ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΣΧΟΛΗ ΚΟΙΝΩΝΙΚΩΝ ΚΑΙ ΑΝΘΡΩΠΙΣΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΨΗΦΙΑΚΩΝ ΜΕΣΩΝ
Π.Μ.Σ. «ΑΝΑΠΤΥΞΗ ΨΗΦΙΑΚΩΝ ΠΑΙΧΝΙΔΙΩΝ ΚΑΙ ΠΟΛΥΜΕΣΙΚΩΝ ΕΦΑΡΜΟΓΩΝ»

ΤΜΗΜΑ ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΨΗΦΙΑΚΩΝ ΜΕΣΩΝ

ΣΧΟΛΗ ΚΟΙΝΩΝΙΚΩΝ ΚΑΙ ΑΝΘΡΩΠΙΣΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Εξερευνώντας τη Γνώση με Python: Ένα Εκπαιδευτικό Serious Game με NPCs και Τεχνητή Νοημοσύνη»

Ανθή Δρουγκάνη

Επιβλέπων καθηγητής: Μηνάς Δασυγένης

Μάρτιος,2025

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΣΧΟΛΗ ΚΟΙΝΩΝΙΚΩΝ ΚΑΙ ΑΝΘΡΩΠΙΣΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΨΗΦΙΑΚΩΝ ΜΕΣΩΝ
Π.Μ.Σ. «ΑΝΑΠΤΥΞΗ ΨΗΦΙΑΚΩΝ ΠΑΙΧΝΙΔΙΩΝ ΚΑΙ ΠΟΛΥΜΕΣΙΚΩΝ ΕΦΑΡΜΟΓΩΝ»

ΤΜΗΜΑ ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΨΗΦΙΑΚΩΝ ΜΕΣΩΝ

ΣΧΟΛΗ ΚΟΙΝΩΝΙΚΩΝ ΚΑΙ ΑΝΘΡΩΠΙΣΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Εξερευνώντας τη Γνώση με Python: Ένα Εκπαιδευτικό Serious Game με NPCs και Τεχνητή Νοημοσύνη»

Ανθή Δρουγκάνη

Επιβλέπων καθηγητής: Μηνάς Δασυγένης

Μέλη της Τριμελούς Εξεταστικής Επιτροπής:

1)Πλόσκας Νικόλαος

2)Πρωτοψάλτης Αντώνιος

Μάρτιος, 2025

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Μεταπτυχιακή Διπλωματική Εργασία με τίτλο: «Εξερευνώντας τη Γνώση με Python: Ένα Εκπαιδευτικό Serious Game με NPCs και Τεχνητή Νοημοσύνη» καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Πρόγραμμα Μεταπτυχιακών Σπουδών «Ανάπτυξη Ψηφιακών Παιχνιδιών και Πολυμεσικών Εφαρμογών» του Τμήματος Επικοινωνίας & Ψηφιακών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του Μηνά Δασυγένη αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και μόνο.

Υπογραφή Φοιτητή/τριας

Ανθή Δρουγκάνη

Περίληψη

Η παρούσα διπλωματική εργασία παρουσιάζει το “A Quest of Knowledge” (Εξερευνώντας τη Γνώση), ένα εκπαιδευτικό 2D παιχνίδι σοβαρού σκοπού (serious game), το οποίο συνδυάζει τη δυναμική μάθηση με τη διαδραστική εμπειρία παιχνιδιού. Σχεδιασμένο για παίκτες όλων των ηλικιών, το παιχνίδι συνδυάζει εξερεύνηση και στρατηγική σε έναν κόσμο ανοιχτού τύπου (open world) με αισθητική γραφικών σε μορφή πλακιδίων (pixel art), εμπνευσμένο από κλασικούς τίτλους όπως το The Legend of Zelda (1986). Οι παίκτες εξερευνούν μια ποικιλία θεματικών ενοτήτων, αλληλεπιδρούν με μη παικτικούς χαρακτήρες (Non Playable Characters, NPCs), οι οποίοι είναι βασισμένοι σε ιστορικές προσωπικότητες και απαντούν σε δυναμικά δημιουργημένες ερωτήσεις από κλάδους όπως τα μαθηματικά, η γεωγραφία, η βιολογία και άλλους. Με την προσαρμογή της δυσκολίας στον αριθμό των ερωτήσεων και στην ηλικία του παίκτη, το παιχνίδι ενισχύει την προσβασιμότητα, την αναπαιξιμότητα και την ενεργή συμμετοχή- “δέσμευση”. Τα στάδια του παιχνιδιού περιλαμβάνουν την εξερεύνηση, όπου οι παίκτες ανακαλύπτουν NPCs και συλλέγουν θεματικά αντικείμενα, τη στρατηγική, όπου ο παίκτης αντιμετωπίζει εχθρούς και τις διαδραστικές συνομιλίες, με τον παίκτη να συλλέγει γνώσεις από τους NPCs. Χρησιμοποιώντας προηγμένες τεχνολογίες τεχνητής νοημοσύνης, όπως μεγάλα γλωσσικά μοντέλα (LLMs) και δυναμική δημιουργία περιεχομένου σε πραγματικό χρόνο, το παιχνίδι δημιουργεί μοναδικούς διαλόγους και προσαρμοσμένο εκπαιδευτικό περιεχόμενο, εμπλουτίζοντας την εμπειρία του παίκτη. Δομημένο σε Pygame, είναι κατάλληλο για χρήστες που φέρουν τα πιο δημοφιλή λειτουργικά συστήματα (Windows, Linux, Mac). Συνδυάζοντας την εκπαίδευση, την ψυχαγωγία και διαλόγους με τεχνητή νοημοσύνη, το παιχνίδι προσφέρει ένα καινοτόμο μαθησιακό περιβάλλον, το οποίο μετατρέπει την αναζήτηση της γνώσης σε μια διαδραστική περιπέτεια.

Λέξεις-κλειδιά

Εκπαιδευτικό παιχνίδι σοβαρού σκοπού, τεχνητή νοημοσύνη, Python, Pygame, μεγάλα γλωσσικά μοντέλα (LLMs)

Abstract

This diploma thesis presents A Quest of Knowledge, a 2D serious educational game that combines dynamic learning with an interactive gaming experience. Designed for players of all ages, the game blends exploration and strategy within an open-world environment featuring pixel art graphics, inspired by classic titles such as The Legend of Zelda (1986). Players explore a variety of thematic areas, interact with non-playable characters (NPCs) based on historical figures, and answer dynamically generated questions from subjects such as mathematics, geography, biology, and more. By adjusting difficulty levels based on the number of questions and the player's age, the game enhances accessibility, replayability, and engagement. The gameplay consists of exploration, where players discover NPCs and collect thematic items; strategy, where they face enemies; and interactive dialogues, where they gain knowledge from NPCs. Leveraging advanced artificial intelligence technologies, such as large language models (LLMs) and real-time dynamic content generation, the game creates unique dialogues and personalized educational content, enriching the player's experience. Developed using Pygame, the game is compatible with Windows, Linux, and Mac computers. By integrating education, entertainment, and AI-driven dialogues, A Quest of Knowledge offers an innovative learning environment that transforms the pursuit of knowledge into an interactive adventure.

Keywords

serious educational games, Artificial Intelligence, Python, Pygame, Large Language Models (LLMs)

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, Δρ. Μηνά Δασυγένη, για την καθοδήγηση, την υποστήριξη και τις πολύτιμες συμβουλές του καθ' όλη τη διάρκεια εκπόνησης της εργασίας. Επιπλέον, ευχαριστώ την οικογένειά μου για τη συνεχή στήριξη, την κατανόηση και την ενθάρρυνση που μου προσέφερε κατά τη διάρκεια των σπουδών μου.

Τέλος, θα ήθελα να ευχαριστήσω όλους όσους, με οποιονδήποτε τρόπο, συνέβαλαν στην ολοκλήρωση αυτής της εργασίας.

Περιεχόμενα

Περίληψη i

Abstract ii

Ευχαριστίες iii

Λίστα Εικόνων vi

Πίνακας Ακρωνυμίων vii

Λίστα Γραφημάτων viii

Εισαγωγή 1

1. Η Τεχνητή Νοημοσύνη στα Ψηφιακά Παιχνίδια Σοβαρού Σκοπού 3

1.1. Τι είναι τα ψηφιακά παιχνίδια Video Games 3

1.2. Τι είναι τα ψηφιακά παιχνίδια σοβαρού σκοπού - Serious Games και ποια η σχέση τους με την εκπαίδευση 3

1.3. Τεχνητή Νοημοσύνη - Ιστορική Αναδρομή 4

1.4. Μεγάλα Γλωσσικά Μοντέλα -Large Language Models (LLMs) 5

1.5. Η ενσωμάτωσή των LLMs στα Ψηφιακά παιχνίδια 5

2. Τεχνολογίες που χρησιμοποιήθηκαν 7

2.1. Python 7

2.2. Llama3 8

2.3. Bing AI Image Creator 9

2.4. Photopea 10

2.5. Eleven Labs 10

2.6. Github 11

2.7. Visual Studio Code 12

2.8. Pixabay 13

3. Game Design Document 14

3.1. Επισκόπηση του παιχνιδιού 14

3.2. Γενική Παρουσίαση 14

3.2.1. Στόχοι Εμπειρίας Παίκτη/Χρήστη (Player Experience Goals): 14

3.2.2. Επιρροές 15

3.3. Καινοτομία και Μοναδικότητα 15

3.3.1. Χαρακτηριστικά και Πρωτοτυπία 15

3.3.2. Συναφή έργα 16
3.5. Περιγραφή του παιχνιδιού 17
3.6. Χειρισμοί 19
3.7. Σενάριο 20
3.8. Concept Art 21
4. Υλοποίηση 29
4.1. Πρώτη Φάση Ανάπτυξης – Έκδοση 1 της εφαρμογής 29
4.2. Δεύτερη Φάση Ανάπτυξης – Έκδοση 2 της εφαρμογής 33
4.3. Τρίτη Φάση Ανάπτυξης – Έκδοση 3 της εφαρμογής 39
4.4. Τέταρτη Φάση Ανάπτυξης – Έκδοση 4 της εφαρμογής 57
5. Ερευνητικός σχεδιασμός και Μεθοδολογία εργασίας 74
5.1. Μεθοδολογία Έρευνας 74
5.2. Αποτελέσματα Έρευνας 76
6. Προκλήσεις, Αντιμετώπιση, Συμπεράσματα και Μελλοντικές Προεκτάσεις 88
6.1. Προκλήσεις υλοποίησης 88
6.2. Αντιμετώπιση προκλήσεων 89
6.3. Συμπεράσματα 90
6.4. Μελλοντικές Επεκτάσεις 91
Βιβλιογραφία 93
Παράρτημα Ι : Οδηγίες εγκατάστασης 94
Παράρτημα ΙΙ : Έντυπο Συμμόρφωσης 95
Παράρτημα ΙΙΙ : Ερωτηματολόγιο 97
Παράρτημα IV : Κώδικας 100

Λίστα Εικόνων

Εικόνα 1.Python 7
Εικόνα 2.LLama3 8
Εικόνα 3.Bing AI Image Creator 9
Εικόνα 4.Photopea 10
Εικόνα 5.Eleven Labs 10
Εικόνα 6.Github 11
Εικόνα 7.Visual Studio Code 12
Εικόνα 8.Pixabay 13
Εικόνα 9.Αρχικό Μενού 21
Εικόνα 10.Intro 22
Εικόνα 11.Χάρτης του παιχνιδιού 23
Εικόνα 12.Tiles- Πλακίδια για κάθε θεματική του χάρτη 23
Εικόνα 13.NPCs (1η επιλογή) 24
Εικόνα 14.NPCs (2η επιλογή) 24
Εικόνα 15.Αντικείμενα για συλλογή 25
Εικόνα 16.Εχθρός 25
Εικόνα 17.Επιλογή ηρώων 26
Εικόνα 18.Όπλο 26
Εικόνα 19.Συνθήκη ήττας 27
Εικόνα 20.Συνθήκης νίκης 28
Εικόνα 21. Σύνδεση LLM και Chat μέσω API 32
Εικόνα 22.Δημιουργία fun fact 40
Εικόνα 23.Διεξαγωγή διαλόγου 41
Εικόνα 24.Εισαγωγή ονόματος 53
Εικόνα 25.Ασφάλεια εισαγωγής ηλικίας 54
Εικόνα 26.Όριο αριθμού ερωτήσεων 55
Εικόνα 27.Επιλογή χαρακτήρα 56
Εικόνα 28.Καταδίωξη από εχθρό 63
Εικόνα 29.Διαθέσιμα αντικείμενα στο Inventory 67
Εικόνα 30.Βοηθητικά μηνύματα στο περιβάλλον του παιχνιδιού 68

Πίνακας Ακρωνυμίων

AI	Artificial Intelligence
API	Application Programming Interface
Co-Op	Cooperative
GPT	Generative Pre-Trained Transformer
LLMs	Large Language Models
NPCs	Non-Playable Characters
URL	Uniform Resource Locator

Λίστα Γραφημάτων

Γράφημα 1.Απαντήσεις 1ης ερώτησης 77
Γράφημα 2.Απαντήσεις 2ης ερώτησης 78
Γράφημα 3.Απαντήσεις 3ης ερώτησης 78
Γράφημα 4.Απαντήσεις 4ης ερώτησης 78
Γράφημα 5.Απαντήσεις 5ης ερώτησης 79
Γράφημα 6.Απαντήσεις 6ης ερώτησης 79
Γράφημα 7.Απαντήσεις 7ης ερώτησης 80
Γράφημα 8.Απαντήσεις 8ης ερώτησης 80
Γράφημα 9.Απαντήσεις 9ης ερώτησης 81
Γράφημα 10.Απαντήσεις 10ης ερώτησης 81
Γράφημα 11.Απαντήσεις 11ης ερώτησης 82
Γράφημα 12.Απαντήσεις 12ης ερώτησης 82
Γράφημα 13.Απαντήσεις 13ης ερώτησης 83
Γράφημα 14.Απαντήσεις 14ης ερώτησης 83
Γράφημα 15.Απαντήσεις 15ης ερώτησης 84
Γράφημα 16.Απαντήσεις 16ης ερώτησης 84
Γράφημα 17.Απαντήσεις 17ης ερώτησης 84
Γράφημα 18.Απαντήσεις 24ης ερώτησης 87

Εισαγωγή

Τα ψηφιακά παιχνίδια αποτελούν πλέον ένα σημαντικό μέσο και συμβάλλουν τόσο στην εκπαίδευση όσο και την προσωπική ανάπτυξη. Ο συνδυασμός της περιπέτειας και της στρατηγικής με την απόκτηση γνώσης προσφέρει νέες δυνατότητες στην κατηγορία των παιχνιδιών σοβαρού σκοπού (serious games), τα οποία αποσκοπούν όχι μόνο στη διασκέδαση, αλλά και στη μετάδοση εκπαιδευτικού περιεχομένου.

Η παρούσα εργασία, με τίτλο «Εξερευνώντας τη Γνώση με Python: Ένα Εκπαιδευτικό Serious Game με NPCs (Non Playable Characters) και Τεχνητή Νοημοσύνη», παρουσιάζει ένα πρωτοποριακό 2D παιχνίδι, το "A Quest of Knowledge", που αξιοποιεί σύγχρονες τεχνολογίες, όπως τη γλώσσα προγραμματισμού Python, τη χρήση Μεγάλων Γλωσσικών Μοντέλων (Large Language Models, LLMs) και API, για τη δημιουργία ενός δυναμικού περιβάλλοντος μάθησης. Το παιχνίδι «A Quest of Knowledge» αναπτύχθηκε για να συνδέσει τη μάθηση με την περιπέτεια και την εξερεύνηση, δίνοντας έμφαση στην ολοκλήρωση της "αποστολής" του παίκτη, δεσμεύοντάς τον με το παιχνίδι, μέσω της αλληλεπίδρασης με NPCs που εκπροσωπούν ιστορικές προσωπικότητες και εκπαιδευτικά αντικείμενα.

Μέσα από έναν ανοιχτό, δισδιάστατο κόσμο σε pixel art αισθητική (γραφικά σε μορφή πλακιδίων), ο παίκτης καλείται να απαντήσει σε ερωτήσεις, να συλλέξει αντικείμενα γνώσης και να επιβιώσει από προκλήσεις, όπως η συνάντηση με εχθρούς. Το παιχνίδι προσφέρει ευελιξία, καθώς επιτρέπει την επιλογή του αριθμού ερωτήσεων και τον ορισμό της ηλικίας του παίκτη, προσαρμόζοντας τη δυσκολία με βάση αυτές τις παραμέτρους.

Η ενσωμάτωση των LLMs, όπως το Llama3, δίνει τη δυνατότητα στους NPCs να παράγουν δυναμικούς διαλόγους, ερωτήσεις και βοηθητικά μηνύματα (hints) σε πραγματικό χρόνο, καθιστώντας την εμπειρία μοναδική σε κάθε νέο παιχνίδι. Επιπλέον, τα αντικείμενα που συλλέγει ο παίκτης και οι ερωτήσεις είναι θεματικά προσαρμοσμένα και τα σημεία που εμφανίζονται οι NPCs διαφοροποιούνται με κάθε νέα εκκίνηση του παιχνιδιού. Έτσι δεν προσφέρεται μόνο γνώση, αλλά και αναπαιξιμότητα (replayability), ενισχύοντας την ψυχαγωγική αξία του παιχνιδιού.

Στην εργασία αυτή εξετάζεται η ανάπτυξη και η υλοποίηση του παιχνιδιού, παρουσιάζοντας την ενσωμάτωση καινοτόμων τεχνολογιών και εργαλείων που κάνουν κυρίως χρήση τεχνητής νοημοσύνης, όπως το Bing AI Image Creator, το Photopea και το Eleven Labs, για τη δημιουργία γραφικών, ηχητικών εφέ και περιεχομένου. Επιπλέον, αναλύεται ο τρόπος

που οι τεχνολογίες αυτές συνδυάζονται, ώστε να δημιουργηθεί μια καινοτόμα εμπειρία που αποτελεί σύνδεσμο της εκπαίδευσης με την ψυχαγωγία.

Το παιχνίδι A Quest of Knowledge σχεδιάστηκε για να αποτελέσει εργαλείο μάθησης και εξερεύνησης, προσελκύοντας παίκτες όλων των ηλικιών και αναδεικνύοντας τις δυνατότητες που προσφέρει η σύγχρονη τεχνολογία στον εκπαιδευτικό τομέα.

1. Η Τεχνητή Νοημοσύνη στα Ψηφιακά Παιχνίδια Σοβαρού Σκοπού

Σε αυτό το κεφάλαιο εξετάζεται η σημαντική σύνδεση μεταξύ της τεχνητής νοημοσύνης και των ψηφιακών παιχνιδιών σοβαρού σκοπού (serious games). Αναλύεται η φύση των ψηφιακών παιχνιδιών και ο τρόπος με τον οποίο τα παιχνίδια σοβαρού σκοπού συμβάλλουν στην εκπαίδευση, χρησιμοποιώντας σύγχρονες τεχνολογίες, όπως τα Μεγάλα Γλωσσικά Μοντέλα (LLMs). Επιπλέον, παρακολουθείται η ιστορική πορεία της τεχνητής νοημοσύνης και η ενσωμάτωσή της στα ψηφιακά παιχνίδια, εστιάζοντας στον αντίκτυπο που έχει στην εμπειρία του χρήστη και στην εκπαίδευση.

1.1. Τι είναι τα ψηφιακά παιχνίδια Video Games

Ψηφιακό παιχνίδι ή αλλιώς βιντεοπαιχνίδι είναι ένα παιχνίδι το οποίο λειτουργεί και υλοποιείται σε κάποια ηλεκτρονική συσκευή. Ένα ψηφιακό παιχνίδι αποτελεί ένα συνδυασμό ψυχαγωγίας, τέχνης και τεχνολογίας, δίνοντας τη δυνατότητα στους χρήστες να αλληλεπιδρούν με έναν εικονικό κόσμο.

Τα ψηφιακά παιχνίδια προέρχονται από στρατιωτικές προσομοιώσεις και έρευνες που αφορούσαν τη μελέτη της υπολογιστικής ισχύος κατά τη διάρκεια των δεκαετιών 1950 και 1960. Έτσι δημιουργήθηκαν τα πρώτα απλά ψηφιακά παιχνίδια.

Με την πάροδο του χρόνου τα βιντεοπαιχνίδια εξελίχθηκαν αρκετά ώστε να αποτελούν έναν ισχυρό οικονομικό και πολιτιστικό τομέα. Καθοριστικό ρόλο σε αυτό έπαιξε η δημιουργία κονσόλων οικιακής χρήσης, η οποία επέτρεψε στο κοινό να έχει εύκολη πρόσβαση σε αυτά (Metuarau, 2017).

1.2. Τι είναι τα ψηφιακά παιχνίδια σοβαρού σκοπού - Serious Games και ποια η σχέση τους με την εκπαίδευση

Τα σοβαρά παιχνίδια αποτελούν έναν καινοτόμο συνδυασμό ψυχαγωγίας και εκπαίδευσης. Σε αντίθεση με τα παραδοσιακά βιντεοπαιχνίδια, δεν επικεντρώνονται μόνο στο κομμάτι της διασκέδασης, αλλά και στη μετάδοση γνώσης, την ανάπτυξη δεξιοτήτων και την εκπαίδευση των χρηστών σε διάφορους τομείς. Με αυτό το κράμα παιχνιδιού και εκπαιδευτικού περιεχομένου τα serious games ενισχύουν το κίνητρο μάθησης, προσφέροντας βαθύτερη κατανόηση του αντικειμένου που αφορούν, ενώ πολλές φορές μπορούν να προσφέρουν ένα ασφαλές περιβάλλον για εξάσκηση σε πραγματικές καταστάσεις. Η έννοια των παιχνιδιών σοβαρού σκοπού έχει εξελιχθεί σημαντικά. Η αφετηρία τους χρονολογείται γύρω στο 1946 (Noemí, 2014), όταν δημιουργήθηκαν τα πρώτα εργαλεία προσομοίωσης, όπως το έργο MIT Whirlwind, το οποίο έδωσε τη δυνατότητα σε στρατιωτικούς πιλότους να εκπαιδεύονται σε ένα ελεγχόμενο και ασφαλές περιβάλλον. Πλέον, χάρη σε προηγμένα προγραμματιστικά

περιβάλλοντα, όπως το Unity, Python, Unreal και το HTML5, τα serious games μπορούν να αναπτυχθούν σε ποικίλα περιβάλλοντα και να τρέξουν σε διαφορετικές συσκευές. Έτσι, λόγω αυτής της προσαρμοστικότητάς τους ήδη χρησιμοποιούνται σε σχολεία, επιχειρήσεις και άλλους τομείς, για την επίτευξη της βέλτιστης εκπαίδευσης.

Ένα σημαντικό στοιχείο των παιχνιδιών σοβαρού σκοπού είναι η ικανότητά τους στην εξατομίκευση της εμπειρίας μάθησης μέσω της προσαρμοζόμενης υποστήριξης και καθοδήγησης. Ο ρόλος των καθοδηγητών, είτε είναι άνθρωποι είτε εργαλεία τεχνητής νοημοσύνης, είναι ζωτικής σημασίας για την υποστήριξη της πορείας εκμάθησης του χρήστη, καθώς παρακολουθούν την πρόοδο και παρέχουν ανατροφοδότηση για τη διασφάλιση της επίτευξης των στόχων. Με αυτή την προσέγγιση, ενισχύεται η συμμετοχή, αλλά και αντιμετωπίζονται αποτελεσματικά οι ατομικές προκλήσεις των μαθητών. Επιπλέον, τα παιχνίδια σοβαρού σκοπού έχουν αποδειχθεί εξαιρετικά χρήσιμα στην επίσημη εκπαίδευση και την επαγγελματική κατάρτιση, κάνοντας πολύπλοκα θέματα πιο προσιτά και ελκυστικά, καθώς αποτελούν ένα καινοτόμο μέσο προετοιμασίας των μαθητών για μελλοντικές προκλήσεις, συνδυάζοντας εκπαιδευτικούς στόχους με τη διαδραστικότητα και την εμπλοκή που προσφέρουν τα παιχνίδια (Noemí, 2014).

1.3. Τεχνητή Νοημοσύνη - Ιστορική Αναδρομή

Ο όρος Τεχνητή Νοημοσύνη, ΤΝ (Artificial Intelligence, AI) χρησιμοποιήθηκε πρώτη φορά τη δεκαετία του 1950 και συγκεκριμένα το 1956 στο συνέδριο του Ντάρτμουθ. Δημιουργήθηκε αρχικά με στόχο την ανάπτυξη μηχανών που θα μπορούσαν να αναπαράγουν και να μιμηθούν βασικές γνωστικές λειτουργίες που χαρακτηρίζουν τον άνθρωπο, όπως η σκέψη, η μάθηση και η επίλυση προβλημάτων.

Οι πρώτες επιτυχείς προσπάθειες χρήσης της Τεχνητής Νοημοσύνης επικεντρώθηκαν στη συμβολική λογική και την επίλυση προβλημάτων, όπως η μετάφραση ξένων γλωσσών και η επίλυση θεωρημάτων. Τη δεκαετία του 1990 η τεχνητή νοημοσύνη κατόρθωσε να εξελιχθεί χάρη στη Μηχανική Μάθηση (Machine Learning), η οποία επέτρεψε στις μηχανές να μαθαίνουν και να εκπαιδεύονται από δεδομένα, καθώς και να κάνουν προβλέψεις. Έτσι πλέον αντί να προγραμματίζονται αποκλειστικά χειροκίνητα απέκτησαν την ικανότητα να βελτιώνονται μέσω της εμπειρίας και των γνώσεων που αποκτούσαν.

Σήμερα η τεχνητή νοημοσύνη έχει εξελιχθεί ακόμη περισσότερο χάρη στη Βαθιά Μάθηση (Deep Learning), έναν εξειδικευμένο κλάδο της Μηχανικής Μάθησης, βασισμένο σε νευρωνικά δίκτυα. Η Βαθιά Μάθηση επιτρέπει στις μηχανές να αναγνωρίζουν σύνθετες σχέσεις και μοτίβα σε τεράστιες ποσότητες δεδομένων επιτυγχάνοντας λειτουργίες όπως η

αναγνώριση εικόνας, η δημιουργία περιεχομένου και η επεξεργασία και παραγωγή φυσικής γλώσσας (Delipetrev, 2020).

1.4. Μεγάλα Γλωσσικά Μοντέλα -Large Language Models (LLMs)

Τα μεγάλα γλωσσικά μοντέλα (Large Language Models, LLMs) αποτελούν ένα σημαντικό αντικείμενο έρευνας για την επεξεργασία φυσικής γλώσσας (Natural Language Processing, NLP), καθώς διαθέτουν εξαιρετικές ικανότητες παραγωγής κειμένου. Είναι μεγάλου εύρους, προεκπαιδευμένα γλωσσικά μοντέλα, τα οποία βασίζονται σε νευρωνικά δίκτυα. Αρχικά ξεκίνησαν σαν ένα εξειδικευμένο επιστημονικό πεδίο, κυρίως θεωρητικού ενδιαφέροντος, με κύριο αντικείμενο την πρόβλεψη κειμένου με βάση τα σώματα κειμένου από τα οποία έχει εκπαιδευτεί το εκάστοτε μοντέλο.

Ωστόσο το 2019 το Generative Pre-Trained Transformer 2 (GPT 2) μοντέλο με την κυκλοφορία του έφερε μία επανάσταση στον κλάδο της τεχνητής νοημοσύνης, καθώς όχι μόνο είχε τη δυνατότητα να παράγει υψηλής ποιότητας κείμενο, αλλά η δημιουργία του κειμένου αυτού μπορούσε και να καθοδηγηθεί μέσω προτροπών (prompts). Έτσι το GPT 2 πλέον αποτελεί σημείο αναφοράς για νεότερα και μεγαλύτερα μοντέλα, τα οποία μπορούν να εκπαιδεύονται και να βελτιώνονται συνεχώς μέσω εντολών, ενισχυτικής μάθησης και ανατροφοδότησης που λαμβάνουν από τον άνθρωπο (Ενισχυτική Μάθηση μέσω Ανθρώπινης Ανατροφοδότησης - Reinforcement Learning From Human Feedback, RLHF) (Minaee, 2024), (Gallotta, 2024).

1.5. Η ενσωμάτωσή των LLMs στα Ψηφιακά παιχνίδια

Τα LLMs έχουν αρχίσει να ενσωματώνονται σε διάφορους τομείς, όπως αυτόν των ψηφιακών παιχνιδιών. Προγραμματιστές και ερευνητές προσπαθούν να αξιοποιήσουν τα πλέον εξελιγμένα LLMs, καθώς έχουν τη δυνατότητα να ενισχύσουν την αλληλεπίδραση με τους NPCs, μέσω της κατανόησης και παραγωγής φυσικής γλώσσας, παρέχοντας μια πιο ρεαλιστική εμπειρία παιχνιδιού. Οι NPCs, οι οποίοι δεν ελέγχονται από τους παίκτες, αποτελούν πλέον χαρακτήρες με μοναδικές προσωπικότητες και έχουν την ικανότητα να προσαρμόζονται στο παιχνίδι βάσει της αλληλεπίδρασης με τον παίκτη.

Τα LLMs μπορούν να χρησιμοποιηθούν με διάφορους τρόπους μέσα σε ένα παιχνίδι. Μπορούν να μιμηθούν ανθρώπινους χαρακτήρες θέτοντας στόχους και λειτουργώντας ως “παίκτες” του παιχνιδιού. Αυτή η ικανότητα τα καθιστά ιδανικά για turn-based παιχνίδια (παιχνίδια με εναλλαγή σειράς μεταξύ των παικτών) και επιτραπέζια, όπου οι κινήσεις είναι ορισμένες και γίνεται χρήση στρατηγικής. Είναι επίσης αποδοτικά σε παιχνίδια περιπέτειας, στα οποία όλη η έκβαση της ιστορίας υλοποιείται με τη χρήση κειμένου σε φυσική γλώσσα,

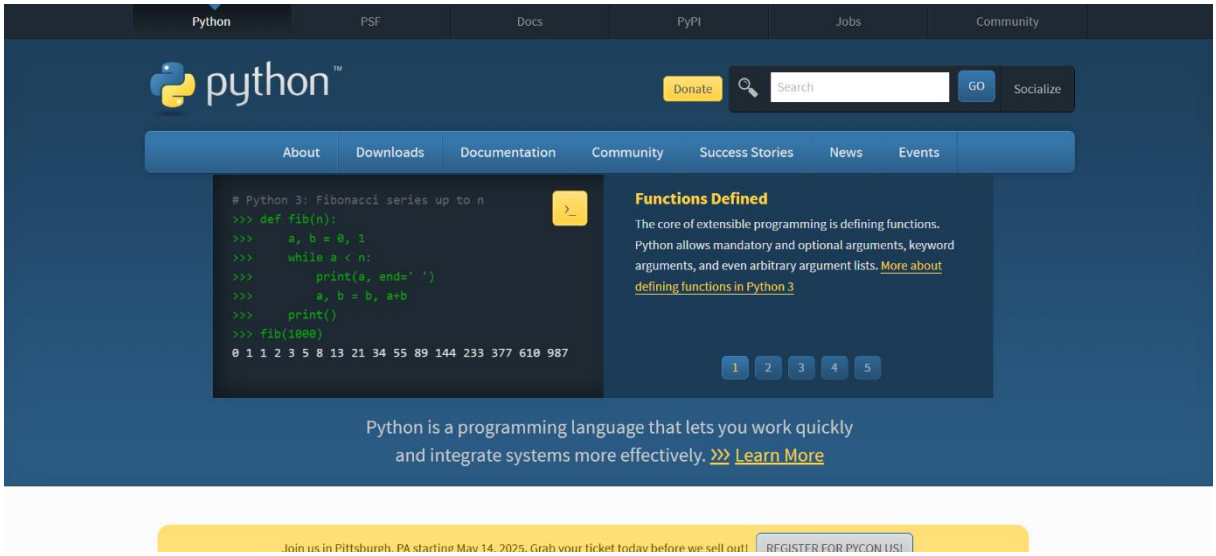
αξιοποιώντας την τεχνική της ενισχυτικής μάθησης (Reinforcement Learning), δηλαδή την ικανότητα εκμάθησης μέσω της αλληλεπίδρασης με τον εκάστοτε παίκτη.

Τέλος, τα LLMs παρέχουν τη δυνατότητα δημιουργίας περιεχομένου, όπως επίπεδα του παιχνιδιού ή ακόμα και ολοκληρωμένα παιχνίδια, προσφέροντας έμπνευση και βελτιώνοντας το περιεχόμενο για τους σχεδιαστές παιχνιδιών. Αυτή η δυνατότητα ενισχύει τη δημιουργικότητα στον τομέα του παιχνιδιού και καθιστά την ανάπτυξή του πιο ευέλικτη, καθώς τα LLMs μπορούν να βοηθήσουν στην αυτόματη δημιουργία περιεχομένου που προσαρμόζεται στις ανάγκες του παιχνιδιού και των παικτών (Gallotta, 2024).

2. Τεχνολογίες που χρησιμοποιήθηκαν

Για την ανάπτυξη του παιχνιδιού αξιοποιήθηκε ένας συνδυασμός σύγχρονων τεχνολογιών, εργαλείων προγραμματισμού και πλατφορμών. Η Python αποτέλεσε τη βασική γλώσσα προγραμματισμού, ενώ χρησιμοποιήθηκαν βιβλιοθήκες όπως η Pygame και η Flask. Επιπλέον, αξιοποιήθηκαν προηγμένα μοντέλα τεχνητής νοημοσύνης, όπως το Llama3, καθώς και εργαλεία δημιουργίας και επεξεργασίας πολυμέσων, όπως το Bing AI Image Creator, το Photopea και το Eleven Labs. Η διαχείριση του κώδικα πραγματοποιήθηκε μέσω του Github, ενώ η ανάπτυξη του παιχνιδιού πραγματοποιήθηκε στο Visual Studio Code. Τέλος, η πλατφόρμα Pixabay χρησιμοποιήθηκε για την ανεύρεση ηχητικών εφέ.

2.1. Python



Εικόνα 1.Python

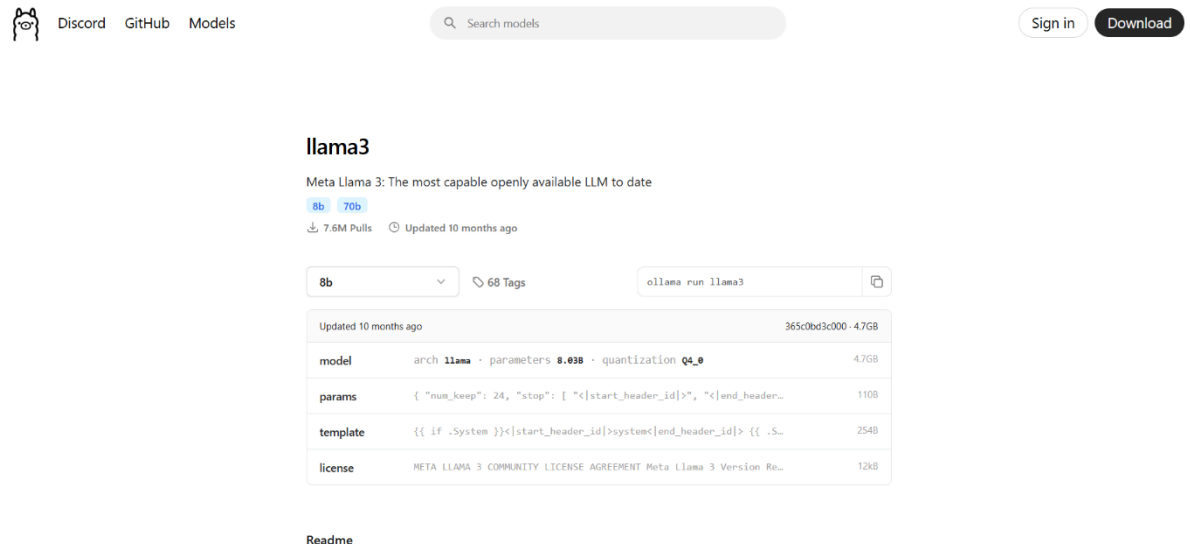
Η Python[1] είναι μια ισχυρή και ευέλικτη γλώσσα προγραμματισμού. Είναι απλή, ευανάγνωστη και εύχρηστη, έτσι αποτελεί μία εξαιρετική επιλογή για μία ποικιλία εφαρμογών. Σχεδιάστηκε από τον Guido van Rossum το 1990. Υποστηρίζει αυτόματη διαχείριση μνήμης και ισχυρές δομές δεδομένων, όπως λίστες και λεξικά. Η σύνταξή της είναι απλή και παραπέμπει αρκετά σε ψευδοκώδικα, επιτρέποντας τη γρήγορη ανάπτυξη και συντήρηση. Ενσωματώνεται άρτια με άλλα εργαλεία και γλώσσες κάτι που καθιστά τη χρήση της ιδανική για σύνθετες ροές εργασίας και επεκτάσιμα συστήματα. Τέλος διαθέτει ένα πλούσιο οικοσύστημα, στο οποίο περιλαμβάνονται βιβλιοθήκες για αριθμητικούς υπολογισμούς, ανάλυση δεδομένων, ανάπτυξη ιστού και πολλά άλλα (Sanner, 1999).

