Σχολή Θετικών Επιστημών και Τεχνολογίας

ΠΛΗΡΟΦΟΡΙΚΗ

Πτυχιακή

«Κατασκευή ενός RAG Chatbot για Ξενοδοχειακή Υποστήριξη»

Νικόλαος Λυμπιτάκης

Επιβλέπων καθηγητής: Μηνάς Δασυγένης

Πάτρα, Ιούλιος 2026

ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Κατασκευή ενός RAG Chatbot για Ξενοδοχειακή Υποστήριξη

Νικόλαος Λυμπιτάκης

Επιτροπή Επίβλεψης Πτυχιακής

Επιβλέπων Καθηγητής:
Μηνάς Δασυγένης
Αναπληρωτής Καθηγητής - Ηλ. Μηχανικός & Μηχανικός Υπολογιστών
Τμ. Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Παν. Δυτικής Μακεδονίας

Συν-Επιβλέπων Καθηγητής
Χρήστος Κούτσικας
ΣΕΠ ΕΑΠ

Συν-Επιβλέπων Καθηγητής
Βασίλειος Στεφανής
Επιστημονικός συνεργάτης Ε.Α.Π

Πάτρα, Ιούλιος 2026

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, Δρ. Μηνά Δασυγένη, για την πολύτιμη καθοδήγηση, τις εύστοχες παρατηρήσεις και την υποστήριξή του σε κάθε στάδιο εκπόνησης της εργασίας.

Ευχαριστώ επίσης την οικογένειά μου για την συνεχή στήριξη, την κατανόηση και την ενθάρρυνση που μου πρόσφερε καθ' όλη τη διάρκεια των σπουδών μου.

Τέλος, ευχαριστώ όλους όσους συνέβαλαν, με οποιονδήποτε τρόπο, στην ολοκλήρωση αυτής της εργασίας.

«Ευχαριστίες ή Αφιέρωση»

Περίληψη

Η παρούσα πτυχιακή εργασία, πραγματεύεται τη σχεδίαση και υλοποίηση ενός έξυπνου chatbot βασισμένου στην αρχιτεκτονική Retrieval-Augmented Generation (RAG), με εφαρμογή στον ξενοδοχειακό κλάδο.

Κίνητρο για την ανάπτυξη του συστήματος αποτέλεσαν οι σημαντικοί περιορισμοί της παραδοσιακής ανθρώπινης εξυπηρέτησης πελατών, όπως το υψηλό λειτουργικό κόστος, οι περιορισμένες ώρες διαθεσιμότητας και η ανομοιογένεια στην ποιότητα της εξυπηρέτησης. Η αρχιτεκτονική RAG επιλέχθηκε ως η πλέον κατάλληλη προσέγγιση, καθώς συνδυάζει τα πλεονεκτήματα των Μεγάλων Γλωσσικών Μοντέλων (LLMs) με μηχανισμούς ανάκτησης πληροφοριών από εξωτερικές βάσεις γνώσης, ξεπερνώντας έτσι τους περιορισμούς της στατικής γνώσης των παραδοσιακών μοντέλων.

Για την υλοποίηση αξιοποιήθηκαν σύγχρονες τεχνολογίες όπως: η Python ως κύρια γλώσσα προγραμματισμού, το LangChain για τη διαχείριση της ροής δεδομένων και τη διασύνδεση με το γλωσσικό μοντέλο DeepSeek, η βιβλιοθήκη FAISS για αποθήκευση διανυσματικών αναπαραστάσεων (embeddings) και ταχεία σημασιολογική αναζήτηση, το Streamlit για τη δημιουργία φιλικού γραφικού περιβάλλοντος και η SQLite ως ελαφριά βάση δεδομένων για τα στοιχεία του ξενοδοχείου και τις κρατήσεις.

Το σύστημα σχεδιάστηκε με βάση συγκεκριμένες προδιαγραφές και περιπτώσεις χρήσης, και αξιολογήθηκε μέσω δομημένης μεθοδολογίας με ποσοτικές μετρικές. Το τελικό chatbot είναι ικανό να απαντά σε ερωτήσεις σχετικά με τις παροχές, τις πολιτικές και τη διαθεσιμότητα δωματίων ενός ξενοδοχείου, τόσο στα Ελληνικά όσο και στα Αγγλικά.

Τα αποτελέσματα επιβεβαιώνουν την αποτελεσματικότητα της αρχιτεκτονικής RAG ως ολοκληρωμένης λύσης για ξενοδοχειακές εφαρμογές, με άμεση μείωση λειτουργικού κόστους και βελτίωση της εμπειρίας χρήστη. Η εργασία ολοκληρώνεται με ανάλυση SWOT και πρόταση μελλοντικών επεκτάσεων, θέτοντας τις βάσεις για περαιτέρω ανάπτυξη έξυπνων συστημάτων εξυπηρέτησης στον τουριστικό τομέα.

Λέξεις – Κλειδιά

RAG, Chatbot, Μεγάλα γλωσσικά μοντέλα, Τεχνητή νοημοσύνη

Abstract

This undergraduate thesis addresses the design and implementation of an intelligent chatbot based on the Retrieval-Augmented Generation (RAG) architecture, applied to the hospitality sector.

The motivation behind the development of this system lies in the significant limitations of traditional human customer service, such as high operational costs, restricted availability hours, and inconsistency in service quality. The RAG architecture was selected as the most suitable approach, as it combines the advantages of Large Language Models (LLMs) with information retrieval mechanisms from external knowledge bases, thereby overcoming the static knowledge constraints of conventional models.

The implementation leverages modern technologies including: Python as the primary programming language, LangChain for managing the data flow and interfacing with the DeepSeek language model, the FAISS library for storing vector representations (embeddings) and enabling fast semantic search, Streamlit for building a user-friendly graphical interface, and SQLite as a lightweight database for hotel information and reservations.

The system was designed according to specific requirements and use cases, and evaluated through a structured methodology using quantitative metrics. The final chatbot is capable of answering questions related to hotel amenities, policies, and room availability, in both Greek and English.

The results confirm the effectiveness of the RAG architecture as a complete solution for hospitality applications, delivering direct reductions in operational costs and an improved user experience. The thesis concludes with a SWOT analysis and proposals for future extensions, laying the groundwork for further development of intelligent customer service systems in the tourism sector.

Keywords

RAG, Chatbot, Large Language Models, Artificial Intelligence

Περιεχόμενα

Συντομογραφίες & Ακρωνύμια

RAG	Retrieval Augmented Generation
LLM	Large Language Model
FAQs	Frequently Asked Questions
AI	Artificial Intelligence
TN	Τεχνητή Νοημοσύνη
NLP	Natural Language Processing
TTS	Text To Speech
CSV	Comma-Separated Values
FAISS	Facebook AI Similarity Search
UI	User Inteface

1. Εισαγωγή

Στην εποχή του ψηφιακού μετασχηματισμού, οι επιχειρήσεις αναζητούν διαρκώς καινοτόμους τρόπους για να βελτιστοποιήσουν τις λειτουργίες τους και να ενισχύσουν την εμπειρία των πελατών τους. Ένα από τα πιο δυνατά εργαλεία που έχουν αναδειχθεί τα τελευταία χρόνια είναι τα chatbots. Ένα chatbot είναι μια εφαρμογή λογισμικού σχεδιασμένη να προσομοιώνει την ανθρώπινη συνομιλία μέσω κειμένου ή φωνής. Η εξέλιξή τους ήταν ραγδαία. Από τα πρώιμα στάδια, βασισμένα σε κανόνες (rule-based) συστήματα που απαντούσαν σε συγκεκριμένες λέξεις-κλειδιά, φτάσαμε στους σημερινούς ευφυείς πράκτορες που αξιοποιούν την Τεχνητή Νοημοσύνη, την Επεξεργασία Φυσικής Γλώσσας (NLP) [1] και τα Μεγάλα Γλωσσικά Μοντέλα (LLMs) για να κατανοούν την πρόθεση του χρήστη και να διεξάγουν πολύπλοκες, φυσικές συζητήσεις.

Η χρησιμότητα των chatbots για τις σύγχρονες επιχειρήσεις είναι στρατηγικής σημασίας. Το πιο άμεσο και προφανές πλεονέκτημα είναι η συνεχής διαθεσιμότητα. Σε αντίθεση με τους ανθρώπινους εκπροσώπους, τα chatbots μπορούν να εξυπηρετούν πελάτες ανά πάσα στιγμή, προσφέροντας άμεσες απαντήσεις και εξαλείφοντας τους χρόνους αναμονής. Αυτό οδηγεί σε σημαντική μείωση του λειτουργικού κόστους, καθώς αυτοματοποιούνται επαναλαμβανόμενες ερωτήσεις και εργασίες, επιτρέποντας στο ανθρώπινο δυναμικό να εστιάσει σε πιο σύνθετα και κρίσιμα ζητήματα που απαιτούν ανθρώπινη παρέμβαση. Ένα άλλο πλεονέκτημα είναι ότι διαθέτουν μεγάλη κλιμακωσιμότητα (scalability), καθώς μπορούν να διαχειριστούν ταυτόχρονα χιλιάδες συνομιλίες, μια δυνατότητα ανέφικτη για οποιαδήποτε ομάδα εξυπηρέτησης πελατών.
Επιπλέον, τα chatbots δεν περιορίζονται μόνο στην υποστήριξη πελατών. Εφαρμόζονται σε ένα ευρύ φάσμα τομέων, όπως το ηλεκτρονικό εμπόριο, η εκπαίδευση, ο τουρισμός και η υγεία, παρέχοντας πολύτιμες υπηρεσίες όπως διαχείριση κρατήσεων, ενημέρωση χρηστών και υποστήριξη σε κρίσιμες αποφάσεις. Με αυτό τον τρόπο, τα chatbots αποτελούν ένα εργαλείο εξοικονόμησης πόρων και καινοτομίας που ενισχύει την ανταγωνιστικότητα των οργανισμών στη σύγχρονη αγορά.

Τέλος, μια από τις πιο πρόσφατες εξελίξεις στον χώρο αποτελεί η προσέγγιση Retrieval-Augmented Generation (RAG) [2], η οποία συνδυάζει τα πλεονεκτήματα των Μεγάλων Γλωσσικών Μοντέλων με εξωτερικές πηγές γνώσης. Τα RAG chatbots έχουν τη δυνατότητα να αντλούν πληροφορίες σε πραγματικό χρόνο από βάσεις δεδομένων ή εξειδικευμένα έγγραφα και να δημιουργούν απαντήσεις που είναι φυσικές, ακριβείς και

επικαιροποιημένες. Έτσι, ξεπερνιούνται οι περιορισμοί της στατικής γνώσης ενός LLM και παρέχεται ένα εργαλείο που μπορεί να προσαρμόζεται δυναμικά στις ιδιαίτερες ανάγκες κάθε επιχείρησης ή οργανισμού.

1.1 Επισκόπηση τρέχουσας κατάστασης και αναγκαιότητα λύσης

Σήμερα η εξυπηρέτηση πελατών στις περισσότερες επιχειρήσεις γίνεται μέσω ανθρώπινου προσωπικού. Αυτό όμως χαρακτηρίζεται από σημαντικούς περιορισμούς όπως: απαιτεί υψηλό λειτουργικό κόστος για μισθούς, εκπαίδευση και ασφαλιστικές εισφορές, οι χρόνοι απόκρισης εξαρτώνται από τα ωράρια και τις βάρδιες με συνέπεια καθυστερήσεις στην κάλυψη των αιτημάτων, η κλιμάκωση της υπηρεσίας σε ώρες αιχμής επιβάλλει υπερωρίες ή νέες προσλήψεις αυξάνοντας περαιτέρω τα έξοδα και τέλος, η ποιότητα της εξυπηρέτησης ποικίλλει ανάλογα με την εμπειρία, τη διάθεση και το επίπεδο εξάντλησης των υπαλλήλων οδηγώντας σε ανομοιόμορφα επίπεδα ικανοποίησης πελατών. Αντίθετα, η υιοθέτηση customer support chatbots επιτρέπει τη μείωση των λειτουργικών εξόδων κατά περίπου 20% μέσω αυτοματοποίησης επαναλαμβανόμενων εργασιών, παρέχοντας παράλληλα συνεχή διαθεσιμότητα, ταχεία ανταπόκριση εντός δευτερολέπτων και συνέπεια στην ποιότητα της πληροφορίας [3]. Τα chatbots μπορούν να χειρίζονται μεγάλο όγκο αιτημάτων χωρίς πρόσθετο προσωπικό ή υπερωριακή απασχόληση, διασφαλίζοντας ομοιογενή εμπειρία πελατών και άμεση εξυπηρέτηση, ανεξάρτητα από φόρτο εργασίας και κόπωση. Η ανάγκη για ανάπτυξη τέτοιων συστημάτων εντείνεται ακόμη περισσότερο στον ξενοδοχειακό κλάδο, όπου οι απαιτήσεις για άμεση και αξιόπιστη ενημέρωση των επισκεπτών είναι διαρκείς. Οι πελάτες αναμένουν πλέον εξατομικευμένες απαντήσεις σε πραγματικό χρόνο και σε πολλαπλές γλώσσες, γεγονός που καθιστά τα παραδοσιακά μέσα επικοινωνίας ανεπαρκή. Η εφαρμογή τεχνολογιών Τεχνητής Νοημοσύνης, και ειδικότερα της αρχιτεκτονικής Retrieval Augmented Generation (RAG), προσφέρει μια σύγχρονη και αποδοτική λύση στο πρόβλημα αυτό. Τα RAG chatbots συνδυάζουν τις δυνατότητες των LLMs με την αξιοποίηση εξωτερικών βάσεων γνώσης, μειώνοντας τα φαινόμενα παραισθήσεων (hallucinations) και παρέχοντας τεκμηριωμένες, επικαιροποιημένες πληροφορίες.

Ειδικότερα, η μετάβαση του τουριστικού τομέα στην ψηφιακή εποχή επιβάλλει λύσεις που ενισχύουν την αυτοματοποίηση και την αποδοτικότητα. Όπως επισημαίνεται στην [4], τα σύγχρονα ξενοδοχεία οφείλουν να προσφέρουν εμπειρίες άμεσης αλληλεπίδρασης και εξατομίκευσης, αξιοποιώντας συστήματα συνομιλίας που λειτουργούν 24/7 και σε διαφορετικές γλώσσες. Παράλληλα, τα RAG chatbots συμβάλλουν στη συλλογή και ανάλυση δεδομένων προτιμήσεων των πελατών, επιτρέποντας τη συνεχή βελτίωση των παρεχόμενων υπηρεσιών και τη διατήρηση ανταγωνιστικού πλεονεκτήματος.

1.2 Σκοπός

Ο κύριος σκοπός της παρούσας πτυχιακής εργασίας είναι η σχεδίαση, η ανάπτυξη και η αξιολόγηση ενός RAG chatbot για τον ξενοδοχειακό κλάδο. Το chatbot στοχεύει στην ενίσχυση της εμπειρίας των δυνητικών πελατών και στην αύξηση των κρατήσεων, παρέχοντας άμεση, εξατομικευμένη και πειστική πληροφόρηση

Επιμέρους Στόχοι:

Για την επίτευξη του κύριου σκοπού, τέθηκαν οι ακόλουθοι επιμέρους στόχοι:

1. **Βελτίωση της Εξυπηρέτησης Πελατών:** Η δημιουργία ενός εικονικού βοηθού που απαντά σε ερωτήσεις πελατών, μειώνοντας τον φόρτο εργασίας του προσωπικού και παρέχοντας άμεση πρόσβαση σε πληροφορίες για τις παροχές, τις πολιτικές του ξενοδοχείου και τα κοντινά σημεία ενδιαφέροντος.
2. **Πειθώ και Εξατομίκευση**: Η ανάπτυξη ενός chatbot που δεν περιορίζεται στην παροχή πληροφοριών, αλλά προσπαθεί ενεργά να πείσει τον χρήστη να επιλέξει το συγκεκριμένο ξενοδοχείο για τη διαμονή του, προβάλλοντας τα ανταγωνιστικά του πλεονεκτήματα.
3. **Εφαρμογή Σύγχρονων Τεχνολογιών AI:** Η χρήση τεχνολογιών αιχμής, όπως το LLM-DeepSeek, το framework LangChain, διανυσματικές βάσεις δεδομένων (vector databases) για την ανάκτηση πληροφοριών από πολλαπλές πηγές (FAQs, κριτικές πελατών, σημεία ενδιαφέροντος, κτλ).
4. **Πολυγλωσσική Υποστήριξη:** Η υλοποίηση ενός πολύγλωσσου chatbot (Ελληνικά και Αγγλικά) για την αποτελεσματική εξυπηρέτηση ενός ευρύτερου πελατολογίου.
5. **Αξιολόγηση της Απόδοσης:** Ο καθορισμός μιας μεθοδολογίας για την αξιολόγηση της αποτελεσματικότητας του chatbot, με έμφαση σε μετρήσεις όπως η ικανοποίηση

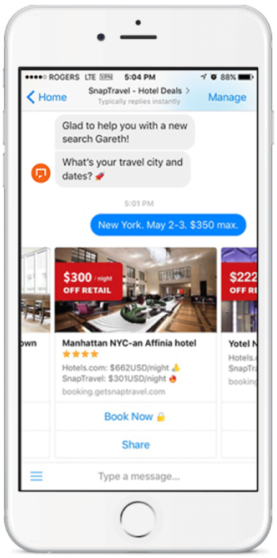
του χρήστη, η ακρίβεια των απαντήσεων και η δυνητική του επίδραση στις κρατήσεις.

1.3 Επισκόπηση υπάρχοντων Chatbot λύσεων

Το παραγωγικό chatbot κρατήσεων της Snaptravel

Η αρχιτεκτονική του συστήματος της Snaptravel [5] βασίζεται σε ένα σύστημα διαχείρισης διαλόγου με πλαίσιο (frame-based), το οποίο χρησιμοποιεί μοντέλα μηχανικής μάθησης για την ταξινόμηση προθέσεων, την αναγνώριση οντοτήτων και την ανάκτηση πληροφοριών. Το chatbot έχει αναπτυχθεί σε εμπορική κλίμακα, διαχειριζόμενο δεκάδες χιλιάδες αναζητήσεις ξενοδοχείων καθημερινά. Το σύστημα επιτρέπει στους χρήστες να αλληλεπιδρούν με το chatbot μέσω τρίτων πλατφορμών μηνυμάτων, όπως το Facebook Messenger, το Amazon Alexa και το WhatsApp. Ο χρήστης παρέχει πληροφορίες όπως ημερομηνίες ταξιδιού, προορισμό και προτιμήσεις ξενοδοχείου, και το chatbot προτείνει μια σειρά από κατάλληλα ξενοδοχεία που ο χρήστης μπορεί στη συνέχεια να κλείσει. Η διαχείριση του διαλόγου επιτυγχάνεται μέσω ενός διαχειριστή διαλόγου που ενσωματώνει συνδυασμό μοντέλων φυσικής γλώσσας (NLP) για την επεξεργασία των πιο συνηθισμένων σεναρίων, ενώ για πιο δύσκολες καταστάσεις παραπέμπει σε ανθρώπινους υποστηρικτές.

Η βάση δεδομένων του συστήματος περιλαμβάνει περίπου 100.000 πόλεις και 300.000 ξενοδοχεία, που αντλούνται από συνεργάτες. Κάθε εγγραφή περιέχει το όνομα της πόλης ή του ξενοδοχείου, γεωγραφικές πληροφορίες (π.χ., διεύθυνση, πολιτεία, χώρα) και διάφορα μεταδεδομένα (π.χ., βαθμολογία κριτικών, αριθμός κρατήσεων). Η αρχιτεκτονική του συστήματος βασίζεται σε ένα σύστημα διαχείρισης διαλόγου με πλαίσιο (frame-based), το οποίο χρησιμοποιεί μοντέλα μηχανικής μάθησης για την ταξινόμηση προθέσεων, την αναγνώριση οντοτήτων και την ανάκτηση πληροφοριών[5].



