα με την πολιτική καταγραφής (logging). Οι αλλαγές στις παραμέτρους εφαρμόζονται, όπου αυτό είναι δυνατό, άμεσα σε ενεργά sessions χρηστών. Τα αρχεία λογοτύπου αποθηκεύονται σε ασφαλή διαδρομή με έλεγχο ακεραιότητας.
Κατάσταση εξόδου: Οι παράμετροι του συστήματος έχουν ενημερωθεί και οι σχετικές αλλαγές εφαρμόζονται στην εμφάνιση και στη λειτουργία της πλατφόρμας για όλους τους χρήστες.

Πίνακας Δ-2-UC10: Καθορισμός παραμέτρων πλατφόρμας

Τίτλος: Καταχώριση αναπλήρωσης μαθήματος από Διδάσκοντα **ID:** UC11
Σύντομη περιγραφή: Η περίπτωση χρήσης περιγράφει τη διαδικασία με την οποία ένας Διδάσκων καταχωρεί αναπλήρωση μαθήματος στο ημερολογιακό πρόγραμμα. Ο χρήστης επιλέγει μάθημα, ημερομηνία, ώρα και αίθουσα, ενώ το σύστημα ελέγχει τη διαθεσιμότητα και αποθηκεύει την αλλαγή.
Χειριστές: Διδάσκων
Κατάσταση Εισόδου (Προϋποθέσεις): Ο χρήστης έχει συνδεθεί στην πλατφόρμα ως Διδάσκων. Ο Διδάσκων είναι συσχετισμένος με το επιλεγμένο μάθημα. Το πρόγραμμα έχει δημοσιευθεί ή υπάρχει διαθέσιμο ημερολογιακό πρόγραμμα μαθημάτων. Η επιλεγμένη ημερομηνία δεν αφορά παρελθοντική ημέρα.
Βασική ροή:

1. Ο χρήστης μεταβαίνει στο ημερολογιακό πρόγραμμα μαθημάτων. 2. Το σύστημα εμφανίζει το πρόγραμμα σε μορφή ημερολογίου. 3. Ο χρήστης επιλέγει συγκεκριμένη ημερομηνία και μάθημα στο ημερολόγιο. 4. Το σύστημα εμφανίζει φόρμα επιλογής αίθουσας και διαθέσιμων ωρών. 5. Ο χρήστης επιλέγει διαθέσιμες ώρα και αίθουσα. 6. Το σύστημα ελέγχει ότι δεν υπάρχει σύγκρουση με άλλο μάθημα, διδάσκοντα ή αίθουσα. 7. Ο χρήστης επιλέγει «Αποθήκευση Αναπλήρωσης». 8. Το σύστημα αποθηκεύει την αναπλήρωση. 9. Το σύστημα ενημερώνει το ημερολογιακό πρόγραμμα και εμφανίζει μήνυμα επιτυχούς καταχώρισης. **Εναλλακτική ροή 1 – «Μη διαθέσιμη αίθουσα ή ώρα»:** 6α. Η επιλεγμένη αίθουσα ή ώρα χρησιμοποιείται ήδη. 6β. Το σύστημα εμφανίζει μήνυμα σύγκρουσης. 6γ. Η ροή μεταφέρεται στο βήμα 6. **Εναλλακτική ροή 2 – «Επικάλυψη Μαθήματος ή Διδάσκοντος»:** 7α. Εντοπίζεται επικάλυψη μαθήματος ή ο διδάσκων είναι δηλωμένος σε άλλο μάθημα την ίδια ημέρα και ώρα. 7β. Το σύστημα εμφανίζει μήνυμα σφάλματος με επεξήγηση και η διαδικασία διακόπτεται. 7γ. Η ροή μεταφέρεται στο βήμα 5.
Μη λειτουργικές απαιτήσεις: Η καταχώριση πραγματοποιείται μόνο από εξουσιοδοτημένο χρήστη. Οι έλεγχοι συγκρούσεων εκτελούνται πριν την αποθήκευση. Οι αλλαγές αποτυπώνονται άμεσα στο ημερολογιακό πρόγραμμα. Οι ενέργειες καταγράφονται σύμφωνα με την πολιτική καταγραφής.
Κατάσταση εξόδου: Η αναπλήρωση έχει καταχωρηθεί και εμφανίζεται στο ημερολογιακό πρόγραμμα μαθημάτων.

Πίνακας Δ-3-UC11: Καταχώριση αναπλήρωσης μαθήματος από Διδάσκοντα

Τίτλος: Εξαγωγή ωρολογίου προγράμματος **ID:** UC12
Σύντομη περιγραφή: Η περίπτωση χρήσης περιγράφει τη διαδικασία εξαγωγής του ωρολογίου προγράμματος σε αρχείο, ώστε ο χρήστης να μπορεί να το αποθηκεύσει, να το εκτυπώσει ή να το επεξεργαστεί περαιτέρω.

Χειριστές: Διαχειριστής, Διδάσκων, Φοιτητής, Μη Εγγεγραμμένος Χρήστης
Κατάσταση Εισόδου (Προϋποθέσεις): Υπάρχει διαθέσιμο ωρολόγιο πρόγραμμα. Ο χρήστης έχει πρόσβαση στη σελίδα προβολής προγράμματος. Έχουν επιλεγεί τα απαραίτητα φίλτρα, όπου απαιτείται.
Βασική ροή: 1. Ο χρήστης μεταβαίνει στη σελίδα προβολής ωρολογίου προγράμματος. 2. Το σύστημα εμφανίζει το πρόγραμμα με βάση τα διαθέσιμα ή επιλεγμένα φίλτρα. 3. Ο χρήστης επιλέγει μορφή εξαγωγής. 4. Το σύστημα δημιουργεί το αντίστοιχο αρχείο. 5. Το σύστημα επιστρέφει το αρχείο στον χρήστη για λήψη. 6. Ο χρήστης αποθηκεύει ή ανοίγει το αρχείο στη συσκευή του. **Εναλλακτική ροή 1 – «Δεν υπάρχουν διαθέσιμα δεδομένα»:** 2α. Δεν υπάρχουν εγγραφές προγράμματος για τα επιλεγμένα κριτήρια. 2β. Το σύστημα εμφανίζει σχετικό μήνυμα ενημέρωσης. 2γ. Η περίπτωση χρήσης τερματίζεται. **Εναλλακτική ροή 2 – «Αποτυχία δημιουργίας αρχείου»:** 4α. Το αρχείο δεν μπορεί να δημιουργηθεί λόγω τεχνικού σφάλματος. 4β. Το σύστημα εμφανίζει μήνυμα αποτυχίας εξαγωγής. 4γ. Η περίπτωση χρήσης τερματίζεται.
Μη λειτουργικές απαιτήσεις: Η εξαγωγή πρέπει να ολοκληρώνεται σε εύλογο χρόνο. Το παραγόμενο αρχείο πρέπει να περιέχει τα ίδια δεδομένα με την τρέχουσα προβολή ή τα επιλεγμένα φίλτρα. Η πρόσβαση στα δεδομένα εξαγωγής πρέπει να ακολουθεί τα δικαιώματα του χρήστη. Η μορφοποίηση του αρχείου πρέπει να είναι κατάλληλη για ανάγνωση και εκτύπωση.
Κατάσταση εξόδου: Το ωρολόγιο πρόγραμμα έχει εξαχθεί επιτυχώς σε αρχείο και είναι διαθέσιμο στον χρήστη.

Πίνακας Δ-4-UC12: Εξαγωγή ωρολογίου προγράμματος

Παράρτημα Ε: «Πρότυπα Κώδικα & Διασφάλιση Ποιότητας»

Τεχνολογία / Εργαλείο	Σκοπός χρήσης
HTML5	Δομή και σημασιολογική οργάνωση ιστοσελίδων.
CSS3	Μορφοποίηση, σχεδιασμός διεπαφής και responsive εμφάνιση σε διαφορετικές συσκευές.
JavaScript, AJAX, JSON	Διαδραστικότητα διεπαφής, ασύγχρονη επικοινωνία.
Bootstrap 5	Front-end framework για responsive σχεδίαση και επαναχρησιμοποιήσιμα στοιχεία διεπαφής.
PHP 8.1+	Γλώσσα προγραμματισμού για την υλοποίηση της λογικής της εφαρμογής από πλευράς διακομιστή.
Session PHP	Διαχείριση συνεδριών χρηστών και ταυτοποίησης.
PDO	Ασφαλής πρόσβαση και επικοινωνία με τη βάση δεδομένων μέσω prepared statements.
MariaDB 10.5+ / MySQL 8+	Σύστημα διαχείρισης σχεσιακής βάσης δεδομένων για αποθήκευση και διαχείριση δεδομένων.
PHPStan, Psalm	Static analysis εργαλεία για έλεγχο τύπων και εντοπισμό πιθανών σφαλμάτων στον κώδικα.
PHP_CodeSniffer (PSR-12)	Έλεγχος συμμόρφωσης του κώδικα με πρότυπα μορφοποίησης και συγγραφής.
PHP Metrics	Ανάλυση δομής κώδικα και υπολογισμός μετρικών όπως η κυκλωματική πολυπλοκότητα.
QCacheGrind	Ανάλυση εκτέλεσης και εντοπισμός σημείων υψηλού υπολογιστικού κόστους.
GitHub	Διαχείριση εκδόσεων κώδικα.
Visual Studio Code	Κύριο περιβάλλον ανάπτυξης κώδικα (IDE).
Visual Paradigm	Σχεδίαση διαγραμμάτων, μοντελοποίηση αρχιτεκτονικής.
phpMyAdmin, HeidiSQL	Διαχείριση και επεξεργασία της βάσης δεδομένων.
Apache JMeter	Έλεγχος απόδοσης και προσομοίωση φόρτου χρηστών.
IIS Express	Web server για ανάπτυξη και δοκιμή της εφαρμογής.

Παράρτημα ΣΤ: «Ανάλυση Επιλυτή Γενετικού Αλγορίθμου»

Ο επιλυτής ωρολογίου προγράμματος αποτελεί πειραματική αλγοριθμική επέκταση της πλατφόρμας, η οποία στοχεύει στην υποβοήθηση της διαδικασίας δημιουργίας προγράμματος εξαμήνου μέσω γενετικού αλγορίθμου. Η λειτουργία του βασίζεται στη μετατροπή των δεδομένων του συστήματος σε κατάλληλη δομή εισόδου, στη δημιουργία αρχικού πληθυσμού πιθανών λύσεων και στη σταδιακή βελτίωση των λύσεων μέσα από επαναλαμβανόμενες γενιές.

Βασική προϋπόθεση για την ορθή λειτουργία του επιλυτή είναι ο ορισμός των βασικών, απαραίτητων και όσο το δυνατόν πληρέστερων δεδομένων από τους διαχειριστές της πλατφόρμας. Επιπλέον, η ύπαρξη προηγούμενων προγραμμάτων, είτε μέσω αρχικής καταχώρησης είτε μέσω της φυσικής εξέλιξης του συστήματος με την πάροδο του χρόνου, μπορεί να συμβάλει στη βελτίωση τόσο της απόδοσης όσο και της ποιότητας των παραγόμενων λύσεων. Παρ' όλα αυτά, ο επιλυτής μπορεί να λειτουργήσει ικανοποιητικά ακόμη και χωρίς την ύπαρξη ιστορικών δεδομένων.

Η υλοποίηση έχει σχεδιαστεί σε διακριτά στάδια, ώστε κάθε μέρος της διαδικασίας να μπορεί να εκτελείται αυτόνομα, να ελέγχεται ευκολότερα και να αποφεύγονται περιορισμοί χρόνου εκτέλεσης από τον server. Η συνολική διαδικασία διακρίνεται στα εξής βασικά βήματα:

- δημιουργία δεδομένων εισόδου,
- παραγωγή αρχικού πληθυσμού,
- επαναληπτική εκτέλεση γενετικού αλγορίθμου,
- αποθήκευση αποτελεσμάτων, προβολή και αναφορά.

1. Δημιουργία δεδομένων εισόδου

Το αρχείο *schedule_solver_input.php* δημιουργεί το αρχικό σύνολο δεδομένων που απαιτείται από τον επιλυτή. Η λειτουργία του είναι ανεξάρτητη από την ίδια την εκτέλεση του αλγορίθμου και μπορεί να χρησιμοποιηθεί είτε ως REST API είτε ως εσωτερική διαδικασία παραγωγής αρχείου JSON, με τον ορισμό κατάλληλης παραμέτρου κλήσης.

Τα δεδομένα εισόδου περιλαμβάνουν τις ρυθμίσεις του εξαμήνου, τα μαθήματα, τις διαθέσιμες αίθουσες, τις προτεινόμενες αίθουσες με βάση προηγούμενη χρήση, το ήδη υπάρχον πρόγραμμα σε περίπτωση μερικής επίλυσης, καθώς και τη διαθεσιμότητα των διδασκόντων (περιορισμούς και προτιμήσεις). Με αυτόν τον τρόπο ο γενετικός αλγόριθμος δεν εργάζεται απευθείας πάνω στη βάση δεδομένων, αλλά πάνω σε ένα ενιαίο και προκαθορισμένο σύνολο δεδομένων. Η επιλογή αυτή μειώνει την πολυπλοκότητα κατά την εκτέλεση του αλγορίθμου, καθώς τα απαραίτητα δεδομένα έχουν ήδη εξαχθεί και οργανωθεί πριν ξεκινήσει η διαδικασία βελτιστοποίησης.

2. Αναπαράσταση λύσης

Κάθε πιθανή λύση του προβλήματος αναπαρίσταται ως χρωμόσωμα. Το χρωμόσωμα αποτελείται από γονίδια, όπου κάθε γονίδιο αντιστοιχεί σε έναν συγκεκριμένο τύπο μαθήματος, όπως θεωρία, εργαστήριο ή άσκηση, μαζί με το αντίστοιχο πλήθος απαιτούμενων διδακτικών ωρών. Κάθε γονίδιο συνδέεται με συγκεκριμένο μάθημα και τύπο, καθώς και με το αντίστοιχο σύνολο διδακτικών ωρών που απαιτούνται, την αίθουσα και τους διδάσκοντες. Η τιμή κάθε γονιδίου αντιστοιχεί σε μία πλήρη επιτρεπτή επιλογή τοποθέτησης του μαθήματος στο πρόγραμμα. Η επιλογή αυτή περιλαμβάνει συνδυασμό ημερών, ωρών, χρονικών μοτίβων, αιθουσών και διδασκόντων. Κάθε επιλογή μπορεί να περιλαμβάνει:

- το μάθημα,
- τον τύπο διδασκαλίας,
- το πλήθος των απαιτούμενων ωρών,
- το μοτίβο κατανομής των ωρών,
- τις αντίστοιχες ημέρες και ώρες,
- την αίθουσα,
- έναν ή περισσότερους διδάσκοντες.