Βιβλιοθήκες Python:

[1] https://www.python.org/

- **Pygame**: Η Pygame είναι ένα σύνολο από Modules της γλώσσας προγραμματισμού της Python, ειδικά σχεδιασμένη για τη δημιουργία απλών παιχνιδιών με διεπαφές γραφικών. Δομημένη στις βιβλιοθήκες SDL (Simple DirectMedia Layer), που παρέχουν πρόσβαση στα πολυμέσα του υπολογιστή, επιτρέπουν στους προγραμματιστές να δημιουργούν παιχνίδια που περιλαμβάνουν ήχους, γραφικά και δυνατότητες εισόδου. Χαρακτηρίζεται από φορητότητα και συμβατότητα με τα περισσότερα λειτουργικά συστήματα, όπως Windows, Linux και MaxOs. Ένα μέρος της είναι γραμμένο στο γλώσσα προγραμματισμού C, καθιστώντας τη ταχύτερη από την Python σε ορισμένες λειτουργίες. Παρέχοντας ένα σύνολο επαναχρησιμοποιούμενων λειτουργιών, η Pygame απλουστεύει τον προγραμματισμό των video games αποτελώντας απαραίτητο εργαλείο για τη δημιουργία παιχνιδιών με χρήση της Python (Shinde, 2021).
- **Flask**: Το Flask είναι ένα απλό και ευέλικτο framework που χρησιμοποιείται για τη δημιουργία web εφαρμογών και APIs στην Python. Όντας εύχρηστο και τροποποιήσιμο, επιτρέπει στους προγραμματιστές να επιλέξουν το κατάλληλο εργαλείο. Δημιουργήθηκε από μία ομάδα με το όνομα Pocoo το 2010 και τώρα συντηρείται από το The Pallets Project. Σήμερα έχει μία ισχυρή κοινότητα για υποστήριξη (Relan, 2019).

2.2. Llama3



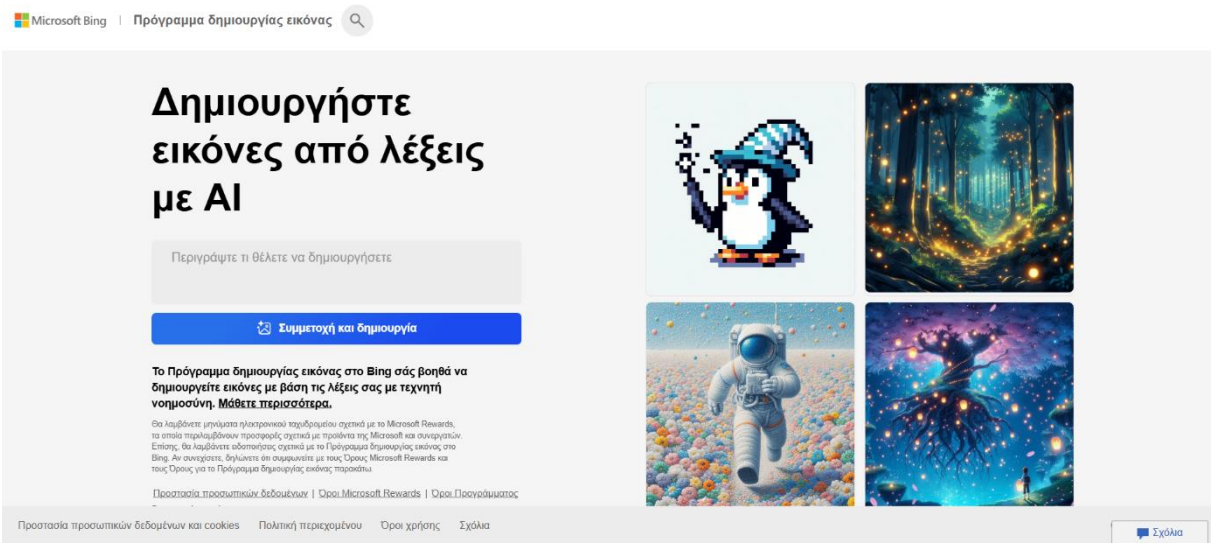
Εικόνα 2.LLama3

Τα Llama3[2] είναι μία σειρά μεγάλων γλωσσικών μοντέλων (LLMs) ανοιχτού κώδικα η οποία αναπτύχθηκε από τη META, τα οποία διατίθενται στην επιστημονική κοινότητα χωρίς εμπορική άδεια. Σε αντίθεση με μοντέλα κλειστού κώδικα όπως το GPT, τα Llama επιτρέπουν την ανάπτυξη μοντέλων με εξειδίκευση σε συγκεκριμένες εφαρμογές. Τα πρώτα μοντέλα Llama κυκλοφόρησαν το Φεβρουάριο του 2023 και το μέγεθος τους κυμαίνεται μεταξύ 7b με 65b (Minaee, 2024).

Το Llama3 είναι διαθέσιμο στα μεγέθη των 8b και των 70b. Υπάρχει η έκδοση του προ-εκπαιδευμένου μοντέλου καθώς και του instruction-based μοντέλου, το οποίο είναι ιδανικό σε περιπτώσεις διαλόγων, συνομιλιών ή κατανόηση οδηγιών με ακρίβεια.

Για να μπορέσει να τρέξει στον υπολογιστή θα χρειαστεί αρχικά η δωρεάν εγκατάσταση του Ollama[3]. Έπειτα στη γραμμή εντολών cmd (command prompt) πρέπει να καταχωρηθεί η εντολή `ollama pull llama3`. Μόλις ολοκληρωθεί η διεργασία πρέπει να καταχωρηθεί η εντολή `ollama run llama3`, ώστε να τρέξει το Llama3. Από εκεί και πέρα δίνεται η δυνατότητα συνομιλίας μαζί του μέσω του cmd.

2.3. Bing AI Image Creator



Εικόνα 3.Bing AI Image Creator

Το Bing AI Image Creator[4] είναι ένα δωρεάν online εργαλείο της Microsoft και της OpenAI. Χρησιμοποιεί τεχνητή νοημοσύνη για τη δημιουργία εικόνων με βάση την περιγραφή που καταχωρεί ο χρήστης.

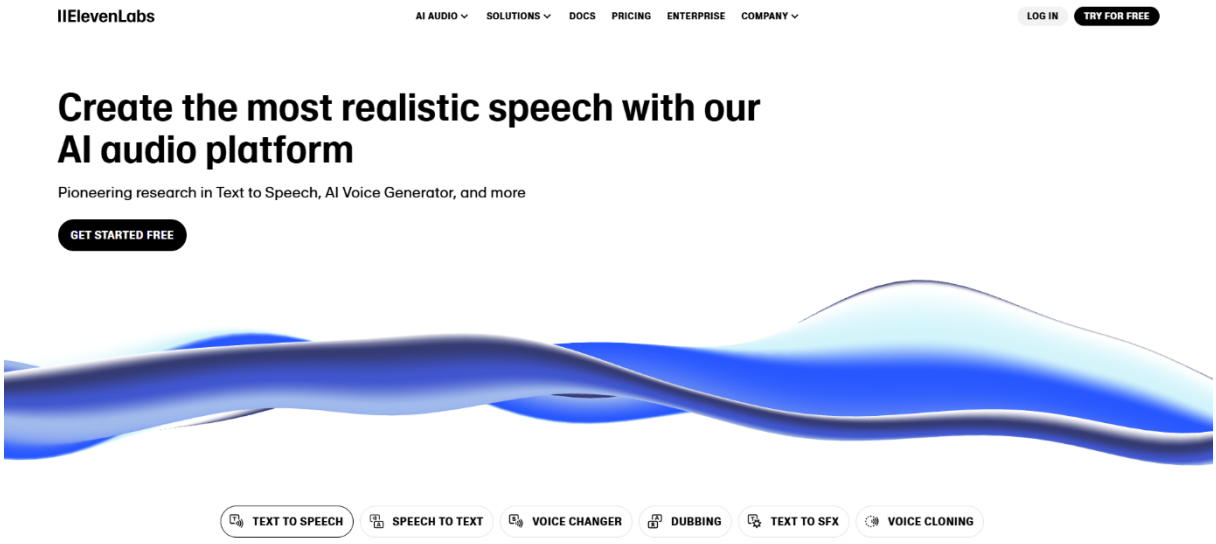
[2] https://ollama.com/library/llama3
[3] Παράρτημα Ι: Οδηγίες εγκατάστασης
[4] https://www.bing.com/images/create

2.4. Photopea

Εικόνα 4.Photopea

Είναι ένα online[5] εργαλείο το οποίο χρησιμοποιείται για την επεξεργασία εικόνας. Υποστηρίζει διάφορα είδη αρχείων, όπως PSD, AI, JPG και PNG. Παρέχει εργαλεία για layers, μάσκες, φίλτρα κ.α. Είναι εξ ολοκλήρου διαδικτυακό και δεν απαιτεί εγκατάσταση. Διαθέτει δωρεάν και premium έκδοση. Είναι ιδανικό τόσο για γρήγορη όσο και για πιο σύνθετη επεξεργασία εικόνων.

2.5. Eleven Labs

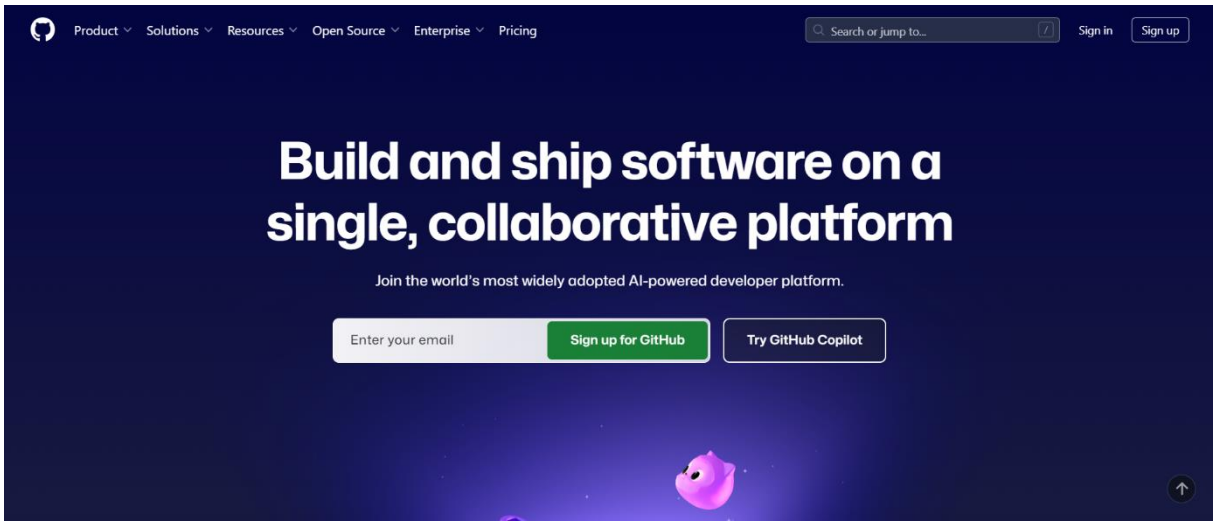


Εικόνα 5.Eleven Labs

[5] https://www.photopea.com/

Το Eleven Labs[6] είναι μια AI πλατφόρμα με εξειδίκευση στη δημιουργία φυσικών φωνών μέσω text-to-speech (TTS), καθώς και στην αντιγραφή φωνής (voice cloning). Παράγει αρκετά ρεαλιστική, ανθρώπινη ομιλία από κείμενο που εισάγει ο χρήστης, υποστηρίζει πολλές γλώσσες, αλλά και διαλέκτους. Είναι χρηστικό σε διάφορους τομείς όπως στην αφήγηση ακουστικών βιβλίων, σε ψηφιακούς βοηθούς, καθώς και στη δημιουργία φωνών για χαρακτήρες animation και video games, και άλλα πολλά.

2.6. Github

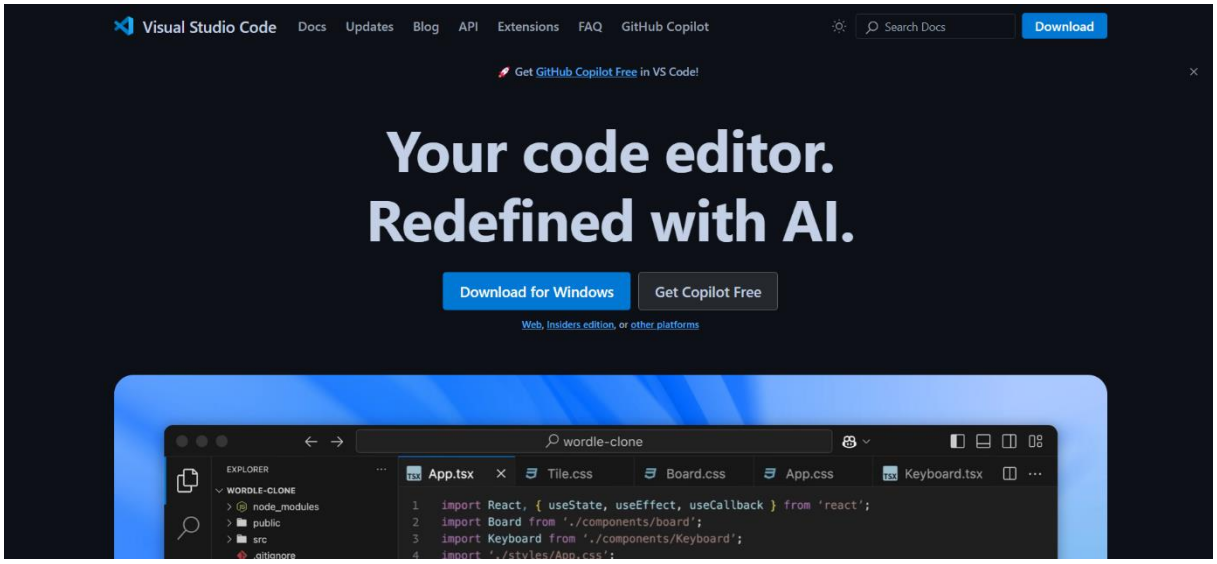


Εικόνα 6.Github

Το Github[7] είναι μια πλατφόρμα διαχείρισης κώδικα που επιτρέπει σε προγραμματιστές να αποθηκεύουν, να διαμοιράζονται και να συνεργάζονται σε έργα λογισμικού. Βασίζεται στο σύστημα ελέγχου εκδόσεων Git και παρέχει λειτουργίες, όπως αποθετήρια (repositories), pull requests, issue tracking και continuous integration. Χρησιμοποιείται ευρέως από εταιρείες, open-source κοινότητες και μεμονωμένους προγραμματιστές για την ανάπτυξη και διαχείριση λογισμικού.

[6] https://elevenlabs.io
[7] https://github.com/

2.7. Visual Studio Code

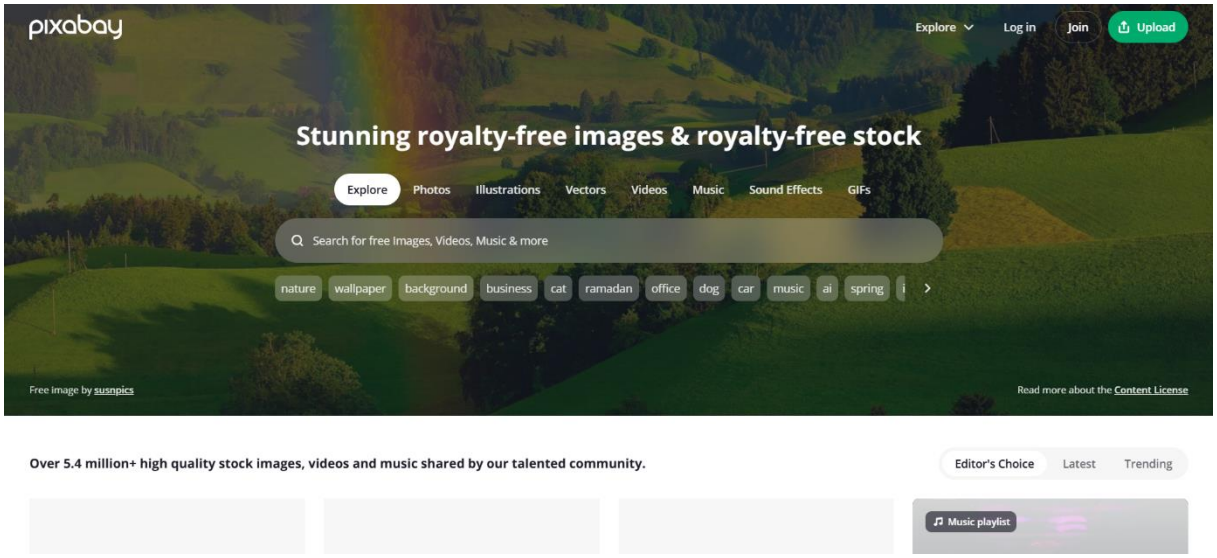


Εικόνα 7. Visual Studio Code

Το Visual Studio Code[8] (VS Code) είναι ένας δωρεάν, ελαφρύς αλλά ισχυρός επεξεργαστής κώδικα της Microsoft, που υποστηρίζει πολλές γλώσσες προγραμματισμού και διαθέτει ενσωματωμένο debugging, Git integration, επεκτάσεις και Live Share για συνεργασία σε πραγματικό χρόνο. Χρησιμοποιείται ευρέως για ανάπτυξη web, mobile και desktop εφαρμογών, καθώς και για machine learning και data science. Είναι διαθέσιμο σε Windows, macOS και Linux, καθιστώντας το μια από τις πιο δημοφιλείς επιλογές για προγραμματιστές παγκοσμίως.

[8] https://code.visualstudio.com/

2.8. Pixabay



Εικόνα 8.Pixabay

Το Pixabay[9] είναι μια διαδικτυακή πλατφόρμα που προσφέρει δωρεάν εικόνες, βίντεο, μουσική και illustrations χωρίς πνευματικά δικαιώματα. Οι χρήστες μπορούν να κατεβάσουν και να χρησιμοποιήσουν το περιεχόμενο για προσωπικούς και εμπορικούς σκοπούς, χωρίς να απαιτείται αναφορά στον δημιουργό. Αποτελεί μια δημοφιλή επιλογή για δημιουργούς περιεχομένου και σχεδιαστές που αναζητούν υψηλής ποιότητας πολυμέσα χωρίς περιορισμούς χρήσης. Χρησιμοποιήθηκε για τη δημιουργία ηχητικών εφέ στο παιχνίδι.

[9] https://pixabay.com/

3. Game Design Document

Αυτό το κεφάλαιο παρέχει μια αναλυτική παρουσίαση του "Game Design Document" (GDD) του παιχνιδιού "A Quest of Knowledge". Εδώ παρατίθεται η γενική επισκόπηση του παιχνιδιού, οι στόχοι της εμπειρίας του παίκτη, οι τεχνολογίες που χρησιμοποιήθηκαν, καθώς και τα καινοτόμα χαρακτηριστικά και η μοναδικότητα του παιχνιδιού. Στη συνέχεια, εξετάζονται οι επιρροές και η καινοτομία του έργου, ενώ παρατίθεται και η περιγραφή του παιχνιδιού, οι χειρισμοί και το σενάριο. Σκοπός του κεφαλαίου είναι να προσφέρει μια πλήρη εικόνα για τη διαδικασία σχεδίασης, ανάπτυξης και υλοποίησης του παιχνιδιού.

3.1. Επισκόπηση του παιχνιδιού

Τίτλος: A Quest of Knowledge

Υπότιτλος: An Educational Serious Game with NPCs and AI

Περιγραφή: Ο χρήστης εξερευνά ένα δισδιάστατο κόσμο αλληλεπιδρώντας με NPCs που εκπροσωπούν διαφορετικά εκπαιδευτικά αντικείμενα. Στην αρχή του παιχνιδιού ο χρήστης θέτει τον επιθυμητό αριθμό ερωτήσεων που θέλει να απαντήσει ανά κατηγορία. Μπορεί να επιλέξει από 0 έως 10 ερωτήσεις για το κάθε εκπαιδευτικό αντικείμενο. Στόχος του παιχνιδιού είναι να βρει όλους τους διαθέσιμους NPCs, να απαντήσει στις ερωτήσεις τους και να συλλέξει όλα τα αντικείμενα που θα εμφανιστούν. Στη διαδρομή θα συναντήσει εχθρούς, τους οποίους είτε μπορεί να αποφύγει είτε να αντιμετωπίσει.

Πρόσθετα στοιχεία:

- Ο παίκτης ξεκινώντας έχει διαθέσιμες 3 ζωές, τις οποίες μπορεί να χάσει από επιθέσεις εχθρών. Οι ζωές αναπληρώνονται απαντώντας σωστά σε 3 αντικείμενα γνώσης.
- Στην περίπτωση που ο αριθμός ερωτήσεων που ορίσει στην αρχή είναι μηδενικός, τότε πρέπει απλά να βρει το συγκεκριμένο NPC και να μιλήσει μαζί του ώστε να ακούσει μία ενδιαφέρουσα πληροφορία (fun fact) σχετικά με το αντικείμενο γνώσης του.
- Ο παίκτης εξερευνά περιοχές, ανακαλύπτει NPCs και συλλέγει αντικείμενα με τη βοήθεια ενός mini map.
- Υπάρχει δυνατότητα απόκτησης όπλου για την αντιμετώπιση των εχθρών

3.2. Γενική Παρουσίαση

3.2.1. Στόχοι Εμπειρίας Παίκτη/Χρήστη (Player Experience Goals):

Εκπαίδευση: Ο παίκτης εμπλουτίζει τις γνώσεις του μέσα από την αλληλεπίδραση με NPCs.

Περιπέτεια: Ανακαλύπτει νέες τοποθεσίες, NPCs και αντικείμενα με την βοήθεια ενός χάρτη

Επιβίωση: Πρέπει να αποφύγει εχθρούς και να διατηρήσει τις διαθέσιμες ζωές του, ώστε να μπορέσει να ολοκληρώσει επιτυχώς το παιχνίδι

Στρατηγική: Μπορεί ανάλογα με την ποσότητα των διαθέσιμων ζωών του και τη διαθεσιμότητα όπλου να επιλέξει αν θα αντιμετωπίσει ή θα αποφύγει εχθρούς

Επίτευξη: Ανακαλύπτει τοποθεσίες, NPCs και συλλέγει αντικείμενα

Επιλογή: Έχει τη δυνατότητα επιλογής του χαρακτήρα και του αριθμού των ερωτήσεων

3.2.2. Επιρροές

Το "A Quest of Knowledge" είναι εμπνευσμένο από το Legend of Zelda του 1986. Ο ήρωας εξερευνά ένα δισδιάστατο κόσμο συλλέγοντας αντικείμενα, λύνοντας γρίφους και αντιμετωπίζοντας εχθρούς.

3.3. Καινοτομία και Μοναδικότητα

Το "A Quest of Knowledge" αποτελεί μια καινοτόμο προσέγγιση στον χώρο των εκπαιδευτικών σοβαρών παιχνιδιών (serious games), αξιοποιώντας τεχνολογίες τεχνητής νοημοσύνης (AI) και δυναμικής παραγωγής περιεχομένου για τη δημιουργία μιας εξατομικευμένης εμπειρίας μάθησης. Η πρωτοπορία του παιχνιδιού έγκειται στη μοναδική ενσωμάτωση δυναμικά παραγόμενου περιεχομένου μέσω LLM και API, επιτρέποντας στους παίκτες να βιώσουν μια εξατομικευμένη εμπειρία μάθησης και εξερεύνησης.

3.3.1. Χαρακτηριστικά και Πρωτοτυπία

Δυναμική Παραγωγή Περιεχομένου μέσω LLM και API

Το παιχνίδι διαφοροποιείται από τα παραδοσιακά εκπαιδευτικά παιχνίδια, καθώς δεν βασίζεται σε προκαθορισμένα σενάρια και διαλόγους. Αντίθετα, οι συνομιλίες με τους NPCs, οι ερωτήσεις, καθώς και τα βοηθητικά μηνύματα παράγονται δυναμικά μέσω μεγάλων γλωσσικών μοντέλων (LLM) και API, προσφέροντας μια μοναδική και εξατομικευμένη εμπειρία στον παίκτη.

NPC που αναπαριστούν υπαρκτά ιστορικά πρόσωπα

Οι χαρακτήρες με τους οποίους συνομιλεί ο παίκτης δεν είναι απλοί καθοδηγητές ή βοηθοί του παίκτη. Αντίθετα, αναπαριστούν υπαρκτά ιστορικά πρόσωπα, που σχετίζονται με το εκάστοτε εκπαιδευτικό αντικείμενο. Αυτός ο σχεδιασμός ενισχύει τη βιωματική μάθηση και παρέχει μια διαδραστική προσέγγιση στην κατανόηση ιστορικών γεγονότων και προσωπικοτήτων.

Συνδυασμός Εκπαίδευσης και Περιπέτειας

Ο παίκτης αποκτά νέες γνώσεις ενώ παράλληλα εξερευνά τον κόσμο του παιχνιδιού συλλέγει αντικείμενα και έρχεται αντιμέτωπος με εχθρούς.

Ενσωμάτωση Στοιχείων Επιβίωσης

Ο παίκτης ξεκινά με περιορισμένο αριθμό ζωών, τις οποίες μπορεί να χάσει από εχθρούς, αλλά και να ανακτήσει μέσω επιτυχημένων απαντήσεων σε ερωτήσεις. Επιπλέον, η δυνατότητα χρήσης όπλων προσθέτει ένα στρατηγικό στοιχείο που διαφοροποιεί το παιχνίδι από τυπικά quiz-based serious games.

Εξατομικευμένη Προσαρμογή Δυσκολίας

Οι ερωτήσεις προσαρμόζονται δυναμικά, λαμβάνοντας υπόψη την ηλικία, το επίπεδο γνώσεων και την προτίμηση δυσκολίας του παίκτη. Αυτό καθιστά το παιχνίδι προσιτό τόσο σε παιδιά όσο και σε ενήλικες, παρέχοντας μια εξατομικευμένη μαθησιακή εμπειρία.

Διαφορετικές εκπαιδευτικές θεματικές

Υπάρχουν 11 διαφορετικές διαθέσιμες εκπαιδευτικές ενότητες, από τις οποίες ο παίκτης λαμβάνει κάποια γνώση (Μαθηματικά, Γεωγραφία, Αστρονομία, Πληροφορική, Φυσική, Χημεία, Ιστορία, Τέχνες, Μουσική, Περιβαλλοντική Εκπαίδευση, Βιολογία)

Συλλεκτικά αντικείμενα- Collectable Items

Τα αντικείμενα που καλείται να συλλέξει ο παίκτης είναι προσαρμοσμένα στη θεματική που αφορά το κάθε ένα.

3.3.2. Συναφή έργα

Σε αντίθεση με τα περισσότερα serious games, τα οποία επικεντρώνονται σε στατικά quiz και προκαθορισμένες εκπαιδευτικές διαδρομές, το "A Quest of Knowledge" αξιοποιεί προηγμένες τεχνολογίες τεχνητής νοημοσύνης για τη δυναμική δημιουργία περιεχομένου. Παράλληλα, αντλεί επιρροές από παραδοσιακά adventure games, όπως το Legend of Zelda (1986), αλλά ανανεώνει την εμπειρία μέσα από τη χρήση εκπαιδευτικών στοιχείων και εξελιγμένων μηχανισμών αλληλεπίδρασης. Παρόμοια παιχνίδια που ενσωματώνουν AI-driven μηχανισμούς και δυναμική αλληλεπίδραση περιλαμβάνουν:

- AI Town[10]: Ένα εικονικό περιβάλλον όπου οι AI χαρακτήρες ζουν, συνομιλούν και αλληλεπιδρούν μεταξύ τους, χωρίς να υπάρχουν στόχοι και συνθήκες νίκης ή ήττας.
- Hidden Door[11]: Ένα AI-driven RPG που δημιουργεί δυναμικές, δημιουργικές ιστορίες.
- AI Dungeon[12]: Ένα text-based adventure game που χρησιμοποιεί GPT για τη δημιουργία ατελείωτων ιστοριών, επιτρέποντας στους παίκτες να διαμορφώνουν την αφήγηση.

[10] https://www.convex.dev/ai-town
[11] https://www.hiddendoor.co/
[12] https://aidungeon.com/

- Gandalf[13]: Ένα παιχνίδι γρίφων όπου ο παίκτης πρέπει να ξεκλειδώσει κωδικούς, ενώ ο AI Gandalf προσαρμόζει τις άμυνές του.

Παρόλο που αυτά τα παιχνίδια ενσωματώνουν AI-driven μηχανισμούς, κανένα δε συνδυάζει τόσο την εξερεύνηση, το gameplay με top-down περιήγηση, όσο την εκπαίδευση και τη δυναμική παραγωγή περιεχομένου μέσω LLM. Η απουσία προκαθορισμένων διαλόγων και η συνεχώς μεταβαλλόμενη φύση των ερωτήσεων καθιστούν το "A Quest of Knowledge" μοναδικό στο είδος του, ενισχύοντας την αναπαιξιμότητα και την εκπαιδευτική του αξία.

3.5. Περιγραφή του παιχνιδιού

Βασική ιδέα και δήλωση σκοπού (Game Concept Statement): Ένα εκπαιδευτικό, 2D top-down παιχνίδι, που συνδυάζει τη μάθηση με τη στρατηγική. Οι παίκτες απαντούν ερωτήσεις σε NPCs ώστε να συλλέξουν αντικείμενα και έρχονται αντιμέτωποι με εχθρούς συνδυάζοντας τη μάθηση με τη διασκέδαση και την περιπέτεια.

Είδος: Εκπαιδευτικό, Γνώσεων, Περιπέτεια, Στρατηγική

Πλατφόρμα: PC (Windows, Linux, Mac) μέσω Pygame

Κοινό-Στόχος (Target Audience): Παίκτες όλων των ηλικιών που προτιμούν εκπαιδευτικά παιχνίδια, παιχνίδια γνώσεων/quiz και παιχνίδια εξερεύνησης και περιπέτειας. Οι ερωτήσεις προσαρμόζονται δυναμικά με βάση την ηλικία.

Γενικά χαρακτηριστικά:

- **Επίπεδα (Levels):** Το παιχνίδι διαδραματίζεται σε έναν ενιαίο και ανοιχτό κόσμο (open world), ο οποίος είναι χωρισμένος σε διάφορες περιοχές (χωριά, δάση, πόλεις κ.τ.λ.)
- **Modes (Λειτουργίες Παιχνιδιού):**
    - **Εξερεύνηση (Exploration Mode):** Ο παίκτης εξερευνά το χάρτη, ανακαλύπτει NPCs και συλλέγει αντικείμενα.
    - **Μάχες (Combat Mode):** Ενεργοποιείται όταν ο παίκτης έρχεται αντιμέτωπος με εχθρούς. Ο παίκτης έχει τη δυνατότητα να επιλέξει αν θα εμπλακεί σε μάχη ή θα αποφύγει τους εχθρούς.
    - **Συνομιλία (Chat Mode):** Ενεργοποιείται η επιφάνεια συνομιλίας (chat surface) και ο παίκτης συνομιλεί με NPCs, ενώ το περιβάλλον του

[13] https://gandalf.lakera.ai/baseline

παιχνιδιού παγώνει, καθιστώντας λειτουργική μόνο την επικοινωνία μέσω του chat.

- **Γραφικά και Ήχος:**
    - **Στυλ:** Pixel Art με έντονα και ζωηρά χρώματα
    - **Προβολή:** 2d top-down view, η οποία παρέχει πανοραμική προβολή του κόσμου του παιχνιδιού αλλά και του χαρακτήρα του παίκτη.
    - **Περιβάλλον:** Περιλαμβάνει διαφορετικές περιοχές (πόλεις, δάση, χωριά)
    - **Ήχος και Μουσική:** Μουσική επένδυση που ταιριάζει με το περιβάλλον και ηχητικά εφέ που ανταποκρίνονται στις αλληλεπιδράσεις του παίκτη με εχθρούς και αντικείμενα.
    - **Γλώσσα:** Η γλώσσα που υποστηρίζεται μεταξύ των διαλόγων και ολόκληρου του παιχνιδιού είναι η Αγγλική.
- **Επίπεδο Δυσκολίας (Difficulty):** Το επίπεδο δυσκολίας ορίζεται από τον παίκτη μέσω της επιλογής του αριθμού ερωτήσεων (0-10) που θέλει να απαντήσει, αλλά και την ηλικία που θα δηλώσει κατά την έναρξη του παιχνιδιού.
- **Αναπαιξιμότητα (Replayability):** Ο χρήστης μπορεί να ξαναπαίξει το παιχνίδι επιλέγοντας διαφορετικό αριθμό ερωτήσεων, διαφορετικό χαρακτήρα ή ακόμα και διαφορετική ηλικία. Η τυχαία και μη προκαθορισμένη δημιουργία ερωτήσεων, βοηθητικών μηνυμάτων και fun facts που παράγονται από LLM, ενισχύει ακόμη περισσότερο την αναπαιξιμότητα του παιχνιδιού.

Βασικά Χαρακτηριστικά Παιχνιδιού (Key Gameplay Features):

- **NPCs με εξειδίκευση σε συγκεκριμένο εκπαιδευτικό αντικείμενο:** Ο κάθε NPC είναι συνδεδεμένος με ένα συγκεκριμένο αντικείμενο γνώσης (Μαθηματικά, Φυσική κ.τ.λ.) και οι ερωτήσεις που θέτει στον παίκτη είναι προσαρμοσμένες στην ηλικία του.
- **Αντικείμενα για συλλογή (Collectable Items):** Τα αντικείμενα που παράγονται και πρέπει να συλλεχθούν μετά την επιτυχή απάντηση των ερωτήσεων σε έναν NPC αναπαριστούν ένα αντικείμενο σχετικό με τη θεματική (π.χ. Πληροφορική, Υπολογιστής)
- **Χάρτης (Mini-map):** Δείχνει τη θέση του παίκτη και των NPC, είναι μία μικρογραφία του κόσμου του παιχνιδιού

- **Εχθροί:** Περιφέρονται τυχαία και κυνηγούν τον παίκτη όταν πλησιάσει. Σε περίπτωση που έρθουν σε επαφή μαζί του, του αφαιρούν μία ζωή.
- **Δυναμική Προσαρμογή Περιεχομένου:** Γίνεται χρήση LLM και API για την παραγωγή δυναμικών ερωτήσεων, διαλόγων και βοηθητικών μηνυμάτων.
- **Ζωές:** Το παιχνίδι ξεκινά με 3 διαθέσιμες ζωές. Χάνεται μία ζωή αν ο παίκτης δεχθεί επίθεση από εχθρό. Αναπληρώνονται αν ο παίκτης απαντήσει σωστά στις ερωτήσεις 3 NPCs και μπορούν να φτάσουν μέχρι 5.
- **Αλληλεπίδραση με LLM NPCs:** Ο παίκτης απαντάει σε ερωτήσεις ή μαθαίνει ενδιαφέρουσες πληροφορίες (fun facts). Οι NPCs βοηθούν τον παίκτη να βρει τη σωστή απάντηση.
- **Όπλο:** Μπορεί να χρησιμοποιηθεί στρατηγικά για να αποφευχθεί η απώλεια ζωών κατά την αντιμετώπιση εχθρών
- **Τερματισμός Παιχνιδιού:**
    - Το παιχνίδι ολοκληρώνεται εφόσον ο παίκτης απαντήσει σωστά σε όλες τις ερωτήσεις και συλλέξει όλα τα θεματικά αντικείμενα.
    - Στην περίπτωση που ο παίκτης χάσει όλες τις διαθέσιμες ζωές του, τότε το παιχνίδι τερματίζεται χωρίς να ολοκληρώσει την αποστολή του.