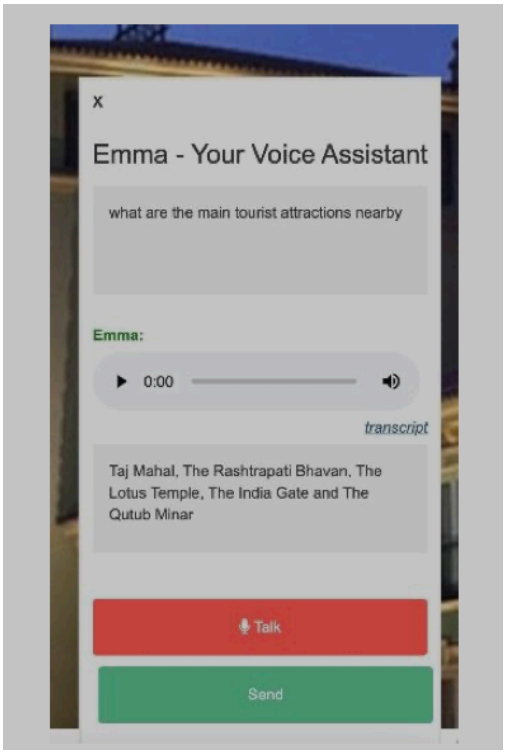
Εικόνα 1: Στιγμιότυπο μιας τυπικής συνομιλίας με το Snaptravel bot στο Facebook Messenger[1]

Το voice chatbot Emma

Το προτεινόμενο σύστημα Emma [4] είναι ένα φωνητικό chatbot ενσωματωμένο σε ξενοδοχειακή web εφαρμογή, το οποίο επιτρέπει αμφίδρομη συνομιλία μέσω φωνής για πλοήγηση, ενημέρωση και επίλυση αποριών σχετικών με τη λειτουργία του ιστότοπου και του ίδιου του ξενοδοχειακού domain. Υλοποιείται ως κλειστού πεδίου ερωτήσεων (Closed Domain QA, cdQA), εκπαιδευμένο πάνω σε έγγραφα φιλοξενίας και υλικό του ξενοδοχείου, ώστε να απαντά με ακρίβεια σε ερωτήσεις που αφορούν διαδικασίες, πληροφορίες δωματίων, παροχές και τοπικές δραστηριότητες. Η αρχιτεκτονική διαχωρίζει modules φωνής (speech-to-text, text-to-speech), NLP ερωτήσεων και web διεπαφή, διευκολύνοντας επαναεκπαίδευση και συντήρηση ανά υποσύστημα. Στην αναγνώριση φωνής αξιοποιούνται δύο προσεγγίσεις: α) browser-native Web Speech API (Chrome) για άμεση μεταγραφή στο περιβάλλον του πελάτη και β) open-source DeepSpeech για προσαρμοστική ακρίβεια μέσω γλωσσικού μοντέλου και βελτιστοποίησης τύπου hybrid parallel training, επιτρέποντας εξισορρόπηση απόδοσης και υποδομών. Για εκφώνηση απαντήσεων, συγκρίνονται λύσεις όπως gTTS (προεκπαιδευμένο TTS με πολυγλωσσική υποστήριξη) και Speech Synthesis API του browser, με τελική επιλογή browser-native μηχανισμού για απλότητα ενσωμάτωσης, ταχύτητα απόκρισης και μικρότερες επιχειρησιακές εξαρτήσεις.

[1] Εικόνα από [5]

Το αποτέλεσμα είναι hands-free εμπειρία χωρίς πρόσθετο backend TTS, που βελτιώνει την προσβασιμότητα και τη φυσικότητα αλληλεπίδρασης μέσα στο περιβάλλον του ξενοδοχειακού ιστότοπου. Ο πυρήνας κατανόησης/απάντησης βασίζεται στο cdQA με δύο στάδια: Retriever και Reader. Η εκπαίδευση έγινε με 50–100 εξειδικευμένου πεδίου (domain-specific) Q/A (ερωτήσεις/απαντήσεις). Το UI ενσωματώνει την Emma, με φιλική έναρξη διαλόγου, οπτική αναπαράσταση μεταγραφής, κουμπί "Talk", ένδειξη επεξεργασίας και αυτόματη αναπαραγωγή της φωνητικής απάντησης, επιτυγχάνοντας ενιαία εμπειρία φωνής-κειμένου. Σε αξιολόγηση, το σύστημα αποδίδει προβλέψιμα σε κείμενα έως ~3000 λέξεις, ενώ σε μακροσκελή κείμενα εμφανίζονται σφάλματα σύνοψης/ακρίβειας. Οι συγγραφείς προτείνουν βελτιώσεις μέσω περισσότερων δεδομένων, feedback loop και ισχυρότερης υλικοτεχνικής υποδομής (GPU), καθώς και συνεκτίμηση παραμέτρων χρηστικότητας/ιδιωτικότητας που καταγράφονται στη σχετική βιβλιογραφία [4].



Εικόνα 2 : Στιγμιότυπο συνομιλίας με το Emma chatbot[2]

[2] Εικόνα από [4]

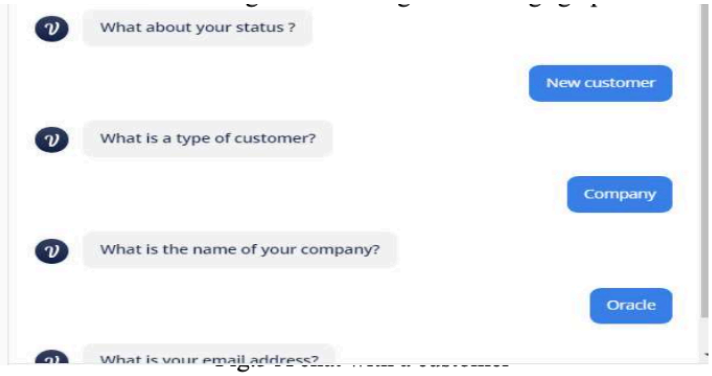
Το EDUBot chatbot

Μια άλλη ενδιαφέρουσα περίπτωση εκτός του ξενοδχειακού κλάδου είναι το το EDUBot [6] το οποίο είναι ένα υποστηρικτικό chatbot βασισμένο στο ChatGPT που σχεδιάστηκε για Customer Relationship Management (CRM) σε εκπαιδευτικούς οργανισμούς, προσφέροντας άμεση, φυσική και εξατομικευμένη εξυπηρέτηση σε ερωτήσεις, πληροφοριακές ανάγκες, υποβολή παραπόνων και καθοδήγηση χρηστών σε υπηρεσίες και διαδικασίες του φορέα. Ως συνομιλιακός βοηθός ενσωματωμένος στον ιστότοπο, δέχεται ελεύθερο κείμενο (και όπου υποστηρίζεται φωνή), αναγνωρίζει πρόθεση και οντότητες, και επιστρέφει κατανοητές απαντήσεις που μειώνουν τον χρόνο αναμονής και την επιβάρυνση των γραμμών υποστήριξης.

Η αρχιτεκτονική του ακολουθεί δοκιμασμένα δομικά στοιχεία για CRM chatbots: Natural Language Understanding για εντοπισμό προθέσεων (intents) και οντοτήτων (entities), Dialog Manager που αξιοποιεί το ChatGPT για συμφραζόμενες απαντήσεις, γνώση/βάση δεδομένων ως πηγή επίσημης πληροφόρησης και Natural Language Generation για φυσική διατύπωση εξόδου. Η υλοποίηση στην πλατφόρμα Voiceflow λειτουργεί ως διαχειριστής διαλόγου και επιτρέπει τον καθορισμό ροών, σεναρίων, γράφο γνώσης και ελέγχων συλλογής δεδομένων, ώστε το chatbot να ανταποκρίνεται με συνέπεια σε πολλαπλά σενάρια πελατειακής αλληλεπίδρασης.

Σε επίπεδο χρήσης, το EDUBot υποστηρίζει ενσωμάτωση νέου χρήστη με συλλογή βασικών στοιχείων, πλοήγηση σε υποδομές/υπηρεσίες (π.χ. αίθουσες, θέατρα, εγκαταστάσεις), αναζήτηση πληροφοριών, υποβολή παραπόνων και αξιολόγηση εμπειριών, καλύπτοντας διαφορετικές φάσεις του κύκλου ζωής του πελάτη. Οι ροές αναγνωρίζουν αν ο χρήστης είναι νέος ή υφιστάμενος, καταγράφουν ελεγχόμενα τα απαραίτητα δεδομένα (π.χ. email, τηλέφωνο) σε κατάλληλη αποθήκη και αξιοποιούν συγκεντρωμένη γνώση στην βάση γνώσης για ακριβή, συνεπή ενημέρωση. Η ολοκλήρωση με CRM επιτρέπει αμφίδρομη ροή δεδομένων: το chatbot αντλεί επίκαιρες πληροφορίες για υπηρεσίες, διαδικασίες και ταυτόχρονα εμπλουτίζει το προφίλ πελάτη με ερωτήματα και προτιμήσεις. Τα λειτουργικά οφέλη περιλαμβάνουν 24/7 διαθεσιμότητα, ταχύτερη επίλυση απορίων, αποφόρτιση προσωπικού από επαναλαμβανόμενες ερωτήσεις, συνεπή επικοινωνία.

Στη μελέτη περίπτωσης "Educational City" χρησιμοποιείται για ενημέρωση χρηστών σχετικά με τα συστατικά και τις υπηρεσίες του φορέα. Η σχεδίαση με γράφο γνώσης ενισχύει την κατανόηση συμφραζομένων και την καθοδήγηση διαλόγου, μειώνοντας ασάφειες και διασφαλίζοντας ευθυγράμμιση με πολιτικές και ύφος επικοινωνίας. Η προσέγγιση είναι επεκτάσιμη καθώς προστίθενται νέα σενάρια, προθέσεις (intents) και οντότητες, εμπλουτίζεται η βάση γνώσης και συνδέεται με πρόσθετα συστήματα (π.χ. ticketing), ώστε οι συνομιλίες να οδηγούν σε ενέργειες και μετρήσιμους δείκτες απόδοσης. Συνολικά, το EDUBot καταδεικνύει πώς ένα ChatGPT-based chatbot, σωστά δεμένο με CRM διαδικασίες και γνώση οργανισμού, αναβαθμίζει την εμπειρία χρήστη, βελτιώνει τη διαχείριση σχέσεων και προσφέρει αποδοτική, κλιμακούμενη υποστήριξη για εκπαιδευτικά ιδρύματα [6].



Εικόνα 3 : Στιγμιότυπο συνομιλίας με το EDUBot chatbot[3]

Συνοψίζοντας, η ανάλυση των υπαρχόντων chatbot λύσεων δείχνει ότι, αν και ορισμένα συστήματα όπως το EDUBot και Emma αξιοποιούν αρχιτεκτονικές τύπου RAG συνδυάζοντας μεγάλα γλωσσικά μοντέλα με βάσεις γνώσης, οι περισσότερες εμπορικές εφαρμογές του ξενοδοχειακού τομέα όπως το Snaptravel παραμένουν περιορισμένες σε προκαθορισμένα σενάρια ή σε στενά πεδία ερωτήσεων. Παρά τις τεχνολογικές εξελίξεις, λείπει μια πλήρως προσαρμοσμένη λύση για τον ξενοδοχειακό κλάδο που να ενοποιεί σε πραγματικό χρόνο γνώση από πολλαπλές πηγές (FAQs, κριτικές, πολιτικές, τοπικές πληροφορίες) και να παρέχει τεκμηριωμένες, πολυγλωσσικές και εξατομικευμένες απαντήσεις. Η ανάγκη αυτή καθιστά ιδιαίτερα χρήσιμη την ανάπτυξη εξειδικευμένων RAG

[3] Εικόνα από [6]

chatbots για τη φιλοξενία, ικανά να καλύψουν το κενό μεταξύ της θεωρητικής προόδου και των πρακτικών απαιτήσεων του τουριστικού τομέα.

1.4 Σύνοψη 1ου Κεφαλαίου

Συνοψίζοντας, στο εισαγωγικό αυτό κεφάλαιο, αναδείχθηκε η σημασία του ψηφιακού μετασχηματισμού και των chatbots για την βελτιστοποίηση της εξυπηρέτησης πελατών, με ιδιαίτερη έμφαση στα πλεονεκτήματα της αρχιτεκτονικής RAG έναντι των παραδοσιακών μεθόδων. Παρουσιάστηκε ο κύριος σκοπός της εργασίας, που αφορά την ανάπτυξη ενός ευφυούς ξενοδοχειακού βοηθού, καθώς και η επισκόπηση υπαρχουσών λύσεων, η οποία κατέδειξε την ανάγκη για πιο εξειδικευμένες και δυναμικές εφαρμογές. Στο επόμενο κεφάλαιο, παρατίθεται το θεωρητικό υπόβαθρο της Τεχνητής Νοημοσύνης και αναλύονται διεξοδικά οι τεχνολογίες και τα εργαλεία που επιλέχθηκαν για την υλοποίηση του συστήματος.

.

2. Θεωρητικό υπόβαθρο και τεχνολογίες

2.1 Η Τεχνητή Νοημοσύνη

Η Τεχνητή Νοημοσύνη (ΤΝ) αποτελεί ένα διεπιστημονικό πεδίο έρευνας που ενσωματώνει γνώσεις και μεθόδους από την πληροφορική, τα μαθηματικά,, τη φιλοσοφία και τη νευροεπιστήμη, με στόχο τη δημιουργία μηχανών ικανών να προσομοιώσουν ή και να αναπαράγουν γνωστικές λειτουργίες του ανθρώπινου νου. Η τεχνητή νοημοσύνη στοχεύει στη σχεδίαση συστημάτων που μπορούν να αντιλαμβάνονται το περιβάλλον τους, να μαθαίνουν από την εμπειρία, να επιλύουν προβλήματα και να λαμβάνουν αποφάσεις με τρόπο που παραπέμπει σε ανθρώπινη νοημοσύνη.

Μία πιο θεωρητική προσέγγιση δίνει ο Dimiter Dobrev στο άρθρο *A Definition of Artificial Intelligence* [7], όπου προτείνει έναν ορισμό της ΤΝ ως πρόγραμμα το οποίο, σε ένα αυθαίρετο περιβάλλον (“world”), δεν ξεπερνά τον άνθρωπο σε κάθε περίπτωση αλλά “τα καταφέρνει όχι χειρότερα από τον άνθρωπο” σε αυτόν τον κόσμο. Δηλαδή, η μηχανή δεν χρειάζεται να κατέχει γνώση από πριν, αρκεί να μπορεί να μαθαίνει και να αντιδρά με τρόπο τον οποίο θα μπορούσε να κάνει ένας άνθρωπος [7].

Ένας από τους πιο σημαίνοντες επιστήμονες στην ιστορία της πληροφορικής και της τεχνητής νοημοσύνης είναι ο Alan Turing (1912–1954), ο οποίος υπήρξε μαθηματικός, κρυπτοαναλυτής και φιλόσοφος. Θεωρείται από τους ιδρυτές της επιστήμης των υπολογιστών, καθώς το έργο του έθεσε τις θεωρητικές βάσεις της υπολογισιμότητας και της μηχανικής νοημοσύνης. Εισήγαγε τη θεωρητική έννοια της “Μηχανής Turing”, ενός αφηρημένου υπολογιστικού μοντέλου που περιγράφει τον τρόπο με τον οποίο μια μηχανή μπορεί να εκτελεί οποιαδήποτε αλγοριθμική διαδικασία. Αυτή η ιδέα καθόρισε την έννοια της υπολογισιμότητας και αποτέλεσε θεμέλιο για την ανάπτυξη των ψηφιακών υπολογιστών. Η συμβολή του Turing, ωστόσο, δεν περιορίζεται στη θεωρητική υπολογισιμότητα. Στο άρθρο του *Computing Machinery and Intelligence* [8], έθεσε το διάσημο ερώτημα “Μπορούν οι μηχανές να σκεφτούν;”. Αντί να επιχειρήσει έναν ακριβή ορισμό της “σκέψης”, ο Turing πρότεινε ένα πρακτικό κριτήριο: το λεγόμενο “παιχνίδι μίμησης”, που σήμερα είναι γνωστό ως Δοκιμή Turing. Σύμφωνα με αυτήν, μια μηχανή μπορεί να θεωρηθεί “νοήμων” εάν ένας ανθρώπινος εξεταστής δεν μπορεί να διακρίνει τις απαντήσεις της από εκείνες ενός ανθρώπου σε μια τυφλή επικοινωνία. Το τεστ αυτό

αποτέλεσε τη βάση για τη μετέπειτα ανάπτυξη των chatbots, δηλαδή προγραμμάτων σχεδιασμένων να αλληλεπιδρούν γλωσσικά με τον άνθρωπο. Πρόσφατα, στο [9], παρουσιάζονται πειραματικά αποτελέσματα που υποδεικνύουν ότι ένα σύγχρονο LLM μπόρεσε, σε ελεγχόμενη εκδοχή του τριμερούς Turing test, να αξιολογηθεί ως "ανθρώπινο", δηλαδή να περάσει τη δοκιμή Turing από τους εξεταστές στο μεγαλύτερο ποσοστό των δοκιμών, παρέχοντας έτσι μια αρχική ένδειξη ότι τα μεγάλα γλωσσικά μοντέλα ενδέχεται να ανταποκρίνονται με επιτυχία σε σύγχρονες μορφές της δοκιμής.

2.2 Κατηγορίες Chatbot

Υπάρχουν διάφορες κατηγορίες chatbot ανάλογα με τη μεθοδολογία επεξεργασίας εισόδου–εξόδου που χρησιμοποιούν. Σε αυτό το κεφάλαιο εξετάζονται οι τέσσερις κύριες κατηγορίες: 1) βάσει κανόνων (rule-based), 2) βάσει τεχνικών φυσικής γλώσσας (NLP-based), 3) βάσει παραγωγής κειμένου με μηχανική/βαθιά μάθηση (Generative), 4) αρχιτεκτονική Retrieval-Augmented Generation (RAG). Κάθε τύπος περιγράφεται συνοπτικά με τα κύρια χαρακτηριστικά, τα πλεονεκτήματά του και τους περιορισμούς του, όπως αναφέρονται στη βιβλιογραφία.

Chatbots βάσει κανόνων (Rule-based)

Τα rule-based chatbots λειτουργούν αποκλειστικά με ένα σταθερό σύνολο κανόνων ή μοτίβων (if-then κανόνες) που καθορίζει τις απαντήσεις τους. Για κάθε είδος εισόδου χρηστών υπάρχει προκαθορισμένη απάντηση στο σύστημα, συνήθως χρησιμοποιώντας τεχνικές αντιστοίχισης προτύπων (pattern matching). Παραδείγματα τέτοιων συστημάτων είναι οι κλασικοί διάλογοι του ELIZA (1966) και του ALICE (1995), οι οποίοι βασίζονταν σε πολυάριθμες προκαθορισμένες κατηγορίες ερωτήσεων–απαντήσεων [10]. Ένα απ' τα πλεονεκτήματα των chatbots κανόνων είναι ότι είναι απλά στην κατασκευή και παρέχουν πολύ προβλέψιμες, συνεπείς απαντήσεις όταν οι ερωτήσεις του χρήστη αντιστοιχούν ακριβώς στα καθορισμένα μοτίβα. Λόγω της απλότητάς τους, μπορούν να αναπτυχθούν γρήγορα με περιορισμένους πόρους, κατάλληλα για εργασίες όπως συχνές ερωτήσεις (FAQs) ή βήμα-βήμα οδηγίες. Είναι αξιόπιστα σε καλά καθορισμένα σενάρια χρήσης, καθώς οι απαντήσεις τους είναι πλήρως προκαθορισμένες και δεν διαφοροποιούνται [11]. Ωστόσο, εμφανίζουν σοβαρούς περιορισμούς στην ευελιξία. Δε μπορούν να χειριστούν ερωτήσεις που δεν προβλέπονται από τους κανόνες τους, ούτε να προσαρμοστούν σε νέες καταστάσεις χωρίς χειροκίνητη επέμβαση. Δεν διαθέτουν μηχανισμό μάθησης, οι

απαντήσεις τους δεν βελτιώνονται αυτόματα και δεν κατανοούν συντακτικές ή μορφολογικές παραλλαγές που δεν ταιριάζουν με τα μοτίβα τους. Κατά συνέπεια, η συνομιλία παραμένει επιφανειακή και επαναλαμβανόμενη, ενώ συνήθως απαιτείται μεταφορά σε ανθρώπινο χειριστή όταν προκύπτουν άγνωστα ερωτήματα.