Με αυτή τη μορφή, κάθε χρωμόσωμα περιγράφει ένα πλήρες πιθανό ωρολόγιο πρόγραμμα. Ο γενετικός αλγόριθμος δεν επιλέγει μεμονωμένες ώρες ανεξάρτητα, αλλά έτοιμες επιτρεπτές επιλογές για κάθε γονίδιο, γεγονός που περιορίζει τον χώρο αναζήτησης και βοηθά στη δημιουργία πιο ρεαλιστικών λύσεων.

3. Δημιουργία αρχικού πληθυσμού

Το αρχείο *schedule_solver_population.php*, σε συνεργασία με την κλάση *ScheduleSolverPopulation*, δημιουργεί τον αρχικό πληθυσμό του αλγορίθμου. Ο πληθυσμός αποτελείται από πολλαπλά χρωμοσώματα, δηλαδή από πολλές διαφορετικές πιθανές λύσεις του ίδιου προβλήματος.

Για κάθε μάθημα και τύπο διδασκαλίας δημιουργούνται ομάδες πιθανών επιλογών. Οι επιλογές αυτές παράγονται λαμβάνοντας υπόψη τις διαθέσιμες ώρες, τα επιτρεπτά μοτίβα κατανομής, τις αίθουσες που είναι κατάλληλες για το μάθημα, καθώς και τους διδάσκοντες που σχετίζονται με το μάθημα.

Κατά τη δημιουργία του πληθυσμού εφαρμόζεται και διαδικασία επιδιόρθωσης. Η επιδιόρθωση δεν εγγυάται ότι κάθε λύση θα είναι πλήρως έγκυρη, όμως μειώνει άμεσα αρκετές εμφανείς συγκρούσεις, όπως επικαλύψεις αιθουσών, διδασκόντων ή μαθημάτων. Έτσι ο αλγόριθμος ξεκινά από καλύτερη αρχική κατάσταση σε σχέση με έναν εντελώς τυχαίο πληθυσμό.

4. Συνάρτηση αξιολόγησης

Η αξιολόγηση κάθε χρωμοσώματος γίνεται με τη συνάρτηση καταλληλότητας, η οποία υπολογίζει αριθμητικό σκορ για κάθε πιθανή λύση. Όσο μικρότερο είναι το σκορ, τόσο καλύτερη θεωρείται η λύση. Η αξιολόγηση διακρίνει τους περιορισμούς σε δύο βασικές κατηγορίες:

- Σκληροί περιορισμοί, οι οποίοι δεν πρέπει να παραβιάζονται:
    1. σύγκρουση μαθημάτων στο ίδιο εξάμηνο ή στην ίδια κατεύθυνση,
    2. χρήση της ίδιας αίθουσας την ίδια ημέρα και ώρα,
    3. ανάθεση του ίδιου διδάσκοντα σε διαφορετικά μαθήματα την ίδια ώρα,
    4. παραβίαση μη διαθέσιμης ώρας διδάσκοντα,
    5. εσωτερική επικάλυψη του ίδιου μαθήματος.
- Μαλακοί περιορισμοί, οι οποίοι επιτρέπονται αλλά επιβαρύνουν τη λύση:
    1. χρήση ώρας μεσημεριανού διαλείμματος,
    2. μη τήρηση προτίμησης διδάσκοντα,
    3. επιτρεπόμενη αλλά ανεπιθύμητη επικάλυψη,

4. πολύ πρωινές ή πολύ αργές ώρες,
5. χρήση λιγότερο επιθυμητών επιλογών τοποθέτησης.

Οι σκληροί περιορισμοί έχουν πολύ μεγάλη ποινή, ώστε μία μη έγκυρη λύση να είναι πάντα χειρότερη από μία έγκυρη λύση με απλές μαλακές ποινές. Με αυτόν τον τρόπο ο αλγόριθμος πρώτα οδηγείται στην αποφυγή κρίσιμων συγκρούσεων και στη συνέχεια προσπαθεί να βελτιώσει την ποιότητα του προγράμματος.

Κατά τη δημιουργία των επιλογών λαμβάνονται επίσης υπόψη προκαθορισμένα μοτίβα κατανομής διδακτικών ωρών, όπως συνεχόμενες ώρες ή διαχωρισμός των ωρών σε πολλαπλές ημέρες. Τα μοτίβα αυτά χρησιμοποιούνται για τη δημιουργία ρεαλιστικών πιθανών λύσεων πριν ξεκινήσει η διαδικασία αξιολόγησης του γενετικού αλγορίθμου.

5. Εξέλιξη πληθυσμού

Το αρχείο *schedule_solver_run.php* εκτελεί τη δημιουργία των γενεών του γενετικού αλγορίθμου. Σε κάθε γενιά αξιολογείται ο τρέχων πληθυσμός, ταξινομούνται οι λύσεις με βάση το σκορ τους και δημιουργείται νέος πληθυσμός. Η διαδικασία περιλαμβάνει:

- διατήρηση των καλύτερων λύσεων μέσω elitism,
- επιλογή γονέων με tournament selection,
- δημιουργία παιδιών με crossover,
- τυχαίες μεταλλάξεις σε γονίδια,
- επιδιόρθωση παιδιών μετά τη μετάλλαξη,
- αποθήκευση της καλύτερης λύσης που έχει βρεθεί συνολικά.

Η χρήση elitism εξασφαλίζει ότι οι καλύτερες λύσεις δεν χάνονται από τη μία γενιά στην επόμενη. Η επιλογή tournament selection ευνοεί τις καλύτερες λύσεις, χωρίς όμως να αποκλείει πλήρως τη συμμετοχή λιγότερο καλών λύσεων, διατηρώντας έτσι ποικιλία στον πληθυσμό. Το crossover δημιουργεί νέα χρωμοσώματα συνδυάζοντας τμήματα από δύο γονείς. Η μετάλλαξη επιτρέπει την τυχαία αλλαγή ορισμένων γονιδίων, ώστε ο αλγόριθμος να μπορεί να εξερευνήσει νέες περιοχές του χώρου λύσεων και να μη μένει εύκολα εγκλωβισμένος σε τοπικά βέλτιστες λύσεις.

Παράμετρος	Τυπική τιμή
Μέγεθος πληθυσμού (population)	50 ~ 100
Μέγιστες γενιές (generations)	200 ~ 300
Γονίδια (genes)	~100+
Αναπαράσταση Χρωμοσώματος	Integer (as array)
Μέγεθος Χρωμοσώματος	~1800 bits
Μαθήματα	~75
Ωριαίες Θέσης	80
Selection	Tournament
Mutation	Τυχαία με ποσοστό 5%
Repair	Ναι
Χρόνος Εκτέλεσης	290sec max

Πίνακας ΣΤ-1: Παράμετροι εκτέλεσης επιλυτή γενετικού αλγόριθμου

6. Τμηματική εκτέλεση

Η εκτέλεση του αλγορίθμου γίνεται τμηματικά. Αντί να εκτελούνται όλες οι γενιές σε ένα μεγάλο αίτημα, το σύστημα εκτελεί μικρές παρτίδες γενεών, αποθηκεύει την πρόοδο σε αρχείο JSON και συνεχίζει με επόμενο αίτημα.

Η επιλογή αυτή έγινε για πρακτικούς λόγους, καθώς η δημιουργία και αξιολόγηση μεγάλου αριθμού πιθανών προγραμμάτων μπορεί να απαιτεί σημαντικό χρόνο εκτέλεσης. Με την τμηματική εκτέλεση αποφεύγονται τα server timeouts και επιτρέπεται η σταδιακή παρακολούθηση της προόδου του solver.

Στην παρούσα φάση η εκτέλεση βρίσκεται ακόμη σε στάδιο βελτιστοποίησης, με στόχο τη μείωση του χρόνου ανά γενιά ώστε η λειτουργία να μπορεί να ενσωματωθεί αποτελεσματικά στο τελικό διαχειριστικό περιβάλλον.

7. Αποθήκευση αποτελεσμάτων και προβολή

Μετά την ολοκλήρωση της εκτέλεσης, οι πέντε καλύτερες λύσεις για όλα τα εξάμηνα της επιλεγμένης περιόδου αποθηκεύονται σε προσωρινούς πίνακες της Βάσης Δεδομένων. Οι πίνακες αυτοί έχουν δομή αντίστοιχη με τους πίνακες δημιουργίας και διαχείρισης του κανονικού ωρολογίου προγράμματος. Με τον τρόπο αυτό οι Διαχειριστές μπορούν να προβάλλουν τις παραγόμενες λύσεις μέσα από το κανονικό περιβάλλον της πλατφόρμας, όπως ακριβώς εμφανίζονται τα ωρολόγια προγράμματα, καθώς και να επεξεργάζονται απευθείας τα δεδομένα μέσω του ημερολογίου διαχείρισης.

Επιπλέον, παρέχεται η δυνατότητα οριστικής αποθήκευσης μίας επιλεγμένης λύσης ως κανονικού ωρολογίου προγράμματος εξαμήνου.

Για την αποφυγή συσσώρευσης μη απαραίτητων δεδομένων και τη διατήρηση απλούστερης διαχείρισης των αποτελεσμάτων, σε κάθε νέα εκτέλεση του solver διαγράφονται τα προηγούμενα προσωρινά δεδομένα που αντιστοιχούν στην ίδια ακαδημαϊκή περίοδο και στο ίδιο ακαδημαϊκό έτος.

8. Συμπληρωματική Μερική Επίλυση

Κατά την εκτέλεση του επιλυτή, ο χρήστης έχει τη δυνατότητα χρήσης ενός ήδη μερικώς συμπληρωμένου ωρολογίου προγράμματος. Στην περίπτωση αυτή, τα ήδη καταχωρημένα δεδομένα μπορούν να θεωρηθούν σταθερά και ο αλγόριθμος να αναζητήσει λύσεις μόνο για τα μαθήματα και τις διδακτικές ώρες που δεν έχουν ακόμη τοποθετηθεί.

Η δυνατότητα αυτή επιτρέπει τη συμπληρωματική ή τμηματική δημιουργία προγράμματος, διευκολύνοντας περιπτώσεις όπου μέρος του προγράμματος έχει ήδη καθοριστεί χειροκίνητα από τη Γραμματεία ή τους Διαχειριστές.

9. Συνολική ροή λειτουργίας

Συνοπτικά, η αλγοριθμική ροή είναι η εξής:

- Εξαγωγή και προετοιμασία δεδομένων από τη βάση.
- Δημιουργία πιθανών επιλογών ανά μάθημα και τύπο διδασκαλίας.
- Δημιουργία αρχικού πληθυσμού χρωμοσωμάτων.
- Αξιολόγηση κάθε χρωμοσώματος με βάση σκληρούς και μαλακούς περιορισμούς.
- Επιλογή καλύτερων λύσεων.
- Παραγωγή νέων λύσεων με crossover και mutation.
- Επιδιόρθωση εμφανών συγκρούσεων.
- Αποθήκευση προόδου.
- Επανάληψη μέχρι να ολοκληρωθεί ο επιθυμητός αριθμός γενεών.
- Αποθήκευση των καλύτερων λύσεων για προβολή και επεξεργασία.

Παράρτημα Ζ: «Εργαλεία ανάλυσης και μετρικών έργου»

Εργαλεία και πηγές λογισμικού:

- PHPStan: εργαλείο στατικής ανάλυσης για PHP.

  https://phpstan.org/

- PHP CodeSniffer (phpcs): εργαλείο ελέγχου συμμόρφωσης κώδικα.

  https://github.com/squizlabs/PHP_CodeSniffer

- PHP Mess Detector (phpmd): ανάλυση ποιότητας κώδικα και πολυπλοκότητας.

  https://phpmd.org/

- PHP Metrics: υπολογισμός μετρικών ποιότητας κώδικα.

  https://phpmetrics.org/

- Psalm: εργαλείο στατικής ανάλυσης για PHP και ελέγχου ασφάλειας.

  https://psalm.dev/

- QCacheGrind: εργαλείο για την οπτικοποίηση της σειράς και των χρόνων εκτέλεσης εντολών.

  https://github.com/KDE/kcachegrind

- Apache JMeter: εργαλείο προσομοίωσης για δοκιμές απόδοσης, φόρτου και αντοχής.

  https://jmeter.apache.org/

Παράρτημα Η: «Αποσπάσματα Κώδικα»

Παρακάτω παρουσιάζονται ενδεικτικά αποσπάσματα κώδικα:

Αρχείο admin_main_common.php: Περιλαμβάνει συναρτήσεις κοινώς χρησιμοποιούμενες από σελίδες διεπαφής για την λήψη δεδομένων από τη ΒΔ και τον καθορισμό παραμέτρων (παρουσιάζεται ενδεικτικό μέρος του κώδικα).

```php
<?php

/**
 * admin_main_common.php
 * Commonly shared code and functions of the main admin pages
 * AccessLevel: Administrator
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : admin_account_manageadmin_handler.php, admin_account_createadmin_handler.php
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Classes\Database;
use SMSYS\Classes\AdminSysRepoSelect;
use SMSYS\Classes\AdminUserRepoSelect;
use SMSYS\Enum\AdminPage as NavPage;
use SMSYS\Enum\Roles;

// Prevent direct page access and check access level
$scriptFilename = filter_input(INPUT_SERVER, 'SCRIPT_FILENAME');
$basename_filename = is_string($scriptFilename) ? $scriptFilename : '';
$samePage = (basename(__FILE__) === basename($basename_filename));
// Set roles of Access Level for the guard
$requiredRoles = [Roles::SuperAdmin, Roles::Admin];
// Load page guard, if it fails any of the checks (page, role, initialization)
// it redirects to start page
require_once __DIR__ . '/../includes/_guard.php';

/** @var string $active_frontend_page */

// Get the email domain from the attributes table
$emailDomain = getAttribute('emailDomain');

// Get (if any) previous form field values if validation failed,
// to pre-fill the fields with the original values
/** @var array<int, array<string, string>> $form_posted */
$form_posted[0] = getSessionArrArr('form.' . $active_frontend_page);
// Clear previous form data saved in session if different page
if (getSessionStr('smsys_previous.form') !== $active_frontend_page) {
  unset($_SESSION['form.' . getSessionStr('smsys_previous.form')]);
  $_SESSION['smsys_previous.form'] = $active_frontend_page;
}
```

```php
// Handles email sending
require_once __DIR__ . '/../includes/_mailer.php';

// Get any returned message to show
set_alert_form_div_msg(getSessionMsg());

/**
 * Returns an options list for <select> with all departments.
 *
 * @param string $department_id     The id of the deparment selected
 * @return string                   HTML option list to output for a select element
 */
function getDepartmentOptions(string $department_id): string {
  // Create the database class instance
  $database = initDatabase();
  $options = '';
  // Get all departments with their faculty to list in select option
  $departments = $database->dbHelper()->fetchDepartmentWithFaculty();
  foreach ($departments as $dep) {
    $text = e_html(translate($dep['department_name']));
    $options .= '<option value="' . e_html($dep['dep_id']) . '" ' .
                html_selected($department_id, (string) $dep['dep_id']) . '>' .
                $text .
                '</option>' . "\n";
  }
  return $options;
}

/**
 * Returns an options list for <select> with all professor's ranks.
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @param string $rank_title        The selected rank title
 * @return string                   HTML option list to output for a select element
 */
function getRankOptions(string $rank_title): string {
  // Create the database class instance
  $database = initDatabase();
  $options = '';
  // Get all departments with their faculty to list in select option
  $repoSys = new AdminUserRepoSelect($database);
  $columns = ['rank_title'];
  $selectList = $repoSys->listRanks($columns);
  /** @var string $rank */
  foreach ($selectList as $rank) {
    $text = e_html(translate($rank));
    $options .= '<option value="' . e_html($rank) . '" ' .
                html_selected($rank_title, $rank) . '>' .
                $text .
                '</option>' . "\n";
  }
  return $options;
}

/**
```

```
 * Get list of all active faculties of the selected semester
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @param string $action                    The current action page for schema
 * @param ?string $fac_ver_id_select        If set returns only the selected faculty by id
 * @param bool $count_items                 If true count items of faculty per action page
 * @return array<int, array<string, scalar|null>>   Return a list of the faculties
 */
function getFacultiesSelectList(
  string $action, ?string $fac_ver_id_select, bool $count_items
): array {
  $database = initDatabase();
  $repoSys = new AdminSysRepoSelect($database);
  $columns = mapTableColumns(NavPage::FacultiesListing->name);
  $selectList = $repoSys->listFaculties($fac_ver_id_select, '1', $columns);
  if ($count_items) {
    $selectList = $repoSys->countInFaculties($selectList, $action);
  }
  return $selectList ?? [];
}

/**
 * Get list of all active departments of the selected semester
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @param string $action                  The current action page for schema
 * @param string|null $dep_ver_id_select  If set returns only the selected department by id
 * @param bool $count_items               If true count items of department per action page
 * @return array<int, array<string, scalar|null>>   Return a list of the departments
 */
function getDepartmentSelectList(
  string $action, ?string $dep_ver_id_select, bool $count_items
): array {
  $database = initDatabase();
  $repoSys = new AdminSysRepoSelect($database);
  $columns = mapTableColumns(NavPage::DepartmentsListing->name);
  $selectList = $repoSys->listDepartments($dep_ver_id_select, null, '1', $columns);
  if (!isSuperAdmin()) {
    $dep_id = getSessionInt('smsys_user_department');
    $selectList = array_values(array_filter($selectList, function ($row) use ($dep_id) {
      return (isset($row['dep_id'])) && ((int) $row['dep_id'] === (int) $dep_id);
    }));
  }
  return $selectList;
}

/**
 * Get list of all active professors of the selected department
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @param string $action                  The current action page for schema
 * @param array<int, int>|null $user_ids  If set returns only the professors in ids list
 * @param string|null $dep_id_select      If set returns only the selected dep professors
 * @param bool $count_items               If true count items of per department
 * @return array<int, array<string, scalar|null>>   Return a list of the professors
 */
```

```php
function getProfessorsSelectList(
  string $action, ?array $user_ids, ?string $dep_id_select, bool $count_items
): array {
  $database = initDatabase();
  $repoSys = new AdminUserRepoSelect($database);
  $columns = mapTableColumns(NavPage::ProfessorsListing->name);
  $selectList = $repoSys->listProfessors($user_ids, $dep_id_select, '>0', $columns);
  return $selectList ?? [];
}

/**
 * Get list of all active specialization categories of the selected department
 *
 * @SuppressWarnings("UnusedFormalParameter")
 * @param string $action                    The current action page for schema
 * @param string|null $spe_ver_id_select    If set returns only the selected special by id
 * @param string|null $department_id_select If set returns only the selected dep specials
 * @return array<int, array<string, scalar|null>>   Return a list of the specializations
 */
function getSpecializationsSelectList(
  string $action, ?string $spe_ver_id_select, ?string $department_id_select
): array {
  $database = initDatabase();
  $repoSys = new AdminSysRepoSelect($database);
  $columns = mapTableColumns(NavPage::SpecializationsListing->name);
  $selectList = $repoSys->listSpecializations(
    $spe_ver_id_select, $department_id_select, '1', $columns
  );
  return $selectList ?? [];
}

/**
 * Compare the current record version semester/year with selected semester/year
 *
 * @param array<string, string> $form_data
 * @return bool   True if it's the current record's semester/year same as the selected one
 */
function isSemesterCurrVer(array $form_data): bool {
  $return = false;
  if (
    (isset($form_data['valid_from_year'])) && (isset($form_data['valid_from_semester']))
  ) {
    $curr = 'smsys_curr_semester';
    $semester_select = getSessionArrStr($curr, 'year') .
      '-' . getSessionArrStr($curr, 'semester');
    $return = ($semester_select === ($form_data['valid_from_year'] .
      '-' . $form_data['valid_from_semester']));
  }
  return $return;
}

/**
 * Get a list of classrooms
 *
 * @param string|null $department_id_select
 * @return array<int, array<string, scalar|null>>
```

```
 */
function getClassroomSelectList(string $action, ?string $department_id_select): array {
  $database = initDatabase();
  $repoSys = new AdminSysRepoSelect($database);
  $columns = mapTableColumns(NavPage::ClassroomsListing->name);
  $department_id_select = (
    (getSessionInt('smsys_user_department') > 0)
    ? getSessionStr('smsys_user_department')
    : $department_id_select
  );
  $classroomList = $repoSys->listClassrooms(
    null, null, '1', $department_id_select, $columns
);
  return $classroomList;
}
```

Πίνακας Η-1: Απόσπασμα κώδικα αρχείου "admin_main_common.php"

Αρχείο logincheck.php: Αρχείο ελέγχου σύνδεσης χρήστη, το οποίο εκτελείται κατά την έναρξη κάθε σελίδας που απαιτεί σύνδεση/πιστοποίηση.

```
<?php

/**
 * logincheck.php
 * Check that user is logged in or else redirect to home page
 * AccessLevel: Administrator, RegisteredUser
 * Loads : _common.php
 * Calls : none
 * Redir : none
 * Used  : (all pages that require user to be logged in)
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Enum\Roles;

// Load application constant values
require_once __DIR__ . '/../config_smsys_const.php';

// Session starter
require_once __DIR__ . '/_session_start.php';
smsys_session_start();

// Load common stand-alone functions (if not loaded)
require_once __DIR__ . '/_common.php';

// Load common stand-alone functions (if not loaded)
require_once __DIR__ . '/../Enum/Roles.php';

// Check page ping for idle users
```

```php
$last_ping = getSessionInt('smsys_last_ping');
$pending_ping = getSessionInt('smsys_pending_ping_ok');
$is_ping_ok = ($last_ping > 0) && ((time() - $last_ping) <= LOGIN_PING_CHECK);
if (!$is_ping_ok) {
  if ((int) $pending_ping === 1) {
    unset($_SESSION['smsys_pending_ping_ok']);
    $_SESSION['smsys_last_ping'] = time();
  } else {
    unset($_SESSION['smsys_loggedin']);
  }
}

// Session check: Validates the login sessions and timeout
if (
    (!isset($_SESSION['smsys_loggedin'])) ||
    (($_SESSION['smsys_user_agent'] ?? '') !== ($_SERVER['HTTP_USER_AGENT'] ?? 'none')) ||
    (($_SESSION['smsys_ip'] ?? '') !== getIPAddress()) ||
    (getSessionInt('smsys_timeout') + LOGIN_TIMER_LOGOUT < getTimeStamp())
) {
  // Logout and redirect to login page
  $role = USER;
  if (in_array(Roles::tryFrom(getSessionStr('smsys_user_role')), Roles::adminRoles(), true)
) {
    $role = ADMIN;
  }
  logout_user('Η συνεδρία σας έχει λήξει / Session expired.', $role);
}

// Reset the timeout session in every page load
$_SESSION['smsys_timeout'] = getTimeStamp();
```

Πίνακας Η-2: Απόσπασμα κώδικα αρχείου "logincheck.php"

Αρχείο _guard.php: Αρχείο ελέγχου πρόσβασης σε μεμονωμένη σελίδα του back-end.

```php
<?php

/**
 * _guard.php
 * Backend check for direct page access in backend files, user role access and
 * correct initialization
 * AccessLevel: Administrator, RegisteredUser
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : (on-demand)
 * Linked: none
 */

// phpcs:ignoreFile PSR1.Files.SideEffects.FoundWithSymbols

declare(strict_types=1);
```

```php
use SMSYS\Enum\Roles;

$filename_tmp = filter_input(INPUT_SERVER, 'SCRIPT_FILENAME', FILTER_UNSAFE_RAW);
$filename = $filename_tmp ?? '';

// Prevent direct page access and check access level
/** @var bool $samePage */
if ($samePage !== false) {
  if (function_exists('log_error')) {
    log_error(
      getSessionStr('smsys_userid'),
      'GUARD ER1: ' . $filename . ' (samePage violation)',
      '',
      funcCallFile(2)
    );
  }
  header('Location: ../index.php' . (!IS_IN_PRODUCTION ? '?ER1-' . $filename : ''));
  die('<a href="index.php">Please click here to go to start page</a> [W01]');
}

// Check that app initialization was completed
/** @psalm-suppress ParadoxicalCondition */
if ((!defined('APP_INIT')) || (session_status() !== PHP_SESSION_ACTIVE)) {
  if (function_exists('log_error')) {
    log_error(
      getSessionStr('smsys_userid'),
      'GUARD ER2: ' . $filename . ' (APP_INIT failed)',
      '',
      funcCallFile(2)
    );
  }
  header('Location: ../index.php' . (!IS_IN_PRODUCTION ? '?ER2-' . $filename : ''));
  die('<a href="index.php">Please click here to go to start page</a> [W03]');
}

// Check that user role is set and has Access Level premission
/** @var list<\SMSYS\Enum\Roles> $requiredRoles */
if (!empty($requiredRoles)) {
  $currentRole = Roles::tryFrom(getSessionStr('smsys_user_role'));
  if (($currentRole === null) || (!in_array($currentRole, $requiredRoles, true))) {
    if (function_exists('log_error')) {
      log_error(
        getSessionStr('smsys_userid'),
        'GUARD ER3: ' . $filename . ' (role violation)',
        '',
        funcCallFile(2)
      );
    }
    header('Location: ../index.php' . (!IS_IN_PRODUCTION ? '?ER3-' . $filename : ''));
    die('<a href="index.php">Please click here to go to start page</a> [W04]');
  }
}

/**
 * Check if it is an action page to match the actual page
 *
```

```
 * @SuppressWarnings("ExitExpression")
 * @param string $action_url
 * @param string $action_filename
 * @return void
 */
function actionPageCheck(string $action_url, string $action_filename): void {
  if (basename($action_url) !== $action_filename) {
    log_error(
      getSessionStr('smsys_userid'),
      'GUARD ER0: $action_url !== $action_filename',
      '',
      funcCallFile(2)
    );
    header('Location: ../index.php' . (!IS_IN_PRODUCTION ? '?ER0' : ''));
    die('<a href="index.php">Please click here to go to start page</a> [W02A]');
  }
}

/**
 * Check if it is an action page from a disabled form
 *
 * @SuppressWarnings("ExitExpression")
 * @param bool $semester_disabled
 * @return void
 */
function actionDisabled(bool $semester_disabled): void {
  if ($semester_disabled) {
    $show_msg = 'Δεν επιτρέπεται η ενέργεια για το επιλεγμένο εξάμηνο.';
    log_error(
      getSessionStr('smsys_userid'),
      'GUARD ER4: Semester access violation',
      $show_msg,
      funcCallFile(2)
    );
    $_SESSION['smsys_message'] = $show_msg;
    header('Location: ../admin_main.php');
    die('<a href="../index.php">Please click here to go to start page</a>');
  }
}
```

Πίνακας Η-3: Απόσπασμα κώδικα αρχείου "_guard.php"