3.6. Χειρισμοί

Παιχνίδι

Ο παίκτης πλοηγείται με τη χρήση των arrow keys (βελάκια πληκτρολογίου):

- Για να προχωρήσει ο παίκτης μπροστά, πρέπει να πατηθεί το πλήκτρο με το πάνω βέλος στο πληκτρολόγιο.
- Για να πάει πίσω, πρέπει να πατηθεί το πλήκτρο με το κάτω βέλος στο πληκτρολόγιο.
- Για να κινηθεί αριστερά, πρέπει να πατηθεί το πλήκτρο με το αριστερό βέλος στο πληκτρολόγιο.
- Για να κινηθεί δεξιά, πρέπει να πατηθεί το πλήκτρο δεξί με το βέλος στο πληκτρολόγιο.

Για να αντιμετωπίσει έναν αντίπαλο, ο χρήστης πρέπει να πατήσει το SPACE, ώστε να του προκαλέσει βλάβη.

Για να συνομιλήσει με έναν NPC πρέπει να κάνει κλικ πάνω του με τη χρήση του ποντικιού.

Chat

- Η συνομιλία ξεκινάει με κλικ στο πλαίσιο συγγραφής κειμένου και με το πάτημα του πλήκτρου ENTER.

- Ο χρήστης συμπληρώνει τις απαντήσεις του με το πληκτρολόγιο και μόλις ολοκληρώσει πρέπει να πατήσει ENTER.

- Όταν θέλει να αποχωρήσει από τη συνομιλία πρέπει να πατήσει με τη χρήση του ποντικιού το κουμπί Χ πάνω δεξιά στο Chat.

Μενού

Χρήση Πληκτρολογίου:

- Για να πλοηγηθεί ο χρήστης πάνω, πρέπει να πατήσει το πλήκτρο με το πάνω βέλος στο πληκτρολόγιο.
- Για να μετακινηθεί κάτω, πρέπει να πατήσει το πλήκτρο με το κάτω βέλος στο πληκτρολόγιο.
- Για να επιλέξει πρέπει πατήσει το πλήκτρο ENTER.

Χρήση Ποντικιού:

- Μπορεί να χρησιμοποιηθεί το ποντίκι για πλοήγηση στο μενού και με αριστερό κλικ για να ολοκληρωθεί η επιλογή.

3.7. Σενάριο

Ο ήρωας ξεκινά την περιπέτεια του ώστε να διασώσει όλη την γνώση που υπάρχει στον κόσμο. Ταξιδεύει σε ένα μακρινό κόσμο, αναζητώντας τους Άρχοντες της Σοφίας. Η γνώση θα σωθεί μόνο αν καταφέρει να τους συναντήσει όλους και να δεχθεί τη σοφία τους, ώστε να γίνει ο επόμενος Φύλακας της Γνώσης. Θα πρέπει να είναι προσεκτικός γιατί οι κίνδυνοι που θα συναντήσει στο δρόμο του μπορούν να ανατρέψουν την αποστολή του. Θα καταφέρει άραγε να διαφυλάξει όλη τη γνώση της ανθρωπότητας;

The hero embarks on a quest to save all the knowledge of the world. Their journey takes them to a distant realm, where they seek the Lords of Wisdom. The knowledge can only be preserved if the hero manages to meet all of them and receive their wisdom, becoming the next Keeper of Knowledge.

But they must tread carefully, as the dangers they encounter along the way could jeopardize their mission. Will they succeed in safeguarding the entirety of humanity's knowledge?

3.8. Concept Art

Στη συγκεκριμένη ενότητα ακολουθεί παρουσίαση των γραφικών στοιχείων του παιχνιδιού, τα οποία δημιουργήθηκαν με τη χρήση προτροπών στο Bing AI Image Creator, σε pixel art αισθητική. Η pixel-art, ένα καλλιτεχνικό στυλ εμπνευσμένο από τα βιντεοπαιχνίδια των δεκαετιών του ’80 και ’90, χαρακτηρίζεται από μικρά, ορατά pixels, χαμηλή ανάλυση και έντονα χρώματα, δημιουργώντας μια ρετρό, νοσταλγική αίσθηση. Αν και περιορισμένη σε λεπτομέρειες, επιτρέπει την απόδοση σύνθετων χαρακτήρων, τοπίων και εφέ, παραμένοντας ιδιαίτερα δημοφιλής στα indie games, animations και digital art (Samuelson, 2020).

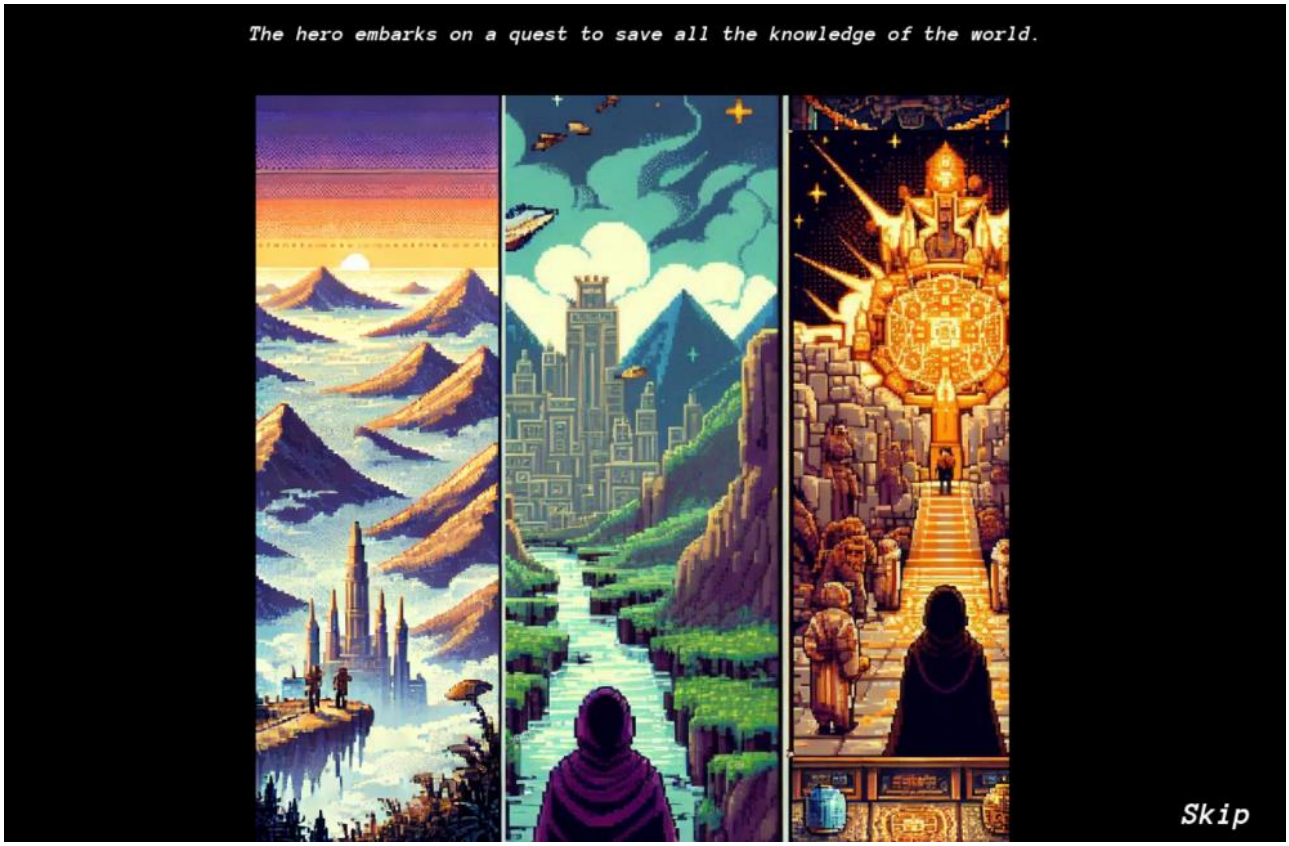
Ζητούμενο: Δημιουργία αρχικού μενού, το οποίο εμφανίζεται πριν από την έναρξη του παιχνιδιού.

Προτροπή για δημιουργία εικόνας: A quest of knowledge 2d game background image medieval pixel art for menu

Εικόνα 9.Αρχικό Μενού

Ζητούμενο: Σύντομη παρουσίαση του σεναρίου του παιχνιδιού μέσω εικονογράφησης.

Προτροπή για δημιουργία εικόνας: medieval 2d game pixel art hero packing for the quest back close pose hooded looking a mountain

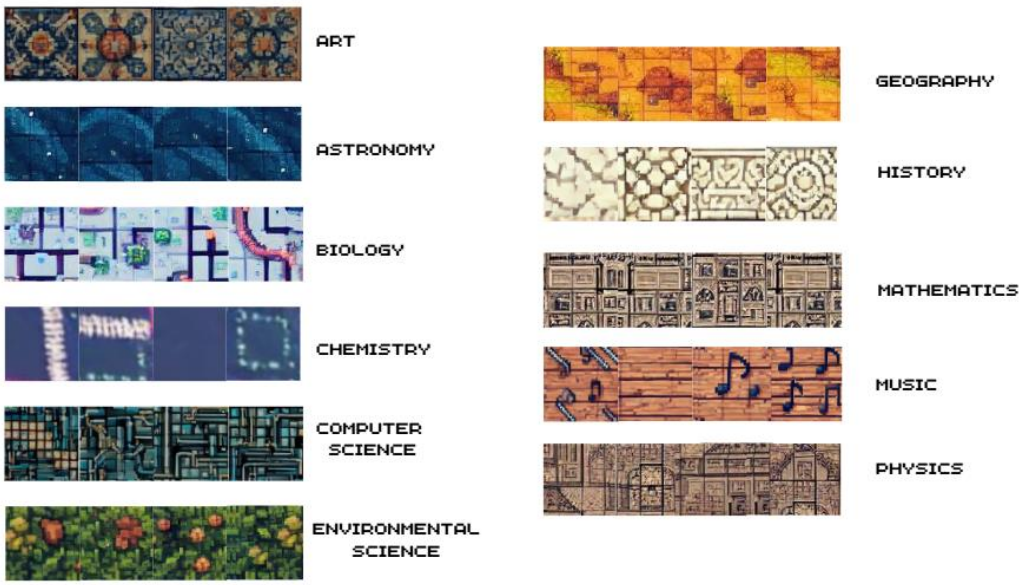


Εικόνα 10.Intro

Ζητούμενο: Δημιουργία 2D χάρτη με διακριτές περιοχές, προσαρμοσμένες στις θεματικές του παιχνιδιού. Στα σημεία ενδιαφέροντος περιλαμβάνονται εικονίδια που εμφανίζονται στο mi-map.

Προτροπή για δημιουργία εικόνας: Create a pixel art tile set for a 2D top-down game featuring {TOPIC} inspired floor tiles.

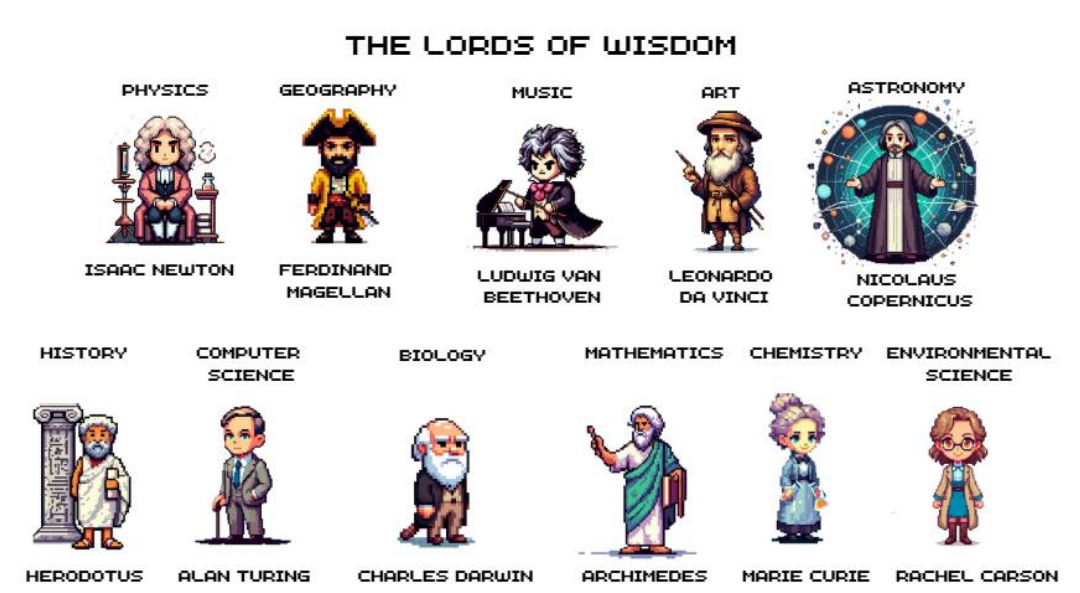
Εικόνα 11.Χάρτης του παιχνιδιού



Εικόνα 12.Tiles- Πλακίδια για κάθε θεματική του χάρτη

Ζητούμενο: Δημιουργία NPCs (μη παικτικών χαρακτήρων) που αναπαριστούν ιστορικά πρόσωπα, σχετικά με τη θεματική του παιχνιδιού. Για κάθε θεματική επιλέγεται τυχαία ένας από τους δύο διαθέσιμους NPCs.

Προτροπή για δημιουργία εικόνας: pixel art sprite {NAME OF NPC} character 2d



Εικόνα 13.NPCs (1η επιλογή)

Εικόνα 14.NPCs (2η επιλογή)

Ζητούμενο: Δημιουργία συλλεκτικών αντικειμένων, το καθένα εκ των οποίων αντιστοιχεί σε μια συγκεκριμένη θεματική κατηγορία του παιχνιδιού.

Προτροπή για δημιουργία εικόνας: {NAME OF EACH TOPIC} related pixel art 2d item

Εικόνα 15.Αντικείμενα για συλλογή

Ζητούμενο: Δημιουργία εχθρικών χαρακτήρων που καταδιώκουν τον παίκτη και εμφανίζονται σε τυχαία σημεία του παιχνιδιού.

Προτροπή για δημιουργία εικόνας: a 2d tiles sprite enemy of knowledge with robotic legs

Εικόνα 16.Εχθρός

Ζητούμενο: Δημιουργία δύο διαθέσιμων χαρακτήρων-παικτών, μεταξύ των οποίων ο χρήστης μπορεί να επιλέξει.

Προτροπή για δημιουργία εικόνας 1: A medieval explorer young hero in 2d pixel art sprite character

Προτροπή για δημιουργία εικόνας 2: A medieval explorer young female hero in 2d pixel art sprite character

Εικόνα 17.Επιλογή ηρώων

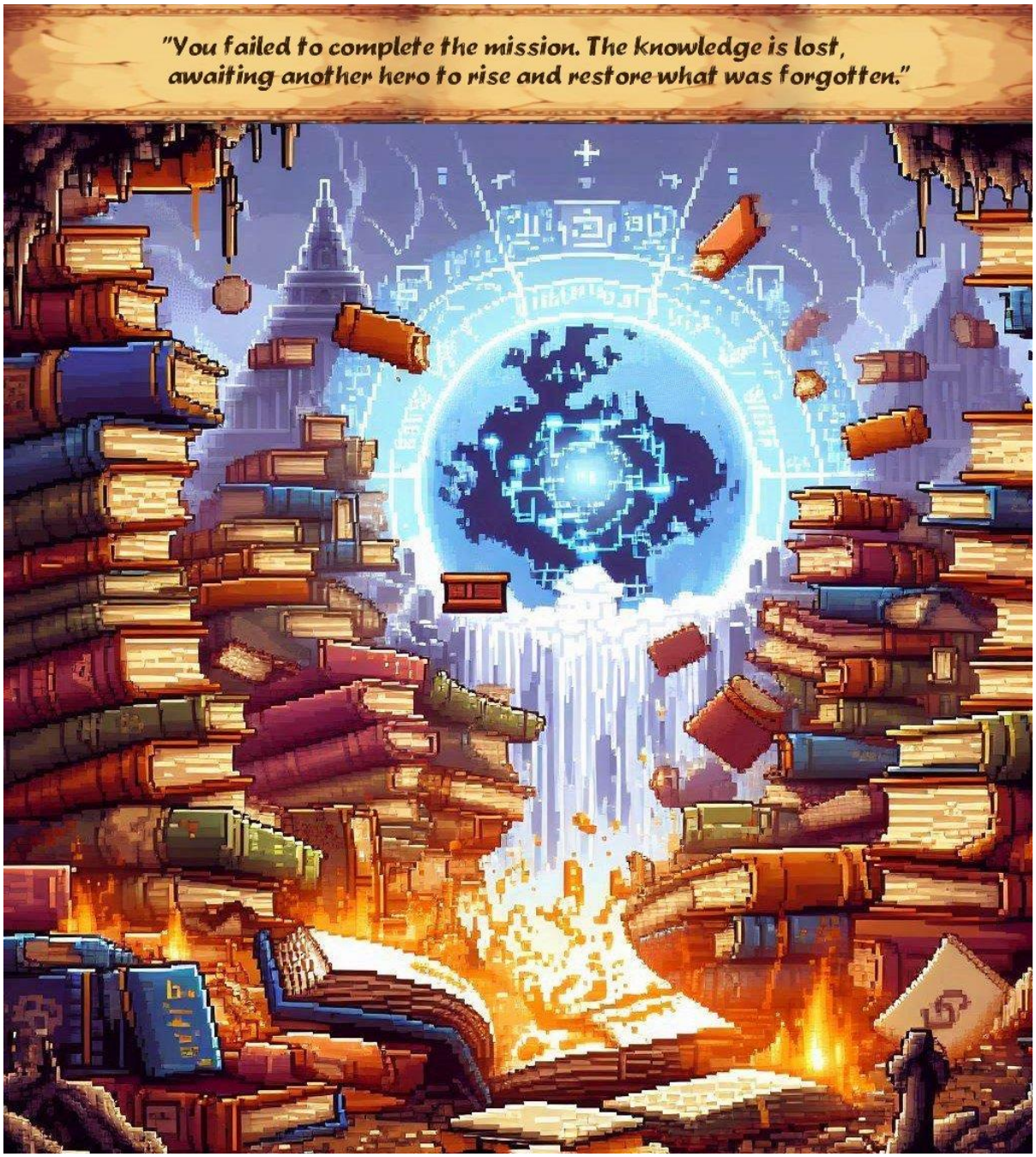
Ζητούμενο: Δημιουργία ενός όπλου, το οποίο θα μπορεί να χρησιμοποιήσει ο παίκτης για να αντιμετωπίσει τους εχθρούς.

Προτροπή για δημιουργία εικόνας: A 2d pixel art sprite medieval sword with blue details in white background

Εικόνα 18.Όπλο

Ζητούμενο: Δημιουργία οθόνης αποτυχίας, η οποία εμφανίζεται όταν ο παίκτης χάσει όλες τις διαθέσιμες ζωές του.

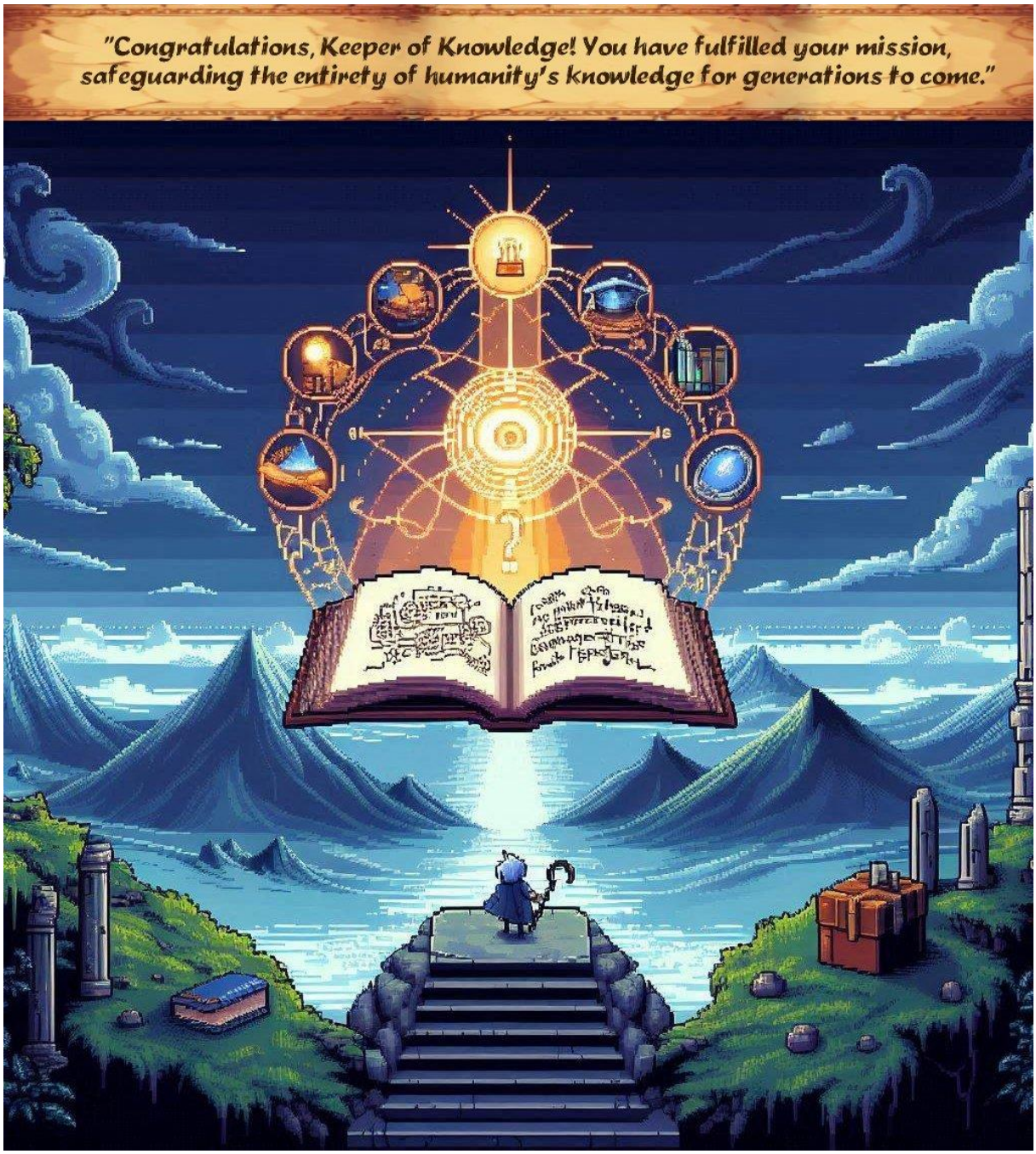
Προτροπή για δημιουργία εικόνας: A quest of knowledge 2d game chaos losing screen pixel art destroyed books



Εικόνα 19.Συνθήκη ήττας

Ζητούμενο: Δημιουργία οθόνης ολοκλήρωσης του παιχνιδιού, η οποία εμφανίζεται όταν ο παίκτης απαντήσει σωστά σε όλες τις ερωτήσεις και συλλέξει όλα τα απαιτούμενα αντικείμενα.

Προτροπή για δημιουργία εικόνας: A quest of knowledge 2d game winning screen medieval pixel art.



Εικόνα 20.Συνθήκης νίκης

4. Υλοποίηση

Η υλοποίηση του έργου ακολούθησε μια σταδιακή προσέγγιση, με την κάθε φάση να ενσωματώνει νέες λειτουργίες και να βελτιώνει τη διαδραστικότητα του συστήματος. Κύριοι στόχοι της υλοποίησης ήταν η δημιουργία μιας λειτουργικής σύνδεσης μεταξύ του Μεγάλου Γλωσσικού Μοντέλου (LLM) και της διεπαφής συνομιλίας (Chat Interface), η ενσωμάτωση αυτής της σύνδεσης σε περιβάλλον παιχνιδιού, η εισαγωγή στοχευμένων θεματικών για την αλληλεπίδραση με τους χρήστες και η συνεχιζόμενη βελτίωση της εμπειρίας του χρήστη μέσω έξυπνων αλληλεπιδράσεων και δυναμικής προσαρμογής του περιεχομένου.

Η πρώτη φάση υλοποίησης επικεντρώθηκε στην ανάπτυξη της σύνδεσης μεταξύ του LLM και της διεπαφής, χρησιμοποιώντας το Flask για την επικοινωνία μέσω API και τη γλώσσα προγραμματισμού Python. Στη δεύτερη φάση, δημιουργήθηκε ένα περιβάλλον παιχνιδιού με ειδικευμένους NPCs που επικεντρώνονταν σε συγκεκριμένα γνωστικά αντικείμενα, προσφέροντας έναν δυναμικό και ενδιαφέρον τρόπο αλληλεπίδρασης. Στην τρίτη φάση, ενσωματώθηκαν λειτουργίες για την εξατομίκευση της εμπειρίας του χρήστη και τη διαχείριση δεδομένων μέσω βάσης δεδομένων. Τέλος, στην τέταρτη φάση, προστέθηκαν ήχοι, γραφικά, νέες θεματικές και NPCs εμπνευσμένοι από ιστορικές προσωπικότητες, εμπλουτίζοντας τον κόσμο του παιχνιδιού και προσφέροντας νέες προκλήσεις στους παίκτες.

4.1. Πρώτη Φάση Ανάπτυξης – Έκδοση 1 της εφαρμογής

Στην αρχική φάση της υλοποίησης, κύριος στόχος ήταν η δημιουργία μιας λειτουργικής σύνδεσης μεταξύ του Μεγάλου Γλωσσικού Μοντέλου (LLM) και της διεπαφής συνομιλίας (Chat Interface) μέσω ενός API.

Εγκατάσταση και Ρύθμιση Περιβάλλοντος

Πρώτο βήμα ήταν η εγκατάσταση του Ollama και η διαμόρφωση του μεγάλου γλωσσικού μοντέλου Llama3 των 8B. Στη συνέχεια με τη χρήση του Visual Studio Code αναπτύχθηκε στη γλώσσα προγραμματισμού Python μία διεπαφή συνομιλίας (Chat Interface), η οποία επικοινωνεί με το LLM μέσω ενός API, που σχεδιάστηκε για αυτό το σκοπό. Για την υλοποίηση αυτής της επικοινωνίας δημιουργήθηκαν δύο βασικά αρχεία Python: api.py και chat.py.

Υλοποίηση API

Το api.py είναι υπεύθυνο για την επικοινωνία με το LLM μέσω του Flask, ενός ελαφριού framework για τη δημιουργία web APIs. Παρακάτω περιγράφονται οι κύριες βιβλιοθήκες που χρησιμοποιήθηκαν.

Βιβλιοθήκες που χρησιμοποιήθηκαν:

- **Flask:** framework για τη δημιουργία του API
- **jsonify:** Μετατροπή των δεδομένων της Python σε αποκρίσεις JSON για HTTP αιτήματα.

- **request:** Λήψη δεδομένων από εισερχόμενα HTTP αιτήματα
- **subprocess:** Εκτέλεση εξωτερικών εντολών στο σύστημα, επιτρέποντας την εκτέλεση του LLM μέσω του τερματικού.

Δημιουργία του Flask App

Αρχικά δημιουργήθηκε ένα αντικείμενο Flask, το οποίο λειτουργεί ως κεντρική εφαρμογή.

```
app = Flask(__name__)
```

Εκτέλεση του LLM

Η συνάρτηση run_the_llm εκτελεί το εξωτερικό LLM μέσω της γραμμής εντολών. Για να πραγματοποιηθεί αυτή η διαδικασία, χρησιμοποιήθηκε το subprocess[14]. Επιπλέον έγινε χρήση του DEVNULL, ώστε να αποφευχθεί το μήνυμα σφάλματος: "failed to get console mode for stdout: The handle is invalid"[15]. Έπειτα, με το try-except πραγματοποιήθηκε ομαλή διαχείριση του σφάλματος “stdout stderr”[16].

```
def run_the_llm(message):
    try:
        llm_run = ['ollama', 'run', 'llama3:latest',  message]
   output = subprocess.check_output(llm_run, encoding='utf-8', stdin=subprocess.DEVNULL)
        return output.strip()
    except subprocess.CalledProcessError as e:
        print("Error running LLM :", e)
        return "Error running LLM"
```

Διαχείριση Αιτημάτων μέσω API

Η διαχείριση της επικοινωνίας πραγματοποιείται μέσω ενός endpoint που λαμβάνει το μήνυμα του χρήστη και επιστρέφει την απάντηση του LLM. Μέσω του app.route στέλνονται τα δεδομένα στο Flask server από τη μέθοδο Post του chat.py (response = requests.post(url, json=data)).

Στη συνάρτηση take_a_response, τα δεδομένα λαμβάνονται και δημιουργείται ένα λεξικό-dictionary, που ονομάζεται data ώστε να είναι προσβάσιμο το αρχείο json. Έπειτα επιστρέφεται το μήνυμα που λήφθηκε.

```
@app.route('/take_a_response', methods=['POST'])
def take_a_response():  #function to take the response from the llm as a text message
    data = request.get_json()
    message = data.get('message')
    response = run_the_llm(message)
    return jsonify({'response': response})
```

[14] https://realpython.com/python-subprocess/
[15] https://stackoverflow.com/questions/39563802/subprocess-calledprocesserror-what-is-the-error
[16] https://stackoverflow.com/questions/40108816/python-running-as-windows-service-oserror-winerror-6-the-handle-is-invalid

Τέλος, το app.run ξεκινάει τη λειτουργεία του τοπικού Flask server. Το debugging χρησιμοποιείται ώστε να μπορούν να εμφανιστούν errors που μπορεί να προκύψουν, αλλά και για να ξεκινήσει ο server εκ νέου, σε περίπτωση που τροποποιηθεί ο κώδικας όσο τρέχει.

```
if __name__ == '__main__':
        app.run(debug=True, host='127.0.0.1', port=5010)
```

Υλοποίηση Διεπαφής Συνομιλίας

Η διεπαφή συνομιλίας αναπτύχθηκε στο αρχείο chat.py και χρησιμοποιεί τη βιβλιοθήκη Pygame για τη διαχείριση των γραφικών, καθώς και τις requests για την επικοινωνία με το API.

Βιβλιοθήκες που χρησιμοποιούνται:

- **Pygame:** module για τη δημιουργία γραφικών και διεπαφών για παιχνίδια σε Python
- **requests:** επιτρέπει την επικοινωνία του LLM με το API μέσω HTTP αιτημάτων.
- **textwrap:** επιτρέπει την αναδίπλωση κειμένου

Επεξεργασία Εισόδου και Πλοήγηση στη Συνομιλία

Η κύρια λογική περιλαμβάνει μία while λούπα, μέσα από την οποία ο χρήστης μπορεί να πληκτρολογήσει το κείμενο που θέλει να στείλει, ώστε να επικοινωνήσει με το LLM. Επιπλέον μπορεί με τη χρήση των up και down arrow keys να γίνει scroll στο τυπωμένο κείμενο, ώστε ο χρήστης να μπορεί να περιηγηθεί στη συνομιλία.

```
while not done:
        if event.type == pygame.KEYDOWN:
            if active:
                if event.key == pygame.K_RETURN:# enter key is pressed
                    response = llm_response(input_text)
                    messages.append(("Me   -> ", input_text))
                    messages.append(("LLM -> ", response))
                    input_text = '' # clear the input box
                elif event.key == pygame.K_BACKSPACE:
                    input_text = input_text[:-1]
                elif event.key == pygame.K_UP: # scroll up in input box
                    scroll_input += 10  #  scroll speed
                elif event.key == pygame.K_DOWN:
                    scroll_input -= 10  # scroll speed
                else:
                    input_text += event.unicode
            if not active:
                if event.key == pygame.K_UP:  # scroll up in messages
                    scroll_messages += 10  #  scroll speed
              elif event.key == pygame.K_DOWN:# scroll down in messages
                    scroll_messages -= 10  # scroll speed
        if event.type == pygame.MOUSEWHEEL:
            scroll_messages += 10*event.y  # mouse scroll speed
```

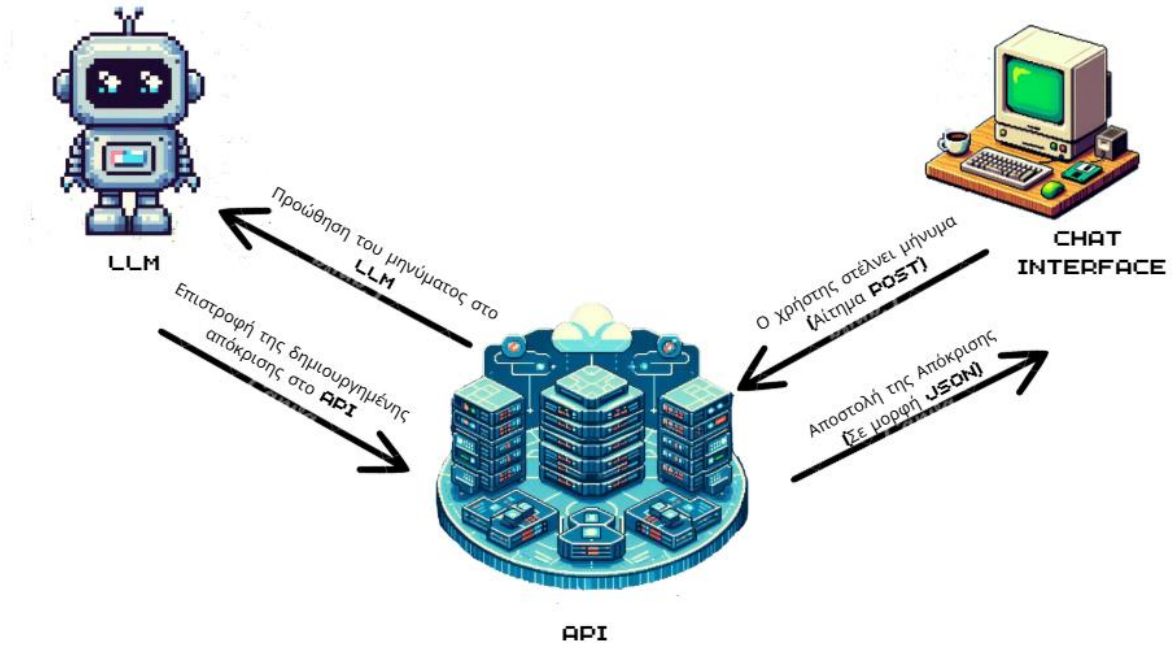
Αποστολή Αιτημάτων στο API

Η llm_response είναι μία συνάρτηση μέσω της οποίας λαμβάνεται η απάντηση που στέλνει το LLM στο χρήστη, με τη χρήση του URL από το api.py. Έπειτα μέσω των requests και σε συνδυασμό με το post, στέλνεται ένα αίτημα, ώστε τα δεδομένα και το μήνυμα του χρήστη να σταλεί στο server και να ληφθεί μία απάντηση. Η απάντηση αυτή καταχωρείται με τη βοήθεια του json.

```
def llm_response(message):
    url = "http://127.0.0.1:5010/take_a_response"
    data = {'message': message}
    print("message:", message)
    response = requests.post(url, json=data)
    response_json = response.json()
    print("Response:", response_json['response'])
    # return the llms response
    return response_json['response']
```

Η επικοινωνία μεταξύ της διεπαφής συνομιλίας και του LLM μέσω του API ολοκληρώνεται με επιτυχία, παρέχοντας έναν διαδραστικό τρόπο για την αλληλεπίδραση του χρήστη με το γλωσσικό μοντέλο.