Chatbots βάσει επεξεργασίας φυσικής γλώσσας (NLP-based)

Τα NLP-based chatbots εφαρμόζουν τεχνικές φυσικής γλώσσας (NLP) για την ανάλυση της εισόδου του χρήστη και την επιλογή απάντησης. Συχνά κάνουν χρήση αναγνώρισης προθέσεων (intent recognition), ανάλυσης συναισθήματος, αναγνώρισης οντοτήτων (entity extraction) και άλλων NLP εργαλείων. Στην πράξη πολλά από αυτά τα συστήματα είναι retrieval-based: δηλαδή αναζητούν την πιο κατάλληλη απάντηση σε μια υπάρχουσα βάση δεδομένων ερωταπαντήσεων ή σε ένα σύνολο εγγράφων και επιστρέφουν αυτή την έτοιμη απάντηση. Σε αντίθεση με τα rule-based, δεν περιορίζονται σε αυστηρά προκαθορισμένα μοτίβα, μπορούν να δεχτούν ποικιλία εκφράσεων και χρησιμοποιούν πιο σύνθετους αλγόριθμους αντιστοίχισης κειμένου για να βρουν την απάντηση [10]. Ένα απ' τα πλεονεκτήματα των chatbots NLP είναι ότι είναι περισσότερο ευέλικτα και ρεαλιστικά σε σύγχρονες εφαρμογές. Συνήθως έχουν μεγάλο ευρετήριο απαντήσεων ή αναφοράς π.χ. βάσεις γνώσεων, FAQs, βάση εγγράφων από όπου αντλούν απαντήσεις. Αυτό σημαίνει ότι μπορούν να επικαλύψουν ευρύ φάσμα ερωτήσεων, εφόσον η απάντηση υπάρχει ήδη καταχωρημένη. Επιπλέον, τείνουν να παράγουν αξιόπιστες και ακριβείς απαντήσεις, αφού δεν φτιάχνουν περιεχόμενο εκ του μηδενός αλλά ανασύρουν προκαθορισμένες απαντήσεις. Ωστόσο, το εύρος των δυνατοτήτων τους περιορίζεται από τη διαθεσιμότητα των δεδομένων τους. Εάν μια ερώτηση δεν καλύπτεται από τη βάση γνώσεων τους, δεν μπορούν να παράγουν ούτε να συμπληρώσουν την απάντηση, η συνομιλία σταματά ουσιαστικά εκεί. Επίσης, η ενημέρωση του συστήματος απαιτεί συχνά χειροκίνητη προσθήκη ή ανανέωση των απαντήσεων, κάτι που δυσκολεύει τη διατήρηση μεγάλων και ενημερωμένων ευρετηρίων. Συνοπτικά, ενώ τα NLP-based bots είναι πιο έξυπνα από τα απλά rule-based, εξακολουθούν να μην μαθαίνουν και βασίζονται σε δοσμένα πρότυπα απαντήσεων [11].

Chatbots παραγωγής κειμένου (Generative)

Τα παραγωγικά chatbots είναι η πιο προχωρημένη κατηγορία συστημάτων συνομιλίας που σχετίζονται με την τεχνητή νοημοσύνη. Σε αντίθεση με τα rule-based που βασίζονται σε προκαθορισμένες απαντήσεις, τα παραγωγικά chatbot έχουν την ικανότητα

να δημιουργούν δυναμικά νέες απαντήσεις, χρησιμοποιώντας τεχνικές βαθιάς και μηχανικής μάθησης. Ένα χαρακτηριστικό παράδειγμα αυτής της μεθόδου είναι τα μοντέλα GPT της OpenAI, όπως το ChatGPT. Αυτά χρησιμοποιούν εξελιγμένα νευρωνικά δίκτυα για να κατανοήσουν τις ερωτήσεις των χρηστών και να παρέχουν απαντήσεις που είναι συνεπείς και φυσικές. Η λειτουργία αυτών των συστημάτων στηρίζεται στην κατανόηση του πλαισίου, χρησιμοποιώντας σύνθετους αλγορίθμους επεξεργασίας φυσικής γλώσσας. Για παράδειγμα, σε μια ερώτηση όπως "Πώς είναι ο καιρός σήμερα;", το μοντέλο έχει την ικανότητα να αναλύσει τη σημασιολογία και να δώσει μια απάντηση όπως: "Ο σημερινός καιρός είναι ηλιόλουστος με θερμοκρασία 25 βαθμούς Κελσίου". Αυτή η ικανότητα των παραγωγικά chatbots να παράγουν απαντήσεις που ταιριάζουν στην ανθρώπινη επικοινωνία τα καθιστά ιδιαίτερα αποτελεσματικά σε ανοιχτού τύπου συζητήσεις, όπου η συνομιλία δεν περιορίζεται από προκαθορισμένες δομές ή συγκεκριμένα θέματα. Το βασικό πλεονέκτημα των παραγωγικά chatbots είναι ότι παρέχουν μια πιο φυσική εμπειρία επικοινωνίας. Μπορούν να συμμετέχουν σε πολλές και αυθόρμητες συνομιλίες με έναν ανθρώπινο τρόπο. Ωστόσο, αυτή η ευελιξία φέρνει μαζί της προκλήσεις στην ακρίβεια, καθώς τα συγκεκριμένα μοντέλα μπορεί να παράγουν ανακριβείς ή "φτιαγμένες" απαντήσεις (Hallucinations). Αυτό αποτυπώνει τις δυσκολίες που σχετίζονται με τη διαδικασία εκπαίδευσης και ρύθμισης τους. Τα μεγάλα γλωσσικά μοντέλα συνεχίζουν να εξελίσσονται και παρ' όλο που έχουν γίνει σημαντικά βήματα στην έρευνα, το θέμα αυτό εξακολουθεί να αποτελεί επίκαιρο αντικείμενο μελέτης [12].

Chatbots με αρχιτεκτονική Retrieval-Augmented Generation (RAG)

Τα RAG chatbots είναι σύγχρονα συστήματα υβριδικής προσέγγισης που συνδυάζουν μηχανισμούς ανάκτησης πληροφοριών (retrieval) με γεννήτριες γλώσσας (generative models). Στα RAG, πριν παραχθεί η απάντηση, το σύστημα εκτελεί ανάκτηση σχετικών κειμένων ή εγγράφων από μια εξωτερική πηγή (π.χ. βάση δεδομένων). Στη συνέχεια, ένα LLM ενσωματώνει αυτές τις πληροφορίες για να γράψει τη τελική απάντηση. Με αυτόν τον τρόπο, οι απαντήσεις βασίζονται σε τεκμηριωμένες, πραγματικές πηγές. Το RAG επιλέχτηκε ως τεχνολογία για την πτυχιακή καθώς παρέχει σημαντικά οφέλη στην ακρίβεια και την επικαιροποίηση των απαντήσεων. Δεδομένου ότι τραβά πληροφορίες από εξωτερικές πηγές, το σύστημα αποφεύγει πολλά φαινόμενα hallucination των καθαρά παραγωγικών μοντέλων, παράγοντας απαντήσεις βασιζόμενο σε πραγματικά δεδομένα

[13]. Επιπλέον, σε αντίθεση με απλά LLMs που είναι εσωτερικά κλειστά μετά την εκπαίδευσή τους, ένα RAG μπορεί να ενημερώνει την βάση γνώσης του συνεχώς χωρίς επανεκπαίδευση, επιτρέποντας ακριβείς απαντήσεις σε ταχύτατα εξελισσόμενα πεδία (π.χ. ιατρική ή ειδησεογραφία). Σε μελέτες έχει βρεθεί ότι τα RAG συστήματα ξεπερνούν σε αξιοπιστία τα καθαρά παραγωγικά chatbots όταν απαιτείται ακριβής γνώση [2].

Παρά τα σημαντικά πλεονεκτήματα της αρχιτεκτονικής Retrieval-Augmented Generation (RAG), υπάρχουν ακόμη αρκετοί περιορισμοί. Ένα από τα βασικά ζητήματα είναι η κλιμακωσιμότητα και αποδοτικότητα, καθώς η χρήση εξωτερικών βάσεων δεδομένων απαιτεί αποδοτικούς μηχανισμούς ανάκτησης και υψηλούς υπολογιστικούς πόρους, κάτι που δυσχεραίνει την ανάπτυξη σε πραγματικό χρόνο. Η ποιότητα και συνάφεια των ανακτώμενων πληροφοριών αποτελεί επίσης πρόκληση, αφού λανθασμένα ή παρωχημένα δεδομένα μειώνουν την ακρίβεια του παραγόμενου περιεχομένου. Παράλληλα, ζητήματα μεροληψίας (bias) παραμένουν σημαντικά, καθώς οι RAG μηχανισμοί ενδέχεται να αναπαράγουν προκαταλήψεις που ενσωματώνονται στα δεδομένα εκπαίδευσης ή ανάκτησης. Επιπλέον, τα RAG συχνά δυσκολεύονται στη συνοχή μεταξύ των ανακτημένων γνώσεων και του παραγόμενου κειμένου, οδηγώντας σε ανακρίβειες ή ασυνέπειες. Τέλος, η έλλειψη διαφάνειας και ερμηνευσιμότητας καθιστά δυσδιάκριτο τον τρόπο με τον οποίο οι ανακτημένες πληροφορίες επηρεάζουν την τελική απάντηση, περιορίζοντας την εμπιστοσύνη των χρηστών [2].

2.3 Αρχιτεκτονική RAG Chatbot

Η αρχιτεκτονική Retrieval-Augmented Generation (RAG) αποτελεί τη σύγχρονη απάντηση στις προκλήσεις των Μεγάλων Γλωσσικών Μοντέλων (LLMs), ιδίως το πρόβλημα της παραίσθησης (hallucination). Η μέθοδος αυτή συνδυάζει δύο διακριτά, αλλά αλληλένδετα, λειτουργικά στάδια, το Retrieval (Ανάκτηση) και το Generation (Παραγωγή), τα οποία λειτουργούν διαδοχικά, επιτρέποντας στο σύστημα να βελτιώσει την ακρίβεια και την αξιοπιστία των απαντήσεων του.

Το Σύστημα Ανάκτησης (The Retriever)

Το πρώτο στάδιο είναι θεμελιώδους σημασίας, καθώς είναι υπεύθυνο για τον εντοπισμό και την εξαγωγή των πλέον σχετικών κομματιών πληροφορίας (chunks) από μια

εξωτερική βάση γνώσης, διασφαλίζοντας ότι η απάντηση θα βασιστεί σε τεκμηριωμένες πληροφορίες. Η διαδικασία αυτή περιλαμβάνει:

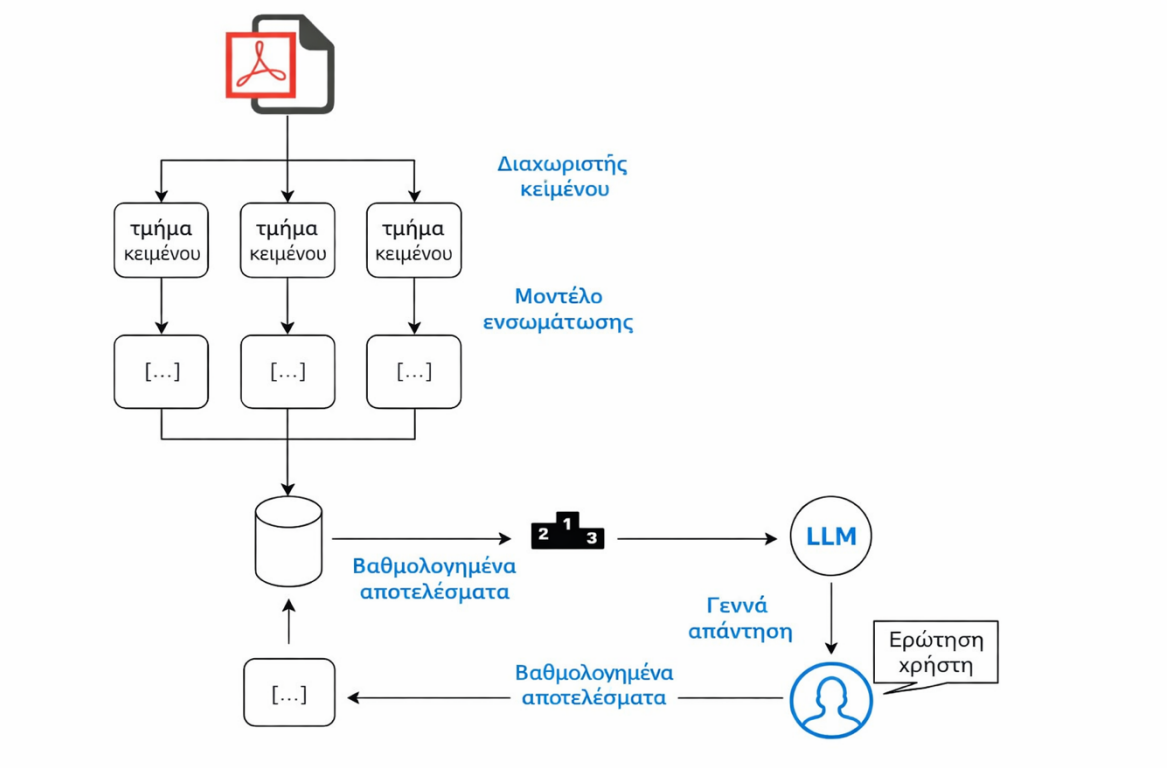
1. Προεπεξεργασία και Διανυσματοποίηση (Vectorization): Τα εξωτερικά έγγραφα πρέπει πρώτα να υποβληθούν σε προεπεξεργασία και να διασπαστούν σε διαχειρίσιμα τμήματα. Στη συνέχεια, ένα μοντέλο ενσωμάτωσης (embedding model) μετατρέπει αυτά τα τμήματα σε διανυσματικές αναπαραστάσεις (vector embeddings), οι οποίες κωδικοποιούν τη σημασιολογική έννοια του κειμένου. Αυτά τα διανύσματα αποθηκεύονται σε μία διανυσματική βάση δεδομένων (Vector Database), η οποία είναι βελτιστοποιημένη για γρήγορη αναζήτηση ομοιότητας.
2. Σημασιολογική Αναζήτηση (Semantic Search): Όταν ο χρήστης υποβάλλει ένα ερώτημα, αυτό μετατρέπεται επίσης σε διάνυσμα. Η βάση δεδομένων πραγματοποιεί αναζήτηση, βρίσκοντας τα κομμάτια κειμένου των οποίων τα διανύσματα είναι σημασιολογικά κοντινότερα στο ερώτημα του χρήστη. Τα ανακτηθέντα αυτά κομμάτια εξάγονται και ονομάζονται ανακτηθέν πλαίσιο (retrieved context). Ο ρόλος του Retriever είναι να ανακτήσει τις πληροφορίες από τις εξωτερικές πηγές με βάση την εισαγωγή του χρήστη.

Το Σύστημα Παραγωγής (The Generator)

Το δεύτερο στάδιο είναι η σύνθεση της τελικής απόκρισης και αξιοποιεί την ικανότητα του LLM (Generator) να παράγει φυσική γλώσσα. Η διαδικασία περιλαμβάνει τα εξής:

1. Ενίσχυση (Augmentation): Το ανακτηθέν πλαίσιο πληροφορίας ενισχύει (augments) το αρχικό ερώτημα του χρήστη. Αυτό το ενισχυμένο ερώτημα (το αρχικό ερώτημα του χρήστη μαζί με τα ανακτηθέντα κείμενα) διοχετεύεται στο Παραγωγικό Μοντέλο (LLM).
2. Τεκμηριωμένη Παραγωγή: Η τελική φάση περιλαμβάνει τη δημιουργία των απαντήσεων χρησιμοποιώντας τις ανακτηθείσες και ενισχυμένες πληροφορίες. Το LLM χρησιμοποιεί το ανακτηθέν κείμενο ως την άμεση πηγή αλήθειας, διασφαλίζοντας ότι οι απαντήσεις που παράγονται είναι συνεκτικές και πραγματικά ορθές, καθώς βασίζονται σε αξιόπιστες πηγές. Με αυτόν τον τρόπο, η RAG επιτυγχάνει τον βασικό της στόχο: να βελτιώσει την ακρίβεια των απαντήσεων που

παρέχονται από τα chatbots, διασφαλίζοντας ότι αυτές βασίζονται σε πληροφορίες που έχουν ανακτηθεί από αξιόπιστες πηγές [14].



Εικόνα 4 : Διάγραμμα αρχιτεκτονικής RAG

2.4 Τεχνολογίες που χρησιμοποιήθηκαν

Για την ανάπτυξη του RAG chatbot χρησιμοποιήθηκαν διάφορες τεχνολογίες όπως η γλώσσα προγραμματισμού Python, το framework Langchain, streamlit, η βιβλιοθήκη Pandas, FAISS για vector databases, βάσεις δεδομένων όπως η SQLite, DB browser for SQLite και φυσικά τα μεγάλα γλωσσικά μοντέλα τεχνητής νοημοσύνης όπως το DeepSeek. Η συγγραφή του κώδικα έγινε μέσω του IDE Pycharm της Jetbrains.