Αρχείο RepoCommon.php: Κλάση με κοινές μεθόδους χρήσης μεταξύ των κλάσεων διαχείρισης της Βάσης Δεδομένων.

```php
<?php

/**
 * RepoCommon.php
 * OOP class for common methods (cache management of table data, old values etc.)
 * AccessLevel: Public
 * Loads : Database.php
 * Calls : none
 * Redir : none
 * Used  : (on-demand)
 * Linked: none
 */

declare(strict_types=1);

namespace SMSYS\Classes;

final class RepoCommon {
  private Database $database;

  /**
   * Constructor of the repo.
   *
   * @param Database $database    The Database class to use
   */
  public function __construct(Database $database) {
    $this->database = $database;
  }

  /**
   * Return current values of fields before an update query
   *
   * @param list<string> $sql_fields,
   * @param string $sql_where
   * @param array<string, scalar> $where_params
   * @param string $table
   * @return array<string, scalar|null>|null
   */
  public function pdoSelectOldValues(
    array $sql_fields,
    string $sql_where,
    array $where_params,
    string $table
  ): array|null {
    $result = null;
    $return = null;
    $sql = '
      SELECT ' . implode(', ', $sql_fields) . '
      FROM ' . $table . '
      WHERE ' . $sql_where . '
      ' . ($table === 'tbl_attributes' ? '' : 'LIMIT 1') . '
    ';
```

```
    $stmt = $this->database->getConnection()->prepare($sql);
    $stmt->execute($where_params);
    if ($table === 'tbl_attributes') {
      /** @var array<string, scalar|null> $map */
      $map = $stmt->fetchAll(\PDO::FETCH_KEY_PAIR) ?: [];
      foreach (array_keys($where_params) as $attr_name) {
        if (array_key_exists($attr_name, $map)) {
          $result[$attr_name] = $map[$attr_name];
        }
        $return = $result ?: null;
      }
    } else {
      $return = null;
      $result = $stmt->fetchAll(\PDO::FETCH_ASSOC);
      if ($result !== false) {
        /** @var array<int, array<string, scalar|null>> $result */
        $return = $result[0] ?? null;
      }
    }
    return $return;
  }

  /**
   * Cache file return to reduce queries.
   * No single-entry/single-exit rule to avoid cyclomatic complexity.
   *
   * @param string $cacheFile  The cache file to load
   * @param array<string, array{
   *     value: array<int, int>|scalar|null, multi: bool
   * }> $params   The fields to compare/return
   * @param bool $cache_is_on
   * @return array<int, array<string, scalar|null>>|null
   */
  public function getFromCache(
    string $cacheFile,
    array $params,
    bool $cache_is_on = CACHE_IS_ON
  ): array|null {
    $return = null;
    if (($cache_is_on === true) && ($this->isCacheFileValid($cacheFile))) {
      $jsonData = $this->readCacheFile($cacheFile);
      if ($jsonData === null) {
        return $return;
      }
      $return = $jsonData;
      foreach ($params as $key => $param_def) {
        $val = $param_def['value'] ?? null;
        $isMulti = $param_def['multi'] ?? false;
        if ((is_array($val)) && (empty($val))) {
          continue;
        }
        if ((!is_array($val)) && (isNullEmpty($val))) {
          continue;
        }
        $val_tmp = $val;
        if (!is_array($val)) {
```

```
          $val_tmp = trim((string) $val);
        }
        $return = array_values(array_filter(
          $return,
          function (array $row) use ($key, $val_tmp, $isMulti): bool {
            $field_value = isset($row[$key]) ? trim((string) $row[$key]) : '';
            if ($isMulti !== true) {
              /** @var string $val_tmp */
              return ($field_value === (string) $val_tmp);
            }
            $haystack = array_map('trim', explode(',', $field_value));
            // If $val_tmp is array
            if (is_array($val_tmp)) {
              $needles = array_map(static function ($val): string {
                return trim((string) $val);
              }, $val_tmp);
              return (count(array_intersect($needles, $haystack)) > 0);
            } else {
              // If $val_tmp is scalar
              return in_array((string)$val_tmp, $haystack, true);
            }
          }
        ));
      }
    }
    return $return;
  }

  /**
   * Cache file: Check that the cache file is valid and not expired,
   * erase file after the time limit (default 1 day).
   *
   * @param string $cacheFile
   * @return bool
   */
  private function isCacheFileValid(string $cacheFile): bool {
    $isValid = false;
    if (file_exists($cacheFile)) {
      $fileTime = filemtime($cacheFile);
      $fileTime = ($fileTime === false) ? 0 : $fileTime;
      if (time() - $fileTime > EXPIRE_CACHE_FILE_SEC) {
        unlink($cacheFile);
      } else {
        $isValid = true;
      }
    }
    return $isValid;
  }

  /**
   * Cache file: Read the json data from the file
   *
   * @param string $cacheFile
   * @return array<int, array<string, scalar|null>>|null
   */
  private function readCacheFile(string $cacheFile): ?array {
```

```
    $result = null;
    $cacheData = file_get_contents($cacheFile);
    $cacheData = ($cacheData === false) ? '' : $cacheData;
    /** @var array<int, array<string, scalar|null>>|null $jsonData */
    $jsonData = json_decode($cacheData, true);
    if (is_array($jsonData)) {
      $result = $jsonData;
    }
    return $result;
  }

  /**
   * Set extra $params if not null/empty
   *
   * @param array<string, scalar|null> $params
   * @param array<string, scalar|null> $new_params
   * @return void   Adds the key/value to the array by ref if not null or empty
   */
  public function addParamNotNullEmpty(array &$params, array $new_params): void {
    foreach ($new_params as $key => $val) {
      if (!isNullEmpty($val)) {
        $params[$key] = $val;
      }
    }
  }

  /**
   * Write query result to cache file
   *
   * @param bool $write
   * @param string $cacheFile
   * @param array<
   *   int, array<string, scalar|null>>|array<string, scalar|null>|list<array<int, string
   * > $return
   * @param bool $cache_is_on
   * @return void
   */
  public function writeCache(
    bool $write, string $cacheFile, array $return, bool $cache_is_on = CACHE_IS_ON
  ): void {
    (
      $write &&
      $cache_is_on &&
      file_put_contents(
        $cacheFile, json_encode($return, JSON_UNESCAPED_UNICODE | JSON_PRETTY_PRINT)
      )
    );
  }
}
```

Πίνακας Η-4: Απόσπασμα κώδικα κλάσης "RepoCommon.php"

Αρχείο DatabaseLogger.php: Κλάση για την αποθήκευση ενεργειών και καταγραφή ιστορικού και σφαλμάτων (logs) στη Βάση Δεδομένων.

```php
<?php

/**
 * DatabaseLogger.php
 * OOP database class for pdo connection and methods
 * AccessLevel: Administrator, RegisteredUser
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : _bootstrap.php
 * Linked: none
 */

declare(strict_types=1);

namespace SMSYS\Classes;

use PDO;

final class DatabaseLogger {
  private PDO $pdo;

  // Hide fields from logging
  private const LOG_HIDE_COLS = [
    'tbl_login_user' => ['user_password', 'user_session_id'],
  ];
  // Encrypt fields in log
  private const LOG_ENCRYPT_COLS = [
    'first_name','last_name','father_name','user_email','mobile_phone','landline_phone'
  ];

  /**
   * Constructor of the database connection class with PDO
   */
  public function __construct(PDO $pdo) {
    $this->pdo = $pdo;
  }

  /**
   * Friendly duplicate data message.
   * No single-entry/single-exit rule to avoid cyclomatic complexity.
   * @param \PDOException $err    The pdo error message to log
   */
  private function buildFriendlyDuplicateMessage(\PDOException $err): string {
    if (($err->getCode() === '23000') && (($err->errorInfo[1] ?? '0') === '1062')) {
      return '';
    }
    $err_msg = $err->getMessage();
    $field_pos = strpos($err_msg, ' for key \'');
    if ($field_pos === false) {
      return '';
    }
```

```
    $field = substr($err_msg, $field_pos + 10);
    $col_pos = strpos($field, '\'');
    if ($col_pos === false) {
      return '';
    }
    $field = substr($field, 0, $col_pos);
    return (string) uniqueDuplicateFriendlyMessage($field);
  }

  /**
   * Database query error logging.
   * @param \PDOException $err    The pdo error message to log
   * @param string $errNum        Custom error numner for debugging
   */
  public function catchError(\PDOException $err, string $errNum): void {
    // If it is a duplicate key error, return friendly message
    $friendly_msg = $this->buildFriendlyDuplicateMessage($err);
    $_SESSION['smsys_db_friendly_error_msg'] = trim(
      getSessionStr('smsys_db_friendly_error_msg') . ' ' . $friendly_msg
    );
    if (!IS_IN_PRODUCTION) {
      $full_msg = 'Database connection failed: ' . $err->getMessage();
      echo $full_msg;
      $_SESSION['smsys_error_messages'] = $full_msg;
    }
    log_error(
      getSessionStr('smsys_userid'),
      'Database connection failed ' . $errNum . ': ' . $err->getMessage(),
      $friendly_msg,
      funcCallFile(3)
    );
  }

  /**
   * Keep database value changes log
   * @param string $table
   * @param array<string, int|string> $pk_json     Primary key (supports composite keys)
   * @param array<string, array{old:mixed, new:mixed}> $changes_json   Map of column
   * @param string $sql_comm                       'UPDATE'|'INSERT'|'DELETE'
   * @return void
   */
  public function logChanges(
    string $table, array $pk_json, array $changes_json, string $sql_comm = 'UPDATE'
  ): void {
    // Remove hidden or secret fields
    foreach (array_keys($changes_json) as $col) {
      if ((isset($changes_json[$col]['old'])) && (is_string($changes_json[$col]['old']))) {
        $value = $changes_json[$col]['old'];
        if (in_array($col, (self::LOG_HIDE_COLS[$table] ?? []), true)) {
          $value = str_repeat('#', mb_strlen($value));
        } elseif (in_array($col, (self::LOG_ENCRYPT_COLS ?? []), true)) {
          $value = encryptAES((string) $value);
        }
        $changes_json[$col]['old'] = $value;
      }
      if ((isset($changes_json[$col]['new'])) && (is_string($changes_json[$col]['new']))) {
```

```
        $value = $changes_json[$col]['new'];
        if (in_array($col, (self::LOG_HIDE_COLS[$table] ?? []), true)) {
          $value = str_repeat('#', mb_strlen($value));
        } elseif (in_array($col, (self::LOG_ENCRYPT_COLS ?? []), true)) {
          $value = encryptAES((string) $value);
        }
        $changes_json[$col]['new'] = $value;
      }
    }
    if (!$changes_json) {
      return; // nothing to log
    }
    $sql = '
      INSERT INTO tbl_change_log
      (ch_user_id, ch_table_name, ch_pk_json, ch_changes_json, ch_comm, ch_user_ip)
      VALUES (
        :ch_user_id, :ch_table_name, :ch_pk_json, :ch_changes_json, :ch_comm, :ch_user_ip
      )
    ';
    $params = [
      ':ch_user_id'      => getSessionStr('smsys_userid'),
      ':ch_table_name'   => $table,
      ':ch_pk_json'      => json_encode(
        $pk_json, JSON_UNESCAPED_UNICODE | JSON_THROW_ON_ERROR
      ),
      ':ch_changes_json' => json_encode(
        $changes_json, JSON_UNESCAPED_UNICODE | JSON_THROW_ON_ERROR
      ),
      ':ch_comm'         => $sql_comm,
      ':ch_user_ip'      => getSessionStr('smsys_ip'),
    ];
    $stmt = $this->pdo->prepare($sql);
    $stmt->execute($params);
  }

  /**
   * Helper to find changes in update commands
   * @param array<string, scalar|null>|null $old_values
   * @param array<string, scalar|null> $new_values
   * @return array<string, array{old:mixed, new:mixed}>
   */
  public function findChangedValues(?array $old_values, array $new_values): array {
    $log_changes = [];
    foreach ($new_values as $col => $new) {
      $old = $old_values[$col] ?? null;
      if ($new !== $old) {
        $log_changes[$col] = ['old' => $old, 'new' => $new];
      }
    }
    return $log_changes;
  }
}
```

Πίνακας Η-5: Απόσπασμα κώδικα κλάσης "DatabaseLogger.php"

Αρχείο _logs.php: Κοινές συναρτήσεις για την αποθήκευση ενεργειών και καταγραφή ιστορικού και σφαλμάτων (logs), με χρήση της κλάσης DatabaseLogger (παρουσιάζεται μέρος του κώδικα).

```php
<?php

/**
 * _logs.php
 * Global log functions to log  actions and errors
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : _bootstrap.php
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Classes\Database;

/**
 * Write to login log.
 *
 * @param string $username      The username/email to store (encrypted)
 * @param int $login_attempt    Number of login attempts to log for security count checks
 * @param string $log_type      Type of login (success, try, login failed, reset, 2fa etc)
 * @param string $user_id_log   User id if associated/logged in else 0
 * @return void                 Only writes log file, no return
 */
function log_login(
  string $username, int $login_attempt, string $log_type, string $user_id_log = '0'
): void {
  $user_id = (is_numeric($user_id_log)) ? (int) $user_id_log : 0;
  try {
    $pdo = initDatabase()->getConnection();
    // Write attempt to log
    $stmt = $pdo->prepare('
      INSERT INTO tbl_login_log (
        log_user_id, log_username, log_ip_address, log_success, log_type
      )
      VALUES (?, ?, ?, ?, ?)
    ');
    $stmt->execute(
      [
        $user_id,
        encryptAES($username),
        encryptAES(getIPAddress()),
        $login_attempt,
        $log_type
      ]
    );
  } catch (PDOException $err) {
    // Keep error log in raw text file in case there is a database connection error
```

```
    text_log_error('log_login failed: ' . $err->getMessage() .
                   ', date/time=' . MYDATE . '@' . date('H:i:s') .
                   ', username=' . encryptAES($username) .
                   ', user_id=' . $user_id .
                   ', log_ip_address=' . encryptAES(getIPAddress()) .
                   ', login_attempt=' . $login_attempt .
                   ', log_type=' . $log_type);
  }
}

/**
 * Write to logout log.
 *
 * @param string $username  The username/email to store (encrypted)
 * @param string $log_type  Type of login (logout + role)
 * @param string $user_id   The user id that logged out
 * @return void             Only writes log file, no return
 */
function log_logout(string $username, string $log_type, string $user_id): void {
  $pdo = initDatabase()->getConnection();
  try {
    $pdo->beginTransaction();
    // Write logout to log
    $stmt = $pdo->prepare('
      INSERT INTO tbl_login_log (
        log_user_id, log_username, log_ip_address, log_success, log_type
      )
      VALUES (?, ?, ?, 1, ?)
    ');
    $stmt->execute(
      [$user_id, encryptAES($username), encryptAES(getIPAddress()), $log_type]
    );
    // Clear session id it login table
    $stmt = $pdo->prepare('
      UPDATE tbl_login_user
      SET user_session_id = NULL
      WHERE user_id = ?
    ');
    $stmt->execute([$user_id]);
    $pdo->commit();
  } catch (PDOException $err) {
    try {
      $pdo->rollBack();
    } catch (\Throwable $skip) {
      // no action
    }
    // Keep error log in raw text file in case there is a database connection error
    text_log_error('log_logout failed: ' . $err->getMessage() .
                   ', date/time=' . MYDATE . '@' . date('H:i:s') .
                   ', username=' . encryptAES($username) .
                   ', user_id=' . $user_id .
                   ', log_ip_address=' . encryptAES(getIPAddress()) .
                   ', log_type=' . $log_type);
  }
}
```

```php
/**
 * Write to error log.
 *
 * @param string $user_id         The user's id
 * @param string $error_msg       The error message either custom or system
 * @param string|null $show_msg   Text to show to the user (or null if no show)
 * @param string|null $source     The error source filename
 * @return void                   Only writes log file, no return
 */
function log_error(
  string $user_id, string $error_msg, ?string $show_msg, ?string $source
): void {
  try {
    $pdo = initDatabase()->getConnection();
    // Write error log
    $stmt = $pdo->prepare('
      INSERT INTO tbl_error_log
        (
          error_log_user_id, error_log_ip_address, error_log_source,
          error_log_msg, error_log_show_msg
        )
      VALUES (?, ?, ?, ?, ?)
    ');
    $stmt->execute([$user_id, encryptAES(getIPAddress()), $source, $error_msg, $show_msg]);
  } catch (PDOException $err) {
    // Keep error log in raw text file in case there is a database connection error
    text_log_error('log_error failed: ' . $err->getMessage() .
                   ', date/time=' . MYDATE . '@' . date('H:i:s') .
                   ', user_id=' . $user_id .
                   ', log_ip_address=' . encryptAES(getIPAddress()) .
                   ', source=' . ($source ?? '') .
                   ', log_msg=' . $error_msg .
                   ', log_show_msg=' . ($show_msg ?? ''));
  }
}

/**
 * Write error log to raw text file in case the database is not accessible.
 *
 * @param string $errorText  The error message to log
 * @return void              Only writes log file, no return
 */
function text_log_error(string $errorText): void {
  try {
    $logDir = __DIR__ . '/../stats/logs';
    // Create directory if it dones not exist
    if (!is_dir($logDir)) {
      mkdir($logDir, 0755, true);
    }
    //
    $filename = $logDir . '/error_log_' . CURRDATE . '.log';
    $file = fopen($filename, 'a+');
    if ($file) {
      fwrite($file, $errorText . PHP_EOL);
      fclose($file);
    } else {
```

```
      if (!IS_IN_PRODUCTION) {
        echo('ERROR: Failed to write to log file: ' . $filename . ', ' . $errorText);
      }
    }
  } catch (Throwable $err) {
    if (!IS_IN_PRODUCTION) {
      echo('ERROR: Logging failed: ' . $err->getMessage() . ', ' . $errorText);
    }
  }
}
```