Στην παρακάτω εικόνα, αναπαρίσταται η σύνδεση και η επικοινωνία του LLM με το chat interface μέσω του API.



Εικόνα 21. Σύνδεση LLM και Chat μέσω API

4.2. Δεύτερη Φάση Ανάπτυξης – Έκδοση 2 της εφαρμογής

Στη δεύτερη φάση της υλοποίησης, στόχος ήταν η ενσωμάτωση του συστήματος συνομιλίας (chat) σε ένα περιβάλλον παιχνιδιού. Το LLM έπρεπε να εξειδικευτεί σε συγκεκριμένες θεματικές, ώστε να επιτυγχάνεται στοχευμένη αλληλεπίδραση. Για τον σκοπό αυτό, δημιουργήθηκαν δύο νέα αρχεία: το level.py και το topics.json. Το πρώτο αποτελεί το πρώιμο περιβάλλον του παιχνιδιού, ενώ το δεύτερο αποθηκεύει τα διαθέσιμα θέματα γνώσης.

Περιβάλλον παιχνιδιού

Το περιβάλλον του παιχνιδιού σχεδιάστηκε ως ένας χάρτης διαστάσεων 1000x1000 pixels, ο οποίος αποτελείται από πλακίδια (tiles). Η τοποθέτηση των tiles γίνεται τυχαία, ώστε κάθε φορά που ξεκινά εκ νέου το παιχνίδι, ο χάρτης να μην είναι πανομοιότυπος. Η δημιουργία και απόδοση του χάρτη επιτυγχάνεται μέσω ενός εμφωλευμένου βρόχου for, ο οποίος υπολογίζει τις θέσεις των tiles βάσει της θέσης του παίκτη και του πεδίου ορατότητας (field of view).

```
     #draw tiles in map with nested loop for x and y range
    for y in range(first_y_pos, last_y_pos):
        for x in range(first_x_pos, last_x_pos):
#calculate screen positions based on players position and field of view for each tiles x and y
coordinate
            screen_x = (x - offset_x) * tile_size
            screen_y = (y - offset_y) * tile_size
            #draw each tile
            pygame.draw.rect(screen, tilemap[y][x], (screen_x, screen_y, tile_size, tile_size))
```

Πεδίο ορατότητας – Field of view

Το πεδίο ορατότητας (field of view) έχει οριστεί σε 30 x 20 πλακίδια (tiles). Το κάθε πλακίδιο έχει μέγεθος 40 pixels. Οι διαστάσεις της οθόνης του παιχνιδιού έχουν οριστεί 1200 x 800, μέγεθος που ταιριάζει ικανοποιητικά στις περισσότερες οθόνες.

```
# set the size of each tile
tile_size = 40
# set the size of the map
map_size = 1000
#set screen dimensions
screen_width, screen_height = 1200, 800
#set the field of view dividing with the use of // to have an integer as a result. It represents
the amount of tiles displayed on x and y axis
field_of_view_x = screen_width // tile_size
field_of_view_y = screen_height // tile_size
```

Η θέση του παίκτη καθορίζει το οπτικό του πεδίο, δημιουργώντας την αίσθηση ότι η κάμερα τον ακολουθεί.

```
    #setting positions within the tilemap based on field of view
    #first_x_pos and first_y_pos won't be less than 0
    first_x_pos = max(0, player_pos[0] - field_of_view_x // 2)
    first_y_pos = max(0, player_pos[1] - field_of_view_y // 2)
    #last_x_pos and last_y_pos won't exceed map_size
    last_x_pos = min(map_size, player_pos[0] + field_of_view_x // 2)
    last_y_pos = min(map_size, player_pos[1] + field_of_view_y // 2)
    #calculate the offset based on the player's position to adjust it with the field of view
    offset_x = player_pos[0] - field_of_view_x // 2
    offset_y = player_pos[1] - field_of_view_y // 2
    #draw tiles in map with nested loop for x and y range
    for y in range(first_y_pos, last_y_pos):
        for x in range(first_x_pos, last_x_pos):
#calculate screen positions based on players position and field of view for each tiles x and y
            screen_x = (x - offset_x) * tile_size
            screen_y = (y - offset_y) * tile_size
            #draw each tile
            pygame.draw.rect(screen, tilemap[y][x], (screen_x, screen_y, tile_size, tile_size))
    # draw the player based on his movement
    player_screen_x = (player_pos[0] - offset_x) * tile_size
    player_screen_y = (player_pos[1] - offset_y) * tile_size
    pygame.draw.rect(screen,  player_color,  (player_screen_x,  player_screen_y,  tile_size,
tile_size))
```

NPCs (Μη παικτικοί χαρακτήρες)

Στο περιβάλλον του παιχνιδιού προστέθηκαν δύο NPCs, καθένας από τους οποίους εξειδικεύεται σε ένα γνωστικό αντικείμενο:

- Ο πρώτος NPC είναι ειδικός στα Μαθηματικά.
- Ο δεύτερος NPC έχει γνώση της Γεωγραφίας.

Η θέση και η θεματική τους αποθηκεύονται σε ένα λεξικό.

```
#a dictionary with npcs information
npcs = {
    "npc1":{"name": "Npc1", "pos":[player_pos[0]+1, player_pos[1]],
    "color": pygame.Color('yellow'),"topic": "mathematics"},
    "npc2":{"name":"Npc2","pos":[player_pos[0]     -     1,     player_pos[1]],     "color":
pygame.Color('green'), "topic": "geography"}}
```

Αντικείμενα Γνώσης – Knowledge topics

Τα αντικείμενα γνώσης (knowledge topics) φορτώνονται από ένα εξωτερικό αρχείο JSON, το οποίο περιλαμβάνει όλες τις σχετικές πληροφορίες. Αυτό εξασφαλίζει την καλύτερη διαχείριση των θεματικών, καθώς επιτρέπει εύκολη προσθήκη νέων ή επεξεργασία των ήδη υπάρχουσων.

```
    "topics": ["mathematics","geography"]
```

Οι θεματικές φορτώνονται από το json αρχείο και μετατρέπονται σε ένα λεξικό, ώστε να μπορούν να χρησιμοποιηθούν μέσα στο py αρχείο. Έπειτα το λεξικό χρησιμοποιήθηκε και στο api.py και στο chat.py. Με αυτό τον τρόπο το LLM, δηλαδή ο NPC, πλέον μπορούσε να συζητάει αποκλειστικά για συγκεκριμένη θεματική.

```
# load the topics from the py by opening the topic.json file (file is assigned to variable f)
with open('topics.json') as f:
    #read the json file and convert it into a dictionary
    topics = json.load(f)["topics"]
# initialize the knowledge topic
knowledge_topic = topics[0]
```

Διαχείριση Θεματικών μέσω API

Το chat.py στέλνει στο API την επιλεγμένη θεματική μέσω της συνάρτησης set_knowledge_topic.

```
        #function to set the knowledge topic
        def set_knowledge_topic(topic):
            # the url to send the topic to api.py
            url = "http://127.0.0.1:5010/get_knowledge_topic"
            # take the current topic as data
            data = {'topic': topic}
            #use requests to send http requests in py, post to send the
            data/users message on the server to get a response
            response = requests.post(url, json=data)
            # take the llms response
            response_json = response.json()
            # print the response in the terminal
            print("Received knowledge topic from level:", topic)
            print(f"Set topic response: {response_json}")
            return response_json
```

api.py

Στο api.py, η συνάρτηση check_topic ελέγχει αν η ερώτηση του χρήστη σχετίζεται με την τρέχουσα θεματική. Στην llm_run δίνεται μία εντολή μέσω prompt στο LLM, μέσω της οποίας λαμβάνει οδηγίες σχετικά με το αντικείμενο γνώσης που θα κατέχει.

Αξίζει να σημειωθεί, ότι στην περίπτωση που ο χρήστης ζητούσε από το LLM να μετρήσει, αυτό το έκανε ακόμη και αν η θεματική δεν ήταν το αντικείμενο των Μαθηματικών. Συνεπώς, χρειάστηκε να δοθεί μία περαιτέρω εντολή μέσω prompt : If message is asking you to count, do it only if knowledge_topic is Mathematics{message}.

```
        #check if the question is related to the current knowledge topic
        def check_topic(message):
            try:
# inform the llm about the rules that must be followed.
                llm_run = ['ollama', 'run', 'llama3:latest', f"Is the following question or
ability clearly and directly related only to {knowledge_topic}?Answer with yes if it is relevant
else dont use it in your answer or no if not relevant. If message is asking you to count, do it
only if knowledge_topic is Mathematics. {message}"]
#subprocess is used to connect an exe running on cmd with a py file
                output       =       subprocess.check_output(llm_run,       encoding='utf-8',
stdin=subprocess.DEVNULL)
                print("Output is",message,output )
#The llm is asked to answer yes or no, to determine if users question is relevant to the current
topic
#With the use of strip case-sensitivity was observed. Lower fixed that
                return "yes" in output.lower()
            # exception for stdout stderr error occurred and print the error
            except subprocess.CalledProcessError as e:
                print("Error running LLM :", e)
                return "Error running LLM"
```

Στη συνάρτηση take_a_response γίνεται έλεγχος, ώστε να διαπιστωθεί αν το περιεχόμενο του μηνύματος που έστειλε ο χρήστης είναι σχετικό με τη θεματική που έχει οριστεί. Σε περίπτωση που σχετίζονται το LLM στέλνει την απάντησή του, αλλιώς απαντάει με ένα προκαθορισμένο μήνυμα, το οποίο ενημερώνει τον χρήστη, σχετικά με ποιο αντικείμενο μπορεί να πραγματοποιηθεί η συζήτηση.

```
             if check_topic(message):
                #take a response from llm using the function running it
                response = run_the_llm(message)
            else:
#setting a default response if question of user is not related to the current knowledge topic
                response = f"I can only answer questions related to {knowledge_topic}."
            print("response:", response)
          # return the llms response as a json file to be received by user
            return jsonify({'response': response})
```

Έπειτα μέσω του app.route λαμβάνεται η post εντολή της συνάρτησης set_knowlege_topic του chat.py. Η μεταβλητή knowledge_topic ορίζεται ως global και μέσω του λεξικού data γίνεται πρόσβαση στο αρχείο json. Στη συνέχεια το knowledge_topic ταυτίζεται με τη θεματική/topic που ορίστηκε στη αρχή.

```
@app.route('/get_knowledge_topic', methods=['POST'])
def get_knowledge_topic():
    # make sure knowlwdge_topic is a global variable
    global knowledge_topic
    # take the data received and create a dict named data to access json file
    data = request.get_json()
    # returns the message received
    topic = data.get('topic')
    #check if topic exists in available topics
    if topic in topics:
        knowledge_topic = topic
        print("Received knowledge topic:", knowledge_topic)
        return jsonify({'message': f'Knowledge topic updated to {knowledge_topic}'})
    else:
        return jsonify({'error': 'Invalid topic provided'})
```

Συνομιλία (Chat) στο παιχνίδι

Η συνομιλία (chat) εμφανίζεται σε μια ειδική επιφάνεια (surface) μέσα στο παράθυρο του παιχνιδιού. Αυτή η επιφάνεια ενεργοποιείται και απενεργοποιείται κάνοντας κλικ πάνω σε κάποιον NPC.

```
chat_surface = pygame.Surface((400, 600))
# set the transparency so the game screen can be semi-visible behind chat_surface
chat_surface.set_alpha(200)
```

Μόλις ο χρήστης πατήσει το κουμπί του ποντικιού, το πρόγραμμα ανιχνεύει το συμβάν MOUSEBUTTONDOWN και αποθηκεύει τις συντεταγμένες του σημείου όπου έγινε το κλικ. Στη συνέχεια, γίνεται προσπέλαση σε όλους τους NPCs που υπάρχουν στο παιχνίδι και οι συντεταγμένες τους στο παιχνίδι (global coordinates) μετατρέπονται σε συντεταγμένες οθόνης (screen coordinates), λαμβάνοντας υπόψη τη θέση του παίκτη και το οπτικό του πεδίο (field of view). Το tile_size αντιπροσωπεύει το μέγεθος ενός τετραγώνου στο grid του παιχνιδιού, και μέσω αυτού ελέγχεται αν το κλικ έγινε μέσα στα όρια του πλακιδίου όπου βρίσκεται ο NPC. Αν το chat ήταν ανενεργό, ενεργοποιείται, ενώ αν ήταν ήδη ενεργό, απενεργοποιείται. Παράλληλα, αποθηκεύονται τα δεδομένα του NPC, όπως το όνομά του και το θέμα γνώσης που θα συζητηθεί, ενώ η set_knowledge_topic (knowledge_topic) καλείται για να ενημερώσει τη θεματική του διαλόγου. Τέλος, αν ο χρήστης έκανε κλικ και απενεργοποίησε το chat, καλείται η restart_chat() ώστε να γίνει επανεκκίνηση της συνομιλίας εφόσον αυτή ενεργοποιηθεί ξανά.

```
        elif event.type == pygame.MOUSEBUTTONDOWN:
            #get the mouse clicking coordinates
            mouse_x, mouse_y = event.pos
            #check if clicking is on the same position as the npc
            for npc_key, npc in npcs.items():
                npc_screen_x = (npc["pos"][0] - (player_pos[0] -
                field_of_view_x // 2)) * tile_size
                npc_screen_y = (npc["pos"][1] - (player_pos[1] -
                field_of_view_y // 2)) * tile_size
                if (npc_screen_x <= mouse_x <= npc_screen_x +
                tile_size) and (npc_screen_y <= mouse_y <=
                 npc_screen_y + tile_size):
                    #activate and deactivate chat on npc clicking
                    chat_active = not chat_active
                    #set the current npcs information
                    current_npc_name = npc["name"]
                    knowledge_topic = npc["topic"]
                    set_knowledge_topic(knowledge_topic)
                    print("The knowledge topic:", knowledge_topic)
                    if not chat_active:
                        restart_chat()
```

Μόλις ενεργοποιηθεί αυτή η επιφάνεια το level.py λαμβάνει όλα τα δεδομένα από το chat.py. Έτσι ο παίκτης έχει πλέον τη δυνατότητα να αλληλεπιδράσει με τους NPCs και να συνομιλήσει μαζί του σχετικά με τα αντικείμενα γνώσης τους.

```
    if chat_active:
        active = True
        chat_surface.fill((0, 0, 0, 0))
        input_text,   active,   run_chat.scroll_messages,   run_chat.scroll_input,   done   =
run_chat(chat_surface, events, input_text, active, done, current_npc_name, knowledge_topic)
        screen.blit(chat_surface, (screen_width - 400, screen_height - 600))
        # print("Received knowledge topic:", knowledge_topic)
    else:
        restart_chat()
```

Κίνηση Παίκτη

Ο παίκτης μπορεί να κινείται ελεύθερα μέσα στα όρια του επιπέδου, εκτός αν η συνομιλία είναι ενεργή. Όταν η συνομιλία είναι ενεργοποιημένη, η κίνηση του παίκτη παγώνει, ώστε να διατηρηθεί η στην αλληλεπίδραση.

```
    if not chat_active:
        if keys[pygame.K_UP] and player_pos[1] > 0:
            player_pos[1] -= 1
        if keys[pygame.K_DOWN] and player_pos[1] < map_size - 1:
            player_pos[1] += 1
        if keys[pygame.K_LEFT] and player_pos[0] > 0:
            player_pos[0] -= 1
        if keys[pygame.K_RIGHT] and player_pos[0] < map_size - 1:
            player_pos[0] += 1
```

Στο δεύτερη φάση υλοποίησης, ενσωματώθηκε επιτυχώς η δυνατότητα διαλόγου με NPCs εντός του παιχνιδιού, με θεματική εξειδίκευση. Η χρήση αρχείου JSON διευκολύνει τη διαχείριση των θεμάτων, ενώ το API διασφαλίζει τον έλεγχο των σχετικών συζητήσεων.

4.3. Τρίτη Φάση Ανάπτυξης – Έκδοση 3 της εφαρμογής

Στην τρίτη φάση της ανάπτυξης του παιχνιδιού, πραγματοποιήθηκαν σημαντικές βελτιώσεις και επεκτάσεις προκειμένου να ενισχυθεί η λειτουργικότητα και η εμπειρία του χρήστη. Η κύρια εστίαση ήταν στην ενσωμάτωση ενός διαδραστικού συστήματος ερωτήσεων και απαντήσεων, τη διαχείριση δεδομένων μέσω βάσης δεδομένων, και την προσαρμογή της δυσκολίας των ερωτήσεων ανάλογα με το προφίλ του χρήστη. Στην παρούσα ενότητα αναλύονται οι βασικές αλλαγές και οι λειτουργίες που προστέθηκαν στο σύστημα, καθώς και η τεχνολογική υποδομή που τις υποστηρίζει.

Ερωτήσεις και Απαντήσεις

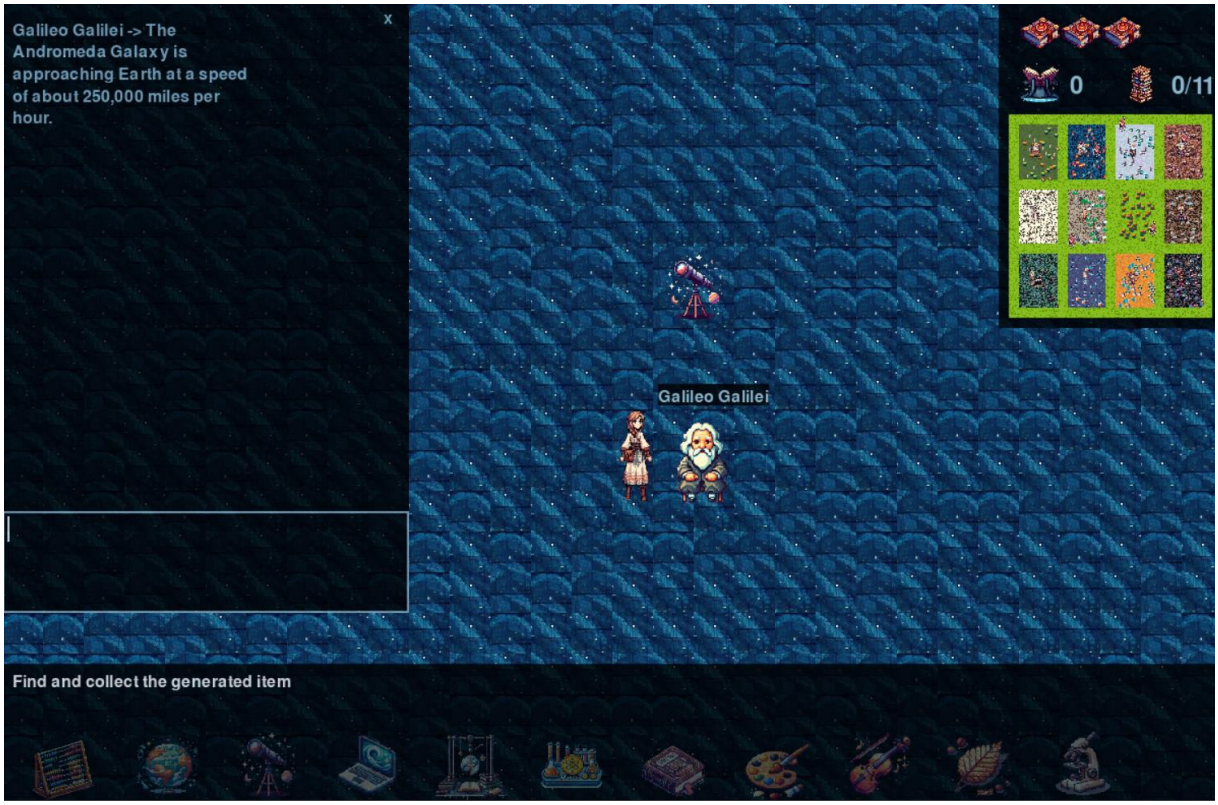
Η αλληλεπίδραση με τον χρήστη βελτιώθηκε με την ενσωμάτωση μιας νέας λειτουργίας για τις ερωτήσεις και τις απαντήσεις, με σκοπό την καλύτερη εξατομίκευση της εμπειρίας του χρήστη. Στο παρελθόν, η συζήτηση ξεκινούσε από τον χρήστη, αλλά πλέον, το LLM αναλαμβάνει την εκκίνηση της συνομιλίας, ζητώντας από το χρήστη να συμφωνήσει στη διεξαγωγή της συζήτησης.

Συγκεκριμένα, όταν ο χρήστης αποδέχεται να δεχθεί ερωτήσεις (δεν έχει ορίσει δηλαδή τον αριθμό των ερωτήσεων ως μηδενικό, τότε το LLM του συστήνεται με το όνομα του εκάστοτε NPC και του ζητάει να επιβεβαιώσει πως είναι έτοιμος να ξεκινήσει η συζήτηση.

```
#set the initial message user receives in case there are questions
    if not initial_response_sent and initial_num_questions >0 and current_num_questions >0:
        response = f"My name is {knowledge_npc}. I know everything about
{knowledge_topic}.Please, click in the input box and press enter to start the conversation."
  #set the initial response flag to true so this message won't appear again
        initial_response_sent = True
```

Επιπλέον, σε περίπτωση που ο χρήστης έχει ορίσει το πλήθος των ερωτήσεων ως μηδενικό (δηλαδή δεν επιθυμεί να απαντήσει σε ερωτήσεις), το LLM τον ενημερώνει ότι δεν απαιτείται να συνεχίσει τη συνομιλία, παρέχοντας του μια γενική πληροφορία για το εκάστοτε θέμα.

```
    elif not initial_response_sent and initial_num_questions <=0 and current_num_questions== 0:
        no_more_questions = True
        fun_fact_text = fun_fact(knowledge_topic)
        # if a fun fact is generated respond the fun fact
        if fun_fact_text:
            response = fun_fact_text
        else:
            response = "No need to talk."
        initial_response_sent = False
```

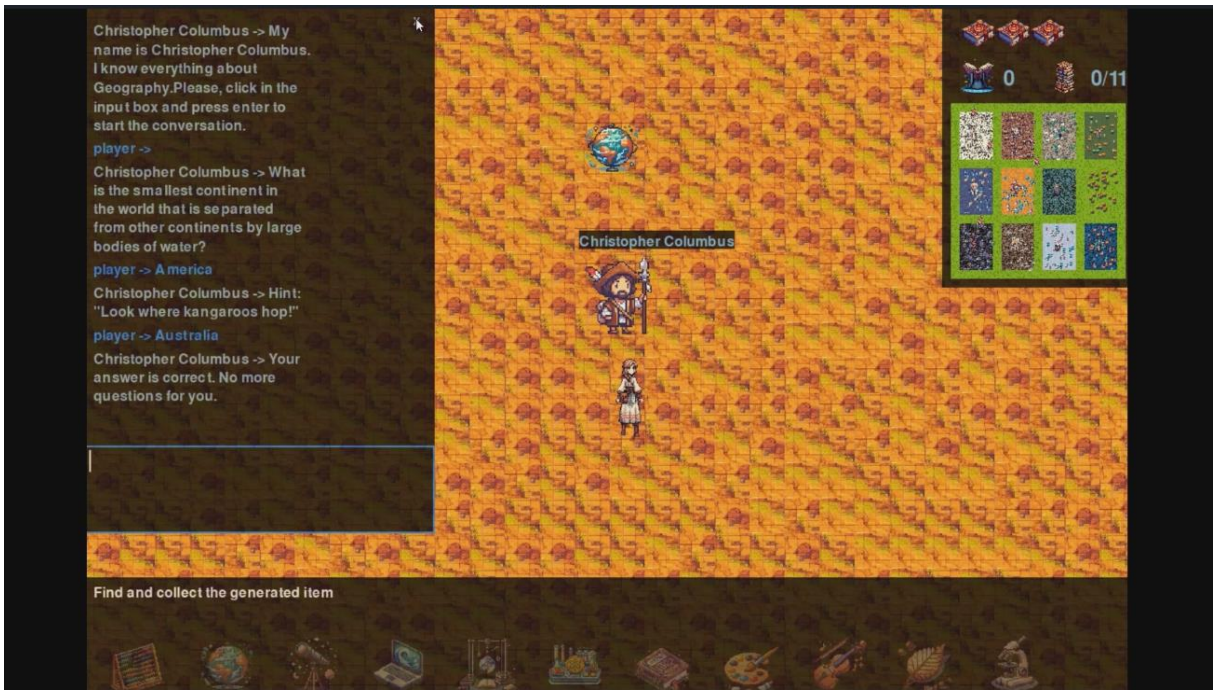


Εικόνα 22.Δημιουργία fun fact

Τέλος ελέγχεται αν εν τέλη ο χρήστης έχει απαντήσει σε όλες τις ερωτήσεις και λαμβάνει το ανάλογο μήνυμα.

```
                question_id,stored_question,correct_answer=get_last_question(knowledge_topic)
        if not stored_question or current_num_questions ==0:
            response = "I don't have any more questions for you."
            follow_up_question_sent = False
        else:
            _, response_message = check_user_answer(message, correct_answer, user_id)
            response = response_message
```

Δυναμική Δημιουργία Ερωτήσεων



Εικόνα 23.Διεξαγωγή διαλόγου

Με την get_unasked_question το LLM αντλεί από τη βάση δεδομένων τις αποθηκευμένες ερωτήσεις και τις απαντήσεις τους και μέσω prompt του δίνεται η εντολή να θέσει μία πρωτότυπη ερώτηση. Η ανάγκη για αυτή τη λειτουργία προέκυψε από το γεγονός ότι χωρίς αυτήν, το LLM είτε επαναλάμβανε την ίδια ερώτηση είτε την παράφραζε οδηγώντας στην ίδια απάντηση με την προηγούμενη. Η συνάρτηση get_answered_questions (user_id, knowledge_topic) ανακτά όλες τις ερωτήσεις που έχει ήδη απαντήσει ο χρήστης για το συγκεκριμένο θέμα και τις μετατρέπει στη λίστα answered_texts, ώστε να δοθούν στο LLM ως αναφορά για την αποφυγή επανάληψης. Παράλληλα, η get_last_user_age() επιστρέφει την τελευταία καταχωρημένη ηλικία του χρήστη, επιτρέποντας στο LLM να προσαρμόσει τη νέα ερώτηση στο κατάλληλο επίπεδο δυσκολίας. Το μοντέλο λαμβάνει την οδηγία να δημιουργήσει μια μοναδική ερώτηση και απάντηση σχετική με το knowledge_topic, αποφεύγοντας να επαναλάβει προηγούμενες ερωταπαντήσεις. Στη συνέχεια, εντοπίζεται η ερώτηση και η απάντηση στην έξοδο του μοντέλου και απομονώνεται το περιεχόμενό τους, αφαιρώντας περιττές πληροφορίες. Αν η παραγόμενη ερώτηση και απάντηση δεν υπάρχει ήδη στη λίστα των απαντημένων, επιστρέφεται ως νέα. Διαφορετικά, επιστρέφεται None, None. Σε περίπτωση σφάλματος κατά την εκτέλεση του LLM, το πρόγραμμα εμφανίζει μήνυμα λάθους και επιστρέφει None, None.

```
def get_unasked_question(user_id, knowledge_topic):
    answered_questions =get_answered_questions(user_id, knowledge_topic)
    answered_texts = [f"Question: {question} Answer: {answer}" for question, answer in
answered_questions]
    user_age = get_last_user_age()
    llm_prompt = (f"Generate a unique question and answer related to {knowledge_topic} suitable
for age {user_age}. "
                  f"Ensure that the generated question and answer do not repeat the following:
{', '.join(answered_texts)} "
                  f"Provide the question starting with 'question:' and the correct answer
starting with 'answer:'.")
    try:
        llm_run = ['ollama', 'run', 'llama3:latest', llm_prompt]
        output = subprocess.check_output(llm_run, encoding='utf-8', stdin=subprocess.DEVNULL)
        print("Output is", output)
  #set the position of what is printed as answer and question without extra information
        question_start_index = output.lower().find('question:')
        answer_start_index = output.lower().find('answer:')
        if question_start_index != -1 and answer_start_index != -1:
            question=output[question_start_index+len('question:'):answer_start_index].strip()
            answer = output[answer_start_index + len('answer:'):].strip()
            if (question, answer) not in answered_questions:
                print(f"Generated New Question: {question}")
                print(f"Generated New Answer: {answer}")
                return question, answer
        return None, None
    except subprocess.CalledProcessError as e:
        print("Error generating new question:", e)
        return None, None
```

Έλεγχος Σημασιολογικής Ομοιότητας Απαντήσεων

Η αναγνώριση σωστής απάντησης πλέον δεν περιορίζεται στην αυστηρά πανομοιότυπη διατύπωση της ήδη αποθηκευμένης. Το LLM ελέγχει τη σημασιολογική ομοιότητα μεταξύ της απάντησης που έδωσε ο χρήστης και αυτής που αποθηκεύτηκε ως ορθή στη βάση δεδομένων. Αυτή η διαδικασία αποδείχθηκε αναγκαία, καθώς η μοναδική απάντηση που θεωρούταν ορθή ήταν η κατά γράμμα απόδοση αυτής που αποθηκεύτηκε στη βάση, συμπεριλαμβανομένων των σημείων στίξης. Πλέον σωστή απάντηση μπορεί να θεωρηθεί, οποιαδήποτε απάντηση έχει την ίδια σημασία και το ίδιο νόημα με την αποθηκευμένη.

```
def check_meaning_with_llm(user_answer, correct_answer, user_id):
    global no_more_questions
    llm_prompt = (f"Determine if the following user answer is semantically equivalent to the
correct answer. "
                  f"User Answer: '{user_answer}' "
                  f"Correct Answer: '{correct_answer}' "
                  f"Answer with 'Yes' or 'No' and provide an explanation if needed. "
                  f"Note: Both answers should be considered correct if they convey the same
meaning even if the formatting is different.")
    try:
        llm_run = ['ollama', 'run', 'llama3:latest', llm_prompt]
        output = subprocess.check_output(llm_run, encoding='utf-8', stdin=subprocess.DEVNULL)
```

Παροχή Βοηθητικών Μηνυμάτων

Όταν ο χρήστης δεν απαντά σωστά, το LLM είναι υπεύθυνο για την παροχή βοηθητικών μηνυμάτων, τα οποία καθοδηγούν τον χρήστη προς τη σωστή απάντηση χωρίς να την αποκαλύπτουν άμεσα. Αυτό ενισχύει τον εκπαιδευτικό χαρακτήρα του παιχνιδιού, δίνοντας στον χρήστη την ευκαιρία να σκεφτεί και να βρει τη σωστή απόκριση με βάση τις παρεχόμενες υποδείξεις. Η get_last_user_age() επιστρέφει την τελευταία καταχωρημένη ηλικία του χρήστη, ενώ αν δεν υπάρχει διαθέσιμη πληροφορία στη βάση δεδομένων, εμφανίζεται σχετικό μήνυμα και η συνάρτηση επιστρέφει False και κενές τιμές. Το βοηθητικό μήνυμα είναι προσαρμοσμένο στην ηλικία του χρήστη, ώστε να είναι κατάλληλο για το επίπεδό του. Το παραγόμενο μήνυμα καθαρίζεται από τυχόν περιττά κενά πριν επιστραφεί. Αν προκύψει σφάλμα κατά την εκτέλεση του LLM, εμφανίζεται μήνυμα λάθους και επιστρέφεται η προεπιλεγμένη απάντηση "Couldn't generate a hint.", διασφαλίζοντας έτσι την ομαλή λειτουργία του.

```
def generate_hint(user_answer, correct_answer):
    try:
        user_age = get_last_user_age()
        if user_age is None:
            print("No user age found in the database.")
            return False, "", ""
        llm_prompt = (f"Provide a helpful hint in a few words, suitable for age {user_age} to
guide the user towards the correct answer, without commenting or checking what you did, just
provide the hint. "
       f"The user's answer is '{user_answer}', and the correct answer is '{correct_answer}'. "
       f"Ensure that the hint leads the user closer to the correct answer without revealing it."
       f"Make sure to provide guidance on understanding the context of the answer.")
       print(f"Generated LLM prompt: {llm_prompt}")
       llm_run = ['ollama', 'run', 'llama3:latest', llm_prompt]
       output = subprocess.check_output(llm_run, encoding='utf-8', stdin=subprocess.DEVNULL)
       hint = output.strip()
       print(f"Generated Hint: {hint}")
       return hint
   except subprocess.CalledProcessError as e:
       print(f"Error generating hint: {e}")
       return "Couldn't generate a hint."
```

Με τη χρήση app.route η set_no_more_questions, ενημερώνει το chat και μηδενίζει τον αριθμό των ερωτήσεων, ώστε να είναι δυνατός ο συγχρονισμός της μεταβλητής στο αρχείο chat.py.

```
@app.route('/set_no_more_quest', methods=['POST'])
def set_no_more_quest():
    global knowledge_topic, knowledge_npc, initial_response_sent, follow_up_question_sent,
no_more_questions
    data = request.get_json()
    message = data.get('message')
    user_id = get_last_user_id()
    if user_id is not None:
        print(f"userid: {user_id}")
    else:
        print("No users in db.")
    initial_num_questions = get_initial_num_questions(user_id, knowledge_topic)
    num_questions = get_user_question_count(user_id, knowledge_topic)
    print(f"num_questions {num_questions}")
    if initial_num_questions <=0:
        no_more_questions = True
    if num_questions <=0:
        no_more_questions = True
    return jsonify({'no_more_questions': no_more_questions})
```

Βάση Δεδομένων

Ο τρόπος αποθήκευσης και διαχείρισης των δεδομένων αναβαθμίστηκε. Το JSON αντικαταστάθηκε με μία βάση δεδομένων (DB) sqlite3. Τα θέματα και τα δεδομένα τους δεν φορτώνονται πλέον από το αρχείο topics.json αλλά από μια βάση δεδομένων db.py. Αυτό παρέχει μεγαλύτερη αποτελεσματικότητα και καλύτερο συγχρονισμό των στοιχείων του παιχνιδιού.

Πίνακας topics: Αποθηκεύει τα διαθέσιμα θέματα γνώσης και το όνομα του NPC που τα εκπροσωπεί. Κάθε θέμα γνώσης έχει ένα μοναδικό όνομα και συνδέεται με έναν NPC, ο οποίος αναλαμβάνει την παρουσίαση των ερωτήσεων.

```
import sqlite3
def init_db():
    conn = sqlite3.connect('topics.db')
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS topics (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            topic_name TEXT UNIQUE NOT NULL,
            npc_name TEXT NOT NULL
        )
    ''')
```

Στον πίνακα topics καταχωρήθηκαν οι προκαθορισμένες θεματικές και τα ονόματα των NPCs.

```
    cursor.execute('''
        INSERT OR IGNORE INTO topics (topic_name, npc_name) VALUES
        ('mathematics', 'Archimedes'),
        ('geography', 'Magellan')
    ''')
```

Πίνακας questions: Αποθηκεύει τις ερωτήσεις που τίθενται από το LLM στον χρήστη και τις αντίστοιχες απαντήσεις τους. Κάθε ερώτηση συνδέεται με ένα συγκεκριμένο θέμα, διευκολύνοντας την ανάκτηση σχετικών δεδομένων ανάλογα με το επιλεγμένο θέμα από τον χρήστη.