2.4.1 Python

Η Python αποτελεί έναν από τις πιο δημοφιλείς προγραμματιστικές γλώσσες στον ακαδημαϊκό και ερευνητικό χώρο, προσφέροντας μοναδικά πλεονεκτήματα για

πτυχιακές εργασίες που εστιάζουν σε επιστημονικούς υπολογισμούς, ανάλυση δεδομένων και έρευνα. Σε έρευνες που αφορούν την εκπαίδευση, έχει αποδειχθεί ότι η εκμάθηση της Python ενισχύει τη μαθησιακή απόδοση και την αποτελεσματικότητα των φοιτητών στον προγραμματισμό, σε σύγκριση με άλλες γλώσσες όπως η Java. Αυτό την καθιστά εξαιρετική επιλογή για φοιτητές που αναπτύσσουν πτυχιακές εργασίες [15]. Κάποιοι απ τους τομείς εφαρμογής της είναι:

Τεχνητή νοημοσύνη και Machine learning

- Μηχανική Μάθηση: Ταξινόμηση, πρόβλεψη, ανάλυση προτύπων
- Επεξεργασία Φυσικής Γλώσσας: Ανάλυση κειμένων, εξόρυξη γνώσης
- Computer Vision: Αναγνώριση εικόνων και οπτική επεξεργασία

Επιχειρήσεις και Οικονομία

- Marketing και Διαφήμιση: Ανάλυση μεγάλων δεδομένων, συστήματα συστάσεων
- Χρηματοοικονομικά: Ανάλυση κινδύνου, πρόβλεψη αγορών

Web Development και εφαρμογές

- Δυναμικές Ιστοσελίδες: Django, Flask frameworks
- APIs και Microservices: RESTful υπηρεσίες
- Big Data Analytics: Επεξεργασία μεγάλων όγκων δεδομένων πχ με τη βιβλιοθήκη Pandas

2.4.2 LangChain

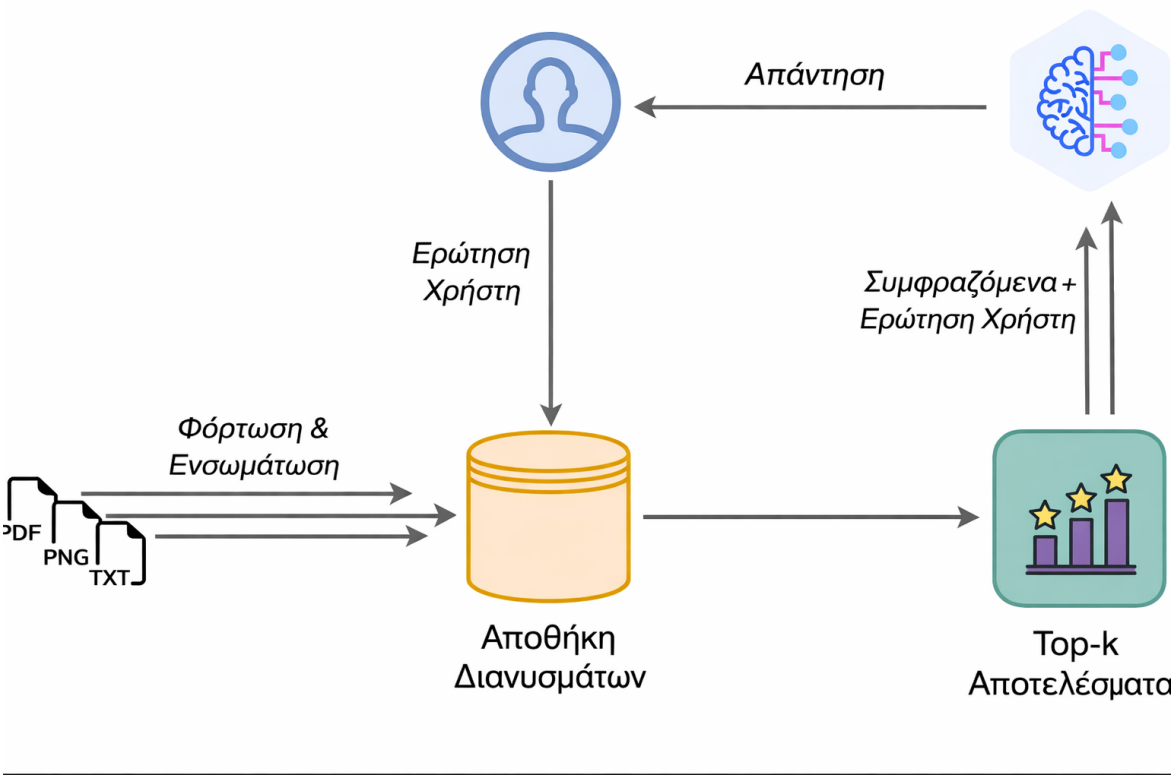
Το LangChain αποτελεί ένα framework ανοιχτού κώδικα που έχει αναπτυχθεί με στόχο τη διευκόλυνση της δημιουργίας εφαρμογών βασισμένων σε μεγάλα γλωσσικά μοντέλα (LLMs). Η βασική του συνεισφορά έγκειται στο ότι οργανώνει και αυτοματοποιεί πολύπλοκες διεργασίες που απαιτούνται για την αποτελεσματική αξιοποίηση των LLMs, παρέχοντας μια δομημένη αλλά ταυτόχρονα ευέλικτη αρχιτεκτονική. Μέσα από αυτήν τη δομή, δίνεται η δυνατότητα στους προγραμματιστές να συνδυάζουν διαφορετικά στοιχεία, όπως επεξεργασία κειμένων, διαχείριση γνώσης, αλληλεπιδράσεις με APIs και συντονισμό εξωτερικών εργαλείων, με τρόπο που να βελτιστοποιεί την αλληλεπίδραση με το εκάστοτε μοντέλο.

Στην αρχιτεκτονική του LangChain βρίσκονται οι αλυσίδες (chains), δηλαδή διαδοχικές ακολουθίες βημάτων που συνδυάζουν κλήσεις σε μοντέλα, προεπεξεργασία δεδομένων και χρήση εξωτερικών εργαλείων. Συμπληρωματικά, το πλαίσιο (context) υποστηρίζει την έννοια της μνήμης (memory), η οποία επιτρέπει στις εφαρμογές να "θυμούνται" προηγούμενες αλληλεπιδράσεις, προσδίδοντας συνέπεια και φυσικότητα σε διαλόγους. Παράλληλα, εισάγει τους πράκτορες (agents), που έχουν τη δυνατότητα να αποφασίζουν δυναμικά ποια ενέργεια ή εργαλείο θα χρησιμοποιηθεί ανάλογα με τα δεδομένα εισόδου.

Επιπλέον, παρέχει κλάσεις για τη φόρτωση και επεξεργασία εγγράφων από διάφορες μορφές (PDF, CSV, HTML κ.ά.) και τη δημιουργία embeddings και ευρετηρίων (vector stores). Ως αποτέλεσμα, διευκολύνει την ταχεία ανάπτυξη σύνθετων εφαρμογών LLM, όπως συστήματα ανάκτησης γνώσης (RAG) που ενσωματώνουν σχετικό περιεχόμενο στα prompts (prompt orchestration). Αυτό καθιστά εφικτή την προσαρμογή των μοντέλων σε συγκεκριμένα πεδία γνώσης, εξασφαλίζοντας πιο ακριβείς και χρήσιμες απαντήσεις.
Ένα ακόμη πλεονέκτημα του LangChain είναι η εκτεταμένη υποστήριξη ενσωματώσεων (integrations) με βιβλιοθήκες επεξεργασίας δεδομένων, βάσεις γνώσης και εξειδικευμένα συστήματα, όπως vector databases. Η modular φιλοσοφία του επιτρέπει την επαναχρησιμοποίηση και τον πειραματισμό με διαφορετικές διαμορφώσεις, καθιστώντας την ανάπτυξη πιο ευέλικτη και παραγωγική.

Συνολικά, το LangChain συνδυάζει την απλότητα με τη λειτουργικότητα, αποτελώντας ένα από τα πιο ισχυρά εργαλεία για την αξιοποίηση των LLMs σε πρακτικές εφαρμογές, όπως chatbots, συστήματα υποστήριξης πελατών, αυτόματη ανάλυση δεδομένων και πλατφόρμες παραγωγής περιεχομένου.

Παρακάτω αναπαρίσταται η αρχιτεκτονική pipeline του LangChain που αναδεικνύει τη διαδικασία RAG. Έγγραφα σε διάφορες μορφές (π.χ., PDF, κείμενο, εικόνες) προφορτώνονται και ενσωματώνονται σε ένα vector store. Όταν ένας χρήστης υποβάλει ένα query, το σύστημα ανακτά τα top-k πιο σχετικά έγγραφα βάσει vector similarity. Αυτά τα έγγραφα συνδυάζονται με το query για να παρέχουν συμφραζόμενα στο LLM, το οποίο στη συνέχεια παράγει μια ακριβή και εμπλουτισμένη με συμφραζόμενα απάντηση. Αυτή η αρχιτεκτονική ενισχύει την ικανότητα του μοντέλου να παράγει απαντήσεις με πραγματική τεκμηρίωση, ενσωματώνοντας σχετική γνώση από το vector store [16][17].



Εικόνα 5 : LangChain pipeline αρχιτεκτονική

2.4.3 Vector Databases - FAISS

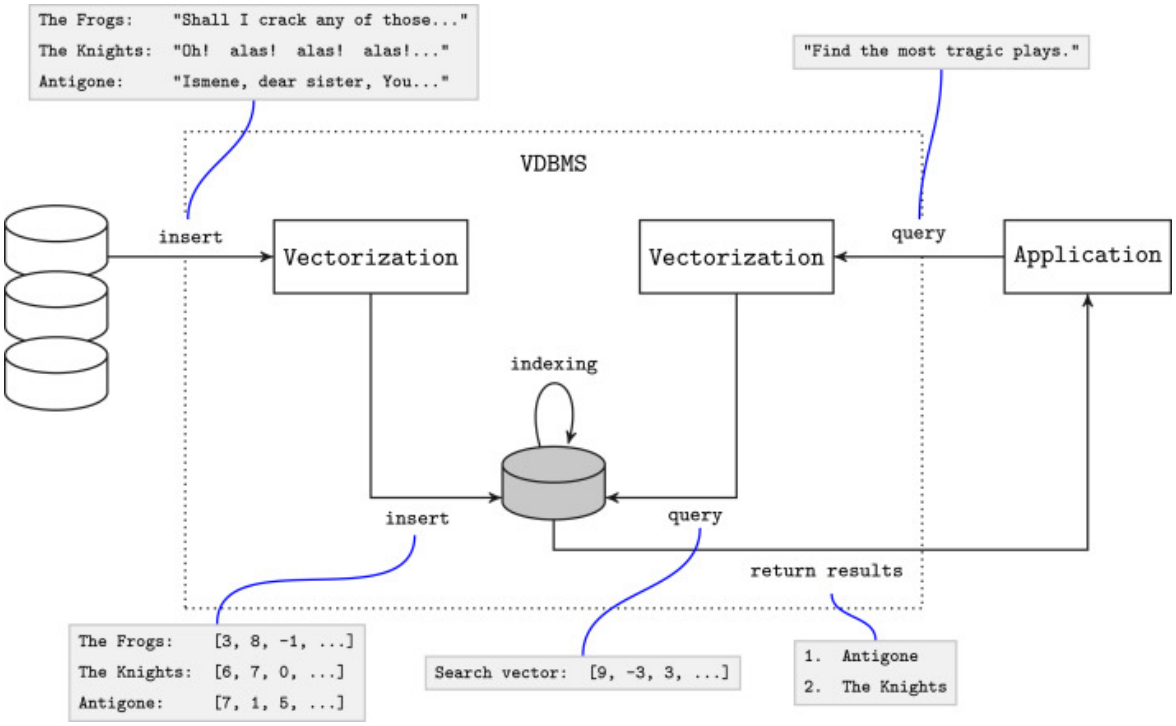
Οι διανυσματικές βάσεις δεδομένων αποτελούν εξειδικευμένο τύπο συστήματος διαχείρισης βάσεων δεδομένων που εστιάζει πρωτίστως στην αποδοτική διαχείριση υψηλοδιάστατων διανυσματικών δεδομένων. Σε αντίθεση με τις παραδοσιακές βάσεις δεδομένων που αποθηκεύουν δομημένα δεδομένα, οι διανυσματικές βάσεις χειρίζονται δεδομένα που έχουν μετατραπεί σε αριθμητικά διανύσματα μέσω μιας διαδικασίας που ονομάζεται vectorization η οποία αποθηκεύει τα χαρακτηριστικά του αντικειμένου δεδομένων σε ένα επεξεργάσιμο αριθμητικό διάνυσμα[18].

Τα διανυσματικά δεδομένα δημιουργούνται μέσω των embeddings, τα οποία είναι διανυσματικές αναπαραστάσεις, συνήθως παραγόμενες από ένα νευρωνικό δίκτυο, που αποτυπώνουν το στοιχείο πολυμέσων σε έναν διανυσματικό χώρο, όπου παρόμοια στοιχεία τοποθετούνται κοντά μεταξύ τους. Αυτή η διαδικασία επιτρέπει σε διάφορες μορφές πολυμέσων όπως λέξεις, κείμενο, εικόνες, να μετατραπούν σε διανυσματική μορφή, ακόμη και να κωδικοποιούν σχέσεις αντικειμένων, για παράδειγμα σχέσεις κειμένου-εικόνας ή κειμένου-ήχου.

Η κύρια δυνατότητα των διανυσματικών βάσεων είναι η παροχή γρήγορης και ακριβούς αναζήτησης ομοιότητας και ανάκτησης, καθώς μπορούν να βρουν τα πιο παρόμοια ή σχετικά δεδομένα βάσει της διανυσματικής απόστασης ή ομοιότητάς τους. Αυτή η ικανότητα αξιοποιείται από το γεγονός ότι ο embeddings extractor (τυπικά ένα νευρωνικό δίκτυο) έχει σχεδιαστεί έτσι ώστε η απόσταση μεταξύ των embeddings να αντικατοπτρίζει την ομοιότητα μεταξύ των αντίστοιχων πολυμέσων τους.

Στη βάση της τεχνολογίας των διανυσματικών βάσεων δεδομένων βρίσκεται το FAISS (Facebook AI Similarity Search), που αποτελεί βιβλιοθήκη αφιερωμένη στην αναζήτηση διανυσματικής ομοιότητας, μια βασική λειτουργικότητα των διανυσματικών βάσεων δεδομένων. Το FAISS είναι μια βιβλιοθήκη μεθόδων ευρετηρίασης σχετικών στοιχείων που χρησιμοποιούνται για την αναζήτηση, ομαδοποίηση, συμπίεση και μετασχηματισμό διανυσμάτων. Από την κυκλοφορία του το 2017, το FAISS έχει αναδειχθεί ως μία από τις πιο δημοφιλείς βιβλιοθήκες διανυσματικής αναζήτησης, συγκεντρώνοντας 30.000 αστέρια στο GitHub και περισσότερες από 4.000 παραπομπές στη δημοσίευσή του για την υλοποίηση GPU. Τεχνολογικά, το FAISS χρησιμοποιεί Approximate Nearest Neighbor (ANNS) τεχνική, η οποία δίνει προτεραιότητα στην επίτευξη γρηγορότερων χρόνων αναζήτησης μέσω εύρεσης του πλησιέστερου γείτονα σε μεγάλα δεδομένα [19].

Στον χώρο της τεχνητής νοημοσύνης, οι διανυσματικές βάσεις δεδομένων αποτελούν βασικό στοιχείο για την ενίσχυση μεγάλων γλωσσικών μοντέλων. Όταν ένας τελικός χρήστης κάνει ένα prompt σε έναν chatbot, το ερώτημα φυσικής γλώσσας διανυσματοποιείται και χρησιμοποιείται ως διάνυσμα ερωτήματος για μια διανυσματική βάση δεδομένων μακροχρόνιας μνήμης για να βρει τις top-k παρόμοιες συνομιλίες. Η προσέγγιση retrieval augmented generation (RAG) χρησιμοποιεί διανυσματικές βάσεις για να αποθηκεύσει έγγραφα τα οποία χρησιμοποιούνται ως επιπλέον input που παρέχει πλαίσιο στο LLM [18].



Εικόνα 6 : Λειτουργία ενός Vector Database Management System[4]

2.4.4 Streamlit

Το Streamlit είναι ένα ανοιχτού κώδικα (open-source) framework για την Python που έχει αποκτήσει μεγάλη δημοτικότητα στην ανάπτυξη διαδραστικών web εφαρμογών βασισμένων σε δεδομένα. Το κύριο πλεονέκτημά του έγκειται στην απλοποίηση της διαδικασίας μετατροπής Python scripts, ιδιαίτερα αυτών που σχετίζονται με ανάλυση δεδομένων και μηχανική μάθηση, σε πλήρως λειτουργικές εφαρμογές web. Η παραδοσιακή ανάπτυξη web για παρόμοιες εφαρμογές απαιτεί γνώσεις HTML, CSS, JavaScript και web frameworks, κάτι που μπορεί να αποτελεί εμπόδιο για data scientists ή μηχανικούς μηχανικής μάθησης. Το Streamlit αφαιρεί αυτή την πολυπλοκότητα, επιτρέποντας στους προγραμματιστές να επικεντρωθούν στα δεδομένα και τα μοντέλα τους αντί για την υποδομή της web εφαρμογής.

[4] Εικόνα από [18]

Ένα από τα βασικά πλεονεκτήματα του Streamlit είναι η ελάχιστη ανάγκη για boilerplate κώδικα. Οι προγραμματιστές δεν χρειάζεται να ορίζουν backend, routes ή να διαχειρίζονται HTTP αιτήματα, αρκεί να γράψουν ένα Python script για να δημιουργηθεί αυτόματα μια ζωντανή web εφαρμογή. Αυτό επιτρέπει ταχεία πρωτοτυποποίηση, μετατρέποντας ιδέες σε λειτουργικές εφαρμογές μέσα σε λίγες ώρες ή ακόμα και λεπτά. Επιπλέον, το Streamlit ενσωματώνεται άψογα με δημοφιλείς Python βιβλιοθήκες δεδομένων όπως pandas, Matplotlib και Plotly, υποστηρίζοντας δυναμικές οπτικοποιήσεις και dashboards που μπορούν να αλληλεπιδρούν μέσω widgets όπως sliders, buttons και checkboxes.

Το Streamlit διαθέτει επίσης τη δυνατότητα live-reload, που ενημερώνει αυτόματα την εφαρμογή όταν αλλάζει ο κώδικας, διευκολύνοντας έναν γρήγορο επαναληπτικό κύκλο ανάπτυξης. Οι πιο προχωρημένες εφαρμογές μπορούν να αξιοποιήσουν custom components και styling, επιτρέποντας την ενσωμάτωση JavaScript λειτουργιών αν χρειαστεί. Υποστηρίζει εύκολη ανάπτυξη σε πλατφόρμες όπως Streamlit Sharing, Heroku, AWS ή Google Cloud και διαθέτει ενεργή κοινότητα με πληθώρα πόρων και πρόσθετων στοιχείων.

Στο πλαίσιο ενός RAG (Retrieval-Augmented Generation) chatbot, το Streamlit προσφέρει ιδανική πλατφόρμα για τη δημιουργία φιλικού προς τον χρήστη interface, επιτρέποντας αλληλεπίδραση σε πραγματικό χρόνο με το μοντέλο, οπτικοποίηση των ανακτημένων εγγράφων και πειραματισμό με διαφορετικά prompts. Η απλότητα, η διαδραστικότητα και η συμβατότητά του με Python-based εργαλεία AI το καθιστούν πρακτική επιλογή για την υλοποίηση και κοινοποίηση προηγμένων chatbot εφαρμογών με ελάχιστο ανάπτυξης overhead [20].

2.4.5 SQLite – DB browser for SQLite

Η SQLite αποτελεί μία ελαφριά και αυτόνομη σχεσιακή βάση δεδομένων, η οποία χρησιμοποιείται ευρέως σε εφαρμογές όπου απαιτείται αποθήκευση δεδομένων χωρίς την ανάγκη υποστήριξης από ξεχωριστό διακομιστή βάσης δεδομένων. Η σχεδίασή της επιτρέπει την εύκολη ενσωμάτωση σε λογισμικό όπως εφαρμογές για κινητές συσκευές, προγράμματα περιήγησης και ενσωματωμένα συστήματα. Βασικό πλεονέκτημα της SQLite είναι η ευχρηστία της, καθώς δεν απαιτείται πολύπλοκη διαδικασία εγκατάστασης ή ρύθμισης, ενώ όλα τα δεδομένα αποθηκεύονται σε ένα μοναδικό αρχείο στη συσκευή του

χρήστη. Αυτό καθιστά τη διαχείρισή της απλή και προσβάσιμη ακόμα και για μη ειδικούς. Η SQLite έχει εξελιχθεί σε μια πλήρως λειτουργική μηχανή βάσης δεδομένων που υποστηρίζει σύνθετα αναλυτικά ερωτήματα, παρόλο που είναι κυρίως σχεδιασμένη για ταχεία επεξεργασία συναλλαγών online (OLTP) [21].

Το DB Browser for SQLite είναι ένα φιλικό προς το χρήστη οπτικό εργαλείο διαχείρισης βάσεων δεδομένων SQLite. Παρέχει τη δυνατότητα δημιουργίας, οργάνωσης, και τροποποίησης των πινάκων και των δεδομένων με απλές ενέργειες μέσω γραφικού περιβάλλοντος. Το εργαλείο αυτό είναι ιδιαίτερα χρήσιμο για φοιτητές, ερευνητές αλλά και επαγγελματίες που θέλουν να εξερευνήσουν ή να τροποποιήσουν τις βάσεις δεδομένων τους χωρίς να χρειάζεται να γράφουν σύνθετες εντολές SQL. Η χρήση του συμβάλλει σημαντικά στην κατανόηση της δομής και της λειτουργίας των σχεσιακών βάσεων δεδομένων, διευκολύνοντας την εκμάθηση και την επεξεργασία δεδομένων σε εκπαιδευτικά ή μικρά επαγγελματικά έργα [22].