Πίνακας Η-6: Απόσπασμα κώδικα αρχείου "_logs.php"

Αρχείο _global.php: Καθολικές συναρτήσεις συστήματος, το οποίο εκτελείται κατά την έναρξη κάθε σελίδας. Περιλαμβάνει συναρτήσεις ελέγχου δεδομένων, σχημάτων, κρυπτογράφησης καθώς και καθολική συσχέτιση κλάσεων μέσω συναρτήσεων για άμεση πρόσβαση από άλλα αρχεία.

```
<?php

/**
 * _global.php
 * Global used functions, constants and global variables
 * AccessLevel: Public
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : _bootstrap.php
 * Linked: none
 */

declare(strict_types=1);

use SMSYS\Classes\Database;
use SMSYS\Classes\FormFields;
use SMSYS\Classes\Translator;
use SMSYS\Enum\Roles;

/**
 * Get the active user's role if is a super admin
 *
 * @SuppressWarnings("StaticAccess")
 * @return bool   True if is it superadmin, else false
 */
function isSuperAdmin(): bool {
  $user_role = Roles::tryFrom(getSessionStr('smsys_user_role'));
  return ($user_role === Roles::SuperAdmin);
}
/**
 * Get the active user's role if is an admin (or super admin)
 *
```

```
 * @SuppressWarnings("StaticAccess")
 * @return bool   True if is it superadmin, else false
 */
function isAdmin(): bool {
  $user_role = Roles::tryFrom(getSessionStr('smsys_user_role'));
  return (($user_role === Roles::SuperAdmin) || ($user_role === Roles::Admin));
}

/**
 * Get the Database class instance
 *
 * @SuppressWarnings("ExitExpression")
 * @return \SMSYS\Classes\Database  Returns the singletone of the database instance
 */
function initDatabase(): \SMSYS\Classes\Database {
  // Exit to home page if database connection is unavailable
  if (!isset($GLOBALS['smsys_database'])) {
    header('Location: ' . LOCAL_SERVER . 'index.php');
    die('<a href="../index.php">Servic is temporarily unavailable. Go to start page</a>');
  }
  /** @var \SMSYS\Classes\Database $database */
  $database = $GLOBALS['smsys_database'];
  return $database;
}

/**
 * Get the FormFields class instance.
 *
 * @return \SMSYS\Classes\FormFields  Returns the class instance
 */
function formFields(): \SMSYS\Classes\FormFields {
  /** @var \SMSYS\Classes\FormFields $formFields */
  $formFields = $GLOBALS['smsys_formFields'];
  return $formFields;
}

/**
 * Initialize translator.
 *
 * @return Translator
 */
function getTranslator(): Translator {
  /** @var Translator $trl_instance */
  $trl_instance = new Translator(getSessionStr('smsys_language'));
  return $trl_instance;
}
/**
 * Call the translator class.
 *
 * @SuppressWarnings("BooleanArgumentFlag")
 * @param string $txt   The text to translate or convert to ELOT743
 * @param bool $noLog   If true then don't log any not found words
 * @return string
 */
function translate(string $txt, bool $noLog = false): string {
  return getTranslator()->translate($txt, $noLog);
```

```php
}
function elot743(string $txt): string {
  return getTranslator()->elot743Convert($txt);
}

/**
 * Create csrf token pool if not set.
 *
 * @return void   Creates session, no return needed
 */
function create_csrf_tokens(): void {
  if (empty($_SESSION['smsys_csrf_token']) || !is_array($_SESSION['smsys_csrf_token'])) {
    $_SESSION['smsys_csrf_token'] = [];
    for ($count = 0; $count < CSRF_TOKEN_POOL; $count++) {
      $_SESSION['smsys_csrf_token'][] = bin2hex(random_bytes(32));
    }
  }
}

/**
 * Form submission verification check for security.
 *
 * @param string $role    The current user page role to manage logout page
 */
function formVerification(string $role = ''): void {
  create_csrf_tokens();
  if (($_SERVER['REQUEST_METHOD'] ?? '') === 'POST') {
    $csrf_token = formFields()->sanitizePost('csrf_token', true, true, true);
    if (!in_array($csrf_token, getSessionArrArr('smsys_csrf_token'), true)) {
      logout_user('Session expired. Please log in or try again.', $role);
    }
  }
}

/**
 * Return a random csrf token, creates the tokens if not set.
 *
 * @return string   One of the random csrf tokens
 */
function get_csrf_token(): string {
  create_csrf_tokens();
  $csrf_token = getSessionArrArr('smsys_csrf_token');
  /** @var string[] $csrf_token */
  return $csrf_token[array_rand($csrf_token)];
}

/**
 * Getter for the form field unique token per session (re-create it if not set)
 *
 * @return string
 */
function get_form_field_token(): string {
  if (empty($_SESSION['smsys_form_field_token'])) {
    $_SESSION['smsys_form_field_token'] = bin2hex(random_bytes(32));
  }
  return getSessionStr('smsys_form_field_token');
```

```
}

/**
 * Create a unique ping csrf token. It is created once per login,
 * to avoid unwanted befavior when user opens other tab.
 *
 * @return string
 */
function set_ping_csrf(): string {
  $existing = $_SESSION['smsys_ping_csrf'] ?? '';
  if ((is_string($existing)) && ($existing !== '')) {
    return $existing;
  }
  $ping_csrf = bin2hex(random_bytes(32));
  $_SESSION['smsys_ping_csrf'] = $ping_csrf;
  return $ping_csrf;
}

/**
 * Single login check, same user cannot be logged in from different devices/browsers.
 *
 * @SuppressWarnings("ErrorControlOperator")
 * @param string $user_id           Current user's id
 * @param string $old_user_session  Last session id the current user had
 */
function forceSingleLogIn(string $user_id, string $old_user_session): void {
  if (!isNullEmpty($old_user_session)) {
    $old_user_session = decryptAES_public($old_user_session);
    $current_session_id = session_id();
    // Sanitizesession id charset/length for PHP
    if ((preg_match('/^[A-Za-z0-9,-]{22,128}$/', $old_user_session))) {
      // Close current session to release lock
      session_write_close();
      session_name(SMSYS_SESSION_NAME);
      // Try switching to the old session id, start it (if it exists), and destroy it
      if (session_id($old_user_session) === false) {
        /* Session does not exists or cannot be set */
      } else {
        $old_session_started = @session_start();
        if (($old_session_started) && (session_id() === $old_user_session)) {
          // Destroy if it is the same user (secures for collision)
          if (
            (isset($_SESSION['smsys_userid'])) && ($_SESSION['smsys_userid'] === $user_id)
          ) {
            $_SESSION = [];
            session_destroy();
          }
        }
        @session_write_close();
      }
      // Restore current session
      if ($current_session_id === false) {
        $current_session_id = null;
      }
      session_id($current_session_id);
      // Session starter
```

```
      smsys_session_start();
    }
  }
}

// Enryption functions
/**
 * @SuppressWarnings("ExitExpression")
 */
function encryptAES_common(string $plainText, string $key): string {
  $cipher = 'AES-256-CBC';
  $iv_length = openssl_cipher_iv_length($cipher);
  if ($iv_length === false || $iv_length <= 0) {
    log_error(
      getSessionStr('smsys_userid'),
      'Error: ' . __FUNCTION__,
      'Something went wrong. [ERR0001A]',
      funcCallFile(1)
    );
    exit;
  }
  try {
    $iv_key = random_bytes($iv_length);
  } catch (Throwable $exception) {
    log_error(
      getSessionStr('smsys_userid'),
      'Error: ' . __FUNCTION__,
      'Something went wrong. [ERR0001B]',
      funcCallFile(1)
    );
    exit;
  }
  $encrypted = openssl_encrypt(
    (string) $plainText, $cipher, $key, OPENSSL_RAW_DATA, $iv_key
  );
  if ($encrypted === false) {
    log_error(
      getSessionStr('smsys_userid'),
      'Error: ' . __FUNCTION__,
      'Something went wrong. [ERR0001]',
      funcCallFile(1)
    );
    exit;
  }
  return base64_encode($iv_key . $encrypted);
}

/**
 * @SuppressWarnings("ExitExpression")
```

```
 * @param string $cipherText
 * @param string $key
 * @return string
 */
function decryptAES_common(string $cipherText, string $key): string {
  $cipher = 'AES-256-CBC';
  $iv_length = openssl_cipher_iv_length($cipher);
  if ($iv_length === false || $iv_length <= 0) {
    log_error(
      getSessionStr('smsys_userid'),
      'Error: ' . __FUNCTION__,
      'Something went wrong. [ERR0002A]',
      funcCallFile(1)
    );
    exit;
  }
  $decoded = base64_decode($cipherText, true);
  if ($decoded !== false && strlen($decoded) > $iv_length) {
    $iv_key = substr($decoded, 0, $iv_length);
    $encrypted = substr($decoded, $iv_length);
    $decrypted = openssl_decrypt($encrypted, $cipher, $key, OPENSSL_RAW_DATA, $iv_key);
    if ($decrypted !== false) {
      return $decrypted;
    }
  }
  /** @var string $legacy_iv_key */
  $legacy_iv_key = ENCRYPTION_IV;
  $decrypted = openssl_decrypt(
    base64_decode($cipherText), $cipher, $key, 0, $legacy_iv_key
  );
  if ($decrypted === false) {
    log_error(
      getSessionStr('smsys_userid'),
      'Error: ' . __FUNCTION__,
      'Something went wrong. [ERR0002]' .
        (!IS_IN_PRODUCTION ? ' decryptAES_:' . $cipherText : ''),
      funcCallFile(1)
    );
    exit;
  }
  return $decrypted;
}
function encryptAES(string $plainText): string {
  return encryptAES_common($plainText, ENCRYPTION_KEY);
}
function decryptAES(string $cipherText): string {
  return decryptAES_common($cipherText, ENCRYPTION_KEY);
}
function encryptAES_public(string $plainText): string {
```

```
  return encryptAES_common($plainText, PUBLIC_KEY);
}
function decryptAES_public(string $cipherText): string {
  return decryptAES_common($cipherText, PUBLIC_KEY);
}
function encryptAES_daily(string $plainText): string {
  return encryptAES_common($plainText, PUBLIC_KEY . CURRDATE);
}
function decryptAES_daily(string $cipherText): string {
  return decryptAES_common($cipherText, PUBLIC_KEY . CURRDATE);
}
```

Πίνακας Η-7: Απόσπασμα κώδικα αρχείου "_global.php"

Αρχείο Database.php: Κύρια κλάση σύνδεσης με τη Βάση Δεδομένων, καθορισμού παραμέτρων PDO Και διαχείρισης συνδέσεων και συναλλαγών (transactions).

```
<?php

/**
 * Database.php
 * OOP database class for pdo connection and methods
 * AccessLevel: Administrator, RegisteredUser
 * Loads : none
 * Calls : none
 * Redir : none
 * Used  : _bootstrap.php
 * Linked: none
 */

declare(strict_types=1);

namespace SMSYS\Classes;

use SMSYS\Enum\Roles;
use LogicException;
use RuntimeException;
use PDO;

final class Database {
  private static ?self $instance = null;
  private DatabaseLogger $dbLogger;
  private DatabaseHelper $dbHelper;
  private PDO $pdo;
  // Retries and delay for connection problems
  private const CONNECTION_MAX_TRIES   = 5;
  private const CONNECTION_DELAY_MS = 150;
  private const CONNECTION_TIMEOUT_S = 5;

  /**
   * Constructor of the database connection class with PDO
   */
  private function __construct() {
```

```
    $charset = 'utf8mb4';
    $dsn = 'mysql:host=' . DB_HOST . ';dbname=' . DB_NAME . ';charset=' . $charset;
    $options = [
      \PDO::ATTR_ERRMODE            => \PDO::ERRMODE_EXCEPTION,
      \PDO::ATTR_DEFAULT_FETCH_MODE => \PDO::FETCH_ASSOC,
      \PDO::ATTR_EMULATE_PREPARES   => false,
      \PDO::ATTR_TIMEOUT            => self::CONNECTION_TIMEOUT_S,
      \PDO::ATTR_PERSISTENT         => false,
      \PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8mb4 COLLATE utf8mb4_unicode_ci',
    ];
    // Try connection for a maximum number of times in case of connection problem
    $pdo = null;
    for ($tries = 1; $tries <= self::CONNECTION_MAX_TRIES; $tries++) {
      try {
        $pdo = new PDO($dsn, DB_USER, DB_PASS, $options);
        // Ensure Greek/UTF-8 is used in dn
        $pdo->exec("SET NAMES utf8mb4");
        $pdo->exec("SET character_set_client = utf8mb4");
        $pdo->exec("SET character_set_connection = utf8mb4");
        $pdo->exec("SET character_set_results = utf8mb4");
        $pdo->exec("SET collation_connection = 'utf8mb4_unicode_ci'");
        break; // connected so exit for
      } catch (\PDOException $err) {
        // Do not retry on non-transient connect errors
        $err_code = (string) $err->getCode();
        $no_retry = in_array($err_code, ['1045', '1044'], true);
        if (($no_retry) || ($tries >= self::CONNECTION_MAX_TRIES)) {
          $errNum = ' [ERR0028]';
          if (!IS_IN_PRODUCTION) {
            $full_msg = 'Database connection failed: ' . $err->getMessage() . $errNum;
            echo($full_msg);
            $_SESSION['smsys_error_messages'] = $full_msg;
          }
          log_error(
            getSessionStr('smsys_userid'),
            'Database connection failed ' . $errNum . ': ' . $err->getMessage(),
            'Database connection failed',
            funcCallFile(3)
          );
          throw $err;
        }
        // Add a delay in each connection re-try
        $connetion_delay = self::CONNECTION_DELAY_MS * $tries;
        usleep($connetion_delay * 1000);
      }
    }
    if ($pdo === null) {
      throw new RuntimeException('PDO was not initialized after the retries');
    }
    $this->pdo = $pdo;
    $this->dbLogger = new DatabaseLogger($this->pdo);
    $this->dbHelper = new DatabaseHelper($this->pdo, $this->dbLogger);
  }

  /**
   * Singletone instance of the database.
```

```
     * @return Database
     */
    public static function getInstance(): self {
      if (self::$instance === null) {
        self::$instance = new self();
      }
      return self::$instance;
    }

    /**
     * Returns the PDO connection of the database for direct usage of pdo prepare
     * @return \PDO
     */
    public function getConnection(): \PDO {
      return $this->pdo;
    }

    /**
     * Use sql query format to write database queries and transform to pdo with prepare.
     * @param string $sql   The sql query to execute
     * @param array<int|string, string|int|float|bool|null> $params
     * @return \PDOStatement
     */
    public function pdoQuery(string $sql, array $params = []): \PDOStatement {
      try {
        $stmt = $this->pdo->prepare($sql);
        $stmt->execute($params);
        return $stmt;
      } catch (\PDOException $err) {
        $this->dbLogger->catchError($err, '[ERR0029]');
        throw $err;
      }
    }

    /** Get helper.
     * @return DatabaseHelper
     */
    public function dbHelper(): DatabaseHelper {
      return $this->dbHelper;
    }

    /** Get logger.
     * @return DatabaseLogger
     */
    public function dbLogger(): DatabaseLogger {
      return $this->dbLogger;
    }

    /**
     * Returns the last insert id in the database
     * @return string   The last inserted id in the database as a string
     */
    public function lastInsertId(): string {
      $last = $this->pdo->lastInsertId();
      $return = '';
      if ($last !== false) {
```

```
      $return = $last;
    }
    return $return;
  }

  // Functions for INSERT/UPDATE/DELETE queries
  /**
   * Begins a database pdo transaction
   */
  public function beginTransaction(): void {
    $this->pdo->beginTransaction();
  }
  /**
   * Commits the sql pdo command
   */
  public function commit(): void {
    $this->pdo->commit();
  }
  /**
   * Rollback the sql pdo in case of an error
   */
  public function rollBack(): void {
    try {
      $this->pdo->rollBack();
    } catch (\Throwable $skip) {
      // no action
    }
  }

  /**
   * Check if a call is within an active transaction
   * @param string $context
   * @return void
   */
  public function checkActiveTransaction(string $context = ''): void {
    if (!$this->pdo->inTransaction()) {
      $msg = $context
       ? "$context requires an active transaction."
       : 'Requires active transaction.';
      throw new LogicException($msg);
    }
  }

  /**
   * Close the database connection manualy if necessary
   * @return void
   */
  public function pdoDisconnect(): void {
    unset($this->pdo);
  }
}
```