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS questions (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        topic TEXT NOT NULL,
        question TEXT NOT NULL,
        answer TEXT NOT NULL)''')
```

Πίνακας users: Αποθηκεύει τα δεδομένα των χρηστών, όπως το όνομα και την ηλικία τους. Οι πληροφορίες αυτές χρησιμοποιούνται για να εξατομικευθεί η εμπειρία του χρήστη, όπως η επιλογή της δυσκολίας των ερωτήσεων.

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_name TEXT NOT NULL,
        user_age INTEGER NOT NULL
    )
''')
```

Πίνακας user_topics: Διαχειρίζεται τη σχέση μεταξύ χρηστών και θεμάτων, καθώς και τον αριθμό των ερωτήσεων που επιθυμεί ο χρήστης να λάβει για κάθε θέμα. Αυτός ο πίνακας εξασφαλίζει την εξατομίκευση της εμπειρίας του παίκτη, επιτρέποντας στο σύστημα να προσαρμόζει τις ερωτήσεις που προτείνονται ανάλογα με τις προτιμήσεις του.

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS user_topics (
        user_id INTEGER,
        topic TEXT,
        num_questions INTEGER,
        initial_num_questions INTEGER,
        PRIMARY KEY (user_id, topic),
        FOREIGN KEY (user_id) REFERENCES users(id),
        FOREIGN KEY (topic) REFERENCES topics(topic_name)
    )
''')
```

Πίνακας user_questions: Εξασφαλίζει ότι δεν θα επαναλαμβάνονται οι ερωτήσεις για το ίδιο θέμα σε έναν χρήστη, αποθηκεύοντας ποια ερωτήματα έχουν ήδη απαντηθεί. Αυτός ο πίνακας είναι κρίσιμος για την αποφυγή επαναλήψεων και την ενίσχυση της δυναμικότητας του παιχνιδιού.

```
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS user_questions (
            user_id INTEGER,
            topic TEXT,
            question_id INTEGER,
            PRIMARY KEY (user_id, topic, question_id),
            FOREIGN KEY (user_id) REFERENCES users(id),
            FOREIGN KEY (topic) REFERENCES topics(topic_name),
            FOREIGN KEY (question_id) REFERENCES questions(id)
        )
    ''')
    conn.commit()
    conn.close()
init_db()
```

Ανάκτηση Δεδομένων από τη Βάση Δεδομένων

Οι περισσότερες λειτουργίες του παιχνιδιού (θέματα, δεδομένα, ερωτήσεις, αποτελέσματα) ελέγχονται από τη βάση δεδομένων για καλύτερη διαχείριση και συγχρονισμό. Έτσι δημιουργήθηκε το αρχείο db_utils.py, το οποίο περιλαμβάνει όλες τις απαραίτητες λειτουργίες, ώστε να διαμοιράζονται τα δεδομένα μεταξύ του παιχνιδιού και της βάσης δεδομένων.
Δημιουργήθηκε μία συνάρτηση, με την οποία μπορεί το παιχνίδι να αντλήσει τις πληροφορίες που αφορούν την κάθε θεματική και το όνομα του NPC της από τον πίνακα topics.

Αρχικά, πραγματοποιείται σύνδεση με τη βάση δεδομένων topics.db με την εντολή sqlite3.connect('topics.db'), χρησιμοποιώντας το with, ώστε να εξασφαλιστεί, ότι η σύνδεση θα κλείσει αυτόματα όταν παύσει να χρησιμοποιείται.

Έπειτα, δημιουργήθηκε ένα αντικείμενο cursor, το οποίο επιτρέπει την εκτέλεση εντολών μέσω sql αλλά και την ανάκτηση δεδομένων από τη db.

Στη συνέχεια, με την εντολή cursor.execute ζητούνται τα δεδομένα του πίνακα topics, τα οποία περιλαμβάνονται στις στήλες topic_name και npc_name.

Τα δεδομένα που λαμβάνονται από τη βάση αποθηκεύονται στη μεταβλητή topics, με τη χρήση λεξικού. Δημιουργείται ένα ζεύγος key: value. Το key (κλειδί) είναι μια μοναδική τιμή που χρησιμοποιείται για την αναφορά σε μια άλλη τιμή (value) μέσα σε ένα λεξικό. Το key είναι το row[0], όπου αποθηκεύεται το topic_name, και value είναι το row[1], όπου αποθηκεύεται το npc_name. Στο τέλος επιστρέφεται το λεξικό topics με τα παραπάνω keys και values.

```
def get_topics():
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        cursor.execute('SELECT topic_name, npc_name FROM topics')
        topics = {row[0]: row[1] for row in cursor.fetchall()}
    return topics
```

Στη συνάρτηση get_user_name, στόχος είναι η ανάκτηση του μεγαλύτερου user id, δηλαδή του τελευταίου χρήστη που καταχωρήθηκε.

Στο κομμάτι SELECT user_name FROM users

WHERE id = (SELECT MAX(id) FROM users),

με την ανάκτηση του τελευταίου id επιστρέφεται εν τέλη το όνομα του χρήστη που αφορά.

Με τη μέθοδο cursor.fetchone αποθηκεύεται το όνομα χρήστη στη μεταβλητή result.

Αν υπάρχει καταχώρηση ονόματος τότε επιστρέφεται result[0] δηλαδή το πρώτο στοιχείο, το user_name, ειδάλλως η τιμή που θα επιστραφεί είναι None.

```
def get_user_name():
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #get the user name from the user with the max id-last user
        cursor.execute('''
            SELECT user_name FROM users
            WHERE id = (SELECT MAX(id) FROM users)
        ''')
        result = cursor.fetchone()
        if result:
            #if there is a result return the user name
            return result[0]
        else:
            #return None if no user is found
            return None
```

Με τον ίδιο τρόπο επιστρέφεται και το ίδιο το id του χρήστη.

```
def get_last_user_id():
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        cursor.execute('SELECT MAX(id) FROM users')
        result = cursor.fetchone()
        if result and result[0] is not None:
            return result[0]
        else:
            return None
```

Η get_answered_questions επιλέγει τις στήλες question (ερώτηση) και answer (απάντηση) από τον πίνακα questions. Το topic και το user_id πρέπει να ταυτίζονται με τις ομώνυμες παραμέτρους τους. Μέσω του cursor.fetchall επιστρέφονται όλα τα αποτελέσματα ως tuples.

Χρησιμεύει στο να διατηρηθεί σωστά ο αριθμός των ερωτήσεων που απαντήθηκαν σε κάθε θεματική από το χρήστη.

```
def get_answered_questions(user_id, topic):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #select questions and answers for a specific topic
        cursor.execute('SELECT question, answer FROM questions WHERE topic=? AND id IN (SELECT
question_id FROM user_questions WHERE user_id=?)', (topic, user_id))
        #return all the results
        return cursor.fetchall()
```

Η save_question αποθηκεύει κάθε νέα ερώτηση που δημιουργεί το LLM και την απάντηση της στη βάση δεδομένων. Η συνάρτηση δέχεται τρεις παραμέτρους:

- topic: Το θέμα στο οποίο ανήκει η ερώτηση.
- question: Η ερώτηση.
- answer: Η απάντηση στην ερώτηση.

Η εντολή INSERT INTO χρησιμοποιείται για την προσθήκη δεδομένων στον πίνακα questions, ενώ τα "?" λειτουργούν ως θέσεις για τις παραμέτρους (topic, question, answer).
Τέλος επιστρέφεται το id της εγγραφής που μόλις προστέθηκε στον πίνακα μέσω του cursor.lastrowid.

```
def save_question(topic, question, answer):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #insert the new question and answer in the questions table
        cursor.execute('INSERT INTO questions (topic, question, answer) VALUES (?, ?, ?)',
(topic, question, answer))
        conn.commit()
        return cursor.lastrowid
```

Στη save_user_question ο χρήστης συνδέεται με την ερώτηση που απάντησε και αποθηκεύεται το id του και το id της ερώτησης στον πίνακα user_questions.

```
def save_user_question(user_id, question_id):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #insert in user_questions a new data row to associate user id with a question
        cursor.execute('INSERT INTO user_questions (user_id, question_id) VALUES (?, ?)',
(user_id, question_id))
        #commit to ensure saving data
        conn.commit()
```

Η get_last_question χρησιμοποιείται για την ανάκτηση της τελευταίας ερώτησης που έκανε το LLM στο χρήστη πάνω σε κάποια συγκεκριμένη θεματική. Η τιμή του topic ορίζεται ως παράμετρος "?", η οποία αντικαθίσταται από το περιεχόμενο του topic.
Η μέθοδος cursor.fetchone() επιστρέφει tuple με τα πεδία id, question και answer στη σειρά αν βρεθεί εγγραφή (if result) και None σε κάθε πεδίο αν δεν βρεθεί καμία εγγραφή.

```
def get_last_question(topic):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #select the id, question and answer of the last added question for the current topic
        cursor.execute('''
            SELECT id, question, answer
            FROM questions
            WHERE id = (SELECT MAX(id) FROM questions WHERE topic = ?)
        ''', (topic,))
        result = cursor.fetchone()
        if result:
             #return the id, question, and answer
            return result[0], result[1], result[2]
        #return None,if no question is found
        return None, None, None
```

Η update_user_question_count χρησιμεύει στο συγχρονισμό και την παρακολούθηση των εναπομείναντων ερωτήσεων που αφορούν τον χρήστη σε συγκεκριμένη θεματική.
Το reduce καθορίζει το αν θα μειωθεί ο αριθμός των ερωτήσεων. Όταν το reduce ισούται με True, τότε μειώνεται κατα μία μονάδα ο αριθμός των ερωτήσεων για τον συγκεκριμένο χρήστη και τη θεματική: SET num_questions = num_questions - 1.
Σε περίπτωση που ο αριθμός των ερωτήσεων πάει να γίνει αρνητικός, αυτό διορθώνεται ορίζοντας αυτό τον αριθμό ίσο με το μηδέν: SET num_questions = 0. Έτσι εξασφαλίζεται ότι η τιμή του πεδίου num_questions δεν θα είναι μικρότερη από 0.
Όταν το reduce είναι False, τότε είτε εισάγεται ή παραλείπεται η εγγραφή. Αν δεν υπάρχει ήδη εγγραφή για τον χρήστη και το θέμα, πραγματοποιείται με τις τιμές: INSERT OR IGNORE INTO user_topics (user_id, topic, num_questions, initial_num_questions). Αν υπάρχει ήδη εγγραφή για το user_id και το topic, η εισαγωγή θα παρακαμφθεί χωρίς την ύπαρξη σφάλματος, χάρη στη χρήση του INSERT OR IGNORE.

```
def update_user_question_count(user_id, topic, reduce=True, initial_num_questions=None):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #try to reduce the question count
        if reduce:
            print(f"reduce num_questions for user_id: {user_id}, topic: {topic}")
            cursor.execute('''
                UPDATE user_topics
                SET num_questions = num_questions - 1
                WHERE user_id = ? AND topic = ?
            ''', (user_id, topic))
            #ff the number of questions is below 0 sit must be set to 0
            cursor.execute('''
                UPDATE user_topics
                SET num_questions = 0
                WHERE user_id = ? AND topic = ? AND num_questions < 0
            ''', (user_id, topic))
        else:
            print(f"set num_questions for user_id: {user_id}, topic: {topic}")
            #insert the number of questions for user and topic
            #if the same user id or topic already exists ignore will ignore the insertion and
preventing an error or a duplication of user id
            cursor.execute('''
                INSERT   OR   IGNORE   INTO   user_topics   (user_id,   topic,   num_questions,
initial_num_questions)
                VALUES (?, ?, ?, ?)
            ''', (user_id, topic, initial_num_questions, initial_num_questions))
        #commit to ensure saving data
        conn.commit()
```

Η get_user_question_count επιστρέφει τον αριθμό των διαθέσιμων ερωτήσεων για τον εκάστοτε χρήστη σε συγκεκριμένη θεματική. Σε περίπτωση που υπάρχει αποτέλεσμα επιστρέφεται ο αριθμός των ερωτήσεων που έχουν απομείνει, διαφορετικά τυπώνεται μήνυμα στο τερματικό που ενημερώνει για τη μη ύπαρξη ερωτήσεων για το συγκεκριμένο χρήστη ή θεματική.

```
def get_user_question_count(user_id, topic):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #select the number of users questions for current topic
        cursor.execute('SELECT num_questions FROM user_topics WHERE user_id = ? AND topic = ?',
(user_id, topic))
        result = cursor.fetchone()
        if result:
            #return the number of remaining questions
            return result[0]
        else:
            print(f"No questions found for user_id {user_id} and topic {topic}.")
            return 0
```

Η get_initial_num_questions επιστρέφει την τιμή των ερωτήσεων που όρισε ο χρήστης κατά την έναρξη του παιχνιδιού.

```
def get_initial_num_questions(user_id, topic):
    with sqlite3.connect('topics.db') as conn:
        cursor = conn.cursor()
        #select the initial number of users questions for a topic
        cursor.execute('SELECT initial_num_questions FROM user_topics WHERE user_id = ? AND
topic = ?', (user_id, topic))
        result = cursor.fetchone()
        if result:
            #return the initial number of questions
            return result[0]
        else:
            return 0
```

Η get_last_user_age επιστρέφει την ηλικία του τελευταίου χρήστη που εισήλθε.

```
def get_last_user_age():
    conn = sqlite3.connect('topics.db')
    cursor = conn.cursor()
    #get user_age of the user with the max id/the last user
    cursor.execute('SELECT user_age FROM users WHERE id = (SELECT MAX(id) FROM users)')
    result = cursor.fetchone()
    conn.close()
    if result:
        #return the last users age
        return result[0]
    else:
        return None
```

Κύριο Μενού

Προστέθηκε αρχικό μενού menu.py με τις επιλογές: Έναρξη παιχνιδιού (Start Game) και Έξοδος από το παιχνίδι (Quit).

Ο χρήστης μπορεί να επιλέξει είτε με χρήση των up και down arrow keys ή να περιηγηθεί και να κάνει κλικ με το ποντίκι.

```
                elif event.type == pygame.KEYDOWN:
                    if current_state == MENU:
                        if event.key == pygame.K_UP:
                            selected_option = max(1, selected_option - 1)
                        elif event.key == pygame.K_DOWN:
                            selected_option = min(2, selected_option + 1)
                        elif event.key == pygame.K_RETURN or event.key == pygame.K_KP_ENTER:
                            if selected_option == 1:
                             # start Flask server and move to GAME state
                                    start_flask_server()
                                    new_user()
                                    current_state = GAME
                            elif selected_option == 2:
                                pygame.quit()
                                sys.exit()
                elif event.type == pygame.MOUSEMOTION:
                    x, y = event.pos
                    hovered_option = None
                    for i, option in enumerate(Game_Menu, start=1):
                        text_rect            =            font.render(option,            True,
brown_red).get_rect(center=(SCREEN_WIDTH // 2, (SCREEN_HEIGHT // 2 + 40) + (i - 1) * 50))
                        if text_rect.collidepoint(event.pos):
                            hovered_option = i
                elif event.type == pygame.MOUSEBUTTONDOWN and event.button == 1:
                    x, y = event.pos
                    for i, option in enumerate(Game_Menu, start=1):
                        text_rect            =            font.render(option,            True,
brown_red).get_rect(center=(SCREEN_WIDTH // 2, (SCREEN_HEIGHT // 2 + 40) + (i - 1) * 50))
                        if text_rect.collidepoint(x, y):
                            if i == 1:
                                start_flask_server()
                                new_user()
                                current_state = GAME
                            elif i == 2:
                                pygame.quit()
                                sys.exit()
```

Εισαγωγή Στοιχείων Χρήστη

Δημιουργήθηκε το αρχείο adduser.py, ώστε ο χρήστης να μπορεί να εισάγει τα στοιχεία του και τον αριθμό των ερωτήσεων που επιθυμεί να δεχθεί ανα θεματική ενότητα.

Όνομα και ηλικία χρήστη

Αρχικά, δημιουργήθηκε η new_user, με την οποία καταχωρούνται το όνομα και η ηλικία του τωρινού χρήστη στη βάση δεδομένων.

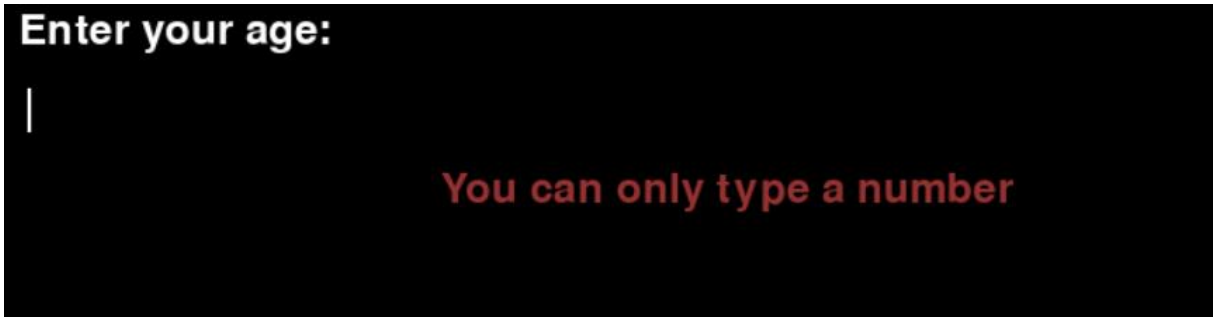
Εικόνα 24.Εισαγωγή ονόματος

```
def new_user():
    def add_user(user_name, user_age):
        conn = sqlite3.connect('topics.db')
        cursor = conn.cursor()

        cursor.execute('''
            INSERT INTO users (user_name, user_age) VALUES (?, ?)
        ''', (user_name, user_age))
        # get the id of the last user inserted
        user_id = cursor.lastrowid

        conn.commit()
        conn.close()
        return user_id
```

Ο χρήστης καλείται να εισάγει το όνομα, την ηλικία και τον αριθμό ερωτήσεων για τον κάθε χρήστη, εξασφαλίζοντας ότι μόνο αριθμητικά δεδομένα εισάγονται για την ηλικία και τον αριθμό των ερωτήσεων.

```
    #function to get the users input from keyboard
    def get_input(screen, font, prompt, default_text="", numeric_only=False):
…
                        #allow user only to type numbers
                        if numeric_only:
                            if event.unicode.isdigit():
                                input_text = input_text[:cursor_pos] + event.unicode +
input_text[cursor_pos:]
                                #move the cursor position
                                cursor_pos += 1
                            else:
#set the message displaying if users types anything else but a number
                                    inform_message = "You can only type a number"
                                inform_time = time.time()
                        else:
                            input_text = input_text[:cursor_pos] + event.unicode +
input_text[cursor_pos:]
                            cursor_pos += 1
            pygame.display.update()

        return input_text
```

Με τη λειτουργία get_input_text ο χρήστης εισάγει τα δεδομένα του στα πεδία του ονόματος και της ηλικίας. Αν στο πεδίο της ηλικίας δεν εισαχθεί αριθμός, αλλά κείμενο, τότε εμφανίζεται μήνυμα, το οποίο ενημερώνει τον χρήστη.



Εικόνα 25.Ασφάλεια εισαγωγής ηλικίας

```
    user_name = get_input(screen, font, "Enter your name:")
    user_age = get_input(screen, font, "Enter your age:", numeric_only=True)
```

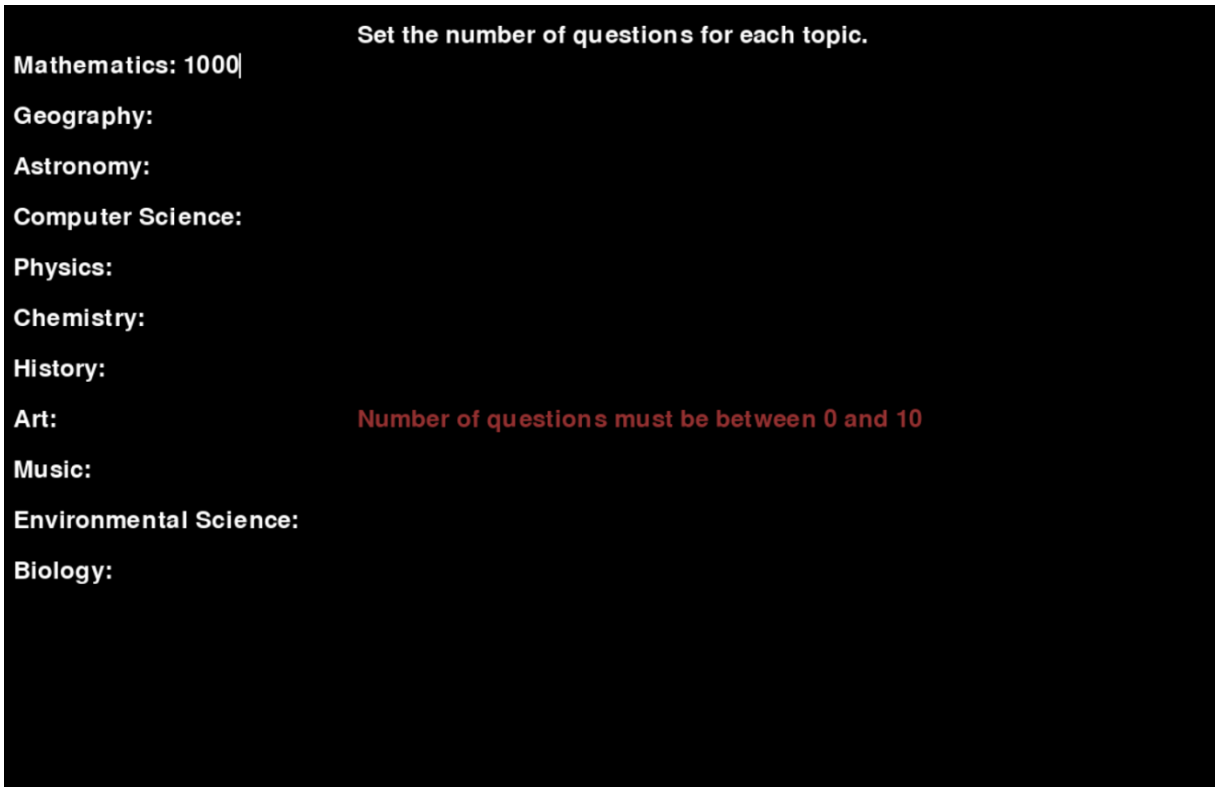
Αν το numeric_only είναι ίσο με True τότε γίνεται ο διαχωρισμός του ονόματος και της ηλικίας, όπου μπορεί να καταχωρηθεί μόνο αριθμός.

Αριθμός Ερωτήσεων

Με την add_user_topic_num_settings καταχωρείται στη βάση δεδομένων ο αριθμός των ερωτήσεων που θέλει να δεχθεί ο χρήστης, ανα θεματική.

```
    def add_user_topic_num_settings(user_id, topics_num_settings):
        conn = sqlite3.connect('topics.db')
        cursor = conn.cursor()
        for topic, num_questions in topics_num_settings.items():
            # set the initial_num_questions as equal to num_questions
            initial_num_questions = num_questions
            cursor.execute('''
                INSERT OR REPLACE INTO user_topics (user_id, topic, num_questions,
initial_num_questions) VALUES (?, ?, ?, ?)
            ''', (user_id, topic, num_questions, initial_num_questions))

        conn.commit()
        conn.close()
```

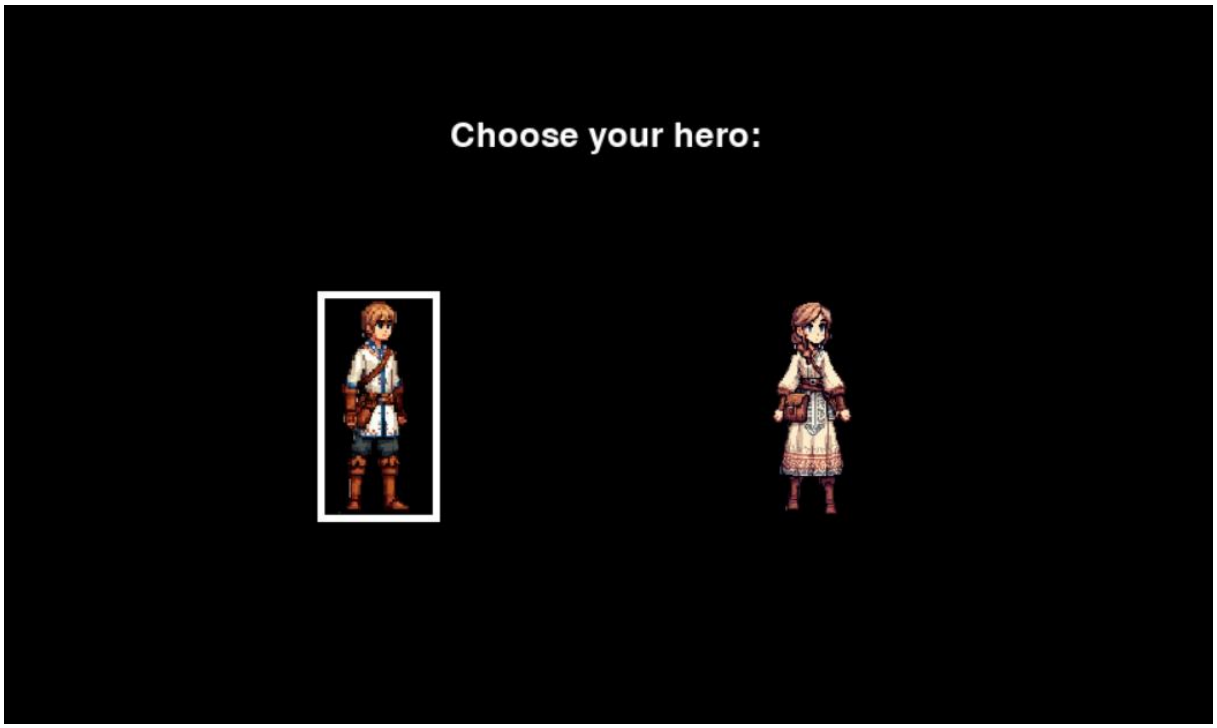
Επιτρέπεται η επιλογή από 0 έως 10 ερωτήσεων. Αν ο αριθμός ξεπεραστεί ή εισαχθεί μη αριθμητικός χαρακτήρας, εμφανίζεται μήνυμα, που ενημερώνει το χρήστη για τη σωστή συμπλήρωση του πεδίου. Το ίδιο συμβαίνει και στην περίπτωση που ο χρήστης δεν εισάγει αριθμό αλλά οποιοδήποτε άλλο σύμβολο.



Εικόνα 26.Όριο αριθμού ερωτήσεων

```
def get_topic_num_settings(screen, font):
                            …
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_TAB:
                    selected = (selected + 1) % len(topics)
                elif event.key == pygame.K_RETURN:
            #allow only numbers.Those numbers must be between 0-10
                    if input_for_topic[selected].isdigit():
                        num_questions = int(input_for_topic[selected])
                        if 0 <= num_questions <= 10:
                            selected = (selected + 1) % len(topics)
                            if selected == 0:
                                active_input = False
                        else:
                            inform_message = "Number of questions must be less than 10"
                            inform_time = time.time()
                    else:
                        inform_message = "You must enter a number"
                        inform_time = time.time()
                                    …
    for i, topic in enumerate(topics):
        if input_for_topic[i].isdigit():
            num_questions = int(input_for_topic[i])
            if 0 <= num_questions <= 10:
                num_settings[topic] = num_questions
            else:
                print(f"Number of questions must be between 0 and 10.")
        else:
            print(f"Wrong num.")
    return num_settings
```

Όταν ο χρήστης απαντήσει σωστά σε όλες τις ερωτήσεις ενός θέματος, εμφανίζεται ένα ορθογώνιο. Σε επόμενη φάση, αυτό θα αντικατασταθεί από ένα sprite που θα αναπαριστά αντικείμενο σχετικό με το θέμα.



Εικόνα 27.Επιλογή χαρακτήρα

```
            if initial_num_questions > 0 and current_num_questions == 0 and not
npc["points_square_collected"]:
                # draw the square two tiles above the npc
                points_square_y = npc_screen_y - 2 * tile_size
                pygame.draw.rect(screen, pygame.Color('red'), (npc_screen_x, points_square_y,
tile_size, tile_size))
```

Βαθμολογία και Δυσκολία

Η βαθμολογία του χρήστη υπολογίζεται ως εξής:

Ο χρήστης κερδίζει πόντους ίσους με το άθροισμα των σωστών απαντήσεων για κάθε θέμα.

Η επιλογή του αριθμού ερωτήσεων και η απόδοση πόντων είναι αυτή που ορίζει ένα μέρος της δυσκολίας του παιχνιδιού.

```
    def check_points_square_collision(player_pos, npcs, tile_size):
        global score
        for npc in npcs.values():
            if not npc["points_square_collected"]:
                npc_screen_x =npc["pos"].x // tile_size * tile_size
                npc_screen_y= npc["pos"].y // tile_size * tile_size - 2 * tile_size
                points_square_rect=pygame.Rect(npc_screen_x,      npc_screen_y,      tile_size,
tile_size)
                player_rect=pygame.Rect(player_pos.x,player_pos.y, tile_size, tile_size)
                if player_rect.colliderect(points_square_rect):
                    npc["points_square_collected"] = True
                    initial_num_questions = get_initial_num_questions(user_id, npc["topic"])
                    score += initial_num_questions
                    print(f"Score: {score}")
                    return
```

Πρόσθετα Χαρακτηριστικά

Επιπλέον χαρακτηριστικά περιλαμβάνουν την υποστήριξη για αντιγραφή και επικόλληση με Ctrl + C και Ctrl + V. Επιπλέον, με τη χρήση της hasattr εξασφαλίζεται η ομαλή λειτουργία της μετάβασης από την μία πλευρά του τυπωμένου κειμένου στην άλλη με τα πλήκτρα αριστερού και δεξιού βέλους, καθώς και για το πλήκτρο backspace[17].

Οι λειτουργίες συνεχίζονται κανονικά μέχρι να απελευθερωθεί το πλήκτρο, εξασφαλίζοντας μια ομαλή εμπειρία χρήσης.

4.4. Τέταρτη Φάση Ανάπτυξης – Έκδοση 4 της εφαρμογής

Στην τέταρτη φάση της υλοποίησης του παιχνιδιού, το έργο άρχισε να αποκτά την τελική του μορφή. Ένα από τα κύρια χαρακτηριστικά που προστέθηκαν ήταν οι περισσότερες θεματικές και NPCs, με τους οποίους πρέπει να συνομιλήσει ο παίκτης. Πλέον διατίθενται 11 θεματικές, κάθε μία από τις οποίες αντιπροσωπεύεται από έναν NPC, ο οποίος είναι εμπνευσμένος από ιστορικές προσωπικότητες που σχετίζονται με το εκάστοτε αντικείμενο γνώσης.

Ορισμός Θεματικών και NPCs

Στον πίνακα topics της βάσης δεδομένων ορίζονται οι θεματικές και οι αντίστοιχοι δύο χαρακτήρες (NPCs), με το όνομα και την εικόνα του κάθε χαρακτήρα να αντιστοιχούν σε ιστορικές προσωπικότητες. Στην φάση αυτή, προστέθηκαν οι εξής θεματικές: Μαθηματικά, Γεωγραφία, Αστρονομία, Πληροφορική, Φυσική, Χημεία, Ιστορία, Τέχνη, Μουσική, Περιβαλλοντική Επιστήμη, Βιολογία.

[17] https://www.toppr.com/guides/python-guide/references/methods-and-functions/methods/built-in/hasattr/python-hasattr/

```
    cursor.execute('''
        INSERT OR REPLACE INTO topics (topic_name, npc_name, npc_image) VALUES
        ('Mathematics', ?, ?),
        ('Geography', ?, ?),
        ('Astronomy', ?, ?),
        ('Computer Science', ?, ?),
        ('Physics', ?, ?),
        ('Chemistry', ?, ?),
        ('History', ?, ?),
        ('Art', ?, ?),
        ('Music', ?, ?),
        ('Environmental Science', ?, ?),
        ('Biology', ?, ?)
    ''',
(*math_npc,*geography_npc,*astronomy_npc,*computer_npc,*physics_npc,*chemistry_npc,*history_np
c,*art_npc, *music_npc,*environmental_npc,*biology_npc))
```

Στον πίνακα topics της βάσης δεδομένων ορίζονται οι θεματικές, ωστόσο το όνομα του κάθε NPC και η εικόνα (sprite) που αναπαριστά το χαρακτήρα είναι ορισμένη με ?. Αυτό συμβαίνει, γιατί για κάθε θεματική, υπάρχουν δύο διαθέσιμοι χαρακτήρες, οι οποίοι αναπαριστούν αληθινά ιστορικά πρόσωπα, που σχετίζονται με το κάθε αντικείμενο γνώσης. Κάθε φορά που ξεκινάει το παιχνίδι επιλέγεται τυχαία ο ένας εκ των δύο αυτών χαρακτήρων. Με την εντολή INSERT OR REPLACE διασφαλίζεται ότι η τυχαία επιλογή του χαρακτήρα, θα ορίζεται εκ νέου κάθε φορά που ξεκινάει το παιχνίδι και δεν χρησιμοποιείται ο χαρακτήρας που επιλέχθηκε τυχαία κατά την πρώτη φορά που έτρεξε το παιχνίδι και δημιουργήθηκε η βάση δεδομένων.

```
    cursor.execute('''
    CREATE TABLE IF NOT EXISTS topics (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        topic_name TEXT UNIQUE NOT NULL,
        npc_name TEXT NOT NULL,
        npc_image TEXT )''')
    music_npc = random.choice([
    ('Ludwig van Beethoven', 'sprites/npcs/Beethoven.png'),
    ('Wolfgang Amadeus Mozart', 'sprites/npcs/Mozart.png')])
    physics_npc = random.choice([
    ('Isaac Newton', 'sprites/npcs/Newton.png'),
    ('Albert Einstein', 'sprites/npcs/Einstein.png')])
    math_npc = random.choice([
    ('Archimedes', 'sprites/npcs/Archimedes.png'),
    ('Euclid', 'sprites/npcs/Euclid.png')])
    geography_npc = random.choice([
    ('Ferdinand Magellan', 'sprites/npcs/Magellan.png'),
    ('Christopher Columbus', 'sprites/npcs/Columbus.png')])
    astronomy_npc = random.choice([
    ('Nicolaus Copernicus', 'sprites/npcs/Copernicus.png'),
    ('Galileo Galilei', 'sprites/npcs/Galileo.png')])
    computer_npc = random.choice([
    ('Alan Turing', 'sprites/npcs/Turing.png'),
    ('Charles Babbage', 'sprites/npcs/Babbage.png')])
    chemistry_npc = random.choice([
```

```
    ('Marie Curie', 'sprites/npcs/Curie.png'),
    ('Dmitri Mendeleev', 'sprites/npcs/Mendeleev.png')])
    history_npc = random.choice([
    ('Herodotus', 'sprites/npcs/Herodotus.png'),
    ('Thucydides', 'sprites/npcs/Thucydides.png')])
    art_npc = random.choice([
    ('Leonardo da Vinci', 'sprites/npcs/Leonardo.png'),
    ('Frida Kahlo', 'sprites/npcs/Frida.png')])
    environmental_npc = random.choice([
    ('Rachel Carson', 'sprites/npcs/Carson.png'),
    ('John Muir', 'sprites/npcs/Muir.png')])
    biology_npc = random.choice([
    ('Charles Darwin', 'sprites/npcs/Darwin.png'),
    ('Alexander Fleming', 'sprites/npcs/Fleming.png')])
```

Διαμόρφωση Χάρτη και Τοποθέτηση NPCs

Ο χάρτης του παιχνιδιού χωρίστηκε σε δώδεκα τμήματα, ένα για κάθε NPC και ένα για να τοποθετηθεί ο παίκτης. Η τοποθέτηση των χαρακτήρων πραγματοποιείται τυχαία κάθε φορά που τρέχει το παιχνίδι.

Η set_npcs() είναι υπεύθυνη για την αρχικοποίηση των NPCs στο παιχνίδι και την τοποθέτησή τους στον χάρτη. Η συνάρτηση συνδέεται στη βάση δεδομένων topics.db, δημιουργεί έναν cursor για την εκτέλεση SQL ερωτημάτων και ανακτά τα topic_name, npc_name και npc_image όλων των NPCs. Ο χάρτης χωρίζεται σε 12 τομείς (3 σειρές × 4 στήλες), με ένα κενό 2000 pixels ανάμεσα στους τομείς και στα εξωτερικά όρια. Τα πραγματικά μεγέθη των τομέων υπολογίζονται λαμβάνοντας υπόψη αυτά τα κενά μεταξύ των τομέων (sector_gap) και τα κενά από τα όρια του χάρτη (outer_gap). Για την τοποθέτηση των NPCs, δημιουργούνται δύο λίστες: η available_sectors, που περιέχει όλους τους διαθέσιμους τομείς, και η occupied_sectors, όπου καταγράφονται οι τομείς που καταλαμβάνονται από NPCs.