2.4.6 DeepSeek

Το DeepSeek LLM αποτελεί ένα νέο ανοιχτού κώδικα μεγάλο γλωσσικό μοντέλο που παρουσιάστηκε από την κινεζική εταιρεία DeepSeek-AI στο πλαίσιο μιας στρατηγικής για ανοικτή έρευνα.. Με αυτή την ανοιχτή διάθεση, το DeepSeek παρέχεται ως εργαλείο σε ερευνητές και επιχειρήσεις χωρίς σημαντικούς περιορισμούς. Σύμφωνα με την επίσημη τεχνική τεκμηρίωση, το DeepSeek υπερβαίνει σε απόδοση προηγούμενα μοντέλα ανοιχτού κώδικα όπως το LLaMA-2 70B και σε γενικές αξιολογήσεις υπερέχει του GPT-3.5 [23].

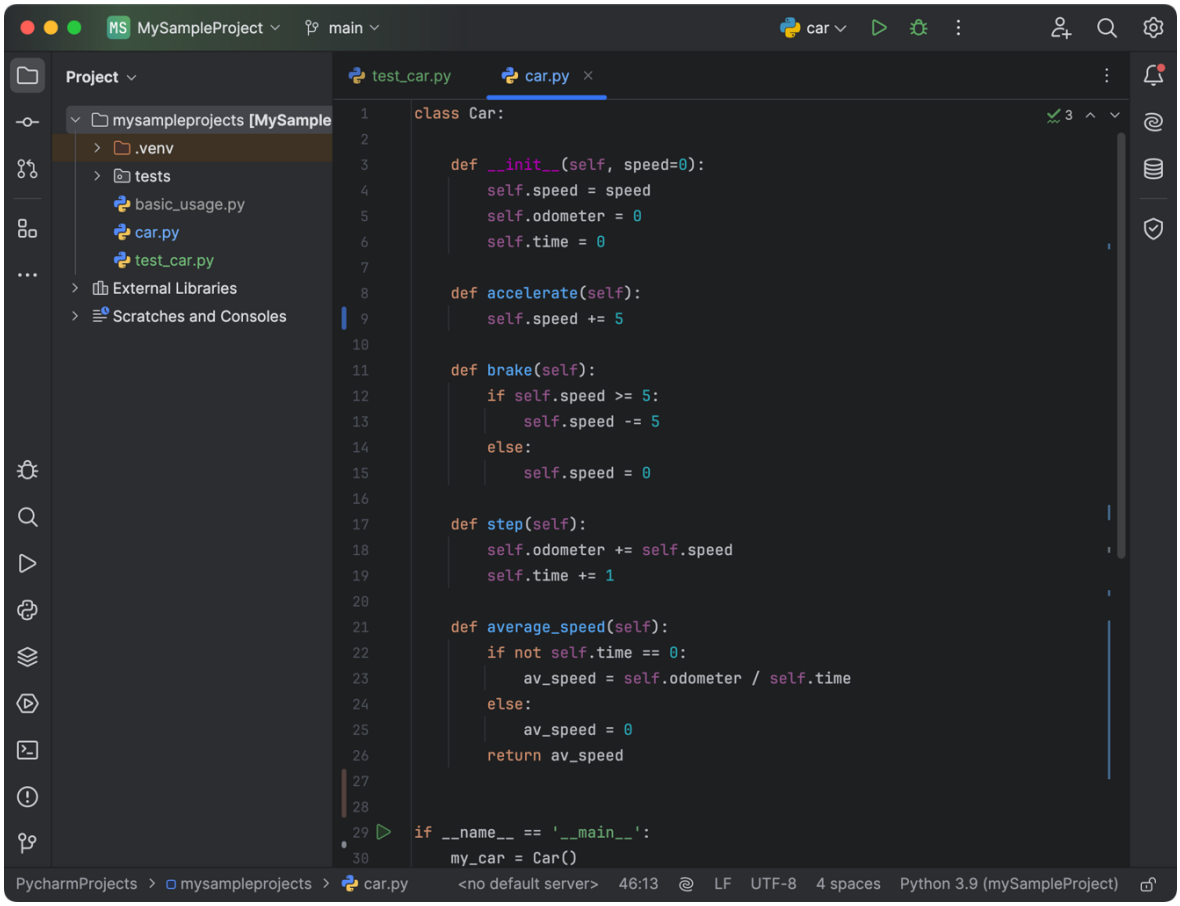
Μία σημαντική καινοτομία του DeepSeek είναι στη χρήση της αρχιτεκτονικής Mixture-of-Experts (MoE), η οποία επιτρέπει στο μοντέλο να ενεργοποιεί μόνο ένα υποσύνολο των παραμέτρων του για κάθε εισερχόμενο token, μειώνοντας δραστικά το υπολογιστικό κόστος χωρίς απώλεια απόδοσης. Το DeepSeek-V3, με συνολικές παραμέτρους 671B αλλά μόνο 37B ενεργές για κάθε token, καταφέρνει να επιτύχει απόδοση συγκρίσιμη με τα καλύτερα κλειστού κώδικα μοντέλα, όπως το GPT-4, με κόστος εκπαίδευσης μόλις 2.788M ώρες GPU[24].

Ιδιαίτερη σημασία έχει το DeepSeek-R1, το οποίο εστιάζει στις δυνατότητες συλλογισμού μέσω τεχνικών ενισχυτικής μάθησης. Αυτό το μοντέλο έχει επιδείξει εξαιρετική απόδοση σε σύνθετες εργασίες που απαιτούν λογική σκέψη, ανταγωνιζόμενο επιτυχώς το OpenAI-o1. Η ικανότητά του να αναλύει πολύπλοκα προβλήματα και να παρέχει λεπτομερείς εξηγήσεις το καθιστά ιδανικό για εφαρμογές που απαιτούν υψηλού επιπέδου κατανόηση και αιτιολόγηση των αποτελεσμάτων [25].

Ένα από τα σημαντικότερα πλεονεκτήματα του DeepSeek είναι το πολύ χαμηλό κόστος εκπαίδευσης και συντήρησης σε σχέση με τα εμπορικά μοντέλα. Η εκπαίδευση του DeepSeek-V3 ολοκληρώθηκε μς εκτιμώμενο κόστος 5 εκατομμυρίων δολαρίων, ποσό σημαντικά χαμηλότερο από τα δεκάδες εκατομμύρια που απαιτούνται για τα GPT-4. Παράλληλα, η ύπαρξη του επίσημου DeepSeek API με χαμηλό κόστος σε σύγκριση μ' αυτά άλλων LLMs δίνει τη δυνατότητα σε προγραμματιστές και ερευνητές να ενσωματώσουν το μοντέλο απευθείας σε εφαρμογές και ερευνητικά έργα, ενώ οι εκδόσεις ανοιχτού κώδικα του μοντέλου είναι πλήρως διαθέσιμες μέσω πλατφορμών όπως το Hugging Face [26].

2.4.7 PyCharm IDE

Για την ανάπτυξη της παρούσας εφαρμογής επιλέχθηκε το PyCharm της JetBrains, ένα από τα πιο δημοφιλή και προηγμένα περιβάλλοντα ολοκληρωμένης ανάπτυξης (IDE) για Python. Το PyCharm παρέχει μια ευρεία γκάμα δυνατοτήτων που καθιστούν την ανάπτυξη Python εφαρμογών πιο αποδοτική και παραγωγική. Οι βασικές λειτουργίες περιλαμβάνουν προηγμένες δυνατότητες auto-completion κώδικα, syntax highlighting, ενσωματωμένο debugger και αναλυτή κώδικα για τον εντοπισμό σφαλμάτων. Για την ανάπτυξη RAG chatbots, το PyCharm παρέχει ισχυρή υποστήριξη για frameworks όπως το LangChain και Streamlit [27].



Εικόνα 7 : GUI PyCharm

2.5 Σύνοψη 2ου Κεφαλαίου

Συνοψίζοντας, στο δεύτερο κεφάλαιο, αναλύθηκαν οι θεμελιώδεις έννοιες της Τεχνητής Νοημοσύνης και οι κατηγορίες των chatbots, με εστίαση στην υπεροχή της αρχιτεκτονικής Retrieval-Augmented Generation (RAG) για την παροχή τεκμηριωμένων απαντήσεων. Επιπροσθέτως, περιγράφηκαν λεπτομερώς τα τεχνολογικά εργαλεία που συνθέτουν το σύστημα, όπως η γλώσσα Python, το framework LangChain, το μοντέλο DeepSeek, καθώς και οι βάσεις δεδομένων FAISS και SQLite. Στο επόμενο κεφάλαιο ακολουθεί ο σχεδιασμός του συστήματος, όπου καθορίζονται οι λειτουργικές προδιαγραφές και αναλύονται οι περιπτώσεις χρήσης της εφαρμογής.

3. Σχεδιασμός συστήματος

3.1 Προδιαγραφές

Οι προδιαγραφές περιγράφουν τις λειτουργίες και τα χαρακτηριστικά ποιότητας που πρέπει να διαθέτει το σύστημα ώστε να ανταποκρίνεται στις ανάγκες του ξενοδοχειακού κλάδου και των τελικών χρηστών. Οι απαιτήσεις διακρίνονται σε λειτουργικές, που καθορίζουν το "τι κάνει" το σύστημα, και σε μη λειτουργικές, που ορίζουν το "πόσο καλά" εκτελεί αυτές τις λειτουργίες.

Μη Λειτουργικές Προδιαγραφές (Non Functional Requirements- NFR)

NFR-1 **Ακρίβεια απαντήσεων**

a. Το chatbot πρέπει να επιτυγχάνει ποσοστό ακρίβειας απαντήσεων ≥ 80%, όπως αξιολογείται με βάση γνωστές σωστές απαντήσεις. Η αξιολόγηση μπορεί να γίνεται περιοδικά, ώστε να παρακολουθείται η απόδοση του συστήματος.

NFR-2 **Αξιοπιστία και διαθεσιμότητα**

a. Το σύστημα πρέπει να λειτουργεί αδιάλειπτα, παρέχοντας συνεχή εξυπηρέτηση 24/7 χωρίς απώλειες δεδομένων ή σφάλματα κατά τη διάρκεια των συνεδριών.

NFR-3 **Αποδοτικότητα**

a. Ο χρόνος απόκρισης δεν πρέπει να υπερβαίνει κατά μέσο όρο τα 15 δευτερόλεπτα ανά ερώτηση υπό κανονικές συνθήκες φόρτου.

NFR-4 **Ευχρηστία**

a. Η σχεδίαση της διεπαφής πρέπει να επιτρέπει εύκολη πλοήγηση και σαφή απεικόνιση της συνομιλίας.

NFR-5 **Επεκτασιμότητα**

a. Το σύστημα πρέπει να μπορεί να επεκταθεί σε πολλαπλά ξενοδοχεία ή αλυσίδες χωρίς επανασχεδίαση, απλώς με εισαγωγή νέων δεδομένων στη βάση γνώσης.

NFR-6 **Ασφάλεια και προστασία δεδομένων**

a. Οι συνομιλίες πρέπει να αποθηκεύονται με ασφάλεια με σεβασμό στα προσωπικά δεδομένα των χρηστών.

NFR-7 **Παρακολούθηση και ανάλυση λειτουργίας συστήματος**

a. Όλες οι συνομιλίες αποθηκεύονται αυτόματα σε αρχείο καταγραφής (logs), περιλαμβάνοντας ημερομηνία, ώρα, μήνυμα χρήστη, απάντηση συστήματος και

αξιολόγηση feedback. Σκοπός είναι η ανάλυση, η αποσφαλμάτωση και η παρακολούθηση της ποιότητας των απαντήσεων.

NFR-8 **Μάθηση και προσαρμοστικότητα**

a. Οι συνομιλίες που έχουν χαρακτηριστεί ως "Καλές" από τους χρήστες χρησιμοποιούνται σε διαδικασία ανατροφοδότησης για τη βελτίωση της ακρίβειας των απαντήσεων. Η διαδικασία εκτελείται περιοδικά και δεν επηρεάζει τη λειτουργία του chatbot σε πραγματικό χρόνο.

Λειτουργικές προδιαγραφές (Functional Requirements-FR)

FR-1 Το chatbot πρέπει να μπορεί να απαντά σε ερωτήσεις στα Αγγλικά και στα Ελληνικά.

FR-2 Πρέπει να παρέχει πληροφορίες για το ξενοδοχείο, τις παροχές, FAQs και τις πολιτικές του στηριζόμενο αποκλειστικά σε εσωτερικά δεδομένα μέσω της τεχνικής RAG.

FR-3 Να μπορεί να απαντά σε ερωτήσεις σχετικά με κοντινά σημεία ενδιαφέροντος μέσω Vector Similarity search.

FR-4 Αν μια ερώτηση δεν σχετίζεται με το ξενοδοχείο, το chatbot πρέπει να αρνείται ευγενικά να απαντήσει, αποφεύγοντας την παραγωγή εσφαλμένων πληροφοριών (hallucinations).

FR-5 Το chatbot πρέπει να διαθέτει μνήμη και θυμάται τα προηγούμενα μηνύματα στη συζήτηση για να παρέχει πιο σχετικές απαντήσεις.

FR-6 Να μπορεί να απαντάει σε ερωτήσεις σχετικά με τη διαθεσιμότητα και τις πληροφορίες των διαθέσιμων δωματίων κάνοντας αναζήτηση σε σχεσιακή βάση δεδομένων με SQL queries.

FR-7 Να μπορεί να ενημερώνει εύκολα τη βάση γνώσης μέσω ξεχωριστών έτοιμων Python scripts.

FR-8 Το σύστημα πρέπει να επιτρέπει την ενσωμάτωσή του σε υπάρχουσες ιστοσελίδες ξενοδοχείων, είτε μέσω iframe, είτε μέσω API.

FR-9 Να μπορεί να εντοπίζει και να επιστρέφει κριτικές που υπάρχουν στη βάση σχετικές με οποιοδήποτε θέμα ρωτήσει ο χρήστης μέσω Vector Similarity search.

FR-10 Μετά από κάθε απάντηση, εμφανίζονται δύο κουμπιά αξιολόγησης απάντησης, “Καλή” / “ Κακή”. Η επιλογή του χρήστη αποθηκεύεται στη βάση δεδομένων ως δείκτης ικανοποίησης, για μελλοντική ανάλυση και βελτίωση του συστήματος.

3.2 Περιπτώσεις χρήσης (Use Cases-UC)

UC-1: *Πολυγλωσσική επικοινωνία*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Ο τρόπος με τον οποίο το chatbot επικοινωνεί με τον χρήστη στα Αγγλικά ή στα Ελληνικά, δίνοντάς του τη δυνατότητα να επιλέξει γλώσσα επικοινωνίας από το αρχικό μενού.

Προϋποθέσεις: Το chatbot πρέπει να διαθέτει μενού επιλογής γλώσσας (Ελληνικά / Αγγλικά). Ο χρήστης πρέπει να έχει ξεκινήσει συνομιλία με το chatbot.

Βασική ροή:

i. Ο χρήστης ανοίγει το chatbot.
ii. Εμφανίζεται μενού επιλογής γλώσσας με επιλογές “Ελληνικά” και “English”.
iii. Ο χρήστης επιλέγει τη γλώσσα που προτιμά.
iv. Το chatbot ρυθμίζει τη γλώσσα επικοινωνίας στη συγκεκριμένη γλώσσα.
v. Ο χρήστης ξεκινά τη συνομιλία και το chatbot απαντά στη γλώσσα που επιλέχθηκε.
vi. Η συνομιλία συνεχίζεται στη γλώσσα που έχει οριστεί, μέχρι να γίνει νέα επιλογή από το μενού.

Εναλλακτικό σενάριο:

Αν ο χρήστης δεν επιλέξει γλώσσα, το chatbot ξεκινά τη συνομιλία στα Αγγλικά ως προεπιλεγμένη γλώσσα.

UC-2: *Παροχή πληροφοριών για το Ξενοδοχείο μέσω RAG*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Το chatbot παρέχει πληροφορίες σχετικά με το ξενοδοχείο, τις παροχές, τις πολιτικές και τις συχνές ερωτήσεις (FAQs), αντλώντας δεδομένα αποκλειστικά από εσωτερικές πηγές γνώσης μέσω της τεχνικής RAG (Retrieval-Augmented Generation).

Προϋποθέσεις: Η βάση δεδομένων γνώσης πρέπει να περιέχει ενημερωμένο περιεχόμενο για τις παροχές, πολιτικές και πληροφορίες του ξενοδοχείου. Το chatbot πρέπει να είναι συνδεδεμένο με το RAG pipeline.

Βασική ροή:

i. Ο χρήστης υποβάλλει ερώτηση σχετική με το ξενοδοχείο, π.χ. “Τι ώρα είναι το check-in;”.
ii. Το chatbot αναλύει το ερώτημα και δημιουργεί embedding του κειμένου.
iii. Εκτελείται αναζήτηση στις εσωτερικές πηγές δεδομένων (Vector Store).
iv. Ανακτώνται τα πιο σχετικά έγγραφα ή αποσπάσματα.
v. Το LLM συνθέτει απάντηση στηριζόμενο αποκλειστικά στο ανακτηθέν περιεχόμενο.
vi. Η απάντηση εμφανίζεται στον χρήστη.

Εναλλακτικό σενάριο: Αν δεν εντοπιστεί σχετικό περιεχόμενο στη βάση γνώσης, το chatbot απαντά ευγενικά: “Δυστυχώς δεν έχω πληροφορίες γι’ αυτό το θέμα. Μπορώ να σας βοηθήσω με κάτι σχετικό με το ξενοδοχείο;”

UC-3: *Αναζήτηση διαθεσιμότητας, τιμών και παροχών δωματίων*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Το chatbot απαντά σε ερωτήσεις που αφορούν τη διαθεσιμότητα δωματίων, τις τιμές και τις παροχές τους, πραγματοποιώντας αναζήτηση στη σχεσιακή βάση δεδομένων του ξενοδοχείου.

Προϋποθέσεις: Η βάση δεδομένων πρέπει να είναι ενημερωμένη με τα διαθέσιμα δωμάτια, τις ημερομηνίες και τις τιμές. Το chatbot πρέπει να έχει πρόσβαση σε αυτή μέσω SQL queries.

Βασική ροή:

i. Ο χρήστης ρωτά για διαθεσιμότητα, π.χ. “Υπάρχουν δωμάτια για 10–12 Αυγούστου;”.

ii. Το chatbot αναγνωρίζει τις ημερομηνίες και τον τύπο αιτήματος.
iii. Δημιουργεί και εκτελεί ερώτημα στη σχεσιακή βάση δεδομένων.
iv. Το σύστημα επιστρέφει τα διαθέσιμα δωμάτια και τις τιμές.
v. Το chatbot διαμορφώνει την απάντηση και την παρουσιάζει στον χρήστη.

Εναλλακτικό σενάριο: Αν δεν υπάρχουν διαθέσιμα δωμάτια για τις επιλεγμένες ημερομηνίες, το chatbot απαντά: "Δυστυχώς δεν υπάρχουν διαθέσιμα δωμάτια για τις ημερομηνίες που ζητήσατε. Θα θέλατε να σας προτείνω εναλλακτικές ημερομηνίες;"

UC-4: *Μνήμη συνομιλίας*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Ο τρόπος με τον οποίο το chatbot θυμάται τις προηγούμενες ερωτήσεις και απαντήσεις στη διάρκεια μιας συνομιλίας, ώστε να παρέχει πιο σχετικές και συνεκτικές απαντήσεις.

Προϋποθέσεις: Το chatbot πρέπει να διαθέτει μηχανισμό προσωρινής μνήμης για την τρέχουσα συνομιλία και κατάλληλη λογική για να συνδυάζει τα προηγούμενα συμφραζόμενα (context).