Πίνακας Η-8: Απόσπασμα κώδικα κλάσης "Database.php"

Παράρτημα Θ: «Σχήμα Βάσης Δεδομένων»

Παρακάτω παρουσιάζεται το σχήμα της Βάσης Δεδομένων, εκτός των οντοτήτων και σχέσεων που αναλύθηκαν στο Κεφάλαιο 4, καθώς και οι όψεις (views):

```
-- Table structure for table `tbl_announcement`
CREATE TABLE IF NOT EXISTS `tbl_announcement` (
  `ann_id` int(11) NOT NULL,
  `ann_uni_id` varchar(16) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`ann_id`),
  UNIQUE KEY `unique_announcement__ann_uni_id` (`ann_uni_id`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_announcement_version`
CREATE TABLE IF NOT EXISTS `tbl_announcement_version` (
  `ann_ver_id` int(11) NOT NULL,
  `ann_id` int(11) NOT NULL,
  `ann_subject` varchar(256) NOT NULL,
  `ann_text` text NOT NULL,
  `ann_url` varchar(2083) DEFAULT NULL,
  `department_id` int(11) DEFAULT NULL,
  `start_from` date DEFAULT current_timestamp(),
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`ann_ver_id`),
  UNIQUE KEY `unique_announcement_version__ann_id` (`ann_id`) USING BTREE,
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_ann_subject` (`ann_subject`),
  KEY `idx_department_id` (`department_id`) USING BTREE,
  KEY `idx_valid_from_semester` (`valid_from_semester`) USING BTREE,
  CONSTRAINT `fk_ann_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_ann_id` FOREIGN KEY (`ann_id`) REFERENCES `tbl_announcement`
(`ann_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_availability_preferences`
CREATE TABLE IF NOT EXISTS `tbl_availability_preferences` (
  `pref_id` int(11) NOT NULL AUTO_INCREMENT,
  `availability_ver_id` int(11) NOT NULL,
  `ava_type` enum('0','1') NOT NULL,
  `ava_day` enum('1','2','3','4','5','6','7') NOT NULL,
  `ava_start_time` time NOT NULL,
  `ava_end_time` time NOT NULL,
  PRIMARY KEY (`pref_id`),
  UNIQUE KEY `unique_availability_prererence`
(`availability_ver_id`,`ava_type`,`ava_day`,`ava_start_time`,`ava_end_time`),
  KEY `idx_availability_ver_id` (`availability_ver_id`) USING BTREE,
  KEY `idx_ava_day` (`ava_day`),
  KEY `idx_ava_type` (`ava_type`),
  CONSTRAINT `fk_availability_ver_id` FOREIGN KEY (`availability_ver_id`)
REFERENCES `tbl_availability_version` (`ava_ver_id`) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=37 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_availability_version`
```

```
CREATE TABLE IF NOT EXISTS `tbl_availability_version` (
  `ava_ver_id` int(11) NOT NULL,
  `ava_professor_id` int(11) NOT NULL,
  `ava_department_id` int(11) NOT NULL,
  `is_final` int(1) NOT NULL DEFAULT 0,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`ava_ver_id`),
  UNIQUE KEY `unique_availability_version__professor_department_year_semester`
(`ava_professor_id`,`ava_department_id`,`valid_from_year`,`valid_from_semester`),
  KEY `idx_ava_professor_id` (`ava_professor_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_ava_department_id` (`ava_department_id`) USING BTREE,
  CONSTRAINT `fk_availability_version_department_id` FOREIGN KEY
(`ava_department_id`) REFERENCES `tbl_department` (`dep_id`),
  CONSTRAINT `fk_availability_version_professor_id` FOREIGN KEY
(`ava_professor_id`) REFERENCES `tbl_professor` (`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_classroom`
CREATE TABLE IF NOT EXISTS `tbl_classroom` (
  `cla_id` int(11) NOT NULL,
  `cla_uni_id` varchar(16) NOT NULL,
  `faculty_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`cla_id`),
  UNIQUE KEY `unique_classroom__cla_uni_id_faculty_id` (`cla_uni_id`,`faculty_id`)
USING BTREE,
  KEY `idx_faculty_id` (`faculty_id`),
  CONSTRAINT `fk_classroom_faculty_id` FOREIGN KEY (`faculty_id`) REFERENCES
`tbl_faculty` (`fac_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_classroom_course_type`
CREATE TABLE IF NOT EXISTS `tbl_classroom_course_type` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `classroom_ver_id` int(11) NOT NULL,
  `classroom_cou_type`
enum('THEORY','LAB','EXERCISE','FIELDWORK','CLINICAL','WORKSHOP') NOT NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_classroom_cou_type` (`classroom_ver_id`,`classroom_cou_type`)
USING BTREE,
  KEY `idx_classroom_ver_id` (`classroom_ver_id`) USING BTREE,
  CONSTRAINT `fk_classroom_course_type_ver_id` FOREIGN KEY (`classroom_ver_id`)
REFERENCES `tbl_classroom_version` (`cla_ver_id`) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=55 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_classroom_department`
CREATE TABLE IF NOT EXISTS `tbl_classroom_department` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `classroom_ver_id` int(11) NOT NULL,
  `classroom_department` int(11) NOT NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_classroom_department`
(`classroom_ver_id`,`classroom_department`),
  KEY `idx_classroom_ver_id` (`classroom_ver_id`) USING BTREE,
  KEY `fk_classroom_department` (`classroom_department`),
  CONSTRAINT `fk_classroom_department` FOREIGN KEY (`classroom_department`)
REFERENCES `tbl_department` (`dep_id`) ON DELETE CASCADE ON UPDATE CASCADE,
```

```
  CONSTRAINT `fk_classroom_department_ver_id` FOREIGN KEY (`classroom_ver_id`)
REFERENCES `tbl_classroom_version` (`cla_ver_id`) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=288 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_classroom_version`
CREATE TABLE IF NOT EXISTS `tbl_classroom_version` (
  `cla_ver_id` int(11) NOT NULL,
  `cla_id` int(11) NOT NULL,
  `classroom_name` varchar(256) NOT NULL,
  `classroom_unique` varchar(256) NOT NULL,
  `faculty_id` int(11) NOT NULL,
  `classroom_category` enum('LECTURE','LABORATORY','MIXED','AUDITORIUM','OTHER')
NOT NULL,
  `max_capacity` int(4) NOT NULL DEFAULT 0,
  `building` varchar(256) DEFAULT NULL,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`cla_ver_id`),
  UNIQUE KEY `unique_classroom_version__cla_id_faculty_year_semester`
(`cla_id`,`faculty_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  UNIQUE KEY `unique_classroom_version__unique_faculty_year_semester`
(`classroom_unique`,`faculty_id`,`valid_from_year`,`valid_from_semester`) USING
BTREE,
  KEY `idx_faculty_id` (`faculty_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_classroom_name` (`classroom_name`),
  CONSTRAINT `fk_cla_id` FOREIGN KEY (`cla_id`) REFERENCES `tbl_classroom`
(`cla_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_classroom_version_faculty_id` FOREIGN KEY (`faculty_id`)
REFERENCES `tbl_faculty` (`fac_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_course_special`
CREATE TABLE IF NOT EXISTS `tbl_course_special` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `course_ver_id` int(11) NOT NULL,
  `course_special` int(11) NOT NULL,
  PRIMARY KEY (`autonum`) USING BTREE,
  UNIQUE KEY `unique_course_special` (`course_ver_id`,`course_special`) USING
BTREE,
  KEY `idx_course_ver_id` (`course_ver_id`) USING BTREE,
  KEY `idx_course_special` (`course_special`) USING BTREE,
  CONSTRAINT `fk_course_special` FOREIGN KEY (`course_special`) REFERENCES
`tbl_specialization` (`spe_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_course_special_ver_id` FOREIGN KEY (`course_ver_id`) REFERENCES
`tbl_course_version` (`cou_ver_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=109 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_ext_course`
CREATE TABLE IF NOT EXISTS `tbl_ext_course` (
  `ext_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`ext_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_ext_course_semester`
CREATE TABLE IF NOT EXISTS `tbl_ext_course_semester` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `extern_ver_id` int(11) NOT NULL,
```

```
  `course_semester` enum('1','2','3','4','5','6','7','8','9','10','11','12') NOT
NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_ext_course_semester` (`extern_ver_id`,`course_semester`),
  KEY `idx_extern_ver_id` (`extern_ver_id`) USING BTREE,
  KEY `idx_course_semester` (`course_semester`),
  CONSTRAINT `fk_ext_course_semester_ver_id` FOREIGN KEY (`extern_ver_id`)
REFERENCES `tbl_ext_course_version` (`ext_ver_id`) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_ext_course_version`
CREATE TABLE IF NOT EXISTS `tbl_ext_course_version` (
  `ext_ver_id` int(11) NOT NULL,
  `ext_id` int(11) NOT NULL,
  `course_id` int(11) NOT NULL,
  `department_id` int(11) NOT NULL,
  `course_category`
enum('MANDATORY','RESTRICTED_ELECTIVE','ELECTIVE','THESIS_SEMINAR','OPTIONAL') NOT
NULL,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`ext_ver_id`),
  UNIQUE KEY `unique_ext_course_version__course_id_department_year_semester`
(`course_id`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `fk_ext_course_version_course_id` (`course_id`),
  KEY `fk_ext_id` (`ext_id`),
  CONSTRAINT `fk_ext_course_id` FOREIGN KEY (`course_id`) REFERENCES `tbl_course`
(`cou_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_ext_course_version_deparment_id` FOREIGN KEY (`department_id`)
REFERENCES `tbl_department` (`dep_id`),
  CONSTRAINT `fk_ext_id` FOREIGN KEY (`ext_id`) REFERENCES `tbl_ext_course`
(`ext_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_holiday`
CREATE TABLE IF NOT EXISTS `tbl_holiday` (
  `hol_id` int(11) NOT NULL,
  `department_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`hol_id`),
  KEY `idx_department_id` (`department_id`),
  CONSTRAINT `fk_holiday_department` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_holiday_version`
CREATE TABLE IF NOT EXISTS `tbl_holiday_version` (
  `hol_ver_id` int(11) NOT NULL,
  `hol_id` int(11) NOT NULL,
  `holiday_name` varchar(256) NOT NULL,
  `holiday_unique` varchar(256) NOT NULL,
  `department_id` int(11) NOT NULL,
  `holiday_date` date NOT NULL,
  `holiday_duration` int(2) NOT NULL DEFAULT 1,
  `is_one_time` tinyint(1) NOT NULL DEFAULT 0,
  `is_moveable` tinyint(1) NOT NULL DEFAULT 0,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
```

```
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`hol_ver_id`) USING BTREE,
  UNIQUE KEY `unique_course_version__unique_department_year_semester`
(`holiday_unique`,`department_id`,`valid_from_year`,`valid_from_semester`) USING
BTREE,
  UNIQUE KEY `unique_holiday_version__hol_id_department_year_semester`
(`hol_id`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_course_name` (`holiday_name`),
  CONSTRAINT `fk_hol_id` FOREIGN KEY (`hol_id`) REFERENCES `tbl_holiday` (`hol_id`)
ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_holiday_version_deparment_id` FOREIGN KEY (`department_id`)
REFERENCES `tbl_department` (`dep_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_message`
CREATE TABLE IF NOT EXISTS `tbl_message` (
  `mss_id` int(11) NOT NULL,
  `mss_uni_id` varchar(16) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`mss_id`),
  UNIQUE KEY `unique_message__mss_uni_id` (`mss_uni_id`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_message_version`
CREATE TABLE IF NOT EXISTS `tbl_message_version` (
  `mss_ver_id` int(11) NOT NULL,
  `mss_id` int(11) NOT NULL,
  `mss_subject` varchar(256) NOT NULL,
  `mss_topic` enum('TECHNICAL ISSUE','DEPARTMENT_MATTERS') NOT NULL,
  `mss_text` text NOT NULL,
  `mss_no_reply` tinyint(1) NOT NULL DEFAULT 0,
  `mss_sender_id` int(11) NOT NULL,
  `mss_recipient_id` int(11) DEFAULT NULL,
  `mss_reply_id` int(11) DEFAULT NULL,
  `mss_root_id` int(11) DEFAULT NULL,
  `sent_datetime` datetime NOT NULL DEFAULT current_timestamp(),
  `read_datetime` datetime DEFAULT NULL,
  `read_by_id` int(11) DEFAULT NULL,
  `department_id` int(11) NOT NULL,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`mss_ver_id`),
  UNIQUE KEY `unique_message_version__mss_id` (`mss_id`) USING BTREE,
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_ann_subject` (`mss_subject`),
  KEY `idx_mss_recipient_id` (`mss_recipient_id`) USING BTREE,
  KEY `idx_mss_sender_id` (`mss_sender_id`) USING BTREE,
  KEY `idx_read_datetime` (`read_datetime`) USING BTREE,
  KEY `idx_mss_reply_id` (`mss_reply_id`) USING BTREE,
  KEY `idx_sent_datetime` (`sent_datetime`) USING BTREE,
  KEY `idx_mss_recipient_read` (`mss_recipient_id`,`read_datetime`) USING BTREE,
  KEY `idx_mss_root_id` (`mss_root_id`) USING BTREE,
  KEY `idx_read_by_id` (`read_by_id`) USING BTREE,
  KEY `idx_valid_from_semester` (`valid_from_semester`) USING BTREE,