```
def set_npcs():
    conn = sqlite3.connect('topics.db')
    cursor = conn.cursor()
    cursor.execute("SELECT topic_name, npc_name, npc_image FROM topics")
    npcs = {}
    tile_size = 40
    map_size = 1000 * tile_size
    sector_gap = 2000
    outer_gap = 2000
    sector_rows = 3
    sector_cols = 4
    total_width_gap = (sector_cols - 1) * sector_gap
    total_height_gap = (sector_rows - 1) * sector_gap
    sector_width = (map_size - total_width_gap - (2 * outer_gap)) // sector_cols
    sector_height = (map_size - total_height_gap - (2 * outer_gap)) // sector_rows
    available_sectors = [(i, j) for i in range(sector_rows) for j in range(sector_cols)]
    occupied_sectors = []
    npc_tilesets = {}
    obstacles = []
```

Για κάθε εγγραφή NPC που ανακτάται από τη βάση, αποθηκεύονται τα topic_name, npc_name και npc_image. Στη συνέχεια, επιλέγεται τυχαία ένας διαθέσιμος τομέας για την τοποθέτησή του, ο οποίος αφαιρείται από τη λίστα available_sectors και προστίθεται στη λίστα occupied_sectors. Υπολογίζεται η ακριβής θέση του τομέα στον χάρτη, ενώ ο NPC τοποθετείται στο κέντρο του, προσαρμόζοντας τη θέση του με βάση τις διαστάσεις του τομέα και τα προκαθορισμένα κενά. Τέλος, φορτώνονται οι εικόνες που αφορούν τα πλακίδια (tiles) του εκάστοτε NPC, και προσαρμόζονται στο καθορισμένο tile_size.

```
    for row in cursor.fetchall():
        topic, npc_name, npc_image = row
        sector = random.choice(available_sectors)
        available_sectors.remove(sector)
        occupied_sectors.append(sector)

        # find the top left position of the sector including gaps
        sector_x = outer_gap +sector[1]*(sector_width + sector_gap)
        sector_y = outer_gap+sector[0]*(sector_height + sector_gap)
        npc_x=sector_x+sector_width//2- tile_size //2-sector_gap
        npc_y =sector_y+sector_height//2-tile_size//2-sector_gap

        # set specific tiles for each npc (images for NPCs)
npc_tileset=[pygame.image.load(resource_path(f'sprites/npc_tiles/{topic}/tile{idx}.png')).conv
ert()
            for idx in range(1, 5)]
        npc_tilesets[npc_name]=[pygame.transform.scale(tile, (tile_size, tile_size)) for tile
in npc_tileset]
```

Οι ιδιότητες του NPC αποθηκεύονται σε μια δομή δεδομένων, περιλαμβάνοντας το όνομά του, τη θέση του στον χάρτη, τις εικόνες του και το αντικείμενο γνώσης του. Παράλληλα, δημιουργείται ένα ορθογώνιο (sector_rect) που αντιπροσωπεύει τον τομέα στον οποίο ανήκει. Αφού ολοκληρωθεί η τοποθέτησή του, καλείται η συνάρτηση set_obs(), η οποία προσθέτει εμπόδια στον τομέα, διασφαλίζοντας ότι ο χάρτης αποκτά ποικιλία και επηρεάζοντας τις κινήσεις του παίκτη.

```
        # enter the NPC data using image paths
        npcs[npc_name.lower()] = {
            "npc_name": npc_name,
            "pos": pygame.Vector2(npc_x, npc_y),
            "topic": topic,
            "npc_image": resource_path(npc_image),
            "collectable_image": resource_path(f"sprites/collectables/{topic}.png"),
            "collectable_item_collected": False,
            "clicked": False,
            "sector_rect":pygame.Rect(sector_x,sector_y, sector_width, sector_height),
            "tileset": npc_tilesets[npc_name],}
   set_obs(topic,sector_x,sector_y,sector_width,sector_height,
tile_size,npc_x,npc_y,obstacles)
```

Αφού τοποθετηθούν όλοι οι NPCs, εντοπίζονται οι τομείς που παραμένουν κενοί, δηλαδή εκείνοι που δεν έχουν καταληφθεί από κάποιον χαρακτήρα. Από αυτούς, επιλέγεται ο πρώτος διαθέσιμος τομέας για την τοποθέτηση του παίκτη. Στη συνέχεια, υπολογίζεται η ακριβής θέση του μέσα στον τομέα, διασφαλίζοντας ότι βρίσκεται στο κέντρο του. Η θέση του αποθηκεύεται ως Vector2, διευκολύνοντας τη διαχείριση των συντεταγμένων του μέσα στο παιχνίδι. Έπειτα, όπως και στους NPCs, έτσι και στον τομέα του παίκτη προστίθενται εμπόδια μέσω της συνάρτησης set_obs(), διαμορφώνοντας το περιβάλλον. Αφού ολοκληρωθεί η διαδικασία τοποθέτησης όλων των χαρακτήρων και των εμποδίων, η σύνδεση με τη βάση δεδομένων κλείνει. Τέλος, η συνάρτηση επιστρέφει τα δεδομένα των NPCs, τη θέση του παίκτη και τη λίστα με όλα τα εμπόδια που έχουν δημιουργηθεί στον χάρτη.

```
    # after all the npcs are placed, the player is placed in the center of the empty sector
    empty_sectors = [sector for sector in available_sectors if sector not in occupied_sectors]
    # select the first available empty sector
    if empty_sectors:
        empty_sector = empty_sectors[0]
        # find the center of the empty sector
        sector_x = outer_gap + empty_sector[1] * (sector_width +
        sector_gap)
        sector_y = outer_gap + empty_sector[0] * (sector_height +
        sector_gap)
        player_x=sector_x+sector_width//2-tile_size//2-sector_gap
        player_y=sector_y+sector_height//2-tile_size//2-sector_gap
        # position the player in the center of the empty sector
        player_pos = pygame.Vector2(player_x, player_y)
        set_obs(topic,sector_x,sector_y,sector_width,
        sector_height,tile_size,npc_x,npc_y,obstacles)
    conn.close()
    return npcs, player_pos, obstacles
```

Τοποθέτηση εμποδίων στο χάρτη

Η τοποθέτηση εμποδίων είναι επίσης ένα σημαντικό στοιχείο του χάρτη, καθώς προσφέρει ποικιλία στο περιβάλλον και επηρεάζει την κίνηση του παίκτη. Η συνάρτηση set_obs τοποθετεί έναν τυχαίο αριθμό εμποδίων μέσα σε έναν συγκεκριμένο τομέα του χάρτη, επιλέγοντας από 30 έως 40 εμπόδια ανά τομέα. Για κάθε εμπόδιο, επιλέγεται τυχαία μία εικόνα από το αντίστοιχο θέμα. Στη συνέχεια, λαμβάνονται οι διαστάσεις της εικόνας και καλείται η random_obs_pos για να βρεθεί μια θέση, εξασφαλίζοντας ότι το εμπόδιο δεν τοποθετείται πολύ κοντά στον NPC (τουλάχιστον 5 tiles μακρυά) και ότι υπάρχει αρκετός χώρος μεταξύ των εμποδίων (τουλάχιστον 10 tiles). Τέλος, το νέο εμπόδιο προστίθεται στη λίστα obstacles, αποθηκεύοντας ένα rect που ορίζει τη θέση και το μέγεθός του, καθώς και την εικόνα του, προσαρμοσμένη στις διαστάσεις του.

```
def set_obs (topic,sector_x, sector_y, sector_width, sector_height,
tile_size,npc_x,npc_y,obstacles):
    #the number of obstacles per sector
    num_obstacles = random.randint(30, 40)
    for _ in range(num_obstacles):
        obstacle_image_path=resource_path(f"sprites/obs/{topic}/obs{
        random.randint(1, 4)}.png")
      obstacle_image=pygame.image.load
       (obstacle_image_path).convert_alpha()
        #get the size of each obstacle from its image dimensions
        obstacle_width, obstacle_height = obstacle_image.get_size()
                    obstacle_x, obstacle_y = random_obs_pos(sector_x, sector_y,
sector_width,sector_height,max(obstacle_width,obstacle_height),pygame.Vector2(npc_x,npc_y),til
e_size,existing_obstacles=[obs['obstacle_rect'] for obs in obstacles],min_distance=5,
min_obstacle_distance=10)
                                obstacles.append({'obstacle_rect': pygame.Rect(obstacle_x,
obstacle_y,obstacle_width,obstacle_height),'obstacle_image':
pygame.transform.scale(obstacle_image,(obstacle_width, obstacle_height)),})
```

Η συνάρτηση random_obs_pos έχει ως σκοπό να βρει μια έγκυρη τυχαία θέση για τοποθέτηση ενός νέου εμποδίου στον χάρτη, διασφαλίζοντας ότι δεν θα συμπίπτει με NPCs ή άλλα εμπόδια. Ορίζει ένα περιθώριο γύρω από τα όρια του τομέα για να αποφεύγεται η τοποθέτηση πολύ κοντά στα σύνορα του τομέα. Επίσης, μετατρέπει τις αποστάσεις από tiles σε pixels, προκειμένου να υπολογίσει τη θέση με μεγαλύτερη ακρίβεια. Η συνάρτηση προσπαθεί να βρει μια έγκυρη θέση με μέγιστο αριθμό προσπαθειών τις 100. Εάν η νέα θέση είναι πολύ κοντά στον NPC, απορρίπτεται και η συνάρτηση επαναλαμβάνει την διαδικασία με νέα τυχαία θέση. Επιπλέον, ελέγχει εάν η θέση αυτή είναι πολύ κοντά σε υπάρχοντα εμπόδια στον τομέα και την απορρίπτει. Αν βρεθεί μια έγκυρη θέση, η συνάρτηση επιστρέφει τις συντεταγμένες της.

```
def random_obs_pos(x_start, y_start, sector_width, sector_height, max_size, npc_pos, tile_size,
existing_obstacles, min_distance=10, min_obstacle_distance=10):
    margin = 2 * max_size  #make sure obstacles are placed within the sector's bounds
    min_distance_px = min_distance * tile_size
    min_obstacle_distance_px = min_obstacle_distance * tile_size
    #make up to 100 attemps to find a valid position
    for _ in range(100):
        x=random.randint(x_start+margin,x_start+sector_width-
        margin-max_size)
        y=random.randint(y_start+margin,y_start+sector_height-
        margin-max_size)
        obstacle_pos = pygame.Vector2(x, y)        #check the distance from each sectors npc
        if obstacle_pos.distance_to(npc_pos) < min_distance_px:
            continue
   #check the distance from the existing obstacles in the sector
        valid = True
        for obs in existing_obstacles:
           if obstacle_pos.distance_to(pygame.Vector2(obs.center)) < min_obstacle_distance_px:
                valid = False
                break
        if valid:
            return x, y  #return the valid position
    raise ValueError("Could not find a position for the obstacle.")
```

Εχθροί

Εικόνα 28.Καταδίωξη από εχθρό

Στη συνέχεια, υλοποιήθηκε η κλάση Enemies(). Η κλάση αυτή είναι υπεύθυνη για τη δημιουργία και τη διαχείριση της συμπεριφοράς των εχθρών του παιχνιδιού.

Η update_animation(self, delta_time) οπτικοποιεί την κίνηση του εχθρού μέσω εναλλαγής των εικόνων animation. Το χρονόμετρο self.animation_timer αυξάνεται με βάση το delta_time, δηλαδή τον χρόνο που έχει περάσει από το τελευταίο καρέ. Όταν το χρονόμετρο υπερβεί την τιμή self.animation_speed, η εικόνα αλλάζει (self.current_image_index), δημιουργώντας εφέ κίνησης, ενώ το self.animation_timer μηδενίζεται. Η χρήση του «%» εξασφαλίζει την ορθή εναλλαγή των εικόνων, προσδίδοντας ομαλή και ρεαλιστική κίνηση του εχθρού.

```
def update_animation(self, delta_time):
    self.animation_timer += delta_time
     if self.animation_timer >= self.animation_speed:
        self.current_image_index = (self.current_image_index + 1) % len(self.images)
         self.animation_timer = 0
```

Η collides_with_obstacle(self, target_pos, obstacles) ελέγχει αν ο εχθρός για κάθε νέα θέση του (target_pos) συγκρούεται με κάποιο εμπόδιο. Αρχικά, δημιουργείται ένα ορθογώνιο (target_rect) που ορίζει την πιθανή νέα θέση του εχθρού. Στη συνέχεια, η colliderect() συγκρίνει αυτό το ορθογώνιο με τα ορθογώνια των εμποδίων (obstacle_rect). Εάν εντοπιστεί κάποια σύγκρουση, η μέθοδος επιστρέφει True, διαφορετικά επιστρέφει False.

```
def collides_with_obstacle(self, target_pos, obstacles):
        target_rect = pygame.Rect(target_pos.x, target_pos.y, self.tile_size * 2, self.tile_size
* 2)
         for obstacle in obstacles:
             if target_rect.colliderect(obstacle['obstacle_rect']):
                 return True
         return False
```

Η random_move(self, delta_time, screen_width, screen_height, obstacles) επιτρέπει στον εχθρό να κινείται τυχαία στον χάρτη όταν δεν ανιχνεύει τον παίκτη. Κάθε random_move_interval δευτερόλεπτα, παράγεται ένας τυχαίος προορισμός (random_target) εντός των ορίων του χάρτη χρησιμοποιώντας τη συνάρτηση random.randint() για τη δημιουργία συντεταγμένων. Εάν η νέα θέση δεν συγκρούεται με κάποιο εμπόδιο, ο στόχος random_target ενημερώνεται. Ο εχθρός κατευθύνεται προς τον τυχαίο προορισμό χρησιμοποιώντας διανύσματα: το διάνυσμα κατεύθυνσης υπολογίζεται ως direction = self.random_target-self.pos και στη συνέχεια κανονικοποιείται direction.normalize() ώστε να εξασφαλιστεί ομαλή κίνηση. Η θέση του εχθρού ενημερώνεται με βάση την ταχύτητά του self.speed και τον χρόνο delta_time που έχει περάσει. Αυτή η συμπεριφορά προσδίδει το εφε περιπλάνησης στους εχθρούς, καθιστώντας τους λιγότερο προβλέψιμους και πιο ρεαλιστικούς στην κίνησή τους.

```
def random_move(self, delta_time, screen_width, screen_height, obstacles):
        self.random_move_timer += delta_time
        if self.random_move_timer >= self.random_move_interval:
            while True:
               random_target=pygame.Vector2(random.randint(0, screen_width - self.tile_size *
2),random.randint(0, screen_height - self.tile_size * 2))
               if not self.collides_with_obstacle(random_target, obstacles):
                 self.random_target = random_target
                 break
            self.random_move_timer = 0
        direction = self.random_target - self.pos
        distance = direction.length()
        if distance > 0:
            direction = direction.normalize()
                target_pos=self.pos+direction* self.speed * delta_time
                if not self.collides_with_obstacle(target_pos,
             obstacles):
               self.pos = target_pos
```

Η move_towards_player(self, player_pos, frame_time, chat_active, obstacles) ελέγχει αν ο παίκτης βρίσκεται εντός εμβέλειας και κατευθύνει τον εχθρό προς αυτόν. Αρχικά, υπολογίζεται το διάνυσμα κατεύθυνσης ως direction = player_pos - self.pos. Εάν η απόσταση distance είναι μικρότερη από self.tile_size * 10, ο εχθρός ξεκινά να ακολουθεί τον παίκτη. Σε αυτήν την περίπτωση, το διάνυσμα κατεύθυνσης κανονικοποιείται direction.normalize(), υπολογίζεται η νέα θέση και πραγματοποιείται έλεγχος για πιθανές συγκρούσεις με εμπόδια.

Εάν δεν εντοπιστεί κάποια σύγκρουση, η θέση του εχθρού ενημερώνεται, επιτρέποντάς του να προσεγγίσει τον παίκτη. Αντίθετα, εάν ο παίκτης βρίσκεται εκτός εμβέλειας, ενεργοποιείται η random_move() για να κινηθεί τυχαία. Αυτή η συμπεριφορά προσδίδει έναν πιο αληθοφανή χαρακτήρα στους εχθρούς, καθιστώντας τους πιο απρόβλεπτους στην κίνηση και την αλληλεπίδραση τους με τον παίκτη.

```
def move_towards_player(self, player_pos, frame_time, chat_active, obstacles):
        global Follow
        if chat_active:
            return
        direction = player_pos - self.pos
        distance = direction.length()
        if distance < self.tile_size * 10:
            if distance != 0:
                direction = direction.normalize()
                target_pos = self.pos + direction * self.speed * frame_time
                if not self.collides_with_obstacle(target_pos, obstacles):
                    self.pos = target_pos
                    Follow = True
        else:
            self.random_move(frame_time, self.tile_size * 40, self.tile_size * 20, obstacles)
            Follow = False
```

Η respawn(self, map_size, obstacles) είναι υπεύθυνη την επανεμφάνιση respawn του εχθρού σε τυχαία θέση στον χάρτη, διασφαλίζοντας ότι δεν θα προκύψουν συγκρούσεις με εμπόδια. Αρχικά, δημιουργούνται τυχαίες συντεταγμένες x και y που βρίσκονται εντός των ορίων του χάρτη χρησιμοποιώντας τη συνάρτηση random.randint(). Αυτές οι συντεταγμένες αποθηκεύονται ως διάνυσμα random_pos. Στη συνέχεια, δημιουργείται ένα ορθογώνιο enemy_rect που αποτελεί τον χώρο που καταλαμβάνει ο εχθρός στην τυχαία θέση. Με τη χρήση της colliderect() ελέγχεται αν το enemy_rect συγκρούεται με κάποιο από τα ορθογώνια obstacle_rect) των εμποδίων. Εάν δεν εντοπιστεί σύγκρουση, η νέα θέση self.pos ενημερώνεται και η διαδικασία σταματά. Μετά την επιτυχή επανεμφάνιση, το self.collision_timer μηδενίζεται και ο εχθρός ενεργοποιείται.

```
def respawn(self, map_size, obstacles):
        while True:
            x = random.randint(0, (map_size - 1) * 40)
            y = random.randint(0, (map_size - 1) * 40)
            random_pos = pygame.Vector2(x, y)
            #make sure enemy collides with obstacles
                 enemy_rect = pygame.Rect(random_pos.x, random_pos.y, self.tile_size * 2,
self.tile_size * 2)
       if not any(enemy_rect.colliderect(obstacle['obstacle_rect']) for obstacle in obstacles):
                self.pos = random_pos
                print(f"Enemy respawned at {self.pos}")
                break  #stop when a valid position is found
        self.collision_timer = 0
        self.active = True
```

Έπειτα γίνεται έλεγχος για συγκρούσεις μεταξύ του παίκτη και των ενεργών εχθρών. Αν ο παίκτης πατήσει SPACE, ο εχθρός θεωρείται ηττημένος, επανεμφανίζεται enemy.respawn σε νέα θέση που δεν συγκρούεται με εμπόδια και αρχίζει να κινείται τυχαία enemy.random_move(). Αν δεν πατηθεί το SPACE, τότε ο παίκτης δέχεται φθορά από τον εχθρό. Το χρονόμετρο του εχθρού αυξάνεται με βάση τον χρόνο που έχει περάσει ανά καρέ και για κάθε 1 δευτερόλεπτο, μειώνεται η ζωή του παίκτη.

```
#check if player collides with enemy
        for enemy in enemies:
          if not enemy.active:
             continue
          enemy_rect = pygame.Rect(enemy.pos.x, enemy.pos.y,
tile_size * 2, tile_size * 2)
          if new_player_rect.colliderect(enemy_rect):
             keys = pygame.key.get_pressed()
             if keys[pygame.K_SPACE]:
                print("Enemy defeated!")
                # Deactivate the enemy if the player attacks
                enemy.respawn(map_size,obstacles)
                enemy.random_move(frame_time, screen_width,
 screen_height,obstacles)
              else:
              #handle damage when colliding with enemy with a timer
                 enemy.collision_timer += frame_time
                 if enemy.collision_timer >= 1:
                    hearts_amount -= 1
                    enemy.collision_timer = 0
                    print("Attacked by enemy!")
```

Αν το χρονόμετρο collision_timer είναι μεγαλύτερο ή ίσο με 1, τότε υπολογίζεται ένα διάνυσμα direction_vector από τον παίκτη προς τον εχθρό και αν το διάνυσμα έχει μήκος μεγαλύτερο από μηδέν, κανονικοποιείται και ο εχθρός απωθείται μακριά από τον παίκτη. Παράλληλα διασφαλίζεται ότι ο εχθρός παραμένει εντός των ορίων του χάρτη, αποτρέποντας την έξοδο από την περιοχή του παιχνιδιού, με τη χρήση των συναρτήσεων max() και min(), οι οποίες χρησιμοποιούνται για να περιοριστούν οι συντεταγμένες x και y του εχθρού.

```
        #push the enemy away
        if enemy.collision_timer >= 1:
          direction_vector=pygame.Vector2(enemy.pos.x- player_pos.x, enemy.pos.y - player_pos.y)
           if direction_vector.length() > 0:
              direction_vector = direction_vector.normalize()
              enemy.pos.x += direction_vector.x * tile_size * 10
              enemy.pos.y += direction_vector.y * tile_size * 10
         #make sure the enemy stays within map boundaries
         enemy.pos.x=max(0,min(enemy.pos.x,(map_size- 1)* tile_size))
         enemy.pos.y=max(0,min(enemy.pos.y,(map_size-1) * tile_size))
```

Inventory

Εικόνα 29.Διαθέσιμα αντικείμενα στο Inventory

Ένα από τα βασικά στοιχεία του παιχνιδιού είναι το Inventory, το οποίο αποθηκεύει τα αντικείμενα που συλλέγονται από τον παίκτη. Η επιφάνεια του Inventory δημιουργήθηκε μέσω της συνάρτησης draw_inventory_surface(), η οποία έχει ως στόχο την εμφάνιση των αντικειμένων που ο παίκτης πρέπει να συλλέξει από τους NPCs. Η επιφάνεια γεμίζει αρχικά με μαύρο χρώμα και στη συνέχεια τα αντικείμενα εμφανίζονται με συγκεκριμένο πλάτος (80 pixels) και απόσταση μεταξύ τους (20 pixels). Η εικόνα κάθε αντικειμένου φορτώνεται δυναμικά, και αν το αντικείμενο δεν έχει συλλεχθεί, εμφανίζεται με ημιδιαφάνεια για να υποδείξει στον παίκτη ότι δεν έχει αποκτηθεί ακόμη.

```
          def draw_inventory_surface(surface, npcs):
               #fill the surface with black colour
               surface.fill(pygame.Color('black'))
               #width of each point rectangle
               rect_width = 80
               spacing = 20
               for i, npc in enumerate(npcs.values()):
                   rect_x = i * (rect_width + spacing) + 20
                   #load the collectable items image
                   collectable_image=pygame.image.load(npc
                   ["collectable_image"]).convert_alpha()
                   #resize the image for the inventory
                 collectable_image  =  pygame.transform.scale(collectable_image,  (rect_width,
rect_width))
                   if not npc["collectable_item_collected"]:
                       collectable_image.set_alpha(100)
                   surface.blit(collectable_image, (rect_x, 10))
```

Βοηθητικά Μηνύματα (Hints)

Εικόνα 30.Βοηθητικά μηνύματα στο περιβάλλον του παιχνιδιού

Η εμφάνιση βοηθητικών μηνυμάτων είναι επίσης απαραίτητη για την καθοδήγηση του παίκτη κατά την αλληλεπίδραση με τους NPCs. Η συνάρτηση draw_chat_hint() είναι υπεύθυνη για την εμφάνιση αυτών των μηνυμάτων. Τα μηνύματα προβάλλονται στην επιφάνεια του παιχνιδιού όταν ο παίκτης πλησιάζει έναν NPC ή όταν πρέπει να ολοκληρώσει έναν στόχο. Η επιφάνεια του βοηθητικού μηνύματος προσαρμόζεται στο μισό ύψος του inventory. Αρχικά, ελέγχεται αν ο παίκτης βρίσκεται κοντά σε κάποιον NPC με την distance_from_npc(). Αν ισχύει αυτό και ο παίκτης δεν έχει ξεκινήσει τη συνομιλία και το σκορ είναι μηδενικό, τότε ενεργοποιείται η εμφάνιση του μηνύματος για 3 δευτερόλεπτα. Ανάλογα με την κατάσταση του παιχνιδιού, το κείμενο του αλλάζει:

- Αν ο παίκτης έχει ήδη μιλήσει με κάποιον NPC και έχει εμφανιστεί το αντικείμενό του προς συλλογή, τότε ενεργοποιείται μήνυμα που καθοδηγεί τον παίκτη να το βρει.
- Αν ο παίκτης δεν έχει μιλήσει με κανέναν ή έχει μιλήσει, αλλά δεν έχει εμφανιστεί αντικείμενο, τότε το κείμενο υποδεικνύει στον παίκτη να συνομιλήσει με τον συγκεκριμένο NPC. Η ονομασία διαφοροποιείται σε Lady of Wisdom για τους θηλυκούς χαρακτήρες, ενώ για τους υπόλοιπους χρησιμοποιείται το Lord of Wisdom.

Αν ο παίκτης δεν βρίσκεται κοντά σε κανέναν NPC, τότε το βοηθητικό μήνυμα προτρέπει τον παίκτη να τους βρει χρησιμοποιώντας το mini-map.

```
def draw_chat_hint(screen, inventory_surface):
    inventory_height = inventory_surface.get_height()
    chat_hint_height = inventory_height/2
        hint_text = ""
        show_chat_hint = False
        chat_hint_timer = 0
        font = pygame.font.Font(None, 24)
        is_near_npc = False
        for npc in npcs.values():
               if distance_from_npc(npc["pos"], player_pos, 10,
            tile_size):
                is_near_npc = True
                if not chat_active and score == 0:
                    show_chat_hint = True
                    #show hint for 3 seconds
                    chat_hint_timer = 3
                break
        if is_near_npc:
            if clicked_npcs != 0 and collectable_generated:
              hint_text="Find and collect the generated item"
            elif clicked_npcs == 0 or (clicked_npcs != 0 and
            not collectable_generated):
                current_npc_name = npc["npc_name"]
                if current_npc_name in ["Marie Curie", "Rachel
               Carson","Frida Kahlo"]:
                 hint_text = f"Click on the Lady of Wisdom
                 {current_npc_name} to chat. You must answer
                 all the questions she asks you. "
                else:
                    hint_text = f"Click on the Lord of Wisdom
                   {current_npc_name} to chat. You must
                   answer all the questions he asks you."
        else:
            if clicked_npcs == 0 or (clicked_npcs != 0 and
            not collectable_generated):
                hint_text = "Your mission is to find the Lords
               of Wisdom.Navigate or click on them on
               minimap."
                chat_hint_timer = 3
```

Εισαγωγική Σκηνή (Intro)

Η εισαγωγική σκηνή του παιχνιδιού, η οποία συνδυάζει αφήγηση, μουσική υπόκρουση, κείμενο και εικόνα, δημιουργείται μέσω της συνάρτησης intro(). Στην αρχή, φορτώνονται οι απαραίτητοι πόροι, όπως η μουσική υπόκρουση και οι εικόνες για να παρουσιαστεί η ιστορία στον παίκτη. Η σκηνή αυτή περιλαμβάνει επίσης ένα κουμπί Skip για να μπορεί ο παίκτης να παραλείψει την εισαγωγή ανά πάσα στιγμή.

```
        font_path =resource_path( "sprites/AnonymousPro.ttf")
        font = pygame.font.Font(font_path, 20)
        intro_music_path = resource_path("sprites/intro.mp3")
        story_image=pygame.image.load(resource_path
      ("sprites/story.png"))
        #scale the image to fit screen
        story_image=.transform.scale(story_image,(700, 700))
```

Έπειτα, με τη χρήση του ήχου πραγματοποιείται η αφήγηση με μουσική υπόκρουση στο παρασκήνιο.

```
      pygame.mixer.init()
      intro_music_path = resource_path("sprites/intro.mp3")
      pygame.mixer.music.load(intro_music_path)
      pygame.mixer.music.set_volume(0.02)
      pygame.mixer.music.play(1)
      narraror_voice_path=resource_path("sprites/narrator.mp3")
      narraror_voice = pygame.mixer.Sound(narraror_voice_path)
```

Το κείμενο της ιστορίας εμφανίζεται διαδοχικά μέσω της λίστας text_blocks, όπου κάθε στοιχείο περιλαμβάνει το κείμενο της αφήγησης και τη διάρκεια εμφάνισης σε χιλιοστά του δευτερολέπτου. Ο συγχρονισμός του κειμένου με τον ήχο της αφήγησης συγχρονίζεται με τη χρήση του pygame.time.get_ticks().

```
      text_blocks = [
         ("The hero …,5000),
         ("Their journey …,6000),…
       for text, duration in text_blocks:
         start_time = pygame.time.get_ticks()
         while pygame.time.get_ticks() - start_time < duration:
…          text_surface = font.render(text, True, WHITE)
           text_rect=text_surface.get_rect(midtop=(WIDTH//2, 20))
                screen.blit(text_surface, text_rect)
```

Τέλος, δίνεται δυνατότητα στο χρήστη να παρακάμψει την εισαγωγή του παιχνιδιού, είτε κάνοντας κλικ με το ποντίκι πάνω στο κουμπί Skip, είτε πατώντας το πλήκτρο Enter. Σταματάει να παράγεται η φωνή του αφηγητή και να τρέχει η intro().

```
    elif event.type == pygame.MOUSEBUTTONDOWN:
       mouse_x, mouse_y = event.pos
      if button_x<=mouse_x <= button_x + button_width and button_y
      <= mouse_y <= button_y + button_height:
      #stop the narrator's sound via click on skip button
         narraror_voice.stop()
         running = False
      elif event.type == pygame.KEYDOWN:
         if event.key == pygame.K_RETURN:
       #stop the narrator's sound via pressing on enter key
            narraror_voice.stop()
            running = False
```

Επιλογή Χαρακτήρα

Η επιλογή του χαρακτήρα γίνεται μέσω της συνάρτησης choose_hero(), η οποία επιτρέπει στον παίκτη να επιλέξει μεταξύ 2 ηρώων που αναπαριστούν το χαρακτήρα του παίκτη. Φορτώνονται 2 διαφορετικές εικόνες και ορίζονται τα μονοπάτια τους.

```
def choose_hero(screen, font):
  hero1_image_path=resource_path('sprites/hero/male/
  male.png')
  hero2_image_path=resource_path('sprites/hero/female/
  female.png')
  hero1_image = pygame.image.load(hero1_image_path)
  hero2_image = pygame.image.load(hero2_image_path)
```

Η επιλογή του ήρωα γίνεται είτε μέσω πληκτρολογίου (αριστερό/δεξί βέλος για πλοήγηση και Enter/Space για επιλογή) είτε με τη χρήση του ποντικιού (κλικ στην εικόνα του ήρωα). Αναλόγως με το ποια εικόνα θα επιλέξει ο χρήστης επιστρέφεται και η αντίστοιχη εικόνα του ήρωα που θα χρησιμοποιηθεί στο παιχνίδι.

```
    elif event.type == pygame.KEYDOWN:
    #select a hero using the arrow keys
        if event.key == pygame.K_LEFT:
           selected_hero = 0 # hero 1
        elif event.key == pygame.K_RIGHT:
           selected_hero = 1 # hero 2
        elif event.key == pygame.K_RETURN or event.key ==
        pygame.K_SPACE:
            #select the hero
            if selected_hero == 0:
              print("Hero 1 selected!")
              return hero1_image_path
             else:
               print("Hero 2 selected!")
               return hero2_image_path
              #use also the mouse to choose the hero
     elif event.type == pygame.MOUSEBUTTONDOWN:
       mouse_x, mouse_y = event.pos
       if hero1_rect.collidepoint(mouse_x, mouse_y):
          print("Hero 1 selected!")
          return hero1_image_path
        elif hero2_rect.collidepoint(mouse_x, mouse_y):
           print("Hero 2 selected!")
           return hero2_image_path
```

Δημιουργία Αρχείου Καταγραφής Ερωτήσεων (db_to_text.py)

Για την ενίσχυση της εκπαιδευτικής αξίας του παιχνιδιού, δημιουργήθηκε το αρχείο db_to_text.py, το οποίο εξάγει τις ερωτήσεις και απαντήσεις που έχουν δοθεί από τον παίκτη

σε μορφή αρχείου κειμένου. Το αρχείο περιλαμβάνει πληροφορίες από τη βάση δεδομένων και είναι οργανωμένο ανά θεματική ενότητα.

Αρχικά ανακτώνται τα απαραίτητα δεδομένα, ερωτήσεις, απαντήσεις και θεματικές από τη βάση δεδομένων, με βάση το id του χρήστη. Γίνεται χρήση του INNER JOIN ώστε να συνδεθούν οι δύο πίνακες της βάσης δεδομένων, user_questions (ο πίνακας που καταγράφει τις ερωτήσεις που απάντησε κάθε χρήστης) και questions (ο πίνακας που περιέχει τις ερωτήσεις και τις απαντήσεις τους).

```
        cursor.execute('''
            SELECT q.topic, q.question, q.answer
            FROM user_questions uq
            INNER JOIN questions q ON uq.question_id = q.id
            WHERE uq.user_id = ?
        ''', (user_id,))
```

Με τη χρήση της open()[18] σε λειτουργία εγγραφής ('w'), δημιουργείται το αρχείο και γράφεται στην κορυφή του μια επικεφαλίδα με το id του χρήστη και τον τίτλο "Questions and Answers", ακολουθούμενη από μια γραμμή διαχωρισμού, ώστε να είναι πιο οργανωμένο και ευπαρουσίαστο. Ελέγχεται αν υπάρχουν διαθέσιμες εγγραφές ερωτήσεων και απαντήσεων. Αν ναι, οι πληροφορίες οργανώνονται ανά θέμα χρησιμοποιώντας το λεξικό topic_questions. Για κάθε θεματική, καταγράφεται ο τίτλος της και οι ερωτήσεις-απαντήσεις που περιλαμβάνει.

```
    records = cursor.fetchall()
        if records:
            file_path = f'user_{user_id}_answers.txt'
            with open(file_path, 'w') as file:
                file.write(f"Users:{user_id}Questions and
                    Answers\n")
                file.write("_____________________\n\n")
                topic_questions = {}
                for topic, question, answer in records:
                  if topic not in topic_questions:
                    topic_questions[topic] = []
                    topic_questions[topic].append((question,
                    answer))
                for topic, qas in topic_questions.items():
                    file.write(f"Topic: {topic}\n")
                    file.write("_____________________\n")
                    for question, answer in qas:
                      file.write(f"Question:{question}\nAnswer:
                      {answer}\n\n")
```

[18] https://www.geeksforgeeks.org/writing-to-file-in-python/

Δημιουργία εκτελέσιμου αρχείου (EXE)

Για να μπορέσει να γίνει εξαγωγή του αρχείου σε εκτελέσιμη μορφή exe δημιουργήθηκε το αρχείο utils.py. Σε αυτό το αρχείο περιλαμβάνεται η συνάρτηση resource_path(), η οποία διευκολύνει τη διαχείριση διαδρομών αρχείων. Η συνάρτηση ελέγχει αν η εφαρμογή τρέχει ως exe χρησιμοποιώντας το sys._MEIPASS, μια προσωρινή τοποθεσία που χρησιμοποιείται από το PyInstaller. Σε αυτήν την περίπτωση, ορίζει ως base_path τη διαδρομή του πακέτου. Εάν τρέχει ως κανονικό Python script, χρησιμοποιεί το τρέχον directory (os.path.abspath(".")) ως base_path[19]. Αυτό εξασφαλίζει ότι οι διαδρομές των αρχείων, όπως εικόνες, ήχοι ή άλλα αρχεία (π.χ. py), λειτουργούν σωστά και στις δύο περιπτώσεις[20].