Βασική ροή:

i. Ο χρήστης ξεκινά συνομιλία με το chatbot.
ii. Ο χρήστης κάνει μια ερώτηση (π.χ. "Ποια είναι η τιμή μιας σουίτας;").
iii. Το chatbot απαντά και αποθηκεύει την ερώτηση και την απάντηση στη μνήμη της συνομιλίας.
iv. Ο χρήστης υποβάλλει νέα ερώτηση (π.χ. "Έχει θέα στη θάλασσα;").
v. Το chatbot αναγνωρίζει ότι η δεύτερη ερώτηση σχετίζεται με την προηγούμενη και απαντά ανάλογα, χωρίς να χρειάζεται ο χρήστης να επαναλάβει το πλαίσιο.

Εναλλακτικό σενάριο: Αν η συνομιλία έχει λήξει, η μνήμη έχει διαγραφεί ή υπήρξε κάποιο σφάλμα, το chatbot ενημερώνει τον χρήστη να επαναδιατυπώσει το ερώτημα.

UC-5: *Αναζήτηση κοντινών σημείων ενδιαφέροντος*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Το chatbot παρέχει πληροφορίες για κοντινά αξιοθέατα, εστιατόρια, παραλίες ή δραστηριότητες, εκτελώντας αναζήτηση σε βάση δεδομένων μέσω Vector Similarity Search.

Προϋποθέσεις: Η βάση δεδομένων πρέπει να περιέχει πληροφορίες για σημεία ενδιαφέροντος (POIs). Το chatbot πρέπει να έχει ενσωματωμένο μηχανισμό υπολογισμού ομοιότητας (vector embeddings).

Βασική ροή:

i. Ο χρήστης ρωτά π.χ. “Τι μπορώ να επισκεφτώ κοντά στο ξενοδοχείο;”.
ii. Το chatbot μετατρέπει το ερώτημα σε διανυσματική αναπαράσταση (embedding).
iii. Εκτελείται Vector Similarity Search στη βάση με τα σημεία ενδιαφέροντος.
iv. Επιστρέφονται τα πιο σχετικά αποτελέσματα (π.χ. Μουσείο, παραλία, ταβέρνα).
v. Το chatbot παρουσιάζει τη λίστα με σύντομες περιγραφές.

Εναλλακτικό σενάριο: Αν δεν βρεθούν κοντινά σημεία, το chatbot προτείνει δημοφιλή αξιοθέατα της περιοχής γενικά.

UC-6: *Αναζήτηση κριτικών πελατών*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Το chatbot μπορεί να εντοπίζει και να επιστρέφει κριτικές επισκεπτών από τη βάση δεδομένων, σχετικές με ένα θέμα που ρωτά ο χρήστης (π.χ. καθαριότητα, φαγητό, εξυπηρέτηση).

Προϋποθέσεις: Η βάση δεδομένων πρέπει να περιέχει κριτικές και το chatbot πρέπει να έχει ενσωματωμένο μηχανισμό υπολογισμού ομοιότητας (vector embeddings).

Βασική ροή:

i. Ο χρήστης ρωτά π.χ. “Τι λένε οι πελάτες για το πρωινό;”.
ii. Το chatbot υπολογίζει το embedding του ερωτήματος.
iii. Εκτελείται αναζήτηση ομοιότητας στις κριτικές.

iv. Επιλέγονται οι πιο σχετικές αξιολογήσεις.
v. Το chatbot συνοψίζει ή εμφανίζει απευθείας τα αποσπάσματα των κριτικών στον χρήστη.

Εναλλακτικό σενάριο: Αν δεν υπάρχουν σχετικές κριτικές, το chatbot ενημερώνει τον χρήστη ότι δεν υπάρχουν διαθέσιμα σχόλια για το συγκεκριμένο θέμα.

UC-7: *Ευγενική άρνηση σε άσχετες ερωτήσεις*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Ο τρόπος με τον οποίο το chatbot αναγνωρίζει ότι μια ερώτηση δεν σχετίζεται με το αντικείμενο (ξενοδοχείο) και απαντά ευγενικά, αποφεύγοντας παραισθήσεις "hallucinations".

Προϋποθέσεις: Το chatbot πρέπει να διαθέτει κατάλληλο prompting ελέγχου συνάφειας και κανόνα απόρριψης εκτός θέματος ερωτήσεων.

Βασική ροή:

i. Ο χρήστης ρωτά ερώτηση άσχετη, π.χ. "Πόσο κοστίζει το καινούργιο iphone;".
ii. Το chatbot αναγνωρίζει ότι η ερώτηση δεν σχετίζεται με το ξενοδοχείο.
iii. Ανιχνεύεται εκτός πεδίου θέμα.
iv. Το chatbot απαντά ευγενικά: "Συγγνώμη, μπορώ να απαντήσω μόνο σε ερωτήσεις που αφορούν το ξενοδοχείο και τις υπηρεσίες του."

Εναλλακτικό σενάριο: Αν η ερώτηση είναι μερικώς σχετική (π.χ. "Υπάρχουν κοντινά μαγαζιά για αγορά κινητών;"), το chatbot προσπαθεί να απαντήσει στο σχετικό μέρος (π.χ. "Υπάρχουν καταστήματα ηλεκτρονικών ειδών σε απόσταση 1km.").

UC-8: *Αξιολόγηση απάντησης (Feedback)*

Χειριστής: Χρήστης (επισκέπτης ξενοδοχείου)

Περιγραφή: Ο χρήστης έχει τη δυνατότητα να αξιολογεί κάθε απάντηση του chatbot μέσω των κουμπιών "Καλή ή "Κακή".

Προϋποθέσεις: Το chatbot πρέπει να διαθέτει μηχανισμό αποθήκευσης αξιολογήσεων σε βάση δεδομένων.

Βασική ροή:

i. Μετά από κάθε απάντηση, εμφανίζονται τα κουμπιά “Καλή ή “Κακή”.
ii. Ο χρήστης επιλέγει ένα από τα δύο.
iii. Το chatbot καταγράφει την αξιολόγηση μαζί με το περιεχόμενο της απάντησης.
iv. Το feedback αποθηκεύεται στο αρχείο καταγραφής (βάση δεδομένων) για μελλοντική ανάλυση.

Εναλλακτικό σενάριο: Αν ο χρήστης δεν πατήσει κάποιο κουμπί, η συνομιλία συνεχίζεται χωρίς να καταγραφεί feedback.

3.3 Σύνοψη 3ου Κεφαλαίου

Συνοψίζοντας, στο τρίτο κεφάλαιο, τέθηκαν οι βάσεις για την ανάπτυξη της εφαρμογής μέσω του καθορισμού των λειτουργικών και μη λειτουργικών προδιαγραφών, διασφαλίζοντας την ακρίβεια, την αποδοτικότητα και την ευχρηστία του chatbot. Παράλληλα, αναλύθηκαν διεξοδικά οι περιπτώσεις χρήσης (Use Cases), οι οποίες καλύπτουν σενάρια όπως η πολυγλωσσική επικοινωνία, η αναζήτηση διαθεσιμότητας και η ανάκτηση πληροφοριών μέσω RAG. Στο επόμενο κεφάλαιο περιγράφεται η τεχνική υλοποίηση του συστήματος, η αρχιτεκτονική των δεδομένων και η ανάπτυξη της λογικής του πράκτορα.

4. Υλοποίηση συστήματος

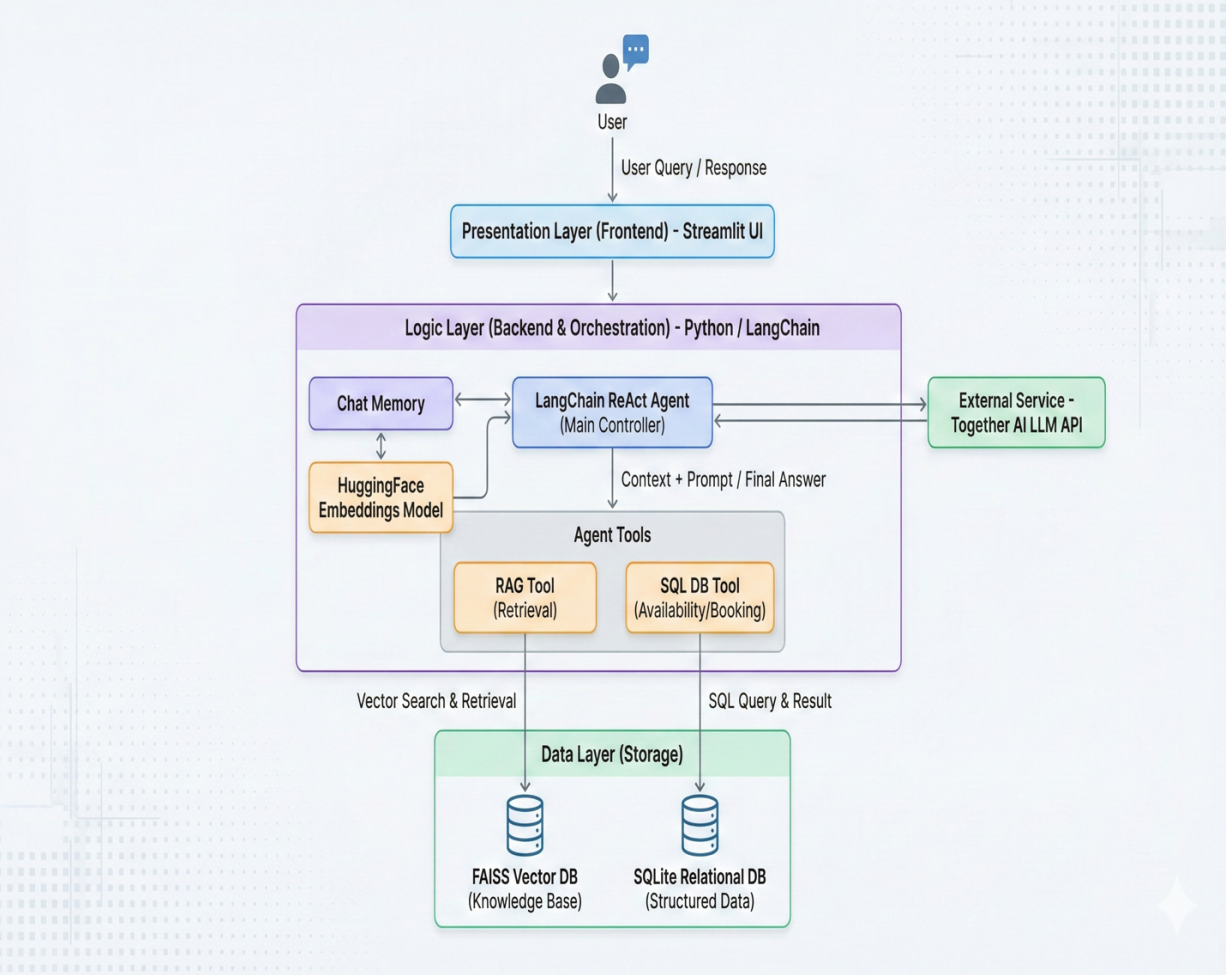
4.1 Αρχιτεκτονική Συστήματος

Η αρχιτεκτονική του συστήματος σχεδιάστηκε με βασικούς στόχους την επεκτασιμότητα, την αποδοτική ανάκτηση πληροφορίας και τον περιορισμό των σφαλμάτων τύπου hallucinations που παρατηρούνται στα Μεγάλα Γλωσσικά Μοντέλα. Για τον λόγο αυτό, το σύστημα υλοποιείται ως διαδικτυακή εφαρμογή (Web Application) και βασίζεται στο πρότυπο RAG, το οποίο συνδυάζει τη γενετική ικανότητα των LLMs με ανάκτηση πληροφορίας από εξωτερικές, ελεγχόμενες πηγές δεδομένων.

Η επιλογή του RAG είναι κρίσιμη, καθώς επιτρέπει στο σύστημα να παράγει απαντήσεις που στηρίζονται σε πραγματικά δεδομένα και όχι αποκλειστικά στη στατιστική γνώση του προεκπαιδευμένου μοντέλου, μειώνοντας έτσι σημαντικά την πιθανότητα ανακριβών ή κατασκευασμένων απαντήσεων. Παράλληλα, η αρχιτεκτονική ενσωματώνει τη λογική των Αυτόνομων Πρακτόρων (Autonomous Agents)[17], επιτρέποντας δυναμική λήψη αποφάσεων ως προς τον τρόπο και την πηγή ανάκτησης της πληροφορίας.

Η συνολική δομή του συστήματος ακολουθεί αρχιτεκτονική πολλαπλών επιπέδων, η οποία διαχωρίζει σαφώς τις αρμοδιότητες και διευκολύνει τη συντήρηση και μελλοντική επέκταση του συστήματος. Συγκεκριμένα, διακρίνονται τρία επίπεδα:

1. **Επίπεδο Παρουσίασης (Presentation Layer)**
2. **Επίπεδο Λογικής Εφαρμογής (Application Logic Layer)**
3. **Επίπεδο Δεδομένων (Data Layer)**



Εικόνα 8 : Διάγραμμα αρχιτεκτονικής

Επίπεδο Παρουσίασης (Presentation Layer)

Το επίπεδο παρουσίασης υλοποιήθηκε με τη βιβλιοθήκη Streamlit, επιλογή που έγινε λόγω της άμεσης διασύνδεσής της με Python-based backend εφαρμογές και της ενσωματωμένης υποστήριξης για διαχείριση κατάστασης (session state). Η συγκεκριμένη τεχνολογία κρίνεται κατάλληλη για εφαρμογές που βασίζονται σε LLMs, καθώς επιτρέπει γρήγορη ανάπτυξη διεπαφών χωρίς την ανάγκη σύνθετης αρχιτεκτονικής frontend.

Το επίπεδο αυτό αποτελεί το μοναδικό σημείο αλληλεπίδρασης του χρήστη με το σύστημα και είναι υπεύθυνο για:

- τη λήψη ερωτημάτων σε φυσική γλώσσα (Ελληνικά ή Αγγλικά),
- τη διατήρηση του ιστορικού της συνομιλίας σε επίπεδο συνεδρίας,

- την παροχή μηχανισμού ανατροφοδότησης (feedback), μέσω του οποίου ο χρήστης αξιολογεί την ποιότητα της απάντησης.

Η συλλογή feedback σε αυτό το στάδιο είναι απαραίτητη για τη μελλοντική ανάλυση της απόδοσης του συστήματος και τη βελτίωση των παραμέτρων ανάκτησης και παραγωγής απαντήσεων.

Επίπεδο Λογικής Εφαρμογής (Application Logic Layer)

Στον πυρήνα της αρχιτεκτονικής βρίσκεται ένας ReAct Agent (Reasoning and Acting) [17][28], υλοποιημένος μέσω του framework LangChain. Ο Agent λειτουργεί ως το κεντρικό στοιχείο λήψης αποφάσεων και διαμεσολαβεί μεταξύ του χρήστη, του γλωσσικού μοντέλου και των διαθέσιμων πηγών δεδομένων. Η λογική ReAct διαχωρίζει ρητά τον συλλογισμό από την εκτέλεση ενεργειών, γεγονός που επιτρέπει:

- την επιλογή της κατάλληλης πηγής πληροφορίας,
- την αποφυγή άσκοπων ή λανθασμένων αναζητήσεων
- και τη συνδυαστική χρήση δομημένων και αδόμητων δεδομένων.

Η ροή εκτέλεσης στο επίπεδο αυτό ακολουθεί τα παρακάτω στάδια:

1. **Reasoning (Συλλογισμός)**: Ο Agent λαμβάνει το ερώτημα του χρήστη και αξιοποιώντας το γλωσσικό μοντέλο DeepSeek, αποφασίζει τη βέλτιστη στρατηγική για την απάντηση.
2. **Tool Selection (Επιλογή Εργαλείου)**: Ο Agent επιλέγει δυναμικά το κατάλληλο εργαλείο, είτε για αναζήτηση στη διανυσματική βάση (π.χ. Συχνές Ερωτήσεις, κριτικές χρηστών, σημεία ενδιαφέροντος), είτε για πρόσβαση στη σχεσιακή βάση δεδομένων (π.χ. έλεγχος διαθεσιμότητας δωματίων).
3. **Acting (Ενέργεια)**: Πραγματοποιείται η εκτέλεση της επιλεγμένης ενέργειας και η ανάκτηση των σχετικών δεδομένων.
4. **Generation (Παραγωγή Απάντησης)**: Τα ανακτηθέντα δεδομένα ενσωματώνονται στο γλωσσικό μοντέλο, το οποίο συνθέτει την τελική απάντηση, προσαρμοσμένη τόσο στο ύφος της εφαρμογής όσο και στη γλώσσα του χρήστη.

Εικόνα 9 : Διάγραμμα ροής ReAct agent

Επίπεδο δεδομένων (Data Layer)

Το επίπεδο δεδομένων βασίζεται σε υβριδική προσέγγιση αποθήκευσης και διαχείρισης πληροφορίας, συνδυάζοντας δομημένα και αδόμητα δεδομένα.

1. **Διανυσματική Βάση Δεδομένων (Vector Store – FAISS)**

Για τη διαχείριση αδόμητης πληροφορίας χρησιμοποιείται η βιβλιοθήκη FAISS (Facebook AI Similarity Search). Η επιλογή της συγκεκριμένης βιβλιοθήκης κρίθηκε απαραίτητη για την υποστήριξη ταχείας και αποδοτικής σημασιολογικής αναζήτησης (semantic search) σε πραγματικό χρόνο. Το FAISS επιτρέπει τον υπολογισμό της εγγύτητας μεταξύ διανυσμάτων (embeddings) με υψηλή ταχύτητα, διασφαλίζοντας χαμηλούς χρόνους απόκρισης ακόμη και σε συστήματα με περιορισμένους υπολογιστικούς πόρους [19]. Η προσέγγιση αυτή επιτρέπει τη σημασιολογική αναζήτηση (semantic search), δηλαδή την ανάκτηση πληροφορίας με βάση το νόημα του ερωτήματος και όχι την απλή αντιστοίχιση λέξεων-κλειδιών.

Τα δεδομένα (Συχνές Ερωτήσεις, Κριτικές Πελατών, Σημεία Ενδιαφέροντος) μετατρέπονται σε διανύσματα (embeddings) μέσω του μοντέλου *all-MiniLM-L6-v2* της HuggingFace. Επιλέχθηκε το συγκεκριμένο μοντέλο, καθώς προσφέρει καλή ισορροπία μεταξύ υπολογιστικού κόστους, ταχύτητας και ακρίβειας στη σημασιολογική αναπαράσταση των προτάσεων [29].

2. **Σχεσιακή Βάση Δεδομένων (SQLite)**

Για τα δομημένα δεδομένα χρησιμοποιείται η βάση SQLite, η οποία είναι κατάλληλη για ερωτήματα που απαιτούν ακρίβεια και ακεραιότητα. Η βάση περιλαμβάνει:

- Τους πίνακα *hotel_rooms, bookings* για τον έλεγχο διαθεσιμότητας, τιμών καθώς και τον τύπο δωματίων.
- Τους πίνακες *conversation_session* και *conversation_turns*, για την καταγραφή των συνομιλιών και του feedback των χρηστών.

Ο διαχωρισμός δομημένων και αδόμητων δεδομένων είναι κρίσιμος, καθώς αποφεύγει τη λανθασμένη χρήση σημασιολογικής αναζήτησης σε δεδομένα που απαιτούν ακριβείς υπολογισμούς ή φίλτρα.

Η συγκεκριμένη αρχιτεκτονική εξασφαλίζει ότι το chatbot σχεδόν εξαλείφει τις παραισθήσεις (hallucinations) των LLM, καθώς κάθε απάντηση στηρίζεται σε ανακτημένα, πραγματικά δεδομένα από τα παραπάνω υποσυστήματα.