  KEY `idx_department_ver_id` (`department_id`) USING BTREE,
  KEY `idx_department_id` (`department_id`) USING BTREE,
```

```
  CONSTRAINT `fk_mss_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department_version` (`dep_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_mss_id` FOREIGN KEY (`mss_id`) REFERENCES `tbl_message` (`mss_id`)
ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_mss_recipient_id` FOREIGN KEY (`mss_recipient_id`) REFERENCES
`tbl_login_user` (`user_id`),
  CONSTRAINT `fk_mss_reply_id` FOREIGN KEY (`mss_reply_id`) REFERENCES
`tbl_message` (`mss_id`),
  CONSTRAINT `fk_mss_root_id` FOREIGN KEY (`mss_root_id`) REFERENCES `tbl_message`
(`mss_id`),
  CONSTRAINT `fk_mss_sender_id` FOREIGN KEY (`mss_sender_id`) REFERENCES
`tbl_login_user` (`user_id`),
  CONSTRAINT `fk_read_by_id` FOREIGN KEY (`read_by_id`) REFERENCES `tbl_login_user`
(`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_professor_rank`
CREATE TABLE IF NOT EXISTS `tbl_professor_rank` (
  `sort_id` int(11) NOT NULL,
  `rank_title` varchar(50) NOT NULL,
  `rank_type` varchar(255) NOT NULL,
  PRIMARY KEY (`rank_title`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_request`
CREATE TABLE IF NOT EXISTS `tbl_request` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `request_type` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `record_id` int(11) NOT NULL,
  `request_user_id` int(11) NOT NULL,
  `request_status` enum('PENDING','APPROVED','REJECTED','COMPLETED') CHARACTER SET
utf8 COLLATE utf8_general_ci NOT NULL,
  `request_datetime` datetime NOT NULL DEFAULT current_timestamp(),
  `response_user_id` int(11) DEFAULT NULL,
  `response_datetime` datetime DEFAULT NULL,
  PRIMARY KEY (`autonum`) USING BTREE,
  KEY `idx_request_lookup` (`request_type`,`record_id`,`request_status`) USING
BTREE,
  KEY `idx_request_user` (`request_user_id`,`request_status`) USING BTREE,
  KEY `idx_response_user` (`response_user_id`) USING BTREE,
  CONSTRAINT `fk_request_request_user_id` FOREIGN KEY (`request_user_id`)
REFERENCES `tbl_login_user` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_request_response_user_id` FOREIGN KEY (`response_user_id`)
REFERENCES `tbl_login_user` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_general_ci;

-- Table structure for table `tbl_semester`
CREATE TABLE IF NOT EXISTS `tbl_semester` (
  `sem_id` int(11) NOT NULL,
  `department_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`sem_id`),
  KEY `idx_department_id` (`department_id`),
  CONSTRAINT `fk_semester_department_id` FOREIGN KEY (`department_id`) REFERENCES
`tbl_department` (`dep_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule` (
  `sch_id` int(11) NOT NULL AUTO_INCREMENT,
  `sem_ver_id` int(11) NOT NULL,
  `semester_number` enum('1','2','3','4','5','6','7','8','9','10','11','12') NOT
NULL,
  `sch_course_id` int(11) NOT NULL,
```

```
  `sch_course_type`
enum('THEORY','LAB','EXERCISE','FIELDWORK','CLINICAL','WORKSHOP','THEORY_LAB') NOT
NULL,
  `day_of_week` enum('1','2','3','4','5','6','7') NOT NULL,
  `hour_slot` time NOT NULL,
  `is_delete` tinyint(1) NOT NULL DEFAULT 0,
  `last_update` datetime NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`sch_id`),
  UNIQUE KEY `unique_semester__ver_number_course_type_day_hour`
(`sem_ver_id`,`semester_number`,`sch_course_id`,`sch_course_type`,`day_of_week`,`ho
ur_slot`,`is_delete`) USING BTREE,
  KEY `idx_sch_course_type` (`sch_course_type`),
  KEY `idx_sch_course_id` (`sem_ver_id`) USING BTREE,
  KEY `fk_semester_course_id` (`sch_course_id`) USING BTREE,
  KEY `idx_semester_number` (`semester_number`) USING BTREE,
  CONSTRAINT `FK_tbl_semester_schedule_tbl_course` FOREIGN KEY (`sch_course_id`)
REFERENCES `tbl_course` (`cou_id`),
  CONSTRAINT `fk_semester_ver_id` FOREIGN KEY (`sem_ver_id`) REFERENCES
`tbl_semester_version` (`sem_ver_id`)
) ENGINE=InnoDB AUTO_INCREMENT=942 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule_change`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule_change` (
  `change_id` int(11) NOT NULL AUTO_INCREMENT,
  `cha_sch_id` int(11) NOT NULL,
  `cha_type` enum('POSTPONE','MAKEUP') NOT NULL,
  `cha_is_edit` tinyint(1) NOT NULL DEFAULT 0,
  `cha_date` date NOT NULL,
  `hour_slot` time NOT NULL,
  `cha_course_type`
enum('THEORY','LAB','EXERCISE','FIELDWORK','CLINICAL','WORKSHOP','THEORY_LAB')
DEFAULT NULL,
  `cha_professor_ids` varchar(255) DEFAULT NULL,
  `cha_classroom_ids` varchar(255) DEFAULT NULL,
  `updated_by_user_id` int(11) NOT NULL,
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`change_id`),
  UNIQUE KEY `unique_semester_schedule_change`
(`cha_sch_id`,`cha_type`,`cha_date`,`hour_slot`),
  KEY `idx_cha_sch_id` (`cha_sch_id`),
  KEY `idx_cha_type` (`cha_type`),
  KEY `idx_cha_date` (`cha_date`),
  KEY `idx_hour_slot` (`hour_slot`),
  KEY `idx_updated_by_user_id` (`updated_by_user_id`),
  KEY `idx_cha_date_hour_slot` (`cha_date`,`hour_slot`),
  KEY `idx_cha_type_date` (`cha_type`,`cha_date`),
  KEY `idx_cha_sch_date` (`cha_sch_id`,`cha_date`),
  CONSTRAINT `fk_semester_schedule_change_sch_id` FOREIGN KEY (`cha_sch_id`)
REFERENCES `tbl_semester_schedule` (`sch_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_semester_schedule_change_updated_by` FOREIGN KEY
(`updated_by_user_id`) REFERENCES `tbl_login_user` (`user_id`) ON DELETE NO ACTION
ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=59 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule_classroom`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule_classroom` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `schedule_id` int(11) NOT NULL,
  `sch_classroom_id` int(11) NOT NULL,
  `last_update` datetime NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_classroom_schedule_id` (`schedule_id`,`sch_classroom_id`)
USING BTREE,
  KEY `idx_schedule_id` (`schedule_id`),
  KEY `idx_sch_classroom_id` (`sch_classroom_id`) USING BTREE,
```

```
  CONSTRAINT `fk_semester_sch_classroom` FOREIGN KEY (`sch_classroom_id`)
REFERENCES `tbl_classroom_version` (`cla_id`),
  CONSTRAINT `fk_semester_sch_id_classroom` FOREIGN KEY (`schedule_id`) REFERENCES
`tbl_semester_schedule` (`sch_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=959 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule_professor`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule_professor` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `schedule_id` int(11) NOT NULL,
  `sch_professor_id` int(11) NOT NULL,
  `last_update` datetime NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_professor_schedule_id` (`schedule_id`,`sch_professor_id`)
USING BTREE,
  KEY `idx_schedule_id` (`schedule_id`),
  KEY `idx_sch_professor_id` (`sch_professor_id`),
  CONSTRAINT `fk_semester_sch_id_professor` FOREIGN KEY (`schedule_id`) REFERENCES
`tbl_semester_schedule` (`sch_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_semester_sch_professor` FOREIGN KEY (`sch_professor_id`)
REFERENCES `tbl_professor` (`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=1017 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule_special`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule_special` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `schedule_id` int(11) NOT NULL,
  `sch_special_id` int(11) NOT NULL,
  `last_update` datetime NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`autonum`) USING BTREE,
  UNIQUE KEY `unique_special_schedule_id` (`schedule_id`,`sch_special_id`) USING
BTREE,
  KEY `idx_schedule_id` (`schedule_id`) USING BTREE,
  KEY `idx_sch_special_id` (`sch_special_id`) USING BTREE,
  CONSTRAINT `fk_semester_sch_id_special` FOREIGN KEY (`schedule_id`) REFERENCES
`tbl_semester_schedule` (`sch_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_semester_sch_special` FOREIGN KEY (`sch_special_id`) REFERENCES
`tbl_specialization_version` (`spe_id`)
) ENGINE=InnoDB AUTO_INCREMENT=724 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_semester_schedule_status`
CREATE TABLE IF NOT EXISTS `tbl_semester_schedule_status` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `sem_ver_id` int(11) NOT NULL,
  `semester_number` enum('1','2','3','4','5','6','7','8','9','10','11','12') NOT
NULL,
  `semester_spe_id` int(11) DEFAULT NULL,
  `is_final` tinyint(1) NOT NULL DEFAULT 0,
  `updated_by_user_id` int(11) DEFAULT NULL,
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_sem_ver_semester_number`
(`sem_ver_id`,`semester_number`,`semester_spe_id`) USING BTREE,
  KEY `fk_sem_schedule_status_updated_by` (`updated_by_user_id`),
  KEY `fk_sem_schedule_semester_spe_id` (`semester_spe_id`),
  KEY `idx_semester_number` (`semester_number`) USING BTREE,
  CONSTRAINT `fk_sem_schedule_semester_spe_id` FOREIGN KEY (`semester_spe_id`)
REFERENCES `tbl_specialization` (`spe_id`),
  CONSTRAINT `fk_sem_schedule_status_sem_ver` FOREIGN KEY (`sem_ver_id`) REFERENCES
`tbl_semester_version` (`sem_ver_id`),
  CONSTRAINT `fk_sem_schedule_status_updated_by` FOREIGN KEY (`updated_by_user_id`)
REFERENCES `tbl_admin` (`user_id`)
) ENGINE=InnoDB AUTO_INCREMENT=41 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_general_ci;
```

```
-- Table structure for table `tbl_semester_version`
CREATE TABLE IF NOT EXISTS `tbl_semester_version` (
  `sem_ver_id` int(11) NOT NULL,
  `sem_id` int(11) NOT NULL,
  `semester_name` varchar(256) NOT NULL,
  `semester_unique` varchar(256) NOT NULL,
  `department_id` int(11) NOT NULL,
  `sem_start_date` date NOT NULL,
  `sem_end_date` date NOT NULL,
  `sem_start_day` enum('1','2','3','4','5','6','7') NOT NULL,
  `sem_end_day` enum('1','2','3','4','5','6','7') NOT NULL,
  `sem_start_time` time NOT NULL,
  `sem_end_time` time NOT NULL,
  `lunch_start_time` time DEFAULT NULL,
  `lunch_end_time` time DEFAULT NULL,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`sem_ver_id`),
  UNIQUE KEY `unique_semester_version__unique_department_year_semester`
(`semester_unique`,`department_id`,`valid_from_year`,`valid_from_semester`) USING
BTREE,
  UNIQUE KEY `unique_semester_version__sem_id_department_year_semester`
(`sem_id`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
  KEY `idx_semester_name` (`semester_name`),
  CONSTRAINT `fk_sem_id` FOREIGN KEY (`sem_id`) REFERENCES `tbl_semester`
(`sem_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_semester_version_deparment_id` FOREIGN KEY (`department_id`)
REFERENCES `tbl_department` (`dep_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_specialization`
CREATE TABLE IF NOT EXISTS `tbl_specialization` (
  `spe_id` int(11) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`spe_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_specialization_version`
CREATE TABLE IF NOT EXISTS `tbl_specialization_version` (
  `spe_ver_id` int(11) NOT NULL,
  `spe_id` int(11) NOT NULL,
  `spe_title` varchar(256) NOT NULL,
  `spe_unique` varchar(256) NOT NULL,
  `department_id` int(11) NOT NULL,
  `semester_from` enum('1','2','3','4','5','6','7','8','9','10','11','12') NOT
NULL,
  `is_active` int(1) NOT NULL DEFAULT 1,
  `valid_from_year` int(4) NOT NULL DEFAULT 2000,
  `valid_from_semester` enum('SPRING','FALL') NOT NULL,
  `version_datetime` timestamp NULL DEFAULT current_timestamp(),
  `last_update` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`spe_ver_id`),
  UNIQUE KEY `unique_specialization_version__unique_department_year_semester`
(`spe_unique`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  UNIQUE KEY `unique_specialization_version__spe_id_department_year_semester`
(`spe_id`,`department_id`,`valid_from_year`,`valid_from_semester`) USING BTREE,
  KEY `idx_department_id` (`department_id`),
  KEY `idx_valid_from_year` (`valid_from_year`),
  KEY `idx_valid_from_semester` (`valid_from_semester`),
```

```
  KEY `idx_spe_title` (`spe_title`) USING BTREE,
  CONSTRAINT `fk_spe_id` FOREIGN KEY (`spe_id`) REFERENCES `tbl_specialization`
(`spe_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fk_specialization_version_deparment_id` FOREIGN KEY (`department_id`)
REFERENCES `tbl_department` (`dep_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_student_course`
CREATE TABLE IF NOT EXISTS `tbl_student_course` (
  `autonum` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `user_course_id` int(11) NOT NULL,