```
        def resource_path(relative_path):
            try:
                base_path = sys._MEIPASS #when running as an exe
            except Exception:
                #when running as a Python script
                base_path = os.path.abspath(".")
            return os.path.join(base_path, relative_path)
```

Έτσι όλα τα αρχεία φορτώνονται μέσα από τη συνάρτηση αυτή π.χ.

```
background_music=resource_path('sounds/background_music.mp3')
enemy_image_path_1 = resource_path('sprites/enemy1.png')
subprocess.Popen(["python", resource_path("api.py")], shell=True)
```

Για να ολοκληρωθεί η διαδικασία της μετατροπής όλης της εφαρμογής του παιχνιδιού σε εκτελέσιμο αρχείο exe δόθηκαν οι παρακάτω εντολές στο terminal του Visual Studio Code.

```
1)-m pip install requests
2)-m pip install pyinstaller
3)-m pip install -U pip setuptools
4)pyinstaller --onefile --name "A Quest Of Knowledge" --add-data "sprites;sprites"  --add-data
"sounds;sounds"--add-data "db_to_text.py;." --add-data "level.py;." --add-data "adduser.py;." -
-add-data "db.py;." --add-data "db_utils.py;." --add-data "utils.py;." --add-data "api.py;." -
-add-data "chat.py;." menu.py
```

Η τελευταία εντολή αποτελεί ουσιαστικά την τελική εντολή που δημιουργεί ένα εκτελέσιμο αρχείο με το όνομα "A Quest Of Knowledge". Το --onefile διασφαλίζει ότι όλα συμπεριλαμβάνονται σε ένα μόνο αρχείο exe. Στην περίπτωση που δεν είναι επιθυμητή η εμφάνιση του παράθυρου κονσόλας, πρέπει να προστεθεί το flag --noconsole.