4.2 Προετοιμασία και Επεξεργασία Δεδομένων

Η ποιότητα και η δομή των δεδομένων αποτελούν κρίσιμους παράγοντες για την απόδοση ενός συστήματος RAG. Πριν από την εισαγωγή τους στη διανυσματική βάση, τα δεδομένα απαιτείται να συλλεχθούν, να καθαριστούν και να μορφοποιηθούν κατάλληλα, ώστε να είναι ανακτήσιμα με ακρίβεια από τον αλγόριθμο αναζήτησης σημασιολογικής ομοιότητας. Στην συγκεριμένη περίπτωση τα δεδομένα ήταν ήδη στη σωστή μορφή καθώς δημιουργήθηκαν μέσω prompt στο LLM, οπότε δεν χρειάστηκε καθαρισμός και μορφοποίηση.

Για τις ανάγκες του παρόντος chatbot χρησιμοποιήθηκαν τρία κύρια σύνολα δεδομένων (datasets), αποθηκευμένα σε μορφή CSV (Comma-Separated Values):

1. **Συχνές Ερωτήσεις (FAQ)**: περιέχει ζεύγη ερωτήσεων–απαντήσεων σχετικών με τις υπηρεσίες του ξενοδοχείου.
2. **Κριτικές Πελατών (Reviews)**: περιλαμβάνει κείμενα σχολίων επισκεπτών και τις αντίστοιχες βαθμολογίες.

3. **Σημεία Ενδιαφέροντος (Points of Interest – POIs)**: περιέχει πληροφορίες για αξιοθέατα και δραστηριότητες στην ευρύτερη περιοχή.

Η μορφή CSV επιλέχθηκε έναντι πιο σύνθετων δομών δεδομένων, όπως JSON ή XML, λόγω της απλότητάς της και της άμεσης συμβατότητάς της με βιβλιοθήκες ανάλυσης δεδομένων, όπως η Pandas.

4.2.1 Επεξεργασία των Datasets

Η διαδικασία επεξεργασίας των δεδομένων υλοποιήθηκε με τη χρήση της βιβλιοθήκης Pandas της Python, η οποία χρησιμοποιήθηκε ως βασικό εργαλείο ανάγνωσης και προεπεξεργασίας των αρχείων CSV.

Η Pandas επιλέχθηκε λόγω της υψηλής αποδοτικότητάς της στη διαχείριση δεδομένων. Παράλληλα, επιτρέπει εύκολους, συχνούς μετασχηματισμούς, όπως φιλτράρισμα δεδομένων, συνένωση πεδίων και εξαγωγή λιστών, με ελάχιστες γραμμές κώδικα. Η επιλογή αυτή συνέβαλε στη μείωση της πολυπλοκότητας της προεπεξεργασίας και στη διατήρηση καθαρού και επεκτάσιμου κώδικα. Τέλος, η επεξεργασία πραγματοποιήθηκε ξεχωριστά για κάθε αρχείου, λαμβάνοντας υπόψη τη σημασιολογική φύση και τον ρόλο των πληροφοριών σε κάθε σύνολο δεδομένων.

Επεξεργασία Συχνών Ερωτήσεων (FAQ Dataset)

Το αρχείο *faq.csv* φορτώθηκε, και ως κύριο κείμενο αναζήτησης ορίστηκε το πεδίο *question*. Η διανυσματοποίηση της ερώτησης, αντί της απάντησης, επιλέχθηκε επειδή η ανάκτηση σε συστήματα RAG βασίζεται στη σημασιολογική ομοιότητα. Ένα ερώτημα χρήστη είναι σαφώς πιθανότερο να παρουσιάζει εννοιολογική εγγύτητα με μια ήδη αποθηκευμένη ερώτηση παρά με το κείμενο της απάντησης. Η προσέγγιση αυτή βελτιώνει την ακρίβεια της ανάκτησης και μειώνει τα άσχετα αποτελέσματα.

```
# Ανάγνωση του αρχείου FAQ
faq_df = pd.read_csv(FAQ_CSV_PATH)

# Χρήση της ερώτησης ως κύριο έγγραφο για embedding
documents = faq_df['question'].tolist()

# Αποθήκευση όλων των πεδίων ως μεταδεδομένα
metadatas = faq_df.to_dict('records')
```

Εικόνα 10 : Κώδικας από setup_vector_stores.py

question	answer
What are the check-in and check-out times?	Check-in is from 3:00 PM, and check-out is until 11:00 AM.
Do you have Wi-Fi?	Yes, we offer complimentary high-speed Wi-Fi for all our guests.
Is breakfast included?	Yes, a complimentary continental breakfast is included with your stay.
Do you have a pool?	Yes, we have a heated indoor pool on the 5th floor.
Is there a gym?	Yes, our fitness center is open 24/7 and is located on the 5th floor.
Do you allow pets?	We are a pet-friendly hotel. A fee of 50 euros per pet, per night applies.
What is your cancellation policy?	You can cancel for free up to 24 hours before your check-in date. After that, a fee equivalent to one night's stay will be charged.
Where the hotel is located?	Hotel address is Daskalogianni 23, near the city center.

Εικόνα 11 : Αρχείο faq.csv

Επεξεργασία Κριτικών Πελατών (Reviews Dataset)

Στο σύνολο δεδομένων των κριτικών, το κύριο ενδιαφέρον επικεντρώνεται στο περιεχόμενο του σχολίου, το οποίο αποτυπώνεται στο πεδίο *review_text*. Ο στόχος εδώ είναι η θεματική ανάκτηση πληροφορίας. Για παράδειγμα, σε ερώτημα όπως "Πώς είναι το πρωινό;", το σύστημα πρέπει να εντοπίσει κριτικές που περιγράφουν σχετικές εμπειρίες. Για τον λόγο αυτό, διανυσματοποιείται το κυρίως κείμενο της κριτικής, καθώς περιέχει τη μεγαλύτερη σημασιολογική πληροφορία.

Στιγμιότυπο κώδικα από *setup_vector_stores.py*

```
# Ανάγνωση του αρχείου Κριτικών
reviews_df = pd.read_csv(REVIEWS_CSV_PATH)

# Χρήση του κειμένου της κριτικής για embedding
documents = reviews_df['review_text'].tolist()

# Τα μεταδεδομένα περιλαμβάνουν βαθμολογία, ημερομηνία κ.ά.
metadatas = reviews_df.to_dict('records')
```

Εικόνα 12 : Κώδικας από setup_vector_stores.py

patient id	name	hotel_name	country of origin	review_text	rating	nights_of_stay
1	James	The Grand Resort Hotel	USA	The staff was incredibly friendly and helpful. The breakfast buffet had a wide variety of delicious options.	9.5	5
2	Mary	The Grand Resort Hotel	UK	The room was spacious, clean, and had a beautiful view. The Wi-Fi in the room was a bit slow at times.	8.2	3
3	John	The Grand Resort Hotel	Canada	The pool area was stunning and very relaxing. The bed was one of the most comfortable I've ever slept in. The check-in process was a little slow.	7.8	7
4	Patricia	The Grand Resort Hotel	Australia	The location is perfect, right on the beach.	9.8	2
5	Robert	The Grand Resort Hotel	Germany	The concierge gave us some excellent recommendations for local restaurants. A truly luxurious experience from start to finish.	9.9	10
6	Jennifer	The Grand Resort Hotel	France	The spa services were top-notch. The prices at the hotel bar were quite high.	7.5	4
7	Michael	The Grand Resort Hotel	Spain	We will definitely be coming back here on our next trip. The air conditioning was a little noisy.	8.1	6
8	Linda	The Grand Resort Hotel	Italy	The staff was incredibly friendly and helpful. The room was spacious, clean, and had a beautiful view. The walls were a bit thin, and we could hear our neighbors.	7.2	1
9	William	The Grand Resort Hotel	Japan	The breakfast buffet had a wide variety of delicious options. The pool area was stunning and very relaxing.	9.6	9
10	Elizabeth	The Grand Resort Hotel	China	The location is perfect, right on the beach. The minibar was not restocked daily.	8.0	3
11	James	The Grand Resort Hotel	USA	The bed was one of the most comfortable I've ever slept in. The gym was a bit small and crowded.	7.9	8
12	Mary	The Grand Resort Hotel	UK	The spa services were top-notch. A truly luxurious experience from start to finish.	9.7	2
13	John	The Grand Resort Hotel	Canada	The staff was incredibly friendly and helpful. The elevators were often busy.	8.4	11
14	Patricia	The Grand Resort Hotel	Australia	The room was spacious, clean, and had a beautiful view. The TV in the room didn't have many channels.	7.6	1
15	Robert	The Grand Resort Hotel	Germany	The breakfast buffet had a wide variety of delicious options. The pool area was stunning and very relaxing. The prices at the hotel bar were quite high.	8.8	12
16	Jennifer	The Grand Resort Hotel	France	The location is perfect, right on the beach. The check-in process was a little slow.	8.3	5
17	Michael	The Grand Resort Hotel	Spain	The concierge gave us some excellent recommendations for local restaurants. The air conditioning was a little noisy.	7.7	7
18	Linda	The Grand Resort Hotel	Italy	The bed was one of the most comfortable I've ever slept in. The walls were a bit thin, and we could hear our neighbors.	7.1	3
19	William	The Grand Resort Hotel	Japan	The spa services were top-notch. The minibar was not restocked daily.	8.5	10
20	Elizabeth	The Grand Resort Hotel	China	A truly luxurious experience from start to finish. The gym was a bit small and crowded.	8.2	4
21	James	The Grand Resort Hotel	USA	The staff was incredibly friendly and helpful. The elevators were often busy.	8.6	6
22	Mary	The Grand Resort Hotel	UK	The room was spacious, clean, and had a beautiful view. The TV in the room didn't have many channels.	7.8	1
23	John	The Grand Resort Hotel	Canada	The breakfast buffet had a wide variety of delicious options. The prices at the hotel bar were quite high.	8.9	13
24	Patricia	The Grand Resort Hotel	Australia	The pool area was stunning and very relaxing. The check-in process was a little slow.	8.7	2
25	Robert	The Grand Resort Hotel	Germany	The location is perfect, right on the beach. The air conditioning was a little noisy.	8.0	14
26	Jennifer	The Grand Resort Hotel	France	The concierge gave us some excellent recommendations for local restaurants. The walls were a bit thin, and we could hear our neighbors.	7.3	6
27	Michael	The Grand Resort Hotel	Spain	The bed was one of the most comfortable I've ever slept in. The minibar was not restocked daily.	8.1	8
28	Linda	The Grand Resort Hotel	Italy	The spa services were top-notch. The gym was a bit small and crowded.	7.9	4

Εικόνα 13 : Αρχείο hotel_reviews_dataset.csv

Επεξεργασία Σημείων Ενδιαφέροντος (POIs Dataset)

Για τα Σημεία Ενδιαφέροντος εφαρμόστηκε συνένωση χαρακτηριστικών, συνδυάζοντας το όνομα (name) και την κατηγορία (category) κάθε εγγραφής. Σε αντίθεση με τις συχνές ερωτήσεις, τα POIs αναζητούνται συνήθως με βάση τον τύπο τους (π.χ. "μουσεία", "εστιατόρια") και όχι αποκλειστικά με το όνομά τους. Ενσωματώνοντας την κατηγορία στο κείμενο που διανυσματοποιείται, το μοντέλο all-MiniLM-L6-v2 μπορεί να συσχετίσει ερωτήματα όπως "places to eat" με εγγραφές τύπου "Taverna X - Restaurant", λόγω της λέξης "Restaurant" βελτιώνοντας αισθητά την ανάκληση.

```
# Ανάγνωση του αρχείου POIS
pois_df = pd.read_csv(POIS_CSV_PATH)

# Συνδυασμός Ονόματος και Κατηγορίας για βελτιωμένη αναζήτηση
# Π.χ. "Palace of Knossos - Historical Site"
documents = (pois_df['name'] + " - " + pois_df['category']).tolist()

# Τα μεταδεδομένα περιλαμβάνουν κατηγορία, όνομα, διεύθυνση.
metadatas = pois_df.to_dict('records')
```

Εικόνα 14 : Κώδικας από setup_vector_stores.py

category	name	address
Museum	Archaeological Museum of Heraklion	2 Xanthoudidou & Tsouderon
Landmark	Venetian Fortress “Κούλες”	Old Harbour
Museum	Natural History Museum of Crete	Sof. Venizelou Avenue
Landmark	Venetian Walls of Heraklion	25th August Street
Landmark	Morosini Fountain (Λιοντάρι)	Pl. El. Venizelou
Religious	Cathedral of St. Minas	Pl. Agiou Titou
Museum	Historical Museum of Crete	Pl. 1866
Historic Site	Palace of Knossos	Knossos Site
Landmark	25th August Street (Main promenade)	25th August Str.
Park	Theotokopoulos Park	Pl. Eleftherias
Beach	Amoudara Beach	Amoudara
Beach	Diktynna Beach	Diktynna Road
Nature	Mount Stroumboulas	Near Heraklion city
Religious	Basilica of Saint Titus	Pl. Agiou Titou
Landmark	Old Venetian Harbor	Old Port Area
Entertainment	Cretaquarium & Thalassocosmos	Ammoudara
Restaurant Area	Restaurant Row (Παλιό Λιμάνι)	Old Harbour Restaurants
Winery	Lyrarakis Winery	Heraklion area
Sports Arena	Heraklion Indoor Sports Arena	Dyo Aorakia
Shopping Area	28th October Street Shopping	28th Oct Str.
Entertainment	Open Air Cinema Heraklion	Beachfront

Εικόνα 15 : Αρχείο pois.csv

4.2.2 Δημιουργία Vector Stores με FAISS

Η υποδομή ανάκτησης πληροφοριών του chatbot βασίζεται στη δημιουργία διανυσματικών βάσεων δεδομένων (Vector Stores) με τη χρήση της βιβλιοθήκης FAISS (Facebook AI Similarity Search).

Η διαδικασία υλοποιείται μέσω του script *setup_vector_stores.py*, ξεκινώντας με την αρχικοποίηση του μοντέλου παραγωγής ενσωματώσεων. Επιλέχθηκε το προ-εκπαιδευμένο μοντέλο *all-MiniLM-L6-v2* (μέσω της βιβλιοθήκης langchain_huggingface). Η παραγωγή των ευρετηρίων υλοποιείται με τη μέθοδο FAISS.from_texts. Η μέθοδος αυτή επιτρέπει την ταυτόχρονη αποθήκευση των κειμένων και των αντίστοιχων μεταδεδομένων (metadata), διασφαλίζοντας ότι κατά την ανάκτηση θα είναι διαθέσιμη η πλήρης

πληροφορία (π.χ. η απάντηση που αντιστοιχεί σε μια ερώτηση). Με τη χρήση της εντολής save_local, τα ευρετήρια αποθηκεύονται μόνιμα στον υπολογιστή.

```
# Δημιουργία και τοπική αποθήκευση του Vector Store
vector_store = FAISS.from_texts(documents, embeddings,
metadatas=metadatas)
vector_store.save_local(FAQ_VECTOR_STORE_PATH)
…
vector_store.save_local(REVIEWS_VECTOR_STORE_PATH)
…
vector_store.save_local(POIS_VECTOR_STORE_PATH)
```

Εικόνα 16 : Κώδικας από setup_vector_stores.py

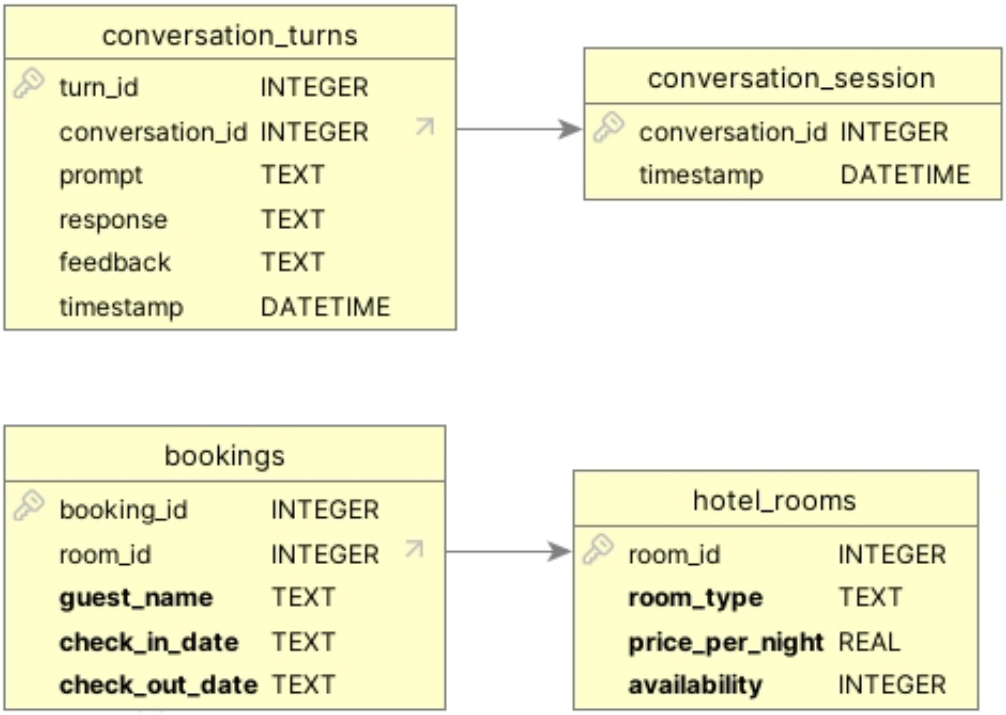
4.3 Σχεδιασμός και ανάπτυξη Βάσης Δεδομένων (SQLite)

Για την αποθήκευση και αποτελεσματική διαχείριση των δομημένων δεδομένων του συστήματος, επιλέχθηκε το Σύστημα Διαχείρισης Βάσεων Δεδομένων SQLite. Η βάση δεδομένων (hotel.db) αποτελεί τη κεντρική βάση αποθήκευσης, εξυπηρετώντας δύο διακριτούς αλλά συμπληρωματικούς επιχειρησιακούς σκοπούς:

1. **Διαχείριση Επιχειρησιακών Δεδομένων:** Αποθήκευση πληροφοριών σχετικά με τη διαθεσιμότητα των δωματίων και τις κρατήσεις των πελατών.
2. **Καταγραφή Ιστορικού και Αξιολόγηση:** Τήρηση αρχείου των συνομιλιών (logging) και των αξιολογήσεων των χρηστών (feedback) για την ανάλυση της απόδοσης του chatbot.

Η επιλογή της SQLite έγινε λόγω της απλότητας της, για την διασφάλιση της ακεραιότητας των συναλλαγών, την συμβατότητα με τη γλώσσα Python (μέσω της βιβλιοθήκης sqlite3) και την ευκολία διαχείρισης που προσφέρει ένα serverless σύστημα.

Ο σχεδιασμός της βάσης δεδομένων ακολουθεί το Διάγραμμα Οντοτήτων-Συσχετίσεων (ER Diagram). Το σχήμα αποτελείται από τέσσερις 4 κύριους πίνακες, οι οποίοι οργανώνονται σε δύο λογικές ενότητες:



Εικόνα 17 : Διάγραμμα οντοτήτων-συσχετίσεων

1. Ενότητα διαχείρισης ξενοδοχειακών πόρων και κρατήσεων

Η ενότητα αυτή υλοποιείται μέσω του script *setup_db.py* και περιλαμβάνει:

- **Πίνακας *hotel_rooms*:** Περιλαμβάνει τα χαρακτηριστικά των δωματίων (τύπος, κόστος ανά διανυκτέρευση) και την κατάσταση διαθεσιμότητάς τους.

  room_id (INTEGER PRIMARY KEY):Μοναδικός κωδικός αναγνώρισης δωματίου

  room_type (TEXT): Κατηγορία δωματίου (π.χ. “Standard”, “Deluxe”, “Suite”).

  price_per_night (REAL): Κόστος διανυκτέρευσης.

  availability (INTEGER): Δείκτης διαθεσιμότητας (τιμή 1 για διαθέσιμο, 0 για κατειλημμένο).