  `user_semester` int(11) NOT NULL,
  PRIMARY KEY (`autonum`),
  UNIQUE KEY `unique_user_course` (`user_id`,`user_course_id`,`user_semester`)
USING BTREE,
  KEY `idx_user_id` (`user_id`) USING BTREE,
  KEY `idx_user_course_id` (`user_course_id`) USING BTREE,
  KEY `idx_user_semester` (`user_semester`) USING BTREE,
  CONSTRAINT `fk_student_course_id` FOREIGN KEY (`user_course_id`) REFERENCES
`tbl_course` (`cou_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_student_course_user_id` FOREIGN KEY (`user_id`) REFERENCES
`tbl_student` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=18 DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

-- Table structure for table `tbl_user_data_seen`
CREATE TABLE IF NOT EXISTS `tbl_user_data_seen` (
  `user_id` int(11) NOT NULL,
  `last_seen_change_id` int(11) NOT NULL DEFAULT 0,
  `last_seen_datetime` datetime DEFAULT NULL,
  `last_seen_schedule_status_update` datetime DEFAULT NULL,
  PRIMARY KEY (`user_id`),
  CONSTRAINT `fk_user_data_seen__user_id` FOREIGN KEY (`user_id`) REFERENCES
`tbl_login_user` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Structure for view `view_admins`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_admins` AS
SELECT
  `a`.`user_id` AS `user_id`,
  `a`.`first_name` AS `first_name`,
  `a`.`last_name` AS `last_name`,
  `a`.`father_name` AS `father_name`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `a`.`uni_id` AS `uni_id`,
  `a`.`mobile_phone` AS `mobile_phone`,
  `a`.`landline_phone` AS `landline_phone`,
  `a`.`department_id` AS `department_id`,
  `a`.`faculty_id` AS `faculty_id`,
  `a`.`created_datetime` AS `created_datetime`,
  `a`.`last_update` AS `last_update`,
  `u`.`user_lastlogin` AS `user_lastlogin`,
  `u`.`user_role` AS `user_role`,
  `u`.`user_isactive` AS `user_isactive`,
  `l`.`activation_email` AS `activation_email`
FROM `tbl_admin` AS `a`
INNER JOIN `tbl_login_user` AS `u`
  ON `a`.`user_id` = `u`.`user_id`
LEFT JOIN `tbl_account_log` AS `l`
  ON `a`.`user_id` = `l`.`user_id`;

-- Structure for view `view_all_users_activation`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_all_users_activation`
AS
SELECT
```

```
  `u`.`user_id` AS `user_id`,
  COALESCE(`a1`.`first_name`, `p1`.`first_name`, `s1`.`first_name`) AS
`first_name`,
  COALESCE(`a1`.`last_name`, `p1`.`last_name`, `s1`.`last_name`) AS `last_name`,
  COALESCE(`a1`.`father_name`, `p1`.`father_name`, `s1`.`father_name`) AS
`father_name`,
  `u`.`user_email` AS `user_email`,
  COALESCE(`a1`.`created_datetime`, `p1`.`created_datetime`,
`s1`.`created_datetime`) AS `created_datetime`,
  `u`.`user_role` AS `user_role`,
  COALESCE(`a1`.`department_id`, `p1`.`department_id`, `s1`.`department_id`) AS
`department_id`,
  `u`.`user_isactive` AS `user_isactive`,
  `l`.`activation_email` AS `activation_email`
FROM `tbl_login_user` AS `u`
LEFT JOIN `tbl_admin` AS `a1`
  ON `a1`.`user_id` = `u`.`user_id`
LEFT JOIN `tbl_professor` AS `p1`
  ON `p1`.`user_id` = `u`.`user_id`
LEFT JOIN `tbl_student` AS `s1`
  ON `s1`.`user_id` = `u`.`user_id`
LEFT JOIN `tbl_account_log` AS `l`
  ON `l`.`user_id` = `u`.`user_id`
WHERE `u`.`user_isactive` = 2;

-- Structure for view `view_all_users_mobile`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_all_users_mobile` AS
SELECT
  `a`.`user_id` AS `user_id`,
  `a`.`mobile_phone` AS `mobile_phone`,
  CAST('admin' AS CHAR(20)) AS `user_type`
FROM `tbl_admin` AS `a`
UNION ALL
SELECT
  `s`.`user_id` AS `user_id`,
  `s`.`mobile_phone` AS `mobile_phone`,
  CAST('student' AS CHAR(20)) AS `user_type`
FROM `tbl_student` AS `s`
UNION ALL
SELECT
  `p`.`user_id` AS `user_id`,
  `p`.`mobile_phone` AS `mobile_phone`,
  CAST('professor' AS CHAR(20)) AS `user_type`
FROM `tbl_professor` AS `p`;

-- Structure for view `view_course_professor`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_course_professor` AS
SELECT
  `cp`.`course_ver_id` AS `course_ver_id`,
  `cp`.`course_professor` AS `course_professor`,
  `p`.`user_id` AS `user_id`,
  `p`.`first_name` AS `first_name`,
  `p`.`last_name` AS `last_name`,
  `p`.`father_name` AS `father_name`,
  `p`.`uni_id` AS `uni_id`,
  `p`.`department_id` AS `department_id`,
  `p`.`rank_title` AS `rank_title`
FROM `tbl_course_professor` AS `cp`
LEFT JOIN `tbl_professor` AS `p`
  ON `p`.`user_id` = `cp`.`course_professor`;

-- Structure for view `view_department_active`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_department_active` AS
SELECT
  `x`.`dep_id` AS `dep_id`,
  `x`.`department_name` AS `department_name`,
```

```
  `x`.`faculty_id` AS `faculty_id`,
  `x`.`is_active` AS `is_active`
FROM (
  SELECT
    `v`.`dep_ver_id` AS `dep_ver_id`,
    `v`.`dep_id` AS `dep_id`,
    `v`.`department_name` AS `department_name`,
    `v`.`faculty_id` AS `faculty_id`,
    `v`.`is_active` AS `is_active`,
    `v`.`valid_from_year` AS `valid_from_year`,
    `v`.`valid_from_semester` AS `valid_from_semester`,
    ROW_NUMBER() OVER (
      PARTITION BY `v`.`dep_id`
      ORDER BY
        `v`.`valid_from_year` DESC,
        FIELD(`v`.`valid_from_semester`, 'SPRING', 'FALL')
    ) AS `rn`
  FROM `tbl_department_version` AS `v`
) AS `x`
WHERE `x`.`rn` = 1;

-- Structure for view `view_faculty_active`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_faculty_active` AS
SELECT
  `x`.`fac_id` AS `fac_id`,
  `x`.`faculty_name` AS `faculty_name`,
  `x`.`is_active` AS `is_active`
FROM (
  SELECT
    `v`.`fac_ver_id` AS `fac_ver_id`,
    `v`.`fac_id` AS `fac_id`,
    `v`.`faculty_name` AS `faculty_name`,
    `v`.`is_active` AS `is_active`,
    `v`.`valid_from_year` AS `valid_from_year`,
    `v`.`valid_from_semester` AS `valid_from_semester`,
    ROW_NUMBER() OVER (
      PARTITION BY `v`.`fac_id`
      ORDER BY
        `v`.`valid_from_year` DESC,
        FIELD(`v`.`valid_from_semester`, 'SPRING', 'FALL')
    ) AS `rn`
  FROM `tbl_faculty_version` AS `v`
) AS `x`
WHERE `x`.`rn` = 1;

-- Structure for view `view_professors`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_professors` AS
SELECT
  `p`.`user_id` AS `user_id`,
  `p`.`first_name` AS `first_name`,
  `p`.`last_name` AS `last_name`,
  `p`.`father_name` AS `father_name`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `u`.`password_set` AS `password_set`,
  `p`.`uni_id` AS `uni_id`,
  `p`.`mobile_phone` AS `mobile_phone`,
  `p`.`landline_phone` AS `landline_phone`,
  `p`.`department_id` AS `department_id`,
  `dv`.`department_name` AS `department_name`,
  `d`.`dep_uni_id` AS `dep_uni_id`,
  `p`.`rank_title` AS `rank_title`,
  `p`.`created_datetime` AS `created_datetime`,
  `p`.`last_update` AS `last_update`,
  `u`.`user_lastlogin` AS `user_lastlogin`,
  `u`.`user_role` AS `user_role`,
```

```
  `u`.`user_isactive` AS `user_isactive`,
  `l`.`activation_email` AS `activation_email`
FROM `tbl_professor` AS `p`
INNER JOIN `tbl_login_user` AS `u`
  ON `p`.`user_id` = `u`.`user_id`
LEFT JOIN `tbl_account_log` AS `l`
  ON `p`.`user_id` = `l`.`user_id`
LEFT JOIN `tbl_department` AS `d`
  ON `d`.`dep_id` = `p`.`department_id`
LEFT JOIN (
  SELECT
    `dv_ranked`.`dep_id` AS `dep_id`,
    `dv_ranked`.`department_name` AS `department_name`
  FROM (
    SELECT
      `dv`.`dep_id` AS `dep_id`,
      `dv`.`department_name` AS `department_name`,
      ROW_NUMBER() OVER (
        PARTITION BY `dv`.`dep_id`
        ORDER BY
          `dv`.`valid_from_year` DESC,
          `dv`.`valid_from_semester` DESC
      ) AS `rn`
    FROM `tbl_department_version` AS `dv`
  ) AS `dv_ranked`
  WHERE `dv_ranked`.`rn` = 1
) AS `dv`
  ON `d`.`dep_id` = `dv`.`dep_id`;

-- Structure for view `view_students`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_students` AS
SELECT
  `s`.`user_id` AS `user_id`,
  `s`.`first_name` AS `first_name`,
  `s`.`last_name` AS `last_name`,
  `s`.`father_name` AS `father_name`,
  `s`.`birth_date` AS `birth_date`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `u`.`password_set` AS `password_set`,
  `s`.`uni_id` AS `uni_id`,
  `s`.`mobile_phone` AS `mobile_phone`,
  `s`.`landline_phone` AS `landline_phone`,
  `s`.`post_address` AS `post_address`,
  `s`.`post_city` AS `post_city`,
  `s`.`post_zip` AS `post_zip`,
  `s`.`enrollment` AS `enrollment`,
  `s`.`curr_semester` AS `curr_semester`,
  `s`.`user_special` AS `user_special`,
  `s`.`registration_id` AS `registration_id`,
  `s`.`registration_date` AS `registration_date`,
  `s`.`department_id` AS `department_id`,
  `dv_match`.`department_name` AS `department_name`,
  `s`.`created_datetime` AS `created_datetime`,
  `s`.`last_update` AS `last_update`,
  `u`.`user_lastlogin` AS `user_lastlogin`,
  `u`.`user_role` AS `user_role`,
  `u`.`user_isactive` AS `user_isactive`,
  COALESCE(
    CAST(`sc`.`student_course_ids` AS CHAR CHARACTER SET utf8mb4) COLLATE
utf8mb4_general_ci,
    ''
  ) AS `user_course_id`
FROM `tbl_student` AS `s`
INNER JOIN `tbl_login_user` AS `u`
  ON `s`.`user_id` = `u`.`user_id`
```

```
LEFT JOIN (
  SELECT
    `x`.`user_id` AS `user_id`,
    `x`.`department_name` AS `department_name`
  FROM (
    SELECT
      `s2`.`user_id` AS `user_id`,
      `dv`.`department_name` AS `department_name`,
      ROW_NUMBER() OVER (
        PARTITION BY `s2`.`user_id`
        ORDER BY
          `dv`.`valid_from_year` DESC,
          CASE
            WHEN `dv`.`valid_from_year` = `s2`.`enrollment`
              AND `dv`.`valid_from_semester` = 'FALL' THEN 1
            WHEN `dv`.`valid_from_year` = `s2`.`enrollment`
              AND `dv`.`valid_from_semester` = 'SPRING' THEN 2
            WHEN `dv`.`valid_from_semester` = 'FALL' THEN 3
            WHEN `dv`.`valid_from_semester` = 'SPRING' THEN 4
            ELSE 5
          END
      ) AS `rn`
    FROM `tbl_student` AS `s2`
    INNER JOIN `tbl_department_version` AS `dv`
      ON `dv`.`dep_id` = `s2`.`department_id`
      AND `dv`.`valid_from_year` <= `s2`.`enrollment`
  ) AS `x`
  WHERE `x`.`rn` = 1
) AS `dv_match`
  ON `dv_match`.`user_id` = `s`.`user_id`
LEFT JOIN (
  SELECT
    `s3`.`user_id` AS `user_id`,
    GROUP_CONCAT(
      `sc3`.`user_course_id`
      ORDER BY `sc3`.`user_course_id` ASC
      SEPARATOR ','
    ) AS `student_course_ids`
  FROM `tbl_student` AS `s3`
  LEFT JOIN `tbl_student_course` AS `sc3`
    ON `sc3`.`user_id` = `s3`.`user_id`
    AND `sc3`.`user_semester` = `s3`.`curr_semester`
  WHERE `s3`.`curr_semester` IS NOT NULL
  GROUP BY `s3`.`user_id`
) AS `sc`
  ON `sc`.`user_id` = `s`.`user_id`;

-- Structure for view `view_users`
CREATE ALGORITHM = UNDEFINED SQL SECURITY DEFINER VIEW `view_users` AS
SELECT
  `s`.`user_id` AS `user_id`,
  `s`.`first_name` AS `first_name`,
  `s`.`last_name` AS `last_name`,
  `s`.`father_name` AS `father_name`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `s`.`uni_id` AS `uni_id`,
  `s`.`department_id` AS `department_id`,
  `u`.`user_role` AS `user_role`,
  `u`.`user_isactive` AS `user_isactive`,
  CONCAT_WS(' ', `s`.`last_name`, `s`.`first_name`) AS `full_name`
FROM `tbl_student` AS `s`
INNER JOIN `tbl_login_user` AS `u`
  ON `u`.`user_id` = `s`.`user_id`
UNION ALL
SELECT
```

```
  `p`.`user_id` AS `user_id`,
  `p`.`first_name` AS `first_name`,
  `p`.`last_name` AS `last_name`,
  `p`.`father_name` AS `father_name`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `p`.`uni_id` AS `uni_id`,
  `p`.`department_id` AS `department_id`,
  `u`.`user_role` AS `user_role`,
  `u`.`user_isactive` AS `user_isactive`,
  CONCAT_WS(' ', `p`.`last_name`, `p`.`first_name`) AS `full_name`
FROM `tbl_professor` AS `p`
INNER JOIN `tbl_login_user` AS `u`
  ON `u`.`user_id` = `p`.`user_id`
UNION ALL
SELECT
  `a`.`user_id` AS `user_id`,
  `a`.`first_name` AS `first_name`,
  `a`.`last_name` AS `last_name`,
  `a`.`father_name` AS `father_name`,
  `u`.`user_username` AS `user_username`,
  `u`.`user_email` AS `user_email`,
  `a`.`uni_id` AS `uni_id`,
  `a`.`department_id` AS `department_id`,
  `u`.`user_role` AS `user_role`,
  `u`.`user_isactive` AS `user_isactive`,
  CONCAT_WS(' ', `a`.`last_name`, `a`.`first_name`) AS `full_name`
FROM `tbl_admin` AS `a`
INNER JOIN `tbl_login_user` AS `u`
  ON `u`.`user_id` = `a`.`user_id`;
```

Πίνακας Θ-1: Απόσπασμα SQL σχήματος Βάσης Δεδομένων

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.