```
pyinstaller --onefile --name "A Quest Of Knowledge" --noconsole --add-data "sprites;sprites" -
-add-data "sounds;sounds" --add-data "db_to_text.py;." --add-data "level.py;." --add-data
"adduser.py;." --add-data "db.py;." --add-data "db_utils.py;." --add-data "utils.py;." --add-
data "api.py;." --add-data "chat.py;." menu.py
```

[19] https://www.enablegeek.com/tutorial/python-remove-extension-from-filename/
[20] https://stackoverflow.com/questions/31836104/pyinstaller-and-onefile-how-to-include-an-image-in-the-exe-file

5. Ερευνητικός σχεδιασμός και Μεθοδολογία εργασίας

Στο παρόν κεφάλαιο περιγράφεται η ερευνητική μεθοδολογία που ακολουθήθηκε για τη διεξαγωγή της αξιολόγησης του παιχνιδιού "A Quest of Knowledge". Η μέθοδος αυτή περιλάμβανε τη χρήση ενός δομημένου ερωτηματολογίου, το οποίο επιτρέπει τη συλλογή ποσοτικών δεδομένων σχετικά με την εμπειρία των χρηστών από τη χρήση του παιχνιδιού. Στη συνέχεια, αναφέρεται η συμμόρφωση με τον Κώδικα Δεοντολογίας και Βιοηθικής του Πανεπιστημίου Δυτικής Μακεδονίας (Πανεπιστήμιο Δυτικής Μακεδονίας, 2018), διασφαλίζοντας την ηθική ακεραιότητα της έρευνας και την προστασία των συμμετεχόντων. Στο τελευταίο μέρος του κεφαλαίου, παρουσιάζονται τα αποτελέσματα της έρευνας, με την ανάλυση των δεδομένων που συγκεντρώθηκαν μέσω του ερωτηματολογίου.

5.1. Μεθοδολογία Έρευνας

Για την αξιολόγηση του παιχνιδιού "A Quest of Knowledge", υιοθετήθηκε μια ποσοτική μεθοδολογική προσέγγιση, χρησιμοποιώντας δομημένο ερωτηματολόγιο για τη συλλογή δεδομένων από τους χρήστες. Το ερωτηματολόγιο περιλάμβανε κλειστού τύπου ερωτήσεις με κλίμακες Likert[21] και επιλογές πολλαπλής επιλογής, επιτρέποντας τη συγκέντρωση αριθμητικών δεδομένων για στατιστική ανάλυση. Αυτή η μέθοδος διευκολύνει την αντικειμενική μέτρηση της εμπειρίας των χρηστών και την εξαγωγή γενικεύσιμων συμπερασμάτων.

Η ποσοτική αυτή προσέγγιση επιτρέπει την ανάλυση των απαντήσεων με τη χρήση στατιστικών εργαλείων, παρέχοντας σαφή εικόνα για τη χρηστικότητα, την εκπαιδευτική αξία και τη διαδραστικότητα του παιχνιδιού. Επιπλέον, η συλλογή δημογραφικών στοιχείων επιτρέπει τη διερεύνηση πιθανών συσχετίσεων μεταξύ χαρακτηριστικών των χρηστών και της εμπειρίας τους με το παιχνίδι, ενισχύοντας την κατανόηση των αναγκών και των προτιμήσεών τους.

Συμμόρφωση με τον Κώδικα Δεοντολογίας και Καλής Πρακτικής

Η διεξαγωγή της παρούσας έρευνας και η συλλογή δεδομένων μέσω του ερωτηματολογίου πραγματοποιήθηκαν σύμφωνα με τον Κώδικα Δεοντολογίας και Καλής Πρακτικής του Πανεπιστημίου Δυτικής Μακεδονίας (Κεφάλαιο 9: Ειδικότεροι Κανόνες Επιστημονικής και Καλλιτεχνικής Έρευνας, 9.1 Δεσμεύσεις Ερευνητών), διασφαλίζοντας την ηθική ακεραιότητα της ερευνητικής διαδικασίας, την προστασία των συμμετεχόντων και τη διαφάνεια στην παρουσίαση των αποτελεσμάτων.

Σεβασμός στη Βιοηθική και τα Ανθρώπινα Δικαιώματα

Η ερευνητική δραστηριότητα πραγματοποιήθηκε με απόλυτο σεβασμό στη βιοηθική, τη σωματική και πνευματική υπόσταση του ανθρώπου, καθώς και στο φυσικό περιβάλλον. Τηρήθηκαν οι γενικά αναγνωρισμένες αρχές:

- Αξία του ανθρώπου και προστασία των ανθρωπίνων δικαιωμάτων.
- Ελευθερία και ισότητα.

[21]Μέτρηση θετικής ή αρνητικής απάντησης σε κλίμακα από 1 έως 5

- Προστασία της δημόσιας υγείας.
- Προστασία του παιδιού και των ευαίσθητων ομάδων.
- Προστασία των προσωπικών δεδομένων.
- Προστασία της βιοποικιλότητας.

Δεσμεύσεις Ερευνητών

Σύμφωνα με τον Κώδικα Δεοντολογίας, κατά τη διεξαγωγή της παρούσας έρευνας:

- Ακολουθήθηκε η ισχύουσα νομοθεσία και υπήρξε συνεχής ενημέρωση για τις κατευθυντήριες οδηγίες που αφορούν την έρευνα.
- Αναλήφθηκε προσωπική ευθύνη για όλες τις ερευνητικές πράξεις, σύμφωνα με την ισχύουσα νομοθεσία και τις διεθνείς διακηρύξεις για τη βιοηθική και τα ανθρώπινα δικαιώματα.
- Λήφθηκε συγκατάθεση κατόπιν ενημέρωσης από όλους τους συμμετέχοντες, οι οποίοι είχαν το δικαίωμα της ελεύθερης ανάκλησης της συγκατάθεσής τους ή της αποχώρησης από την έρευνα.
- Διασφαλίστηκε η προστασία των προσωπικών δεδομένων, με απόλυτο σεβασμό στις φυλετικές, έμφυλες, θρησκευτικές, πολιτικές και πολιτισμικές ιδιαιτερότητες των συμμετεχόντων.
- Διατηρήθηκε αντικειμενικότητα και αποφεύχθηκε κάθε είδους επηρεασμός από κοινωνικούς, πολιτικούς ή οικονομικούς παράγοντες.
- Παρουσιάστηκαν με διαφάνεια τα αποτελέσματα της μελέτης χωρίς απόκρυψη ή αλλοίωση και τηρήθηκαν πλήρη αρχεία για την εξέλιξη και τα αποτελέσματα του ερευνητικού προγράμματος.
- Τηρήθηκαν οι κανόνες ασφαλείας του Πανεπιστημίου Δυτικής Μακεδονίας στους χώρους της έρευνας, ενώ ακολουθήθηκαν οι αρχές της χρηστής, διαφανούς και αποτελεσματικής χρηματοοικονομικής διαχείρισης.
- Αποφεύχθηκε η σύγκρουση συμφερόντων κατά την αξιολόγηση της έρευνας, και δεν λήφθηκαν αποφάσεις που θα μπορούσαν να επηρεαστούν από προσωπικές ή επαγγελματικές σχέσεις. (Πανεπιστήμιο Δυτικής Μακεδονίας, 2018)

Προστασία Ανηλίκων και Ευάλωτων Ομάδων

Η έρευνα συμμορφώθηκε με τις ειδικές απαιτήσεις που αφορούν τη συμμετοχή παιδιών και εφήβων, λαμβάνοντας υπόψη:

- Τις διαφορετικές αντιλήψεις και ικανότητες των παιδιών και εφήβων σε σχέση με τους ενήλικες.
- Την ευαλωτότητα των παιδιών σε πιθανή εκμετάλλευση κατά την αλληλεπίδρασή τους με ενήλικες.
- Την άνιση σχέση ισχύος μεταξύ ενήλικα ερευνητή και παιδιών/εφήβων που συμμετέχουν στην έρευνα.
- Τη διασφάλιση εχεμύθειας και αποφυγή κοινοποίησης εμπιστευτικών πληροφοριών.
- Την αυστηρή τήρηση της δεοντολογίας ως προς την ενημέρωση και τη συγκατάθεση τόσο του παιδιού/εφήβου όσο και του γονέα για τη συμμετοχή στην έρευνα.

5.2. Αποτελέσματα Έρευνας

Στην παρούσα ενότητα παρουσιάζονται τα αποτελέσματα της έρευνας που πραγματοποιήθηκε σχετικά με την εμπειρία των χρηστών στο serious game. Σκοπός της ανάλυσης είναι να αποτυπωθούν τα δημογραφικά χαρακτηριστικά των συμμετεχόντων, η εμπειρία τους με το παιχνίδι, καθώς και οι αξιολογήσεις τους ως προς διάφορες πτυχές του.

Ο μέσος όρος ηλικίας των χρηστών που συμμετείχαν στην έρευνα είναι 32 έτη.

Όσον αφορά το φύλο, το 55% των συμμετεχόντων ήταν άνδρες, το 45% γυναίκες.

Το εκπαιδευτικό επίπεδο των συμμετεχόντων κατανέμεται ως εξής:

- 40% το Πανεπιστήμιο και
- 60% έχουν Μεταπτυχιακό ή Διδακτορικό τίτλο.

Όσον αφορά την εμπειρία με serious games, το 75% των χρηστών δήλωσε ότι έχει προηγούμενη εμπειρία, ενώ το 25% όχι.

Η συνολική εμπειρία του παιχνιδιού αξιολογήθηκε κατά μέσο όρο ως Εξαιρετική με βαθμολογία 4,75 στα 5.

Η ευκολία πλοήγησης και χειρισμού αξιολογήθηκε κατά μέσο όρο με βαθμολογία 4,9 στα 5.

Το παιχνίδι θεωρήθηκε πολύ διασκεδαστικό από τους χρήστες, με μέσο όρο βαθμολογίας 4,6 στα 5.

Οι ερωτήσεις του παιχνιδιού αξιολογήθηκαν ως ενδιαφέρουσες και εκπαιδευτικές με μέσο όρο 4,7 στα 5.

Το περιβάλλον του παιχνιδιού όσον αφορά τα γραφικά, τον ήχο και το γενικό στυλ αξιολογήθηκε με μέσο όρο 4,55 στα 5.

Οι οδηγίες και τα μηνύματα του παιχνιδιού κρίθηκαν κατανοητά με βαθμολογία 4,85 στα 5.

Η εμπειρία συνομιλίας (Chat Mode) με τους NPCs αξιολογήθηκε κατά μέσο όρο ως 4,75 στα 5.

Η χρησιμότητα του mini map ως εργαλείο πλοήγησης βαθμολογήθηκε με 4,7 στα 5.

Η πιθανότητα να ξαναπαίξουν οι χρήστες το παιχνίδι με διαφορετικές παραμέτρους αξιολογήθηκε με 4,7 στα 5.

Όσον αφορά τη βελτίωση των γνώσεων, το 20% των συμμετεχόντων θεώρησε ότι το παιχνίδι βελτίωσε πολύ τις γνώσεις του, το 70% αρκετά, το 5% μέτρια, το 0% λίγο και το 5% καθόλου.

Η χρήση της τεχνητής νοημοσύνης στους NPCs αξιολογήθηκε κατά μέσο όρο ως Εξαιρετική.

Η προσαρμογή της δυσκολίας του παιχνιδιού κρίθηκε αποτελεσματική από το 85% των συμμετεχόντων, ενώ το 5% δήλωσε ότι θα μπορούσε να βελτιωθεί. Το 10% θεώρησε ότι δεν ήταν ιδιαίτερα αποτελεσματική και το 0% ότι δεν ήταν καθόλου αποτελεσματική.

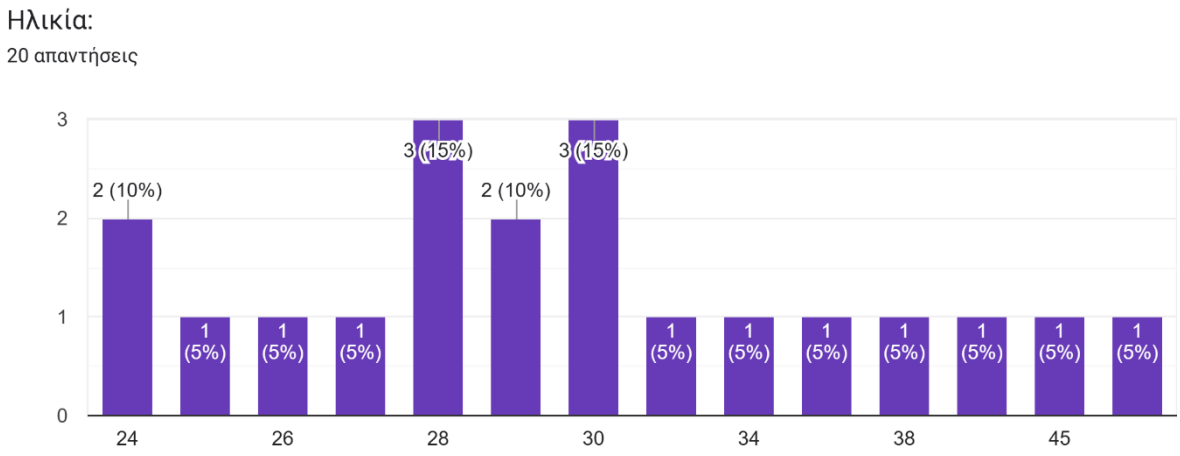
Το 95% των χρηστών θεώρησε ότι οι διαθέσιμες εκπαιδευτικές ενότητες κάλυψαν επαρκώς τις ανάγκες τους, ενώ το 5% όχι.

Το 55% των χρηστών πρότεινε την προσθήκη επιπλέον θεματικών ενοτήτων ή ενσωμάτωση άλλου είδους εκπαιδευτικού περιεχομένου.

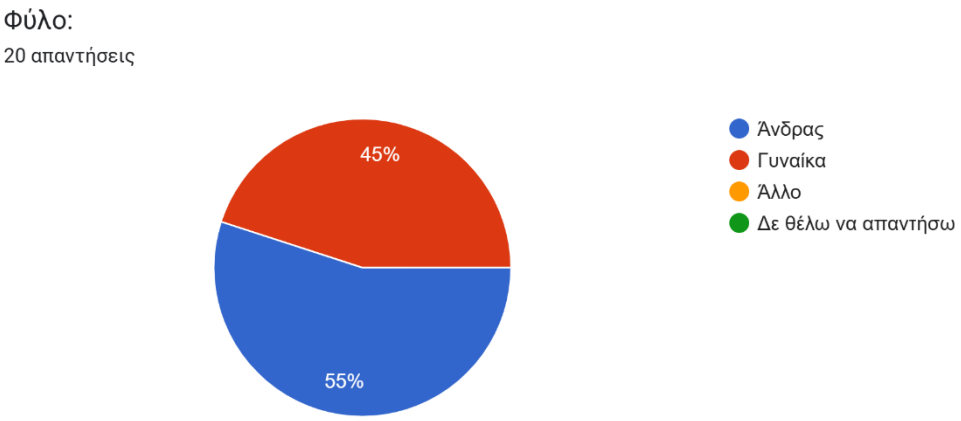
Συνολικά, το 90% των χρηστών θα πρότεινε το παιχνίδι σε άλλους, το 10% δήλωσε ίσως και το 0% όχι.

Στα ακόλουθα γραφήματα αποτυπώνονται οπτικά τα αποτελέσματα της έρευνας, διευκολύνοντας τη σύγκριση και την εξαγωγή συμπερασμάτων. Οι αναλύσεις παρουσιάζονται μέσω διαγραμμάτων που δείχνουν τις απαντήσεις των συμμετεχόντων στα ζητήματα που αξιολογήθηκαν στο πλαίσιο της μελέτης.

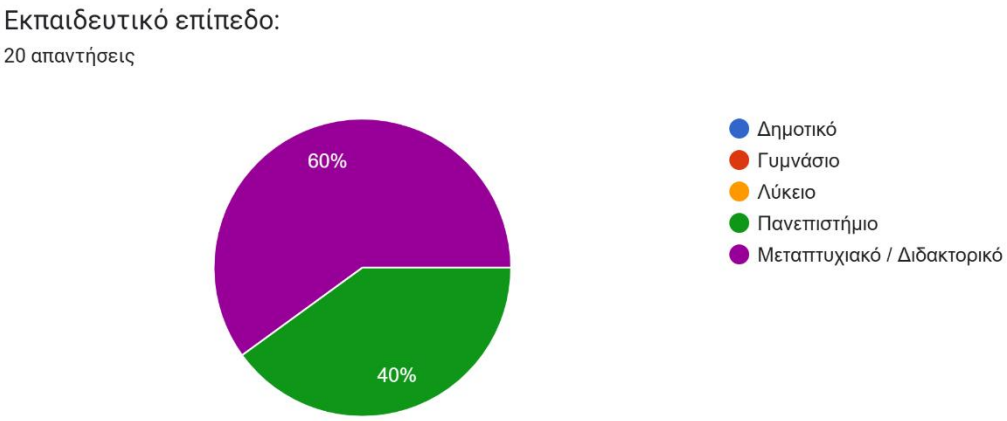
Μέρος 1: Δημογραφικά Στοιχεία



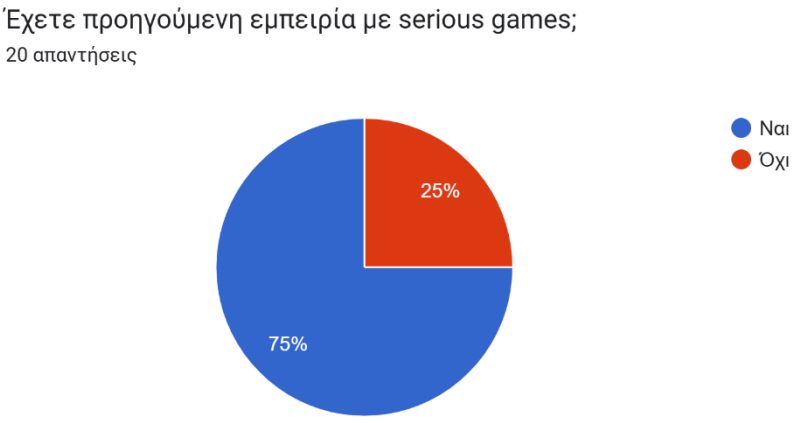
Γράφημα 1.Απαντήσεις 1ης ερώτησης



Γράφημα 2.Απαντήσεις 2ης ερώτησης



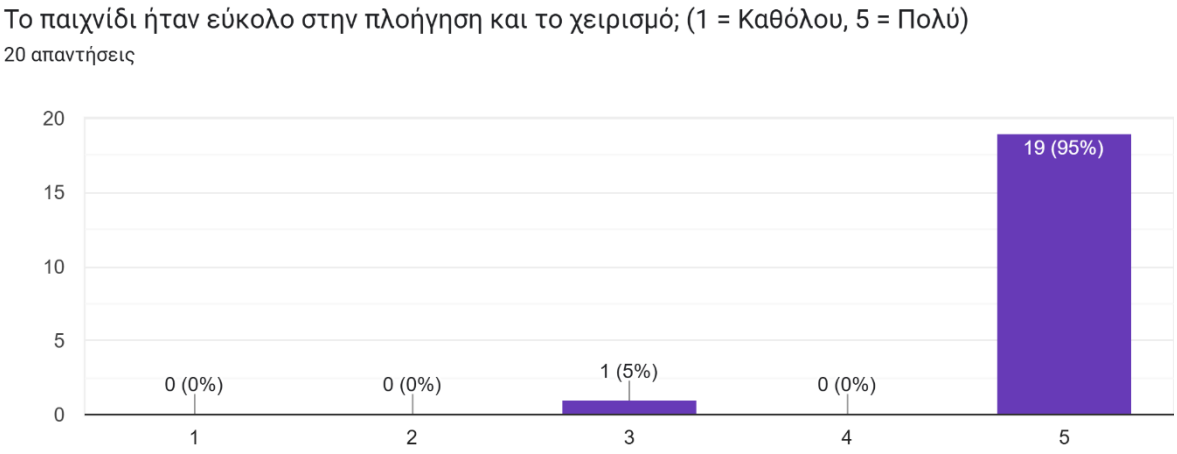
Γράφημα 3.Απαντήσεις 3ης ερώτησης



Γράφημα 4.Απαντήσεις 4ης ερώτησης

Μέρος 2: Χρηστικότητα & Εμπειρία Παιχνιδιού

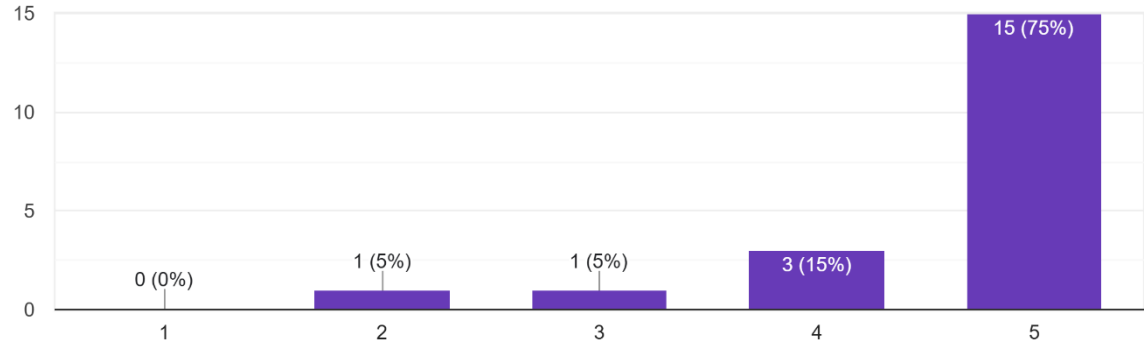
Γράφημα 5.Απαντήσεις 5ης ερώτησης



Γράφημα 6.Απαντήσεις 6ης ερώτησης

Πόσο διασκεδαστικό βρήκατε το παιχνίδι; (1 = Καθόλου, 5 = Πολύ)

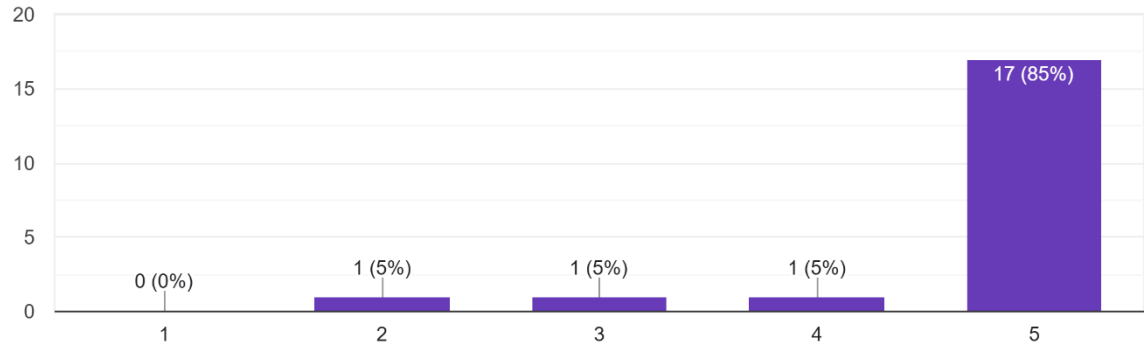
20 απαντήσεις



Γράφημα 7.Απαντήσεις 7ης ερώτησης

Βρήκατε τις ερωτήσεις του παιχνιδιού ενδιαφέρουσες και εκπαιδευτικές; (1 = Καθόλου, 5 = Πολύ)

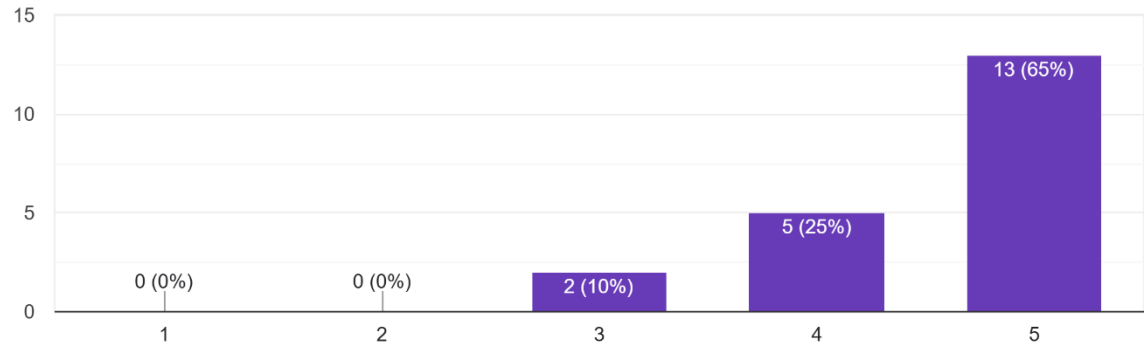
20 απαντήσεις



Γράφημα 8.Απαντήσεις 8ης ερώτησης

Πώς αξιολογείτε το περιβάλλον του παιχνιδιού όσον αφορά τα γραφικά, τον ήχο και το γενικό στυλ;(1 = Πολύ χαμηλά, 5 = Εξαιρετικά)

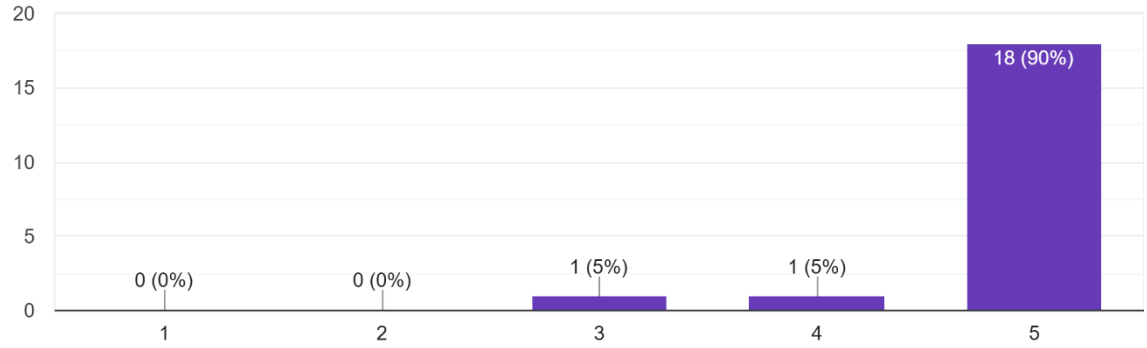
20 απαντήσεις



Γράφημα 9.Απαντήσεις 9ης ερώτησης

Πόσο κατανοητές ήταν οι οδηγίες και τα μηνύματα του παιχνιδιού;(1 = Καθόλου κατανοητές, 5 = Απόλυτα κατανοητές)

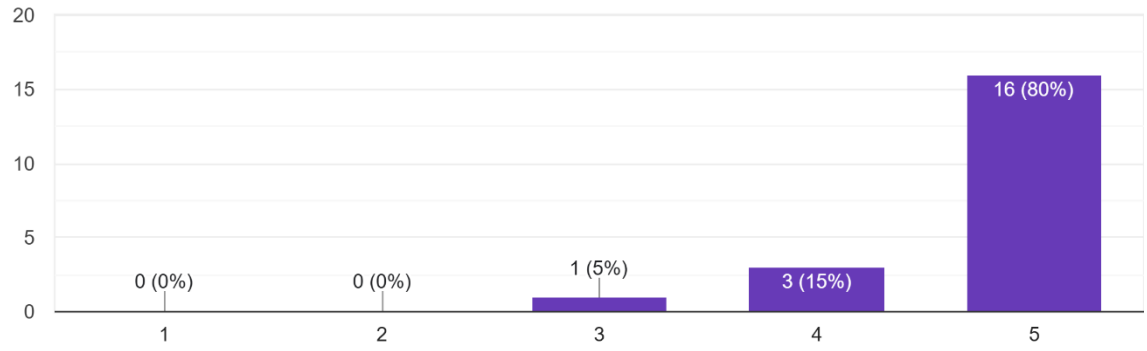
20 απαντήσεις



Γράφημα 10.Απαντήσεις 10ης ερώτησης

Πώς θα αξιολογούσατε την εμπειρία συνομιλίας (Chat Mode) με τους NPCs;(1 = Ανεπαρκής, 5 = Εξαιρετική)

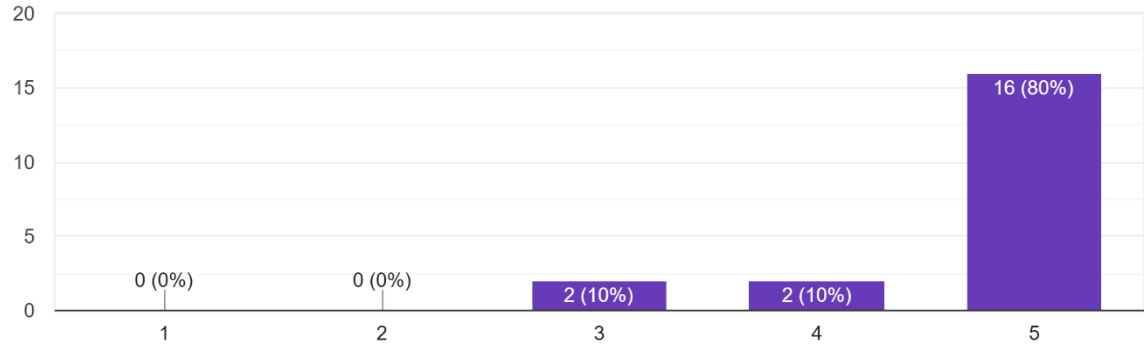
20 απαντήσεις



Γράφημα 11.Απαντήσεις 11ης ερώτησης

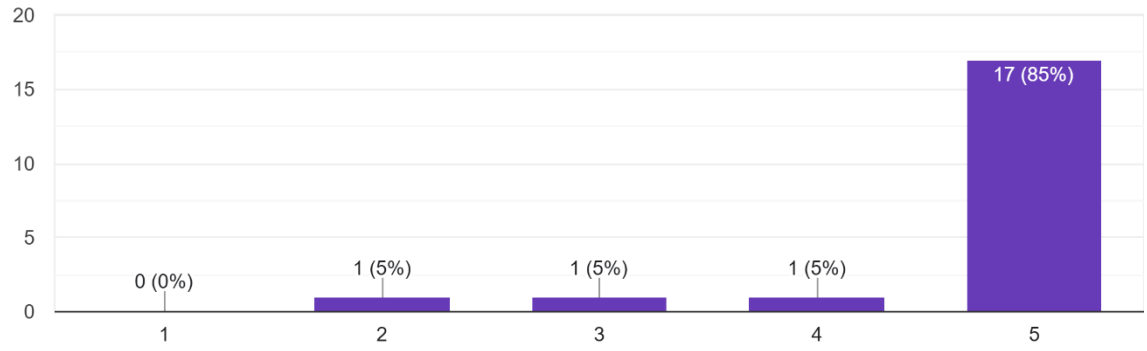
Πόσο αποτελεσματικό θεωρείτε το mini-map ως εργαλείο πλοήγησης;(1 = Καθόλου χρήσιμο, 5 = Πολύ χρήσιμο)

20 απαντήσεις



Γράφημα 12.Απαντήσεις 12ης ερώτησης

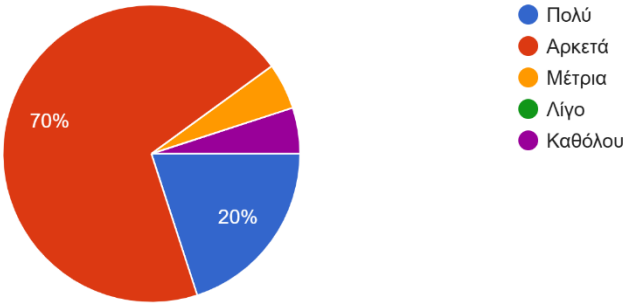
Πόσο πιθανό είναι να ξαναπαίξετε το παιχνίδι με διαφορετικές παραμέτρους (π.χ. αριθμό ερωτήσεων, επιλογή χαρακτήρα, δήλωση ηλικίας);(1 = Καθόλου πιθανό, 5 = Πολύ πιθανό)
20 απαντήσεις



Γράφημα 13.Απαντήσεις 13ης ερώτησης

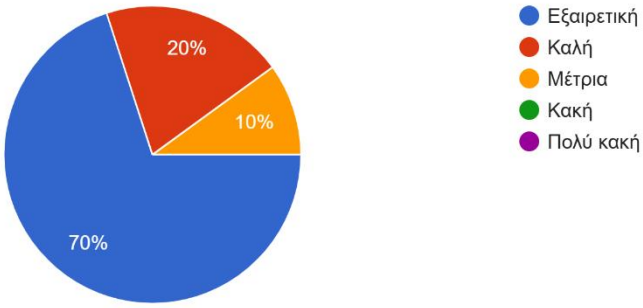
Μέρος 3: Εκπαιδευτική Αξία & Χρήση Τεχνητής Νοημοσύνης

Πιστεύετε ότι το παιχνίδι βελτίωσε τις γνώσεις σας;
20 απαντήσεις



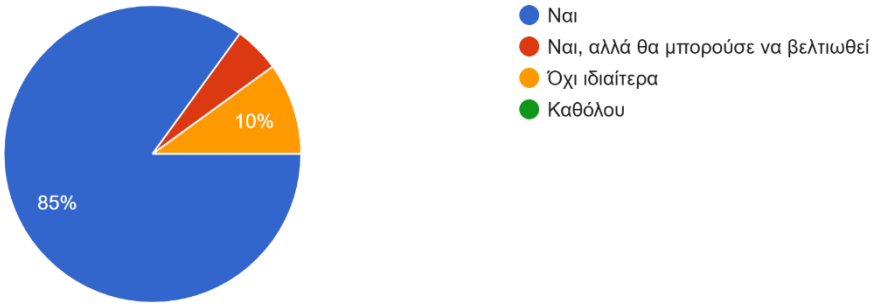
Γράφημα 14.Απαντήσεις 14ης ερώτησης

Πώς αξιολογείτε τη χρήση της τεχνητής νοημοσύνης στους NPCs;
20 απαντήσεις



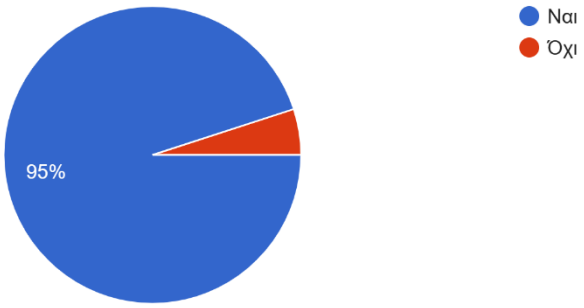
Γράφημα 15.Απαντήσεις 15ης ερώτησης

Πιστεύετε ότι η προσαρμογή της δυσκολίας (με βάση την ηλικία και τον αριθμό ερωτήσεων) ήταν αποτελεσματική;
20 απαντήσεις



Γράφημα 16.Απαντήσεις 16ης ερώτησης

Θεωρείτε ότι οι διαθέσιμες εκπαιδευτικές ενότητες (Μαθηματικά, Γεωγραφία, Αστρονομία, Πληροφορική, Φυσική, Χημεία, Ιστορία, Τέχνες, Μ... Βιολογία) καλύπτουν επαρκώς τις ανάγκες σας;
20 απαντήσεις



Γράφημα 17.Απαντήσεις 17ης ερώτησης

Θα προτείνατε την προσθήκη επιπλέον θεματικών ενοτήτων ή ενσωμάτωση άλλου είδους εκπαιδευτικού περιεχομένου;

11 απαντήσεις

Οποιαδήποτε εκπαιδευτική ενότητα σχετική με τη μάθηση.

-

πολιτική

Ανάλυση λογοτεχνικών έργων, δημιουργία ιστοριών και ποίησης.

Extra Mini games, όπως παιχνίδια μνήμης ή παζλ με βάση τις θεματικές

Ορισμός θεματικών από το χρήστη

Ψυχολογία & Κοινωνιολογία: Ερωτήσεις σχετικά με ανθρώπινη συμπεριφορά, κοινωνικές δομές, συναισθήματα.

Έχει καλυφθεί ένα αρκετά μεγάλο μέρος θεματικών ενοτήτων

Περιεχόμενο σχετικό με την υγεία

ΟΧΙ

Όχι

Μέρος 4: Βελτιώσεις & Σχόλια

Τι θα προτείνατε για να γίνει το παιχνίδι πιο διασκεδαστικό ή εκπαιδευτικό;

8 απαντήσεις

Περισσότερα χρώματα

Περισσότερη μουσική επένδυση

να συμπεριληφθούν ίσως και κάποιες ερωτήσεις πολιτικού περιεχομένου

mini games για έξτρα πόντους

Αναβάθμιση avatar, εξαργύρωση πόντων για εξοπλισμό

Δημιουργία ερωτήσεων από παίκτες

Ένα combat system που να βασίζεται στις ερωτήσεις

Τίποτα καθώς ήταν εξαιρετικό

Υπήρχε κάποιο τεχνικό ή σχεδιαστικό πρόβλημα που συναντήσατε; Αν ναι, περιγράψτε το.

8 απαντήσεις

Όχι

Οχι

-

δεν συνάντησα κάποιο πρόβλημα

Όχι

Ποια σημεία του παιχνιδιού θεωρείτε ότι χρειάζονται περαιτέρω βελτίωση (π.χ. έλεγχοι, γραφικά, ήχος, διάλογοι);

7 απαντήσεις

-

Όλα καλά.

Ήχος

θα ήθελα περισσότερους διαλόγους

Για το είδος του παιχνιδιού δεν χρειάζεται κάποια περαιτέρω βελτίωση.

Τίποτα

Ποια ήταν η πιο θετική πτυχή της εμπειρίας σας στο παιχνίδι;

12 απαντήσεις

Ο εύκολος χειρισμός.

Η εκπαιδευτική χροιά του παιχνιδιου

η ανασκόπηση των ιστορικών μου γνώσεων

Συνδύαζε στοιχεία από διαφορετικά είδη παιχνιδιών με επιτυχημένο τρόπο.

Δεν ένιωθα ότι "διάβαζα", αλλά ότι μάθαινα μέσα από το παιχνίδι.

Το παιχνίδι με κράτησε αφοσιωμένο, συνδυάζοντας έξυπνες ερωτήσεις, διαδραστικούς διαλόγους και προκλήσεις

Η καινοτομία με την προσθήκη AI

Ήταν ένα διαφορετικό παιχνίδι γνώσεων, που ξέφυγε από τα όρια των quiz

Κάθε φορά μπορούσα να παίξω με διαφορετικές επιλογές και να έχω μια νέα εμπειρία.

Ήταν διαφορετικό και ρεαλιστικό

Η ποικιλία των ερωτήσεων

Οι μη τυποποιημένοι διάλογοι

Υπήρξε κάποιο σημείο που σας απογοήτευσε ή δυσκολευτήκατε κατά τη διάρκεια του παιχνιδιού;

11 απαντήσεις

Όχι

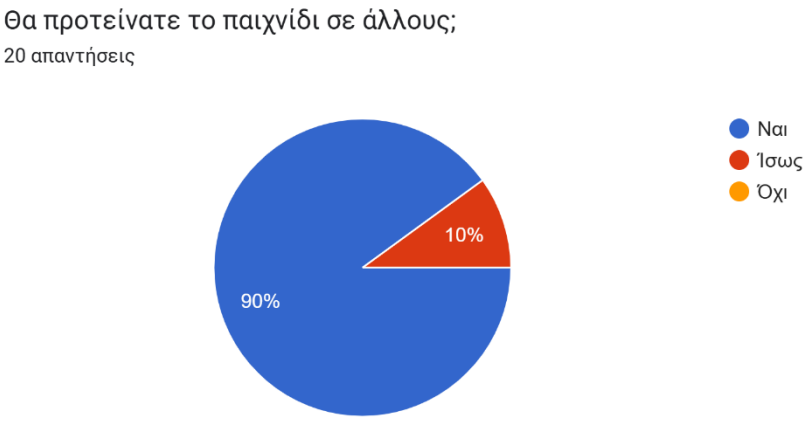
Όχι

όχι

-

οχι

Οχι



Γράφημα 18.Απαντήσεις 24ης ερώτησης

6. Προκλήσεις, Αντιμετώπιση, Συμπεράσματα και Μελλοντικές Προεκτάσεις

Στο παρόν κεφάλαιο αναλύονται οι προκλήσεις που προέκυψαν κατά την ανάπτυξη και υλοποίηση του παιχνιδιού A Quest of Knowledge, οι στρατηγικές που υιοθετήθηκαν για την αποτελεσματική αντιμετώπισή τους, καθώς και τα συμπεράσματα και οι προτάσεις για μελλοντικές επεκτάσεις. Η ενσωμάτωση ενός μεγάλου γλωσσικού μοντέλου (LLM) στην πλατφόρμα Pygame για τη δημιουργία δυναμικών διαλόγων παρουσίασε τεχνικές και οργανωτικές δυσκολίες, οι οποίες αφορούσαν την απόδοση του συστήματος, τη διαχείριση πόρων και τη δομή του κώδικα. Παρά τις προκλήσεις, εφαρμόστηκαν βελτιώσεις και στρατηγικές που εξασφάλισαν τη βέλτιστη εμπειρία χρήστη και τη διατήρηση υψηλής ποιότητας στις αλληλεπιδράσεις. Στη συνέχεια, παρουσιάζεται μια αναλυτική εξέταση των κυριότερων προκλήσεων που αντιμετωπίστηκαν, καθώς και οι λύσεις που εφαρμόστηκαν για την αποτελεσματική διαχείρισή τους. Επιπλέον, προτείνονται μελλοντικές βελτιώσεις και νέες λειτουργίες που μπορούν να ενισχύσουν την εκπαιδευτική αξία και τη διαδραστική εμπειρία των χρηστών, προσφέροντας μια ακόμα πιο ολοκληρωμένη και προσαρμοσμένη εμπειρία μάθησης.

6.1. Προκλήσεις υλοποίησης

Η ενσωμάτωση ενός μεγάλου γλωσσικού μοντέλου(LLM) στην Pygame για τη δημιουργία ενός παιχνιδιού με δυναμικά δημιουργούμενους διαλόγους παρουσίασε μία πληθώρα προκλήσεων που αφορούν την απόδοση, τη διαχείριση μνήμης, τη δομή του κώδικα και την ακρίβεια των αλληλεπιδράσεων. Όσον αφορά την απόδοση, η καθυστέρηση στις απαντήσεις του LLM μπορεί να επηρεάσει την εμπειρία του χρήστη, ενώ η απόδοση του API για τη διαδικασία της επικοινωνίας είναι κρίσιμη για την ταχύτητα της ανταπόκρισης. Επίσης, η χρήση του LLM σε συνδυασμό με τα γραφικά της Pygame απαιτεί προσεκτική διαχείριση των πόρων του συστήματος, καθώς η κατανάλωση μνήμης και ισχύος του επεξεργαστή από το LLM μπορεί να προκαλέσει καθυστερήσεις στο παιχνίδι, επηρεάζοντας την ομαλότητα της εμπειρίας χρήσης.

Από την πλευρά των αλληλεπιδράσεων, η σωστή και ακριβής διατύπωση των απαντήσεων που λαμβάνει ο χρήστης είναι καθοριστική για την ποιότητα της εμπειρίας, με έμφαση στα αντικείμενα των θεματικών και την ακρίβεια των πληροφοριών, με βάση την ηλικία του χρήστη και το διάλογο που έχει ήδη πραγματοποιηθεί. Ταυτόχρονα, εξίσου σημαντικό είναι και το αισθητικό κομμάτι του διαλόγου. Ο σχεδιασμός του user interface (UI) πρέπει να εξασφαλίζει την ομαλή ροή του διαλόγου, χωρίς διακοπές ή προβλήματα στην αναδίπλωση του κειμένου. Η σωστή διαχείριση των δεδομένων του χρήστη σε πραγματικό χρόνο και η διασφάλιση της των δεδομένων κατά την αποθήκευση ή την ανάκτησή τους από τη βάση δεδομένων αποτελούν επίσης σημαντικά ζητήματα, εξασφαλίζοντας τη μη

επαναληψιμότητα των ερωτήσεων, των απαντήσεων ή των βοηθητικών μηνυμάτων που δέχεται ο χρήστης. Τέλος, σημαντική πρόκληση αποτέλεσε και η δομή του κώδικα με τέτοιο τρόπο ώστε να μπορεί εύκολα να δεχτεί νέες προσθήκες, θεματικές, NPCs και γραφικά, χωρίς να απαιτούνται περαιτέρω τροποποιήσεις στο σύνολό του.

6.2. Αντιμετώπιση προκλήσεων

Παρά τις αρχικές προκλήσεις με την απόδοση και τις καθυστερήσεις στις απαντήσεις του LLM, πραγματοποιήθηκαν βελτιστοποιήσεις στο API και τη διαδικασία επικοινωνίας, επιτυγχάνοντας ταχύτερη απόκριση και βελτίωση της εμπειρίας χρήστη. Η διαχείριση των πόρων του συστήματος προσαρμόστηκε έτσι ώστε να επιτρέπει την ομαλή εκτέλεση του παιχνιδιού χωρίς καθυστερήσεις, παρά την απαιτητική κατανάλωση μνήμης και
επεξεργαστικής ισχύος από το LLM.
Συγκεκριμένα:

- **Αποτελεσματική διαχείριση διεργασιών μέσω Subprocess:** Η χρήση του subprocess.check_output για την κλήση του LLM (ollama) επιτρέπει την εκτέλεση αιτημάτων για παραγωγή ερωτήσεων και απαντήσεων σε εξωτερικές διεργασίες, αποφεύγοντας την υπερφόρτωση της κύριας εκτέλεσης και εξασφαλίζοντας τη διαχείριση των υπολογιστικών πόρων χωρίς να εμποδίζονται άλλες λειτουργείες του συστήματος.
- **Διαχείριση Βάσης Δεδομένων (SQLite):** Η αποθήκευση δεδομένων (όπως ερωτήσεις και απαντήσεις χρηστών) στη βάση δεδομένων SQLite εξασφαλίζει ότι οι πόροι χρησιμοποιούνται, χωρίς την ανάγκη επιπλέον εξωτερικών συνδέσεων ή διεργασιών κατά τη διάρκεια της λειτουργίας του παιχνιδιού. Αυτή η διαδικασία ενσωματώνεται με SQL queries για την καταγραφή και την επαναχρησιμοποίηση των δεδομένων χωρίς να απαιτούνται επιπλέον υπολογιστικοί πόροι.
- **Χρήση global μεταβλητών:** Γίνεται χρήση παγκόσμιων μεταβλητών (π.χ. initial_response_sent, follow_up_question_sent, no_more_questions), οι οποίες ελέγχουν τις ενέργειες του συστήματος, ελαχιστοποιώντας την ανάγκη για πρόσθετες διεργασίες και διασφαλίζοντας ότι οι πόροι χρησιμοποιούνται μόνο όταν απαιτείται.
- Α**ποφυγή επαναλαμβανόμενων ερωτήσεων:** Μέσω της αποθήκευσης και επεξεργασίας των ερωτήσεων και των απαντήσεων στην SQLite, αποφεύγεται η επανάληψη της ίδιας ερώτησης, απάντησης ή βοηθητικού μηνύματος.

Επιπλέον, η ακρίβεια και σαφήνεια των απαντήσεων και των ερωτήσεων βελτιώθηκαν, εξασφαλίζοντας την κατάλληλη προσαρμογή τους στο επίπεδο του χρήστη, τη θεματική και το ιστορικό του διαλόγου.

Σχετικά με τη διαχείριση των δεδομένων και τη σχεδίαση του UI, επιτεύχθηκαν σημαντικές βελτιώσεις στη επεξεργασία και προβολή των δεδομένων σε πραγματικό χρόνο, αποτρέποντας την επαναληψιμότητα ερωτήσεων και απαντήσεων. Η αισθητική του διαλόγου εξασφαλίζει την ομαλή ροή και αναδίπλωση του κειμένου χωρίς τεχνικά προβλήματα. Επίσης, ο κώδικας υποστηρίζει την εύκολη προσθήκη νέων θεματικών, NPCs και γραφικών μέσα από τη βάση δεδομένων, χωρίς να απαιτούνται περαιτέρω τροποποιήσεις. Έτσι, με την εξασφάλιση της επεκτασιμότητας του συστήματος καθίσταται ευκολότερη η μελλοντική ανάπτυξη και οι αναβαθμίσεις.

6.3. Συμπεράσματα

Το "A Quest of Knowledge" αποτελεί μια καινοτόμο προσέγγιση στην εκπαιδευτική διαδικασία μέσω των ψηφιακών serious games, αξιοποιώντας τις δυνατότητες της Python και των Μεγάλων Γλωσσικών Μοντέλων (LLMs). Η ενσωμάτωση αυτών των τεχνολογιών συνέβαλε στη δημιουργία μιας διαδραστικής και εξατομικευμένης εμπειρίας μάθησης, μετατρέποντας την εκπαίδευση σε μια συναρπαστική περιπέτεια.

Ένα από τα σημαντικότερα επιτεύγματα του έργου ήταν η επιτυχής ενσωμάτωση έξυπνων NPCs που αλληλεπιδρούν δυναμικά με τους παίκτες. Χάρη στη χρήση του Llama3, οι διάλογοι με ιστορικές προσωπικότητες ξεπέρασαν το πλαίσιο των παραδοσιακών ερωτήσεων-απαντήσεων, προσφέροντας μια πιο φυσική και δημιουργική αλληλεπίδραση. Αυτή η τεχνολογική καινοτομία ενίσχυσε την αυθεντικότητα της εμπειρίας, καθιστώντας το παιχνίδι πιο ελκυστικό και αποτελεσματικό ως εργαλείο μάθησης. Επιπλέον, η επιλογή και σύνδεση θεματικών πεδίων, όπως τα Μαθηματικά, η Ιστορία και η Φυσική, με ιστορικά πρόσωπα δημιούργησε ένα πρωτότυπο και εκπαιδευτικό περιβάλλον. Η δυνατότητα εξατομίκευσης της εμπειρίας μάθησης, μέσω της προσαρμογής του επιπέδου δυσκολίας και του αριθμού ερωτήσεων, προσέφερε μια εμπειρία προσαρμοσμένη στις ανάγκες κάθε παίκτη. Ωστόσο, κατά την ανάπτυξη του παιχνιδιού, παρουσιάστηκαν προκλήσεις που αφορούσαν κυρίως την απόδοση του API και τη διαχείριση της μνήμης λόγω των απαιτήσεων των LLMs και των γραφικών του παιχνιδιού. Οι καθυστερήσεις στις απαντήσεις των NPCs και η υψηλή κατανάλωση πόρων αντιμετωπίστηκαν με βελτιστοποίηση του API, χρήση global μεταβλητών και αποθήκευση δεδομένων σε SQLite. Η επιτυχημένη υλοποίηση των δυναμικών αλληλεπιδράσεων μεταξύ των παικτών και των NPCs αποδεικνύει ότι τα LLMs μπορούν να

αποτελέσουν ισχυρό εργαλείο για την εκπαίδευση, προσομοιώνοντας ρεαλιστικούς και εκπαιδευτικούς διαλόγους. Παράλληλα, η άντληση των δεδομένων από τη βάση διευκολύνει τη μελλοντική ενσωμάτωση περισσότερων θεματικών και χαρακτήρων, χωρίς την ανάγκη εκτεταμένων τροποποιήσεων στον κώδικα.

Συνοψίζοντας, το "A Quest of Knowledge" απέδειξε ότι τα serious games με ενσωματωμένη τεχνητή νοημοσύνη μπορούν να αποτελέσουν ένα ισχυρό και καινοτόμο εργαλείο μάθησης. Ο συνδυασμός διασκέδασης και εκπαίδευσης, η προσαρμοστικότητα στις ανάγκες των παικτών και η δυναμική δημιουργία περιεχομένου καθιστούν το παιχνίδι μια πρωτοποριακή προσθήκη στον χώρο της εκπαιδευτικής τεχνολογίας. Με τη συνεχή εξέλιξη των LLMs και των εργαλείων ανάπτυξης παιχνιδιών, το μέλλον των εκπαιδευτικών serious games διαφαίνεται εξαιρετικά πολλά υποσχόμενο, ανοίγοντας νέες προοπτικές και προσφέροντας νέες δυνατότητες για την εκπαίδευση.

6.4. Μελλοντικές Επεκτάσεις

Το “A Quest of Knowledge” διαθέτει σημαντικές δυνατότητες εξέλιξης, οι οποίες μπορούν να ενισχύσουν περαιτέρω την εκπαιδευτική του αξία και τη διαδραστική εμπειρία των παικτών. Οι προοπτικές επέκτασης περιλαμβάνουν τη δυνατότητα προσαρμογής θεματικών ενοτήτων, τη δυναμική δημιουργία περιεχομένου με τη χρήση τεχνητής νοημοσύνης και την εισαγωγή νέων τρόπων παιχνιδιού. Πιο συγκεκριμένα, το παιχνίδι θα μπορούσε να επεκταθεί ως εξής:

Εξατομίκευση και Δυναμική Δημιουργία Περιεχομένου

- **Προσαρμογή θεματικών ενοτήτων**: Οι χρήστες θα μπορούν να ορίζουν τις δικές τους θεματικές ενότητες, επιλέγοντας εκπαιδευτικό περιεχόμενο που ταιριάζει στις ανάγκες τους.
- **Δυναμική δημιουργία περιβάλλοντος και NPCs:** Χρήση τεχνητής νοημοσύνης για την αυτόματη διαμόρφωση γραφικών και χαρακτήρων ανάλογα με τις επιλεγμένες θεματικές ενότητες, ενισχύοντας τη μοναδικότητα κάθε νέας εμπειρίας.

Επέκταση Χαρτών και Δομής Παιχνιδιού

- **Νέες πίστες και περιοχές:** Προσθήκη περισσότερων χαρτών με διαφορετικά θέματα και βαθμούς δυσκολίας, βελτιώνοντας την εμπειρία εξερεύνησης.
- **Δυναμική δημιουργία πιστών:** Αυτόματη παραγωγή νέων περιοχών με βάση τις θεματικές επιλογές των παικτών.
- **Αναβάθμιση όπλων και δεξιοτήτων:** Οι παίκτες θα μπορούν να βελτιώνουν τις δεξιότητές τους μέσω σωστών απαντήσεων σε ερωτήσεις.

Διεύρυνση Εκπαιδευτικού Περιεχομένου

- **Περισσότερα ιστορικά πρόσωπα:** Εισαγωγή νέων NPCs από διάφορες ιστορικές περιόδους και πολιτισμούς, προσφέροντας εκπαιδευτική αξία και πολυμορφία.
- **Δυναμική δημιουργία NPCs:** Το παιχνίδι θα μπορεί να δημιουργεί τυχαίους χαρακτήρες με ονόματα και χαρακτηριστικά βασισμένα σε ιστορικές προσωπικότητες, καθιστώντας κάθε νέα εμπειρία μοναδική.
- **Δημιουργία ερωτήσεων από χρήστες:** Δυνατότητα στους εκπαιδευτικούς και τους μαθητές να προσθέτουν δικές τους ερωτήσεις και θεματικές ενότητες, εξατομικεύοντας την εμπειρία μάθησης.

Νέες Λειτουργίες και Κοινωνική Αλληλεπίδραση

- **Co-op Mode:** Συνεργατική λειτουργία όπου δύο ή περισσότεροι παίκτες μπορούν να εξερευνούν, να απαντούν ερωτήσεις και να βοηθούν ο ένας τον άλλον στις μάχες.
- **Knowledge Battles:** Ανταγωνιστικό mode, όπου οι παίκτες αναμετρώνται σε μάχες γνώσεων, προσπαθώντας να απαντήσουν πιο γρήγορα και σωστά.
- **Διαγωνιστικά events:** Εβδομαδιαίοι ή μηνιαίοι διαγωνισμοί γνώσεων, με έπαθλα και leaderboard για τους κορυφαίους παίκτες.
- **Online leaderboards:** Προβολή των κορυφαίων παικτών, ενισχύοντας τον ανταγωνισμό και την αναπαιξιμότητα του παιχνιδιού.

Αυτές οι επεκτάσεις θα μπορούσαν να αναβαθμίσουν σημαντικά το “A Quest of Knowledge”, ενισχύοντας ακόμη περισσότερο την εκπαιδευτική αξία του παιχνιδιού, προωθώντας την καινοτομία στη μάθηση μέσω των serious games και των LLMs.

Βιβλιογραφία

Delipetrev, B. T. (2020). Historical evolution of artificial intelligence.

Gallotta, R. T. (2024). *Large language models and games: A survey and roadmap*. arXiv.

Metuarau, T. (2017). *A history of video games*. Wellington: Open Access Te Herenga Waka-Victoria University of Wellington.

Minaee, S. M. (2024). *Large language models: A survey*. arXiv.

Noemí, P. M. (2014). Educational games for learning. *Universal Journal of Educational Research*, σσ. 230-238.

Relan, K. (2019). *Building REST APIs with Flask*.

Samuelson, G. (2020). *Pixel art-The Medium of Limitation: A qualitative study on how experienced artists perceive the relationship between restrictions and creativity*.

Sanner, M. F. (1999). Python: a programming language for software integration and development. *Journal of Molecular Graphics and Modelling*, σσ. 57-61.

Shinde, P. N. (2021). Pygame: Develop games using python. *International Journal of Research in Applied Science and Engineering Technology*, σσ. 4442-4447.

Πανεπιστήμιο Δυτικής Μακεδονίας. (2018). *ΚΑΝΟΝΙΣΜΟΣ ΑΡΧΩΝ ΚΑΙ ΛΕΙΤΟΥΡΓΙΑΣ ΤΗΣ ΕΠΙΤΡΟΠΗΣ ΗΘΙΚΗΣ ΚΑΙ ΔΕΟΝΤΟΛΟΓΙΑΣ ΤΗΣ ΕΡΕΥΝΑΣ ΤΟΥ ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ*. Ανάκτηση από Κανονισμός αρχών και Λειτουργίας ΕΗΔΕ ΠΔΜ: https://ehde.uowm.gr/

Παράρτημα Ι : Οδηγίες εγκατάστασης

Απαιτήσεις Συστήματος

Λειτουργικό Σύστημα: Windows 10 ή νεότερο

Κάρτα Γραφικών: NVIDIA (οδηγός 452.39 ή νεότερος) ή AMD Radeon, με ελάχιστη μνήμη κάρτας γραφικών VRAM (Video Random Access Memory)

Αποθηκευτικός χώρος: 4GB για την εγκατάσταση και 5GB για το μοντέλο, καθώς και 1 GB για το εκτελέσιμο αρχείο exe του παιχνιδιού και τα υπόλοιπα δεδομένα του

Διαθέσιμη RAM: 8GB ή μεγαλύτερη (συνιστάται 16GB για καλύτερη απόδοση)

Οδηγίες Εγκατάστασης και Εκτέλεσης του Ollama και του Llama 3(8b)-latest σε Windows

Βήμα 1ο: Εγκατάσταση του Ollama

Κατεβάστε το Ollama για Windows από την επίσημη σελίδα : https://ollama.com/download

Εκτελέστε το αρχείο OllamaSetup.exe (212 MB)

Ακολουθήστε τον οδηγό εγκατάστασης

Με την επιτυχή ολοκλήρωση της εγκατάστασης, το Ollama θα τρέχει στο παρασκήνιο

Βήμα 2ο: Λήψη και Εκτέλεση του Llama3 (8b):latest

Ανοίξτε το cmd ή το PowerShell

Εκτελέστε την εντολή `ollama run llama3` για λήψη και εκτέλεση του Llama3(8b) latest(Την πρώτη φορά το μοντέλο (4.7GB) κατεβαίνει αυτόματα)

Οδηγίες εκτέλεσης του παιχνιδιού

Ανοίξτε το φάκελο dist, όπου βρίσκετε το εκτελέσιμο αρχείο (exe) του παιχνιδιού

Επιλέξτε το αρχείο A Quest Of Knowledge.exe

Κάνετε κλικ στο αρχείο για να ξεκινήσει το παιχνίδι

Παράρτημα ΙΙ : Έντυπο Συμμόρφωσης

Έντυπο Συγκατάθεσης για Επιστημονική Έρευνα

Τίτλος Έρευνας: Αξιολόγηση του Παιχνιδιού "A Quest of Knowledge"

Σκοπός της Έρευνας: Η παρούσα έρευνα στοχεύει στη συλλογή δεδομένων σχετικά με την εμπειρία των χρηστών που έπαιξαν το παιχνίδι "A Quest of Knowledge". Ειδικότερα, εξετάζονται η χρηστικότητα, η εκπαιδευτική αξία, η διαδραστικότητα και η επίδραση της τεχνητής νοημοσύνης στο παιχνίδι.

Διαδικασία Συμμετοχής: Η συμμετοχή σας περιλαμβάνει τη συμπλήρωση ενός ερωτηματολογίου που αποτελείται από τέσσερα μέρη:

- **Δημογραφικά Στοιχεία:** Ερωτήσεις σχετικά με την ηλικία, το φύλο, το εκπαιδευτικό επίπεδο και την προηγούμενη εμπειρία με serious games**.**
- **Χρηστικότητα & Εμπειρία Παιχνιδιού:** Αξιολόγηση της συνολικής εμπειρίας σας με το παιχνίδι, της ευκολίας πλοήγησης, του επιπέδου διασκέδασης και της εκπαιδευτικής αξίας των ερωτήσεων.
- **Εκπαιδευτική Αξία & Χρήση Τεχνητής Νοημοσύνης:** Εκτίμηση της βελτίωσης των γνώσεών σας μέσω του παιχνιδιού, της χρήσης τεχνητής νοημοσύνης στους μη παικτικούς χαρακτήρες (NPCs) και της αποτελεσματικότητας της προσαρμογής της δυσκολίας.
- **Βελτιώσεις & Σχόλια:** Προτάσεις για τη βελτίωση του παιχνιδιού, αναφορά τυχόν τεχνικών ή σχεδιαστικών προβλημάτων και πρόθεση σύστασης του παιχνιδιού σε άλλους.

Εθελοντική Συμμετοχή: Η συμμετοχή σας είναι απολύτως εθελοντική. Μπορείτε να αρνηθείτε να συμμετάσχετε ή να αποχωρήσετε από την έρευνα οποιαδήποτε στιγμή, χωρίς καμία συνέπεια.

Πιθανοί Κίνδυνοι και Οφέλη: Δεν υπάρχουν γνωστοί κίνδυνοι που συνδέονται με τη συμμετοχή σας σε αυτή την έρευνα. Αν και δεν προβλέπεται άμεσο όφελος για εσάς, η συμβολή σας θα βοηθήσει στη βελτίωση του παιχνιδιού και στην ενίσχυση της εκπαιδευτικής του αξίας.

Εχεμύθεια και Προστασία Προσωπικών Δεδομένων: Όλες οι πληροφορίες που θα συλλεχθούν είναι ανώνυμες και θα χρησιμοποιηθούν αποκλειστικά για ακαδημαϊκούς σκοπούς. Τα δεδομένα θα αποθηκευτούν με ασφάλεια και δεν θα κοινοποιηθούν σε τρίτους.

Δήλωση Συγκατάθεσης: Έχω διαβάσει και κατανοήσει τις πληροφορίες που παρέχονται παραπάνω και συναινώ να συμμετάσχω σε αυτή την έρευνα.

Όνομα Συμμετέχοντα: _____

Υπογραφή:

Ημερομηνία: _____

Σημείωση: Εάν ο συμμετέχων είναι ανήλικος, απαιτείται η συγκατάθεση του γονέα ή κηδεμόνα.

Όνομα Γονέα/Κηδεμόνα: _____

Υπογραφή:

Ημερομηνία: _____

Παράρτημα ΙΙΙ : Ερωτηματολόγιο

Ερωτηματολόγιο Αξιολόγησης του Παιχνιδιού "A Quest of Knowledge"

Αυτή η φόρμα έχει ως στόχο τη συλλογή δεδομένων σχετικά με την εμπειρία των

χρηστών που έπαιξαν το παιχνίδι "A Quest of Knowledge". Το ερωτηματολόγιο εξετάζει τη χρηστικότητα, την εκπαιδευτική αξία, τη διαδραστικότητα και την επίδραση της τεχνητής νοημοσύνης στο παιχνίδι.

Οι απαντήσεις σας είναι ανώνυμες και θα χρησιμοποιηθούν αποκλειστικά για

ακαδημαϊκούς σκοπούς, ώστε να αξιολογηθεί και να βελτιωθεί το παιχνίδι.

Μέρος 1: Δημογραφικά Στοιχεία

1.Ηλικία:

_____

2.Φύλο:

- Άνδρας
- Γυναίκα
- Άλλο
- Δε θέλω να απαντήσω

3. Εκπαιδευτικό επίπεδο:

- Δημοτικό
- Γυμνάσιο
- Λύκειο
- Πανεπιστήμιο
- Μεταπτυχιακό / Διδακτορικό

4. Έχετε προηγούμενη εμπειρία με serious games;

- Ναι
- Όχι

Μέρος 2: Χρηστικότητα & Εμπειρία Παιχνιδιού

5. Πώς θα αξιολογούσατε τη συνολική εμπειρία σας με το παιχνίδι; (1 = Πολύ κακή, 5 = Εξαιρετική)
6. Το παιχνίδι ήταν εύκολο στην πλοήγηση και το χειρισμό; (1 = Καθόλου, 5 = Πολύ)

7. Πόσο διασκεδαστικό βρήκατε το παιχνίδι; (1 = Καθόλου, 5 = Πολύ)

8. Βρήκατε τις ερωτήσεις του παιχνιδιού ενδιαφέρουσες και εκπαιδευτικές; (1 = Καθόλου, 5 = Πολύ)

9. Πώς αξιολογείτε το περιβάλλον του παιχνιδιού όσον αφορά τα γραφικά, τον ήχο και το γενικό στυλ;(1 = Πολύ χαμηλά, 5 = Εξαιρετικά)

10. Πόσο κατανοητές ήταν οι οδηγίες και τα μηνύματα του παιχνιδιού;(1 = Καθόλου κατανοητές, 5 = Απόλυτα κατανοητές)

11. Πώς θα αξιολογούσατε την εμπειρία συνομιλίας (Chat Mode) με τους NPCs;(1 = Ανεπαρκής, 5 = Εξαιρετική)

12. Πόσο αποτελεσματικό θεωρείτε το mini-map ως εργαλείο πλοήγησης;(1 = Καθόλου χρήσιμο, 5 = Πολύ χρήσιμο)

13. Πόσο πιθανό είναι να ξαναπαίξετε το παιχνίδι με διαφορετικές παραμέτρους (π.χ. αριθμό ερωτήσεων, επιλογή χαρακτήρα, δήλωση ηλικίας);(1 = Καθόλου πιθανό, 5 = Πολύ πιθανό)

Μέρος 3: Εκπαιδευτική Αξία & Χρήση Τεχνητής Νοημοσύνης

14. Πιστεύετε ότι το παιχνίδι βελτίωσε τις γνώσεις σας;

- Πολύ
- Αρκετά
- Μέτρια
- Λίγο
- Καθόλου

15.Πώς αξιολογείτε τη χρήση της τεχνητής νοημοσύνης στους NPCs;

- Εξαιρετική
- Καλή
- Μέτρια
- Κακή
- Πολύ κακή

16. Πιστεύετε ότι η προσαρμογή της δυσκολίας (με βάση την ηλικία και τον αριθμό ερωτήσεων) ήταν αποτελεσματική;

- Ναι
- Ναι, αλλά θα μπορούσε να βελτιωθεί
- Όχι ιδιαίτερα
- Καθόλου

17.Θεωρείτε ότι οι διαθέσιμες εκπαιδευτικές ενότητες (Μαθηματικά, Γεωγραφία, Αστρονομία, Πληροφορική, Φυσική, Χημεία, Ιστορία, Τέχνες, Μουσική, Περιβαλλοντική Εκπαίδευση, Βιολογία) καλύπτουν επαρκώς τις ανάγκες σας;

- Ναι
- Όχι

18. Θα προτείνατε την προσθήκη επιπλέον θεματικών ενοτήτων ή ενσωμάτωση άλλου είδους εκπαιδευτικού περιεχομένου;

_____
_____

Μέρος 4: Βελτιώσεις & Σχόλια

19. Τι θα προτείνατε για να γίνει το παιχνίδι πιο διασκεδαστικό ή εκπαιδευτικό;

_____
_____

20. Υπήρχε κάποιο τεχνικό ή σχεδιαστικό πρόβλημα που συναντήσατε; Αν ναι, περιγράψτε το.

_____
_____

21. Ποια σημεία του παιχνιδιού θεωρείτε ότι χρειάζονται περαιτέρω βελτίωση (π.χ. έλεγχοι, γραφικά, ήχος, διάλογοι);

_____
_____

22. Ποια ήταν η πιο θετική πτυχή της εμπειρίας σας στο παιχνίδι;

_____
_____

23. Υπήρξε κάποιο σημείο που σας απογοήτευσε ή δυσκολευτήκατε κατά τη διάρκεια του παιχνιδιού;

_____
_____

24. Θα προτείνατε το παιχνίδι σε άλλους;

- Ναι
- Ίσως
- Όχι

Παράρτημα IV : Κώδικας

https://github.com/Anthi-Drougkani/Knowledge_Exploration_Pyagame