- **Πίνακας *bookings*:** Καταγράφει τις κρατήσεις, συνδέοντας κάθε κράτηση με ένα συγκεκριμένο δωμάτιο μέσω ξένου κλειδιού (room_id).

  booking_id (INTEGER PRIMARY KEY): Μοναδικός κωδικός κράτησης.

  room_id (INTEGER): Ξένο κλειδί (Foreign Key) που συνδέει την κράτηση με τη συγκεκριμένη οντότητα δωματίου στον πίνακα hotel_rooms.

guest_name (TEXT): Ονοματεπώνυμο του πελάτη.
check_in_date (TEXT): Ημερομηνία άφιξης σε μορφή προτύπου (YYYY-MM-DD).
check_out_date (TEXT): Ημερομηνία αναχώρησης.

Η συχέτιση μεταξύ του πίνακα *hotel_rooms* και του πίνακα *bookings* είναι Ένα προς Πολλά (One-to-Many). Ένα δωμάτιο μπορεί να έχει πολλές κρατήσεις, αλλά κάθε κράτηση αντιστοιχεί σε ένα και μόνο δωμάτιο.

2. Ενότητα Ιστορικού Συνομιλιών και Feedback

Η ενότητα αυτή υλοποιείται μέσω του script *setup_feedback_db.py* με σκοπό την καταγραφή και παρακολούθηση της ποιότητας των απαντήσεων:

- **Πίνακας *conversation_session*:** Αποθηκεύει τις μεμονωμένες συνεδρίες συνομιλίας, καταγράφοντας την έναρξη κάθε αλληλεπίδρασης.
  conversation_id (INTEGER PRIMARY KEY): Μοναδικός κωδικός συνεδρίας (Auto-increment).
  timestamp: Η χρονική στιγμή έναρξης της συνομιλίας.

- **Πίνακας *conversation_turns*:** Αποτελεί την βάση του μηχανισμού καταγραφής, αποθηκεύοντας κάθε μήνυμα του διαλόγου (ερώτηση χρήστη και απάντηση συστήματος). Περιλαμβάνει πεδία για την αξιολόγηση του χρήστη (feedback) και τη χρονική σήμανσης της.
  turn_id (INTEGER PRIMARY KEY): Μοναδικός κωδικός σειράς.
  conversation_id (INTEGER FOREIGN KEY): Συσχέτιση με τη συνεδρία (conversation_session)
  prompt (TEXT): Το ερώτημα του χρήστη.
  response (TEXT): Η απάντηση που παρήγαγε το chatbot.
  feedback (TEXT): Η αξιολόγηση του χρήστη (“Good”, “Bad”) για τη συγκεκριμένη απάντηση.
  timestamp (DATETIME): Χρόνος καταγραφής.

Η συχέτιση μεταξύ του πίνακα *conversation_session* και του πίνακα *conversation_turns* είναι Ένα προς Πολλά (One-to-Many). Μία συνεδρία συνομιλίας μπορεί να περιλαμβάνει πολλά βήματα διαλόγου (ερωτήσεις-απαντήσεις), αλλά κάθε βήμα ανήκει σε μία μόνο συνεδρία.

Η υλοποίηση της βάσης δεδομένων γίνεται μέσω δύο ξεχωριστών scripts (*setup_db.py* και *setup_feedback_db.py*) τα οποία δημιουργούν τους πίνακες και εισάγουν αρχικά δεδομένα, όπου χρειάζεται.

Αξιοποίηση ιστορικού συνομιλιών για βελτίωση, ταχύτητα και οικονομία tokens

Ένα σημαντικό χαρακτηριστικό του συστήματος είναι η δυνατότητα αυτοβελτίωσης μέσω του μηχανισμού ανάδρασης. Το script *update_good_conversations_vector_store.py* λειτουργεί ως γέφυρα μεταξύ της βάσης δεδομένων SQLite και του συστήματος RAG. Συγκεκριμένα:

- Αντλεί από τον πίνακα conversation_turns τις συνομιλίες που έχουν λάβει θετική αξιολόγηση (feedback = 'good').
- Δημιουργεί διανυσματικές αναπαραστάσεις (embeddings) των ερωτημάτων (prompts) αυτών των επιτυχημένων ζευγών ερώτησης-απάντησης.
- Ενημερώνει το good_examples_vector_store (FAISS), προσθέτοντας νέα παραδείγματα ή δημιουργώντας το από την αρχή αν δεν υπάρχει.

Με αυτόν τον τρόπο, οι καλές απαντήσεις του παρελθόντος ενσωματώνονται στη μνήμη του συστήματος και τροφοδοτούνται ως παραδείγματα στο Prompt του LLM για μελλοντικές ερωτήσεις, βελτιώνοντας διαρκώς όχι μόνο την ποιότητα των απαντήσεων αλλα και λειτουργώντας ως caching για ταχύτερη απάντηση από το LLM. Πριν κληθεί το LLM, το σύστημα ελέγχει αν υπάρχει ήδη μια παρόμοια ερώτηση στο good_examples_vector_store. Αν βρεθεί υψηλή ομοιότητα, μπορεί να ανασυρθεί απευθείας η αποθηκευμένη απάντηση. Επίσης πολύ σημαντικό είναι και το γεγονός ότι με αυτή τη τεχνική μειώνεται ο αριθμός των tokens που καταναλώνονται στο LLM, μειώνοντας έτσι και το λειτουργικό κόστος.

```
# Επιλέγουμε μόνο τα ζεύγη ερώτησης-απάντησης με θετικό feedback
#('good')

conn = sqlite3.connect(DB_PATH)

cursor = conn.cursor()

cursor.execute("SELECT prompt, response FROM conversation_turns WHERE
feedback = 'good'")

good_conversations = cursor.fetchall()

conn.close()

…

# Προετοιμασία Δεδομένων για το Vector Store. Διαχωρίζουμε τα δεδομένα
#σε λίστες για ευρετηρίαση.

# documents: Οι ερωτήσεις των χρηστών (κείμενο προς
#αναζήτηση/embedding).

# metadatas: Οι απαντήσεις του chatbot (πληροφορία προς ανάκτηση).

documents = []

metadatas = []

for prompt, response in good_conversations:

      documents.append(prompt)

      metadatas.append({"response": response})

# Ενημέρωση ή Δημιουργία του FAISS Index. Φορτώνουμε το υπάρχον vector
#store για να προσθέσουμε τα νέα δεδομένα. Αν δεν υπάρχει,
#δημιουργούμε ένα καινούργιο.

try:

      vector_store = FAISS.load_local(VECTOR_STORE_PATH, embeddings,
      allow_dangerous_deserialization=True)

      vector_store.add_texts(texts=documents, metadatas=metadatas)

except Exception:

      vector_store = FAISS.from_texts(texts=documents,
      embedding=embeddings, metadatas=metadatas)
```

Εικόνα 18 : Κώδικας από update_good_conversations_vector_store.py

4.4 Λογική Chatbot - Υλοποίηση του RAG Pipeline με χρήση LangChain

4.4.1 Δημιουργία Εργαλείων (Tools) για Ανάκτηση Πληροφοριών

Η ικανότητα του πράκτορα να παρέχει ακριβείς και τεκμηριωμένες πληροφορίες στο πλαίσιο της λειτουργίας του ως συστήματος RAG, βασίζεται στην υλοποίηση εξειδικευμένων εργαλείων. Τα εργαλεία αυτά αποτελούν το συνδετικό κρίκο μεταξύ του μεγάλου γλωσσικού μοντέλου και των εξωτερικών πηγών δεδομένων, επιτρέποντας την ανάκτηση πληροφοριών σε πραγματικό χρόνο από διανυσματικές και σχεσιακές βάσεις δεδομένων. Η υλοποίησή τους πραγματοποιήθηκε με τη χρήση του διακοσμητή (decorator) "@tool" της βιβλιοθήκης LangChain, ο οποίος επιτρέπει τη μετατροπή τυπικών συναρτήσεων Python σε λειτουργικές μονάδες που ο πράκτορας μπορεί να καλέσει αυτόνομα. Η επιλογή του κατάλληλου εργαλείου από το μοντέλο δεν περιορίζεται στην ανάγνωση του ονόματος της συνάρτησης, αλλά βασίζεται καθοριστικά στην ανάλυση της περιγραφής (docstring) που συνοδεύει κάθε εργαλείο [17] . Το LangChain μεταφέρει στο LLM μια λίστα με τα ονόματα και τις λεπτομερείς περιγραφές των συναρτήσεων, και το μοντέλο επιλέγει εκείνο που ταιριάζει περισσότερο στο αίτημα του χρήστη βάσει του κειμένου της περιγραφής. Αυτό σημαίνει ότι ακόμη και αν το όνομα μιας συνάρτησης είναι ασαφές, η ύπαρξη ενός λεπτομερούς docstring επιτρέπει στο μοντέλο να κατανοήσει τη λειτουργικότητα και τις παραμέτρους του εργαλείου, διασφαλίζοντας την ορθή λήψη αποφάσεων κατά τη διαδικασία της συλλογιστικής (reasoning).

Το πρώτο και βασικότερο εργαλείο ανάκτησης είναι η συνάρτηση *get_faq_answer()* (εικόνα 19) , η οποία είναι υπεύθυνη για την παροχή απαντήσεων σε συχνά ερωτήματα των χρηστών. Η λειτουργία της βασίζεται στην αναζήτηση σημασιολογικής ομοιότητας εντός της διανυσματικής βάσης δεδομένων FAQ.

```
@tool
def get_faq_answer(question: str) -> str:
    """
    Searches the hotel's FAQ for an answer to a specific question using
vector similarity search.
    """
    if faq_vector_store:
        results = faq_vector_store.similarity_search(question, k=3)
```

```
        if results:
            return results[0].metadata['answer']
        else:
            return "I couldn't find an answer to that question in our
FAQ."
    else:
        return "I'm sorry, I couldn't access the FAQ information right
now."
```

Εικόνα 19 : Κώδικας του tool "get_faq_answer" από το chatbot_logic.py

Πέραν της διαχείρισης των συχνών ερωτήσεων, αναπτύχθηκαν επιπλέον εργαλεία που επεκτείνουν τις δυνατότητες πληροφόρησης του συστήματος:

Η συνάρτηση *get_review_summary()* επιτρέπει στον πράκτορα να αναλύει την εμπειρία προηγούμενων επισκεπτών. Αντί να επιστρέφει μεμονωμένες κριτικές, το εργαλείο ανακτά τα πιο σχετικά αποσπάσματα από τη διανυσματική βάση κριτικών βάσει της θεματολογίας που εισάγει ο χρήστης (π.χ. καθαριότητα, εστίαση). Με αυτόν τον τρόπο, ο πράκτορας μπορεί να διαμορφώσει μια αντικειμενική σύνοψη της ποιότητας των υπηρεσιών.

Για την παροχή πληροφοριών σχετικά με την τοποθεσία του ξενοδοχείου, υλοποιήθηκε η συνάρτηση *find_points_of_interest()*. Το εργαλείο αυτό αναζητά στη διανυσματική βάση δεδομένων POI (Points of Interest) αξιοθέατα, τοπικές επιχειρήσεις κτλ. που ταιριάζουν με το ερώτημα του χρήστη. Τα αποτελέσματα περιλαμβάνουν την ονομασία, τη διεύθυνση και την κατηγορία του σημείου, ενισχύοντας τον ρόλο του chatbot ως ολοκληρωμένου τουριστικού οδηγού.

Η διαχείριση των δυναμικών δεδομένων που αφορούν τη διαμονή πραγματοποιείται μέσω των συναρτήσεων *get_available_rooms()* και *get_room_info()*. Τα εργαλεία αυτά αλληλεπιδρούν απευθείας με τη σχεσιακή βάση δεδομένων SQLite, εκτελώντας δομημένα ερωτήματα SQL. Η πρώτη συνάρτηση ελέγχει σε πραγματικό χρόνο τη διαθεσιμότητα των δωματίων ανά τύπο, ενώ η δεύτερη ανακτά τις ισχύουσες τιμές διανυκτέρευσης. Η χρήση σχεσιακής βάσης για αυτά τα δεδομένα κρίθηκε απαραίτητη, καθώς οι πληροφορίες αυτές μεταβάλλονται συχνά και απαιτούν απόλυτη ακρίβεια, την οποία δεν μπορεί να εγγυηθεί μια διανυσματική βάση. Για την υλοποίηση αυτών των εργαλείων επιλέχθηκε η χρήση προκαθορισμένων ερωτημάτων SQL αντί της δυναμικής παραγωγής κώδικα SQL από το γλωσσικό μοντέλο. Η απόφαση αυτή βασίστηκε σε δύο κύριους πυλώνες:

1. Ασφάλεια δεδομένων: Τα προκαθορισμένα ερωτήματα αποτρέπουν το ενδεχόμενο κακόβουλων ή λανθασμένων SQL ερωτημάτων και διασφαλίζουν ότι ο πράκτορας δεν θα εκτελέσει κατά λάθος εντολές τροποποίησης ή διαγραφής δεδομένων.
2. Βελτιστοποίηση πόρων: Με τη χρήση έτοιμων ερωτημάτων εξοικονομείται σημαντικός αριθμός tokens, καθώς δεν απαιτείται η αποστολή του σχήματος της βάσης δεδομένων στο μοντέλο ούτε η λεπτομερής καθοδήγησή του για τη σύνταξη του ερωτήματος. Παράλληλα, εξασφαλίζεται η ακρίβεια και η ταχύτητα της ανάκτησης.

Τέλος, η συνάρτηση *get_all_room_types()* λειτουργεί υποστηρικτικά, παρέχοντας στον πράκτορα τη δυνατότητα να γνωρίζει το σύνολο των προσφερόμενων κατηγοριών δωματίων. Αυτή η γνώση είναι απαραίτητη ώστε το μοντέλο να μπορεί να καθοδηγήσει τον χρήστη στις διαθέσιμες επιλογές προτού προχωρήσει σε συγκεκριμένες αναζητήσεις διαθεσιμότητας ή τιμών.

Η αρχιτεκτονική αυτή διασφαλίζει τον πλήρη διαχωρισμό της λογικής της ανάκτησης από τη λογική της σύνθεσης της απάντησης. Με αυτόν τον τρόπο, το μοντέλο γλώσσας λειτουργεί ως συντονιστής, επιλέγοντας την κατάλληλη συνάρτηση βάσει της πρόθεσης του χρήστη και επεξεργάζεται τα επιστρεφόμενα δεδομένα για τη διαμόρφωση μιας φυσικής και πειστικής απόκρισης.

4.4.2 Ενσωμάτωση του Γλωσσικού Μοντέλου DeepSeek και Δημιουργία Prompt

Μετά τον ορισμό των εργαλείων ανάκτησης, το επόμενο στάδιο της υλοποίησης αφορά τη συγκρότηση της κεντρικής λογικής του πράκτορα, η οποία βρίσκεται στη συνάρτηση *handle_user_input()*. Ο πράκτορας λειτουργεί ως ο κεντρικός εγκέφαλος του συστήματος, υπεύθυνος για τη λήψη αποφάσεων σχετικά με τη χρήση των εργαλείων και τη σύνθεση της τελικής απόκρισης. Η υλοποίηση βασίζεται στην αρχιτεκτονική ReAct (Reasoning and Acting), η οποία επιτρέπει στο μοντέλο να εκτελεί βήματα λογικής σκέψης προτού προβεί σε οποιαδήποτε ενέργεια. Η τεχνική αυτή υιοθετεί έναν επαναληπτικό κύκλο ο οποίος αποτελείται από τα εξής στάδια:

1. Σκέψη (**Thought**): Το μοντέλο αναλύει το ερώτημα του χρήστη και διατυπώνει έναν συλλογισμό σχετικά με το ποιο είναι το επόμενο απαραίτητο βήμα για την επίλυση του προβλήματος.

3. Ενέργεια (**Action**): Με βάση τον συλλογισμό, ο πράκτορας επιλέγει το καταλληλότερο εργαλείο από το διαθέσιμο σύνολο (π.χ. get_faq_answer).
4. Είσοδος Ενέργειας (**Action Input**): Καθορίζεται η συγκεκριμένη παράμετρος που θα σταλεί στο εργαλείο (π.χ. η ερώτηση προς αναζήτηση).
5. Παρατήρηση (**Observation**): Το μοντέλο λαμβάνει το αποτέλεσμα από την εκτέλεση του εργαλείου και το ενσωματώνει στο πλαίσιο της συνομιλίας [30].

Ο κύκλος αυτός επαναλαμβάνεται όσες φορές κρίνεται απαραίτητο, μέχρι ο πράκτορας να συγκεντρώσει όλες τις απαιτούμενες πληροφορίες. Μόνο τότε προχωρά στη σύνθεση της Τελικής Απάντησης (Final Answer). Η προσέγγιση αυτή εξασφαλίζει τη διαφάνεια στη λήψη αποφάσεων και μειώνει σημαντικά την πιθανότητα παραγωγής εσφαλμένων πληροφοριών (hallucinations).

Ως εγκέφαλος του συστήματος επιλέχθηκε το γλωσσικό μοντέλο DeepSeek-Chat [35]. Η ενσωμάτωσή του στην πλατφόρμα LangChain πραγματοποιήθηκε αξιοποιώντας τη συμβατότητα του API της DeepSeek με το πρότυπο της OpenAI. Αυτό επέτρεψε τη χρήση της κλάσης ChatOpenAI, ορίζοντας απλώς τη διεύθυνση βάσης (base_url) στον διακομιστή της DeepSeek. Η προσέγγιση αυτή προσφέρει ευελιξία και μειώνει την πολυπλοκότητα της υλοποίησης.

```
llm = ChatOpenAI(
        model="deepseek-chat",
        api_key=DEEPSEEK_API_KEY,
        base_url="https://api.deepseek.com"
    )
```

Εικόνα 20 : Κώδικας από chatbot_logic.py

Κεντρικό ρόλο στην απόδοση του πράκτορα παίζει η Μηχανική Προτροπών (Prompt Engineering). Το πρότυπο προτροπής (Prompt Template, εικόνα 21) που σχεδιάστηκε καθορίζει την προσωπικότητα του συστήματος, το ύφος επικοινωνίας και τους περιορισμούς λειτουργίας. Ο πράκτορας ορίζεται ως ένας πειστικός βοηθός του ξενοδοχείου "The Grand Cretan Resort Hotel", με κύριο στόχο την ενθάρρυνση των κρατήσεων.

Ένα κρίσιμο στοιχείο του σχεδιασμού είναι η διαχείριση της πολυγλωσσικότητας. Παρόλο που η βάση γνώσης και τα εργαλεία είναι βελτιστοποιημένα για την αγγλική γλώσσα, ο πράκτορας έχει την οδηγία να μεταφράζει εσωτερικά τα ερωτήματα, να εκτελεί

την ανάκτηση και στη συνέχεια να επιστρέφει την απάντηση στη γλώσσα επιλογής του χρήστη. Η προτροπή (Prompt Template) έχει σχεδιαστεί ώστε να καθοδηγεί το μοντέλο να λειτουργεί ως ένας "πειστικός βοηθός" του ξενοδοχείου, με στόχο την προώθηση των κρατήσεων. Στο prompt δηλώνεται η διαχείριση της ροής σκέψης (Thought-Action-Observation loop). Επίσης, στο template ενσωματώνεται δυναμικά το ιστορικό της συνομιλίας ("chat_history"), επιτρέποντας στον πράκτορα να έχει επίγνωση των προηγούμενων αλληλεπιδράσεων.

Ιδιαίτερη έμφαση δόθηκε στη θέσπιση αυστηρών κανόνων τεκμηρίωσης (Grounding Rules) εντός της προτροπής. Αυτοί οι κανόνες διασφαλίζουν ότι ο πράκτορας βασίζεται αποκλειστικά στα δεδομένα που ανακτώνται από τα εργαλεία και τα παραδείγματα της βάσης γνώσης, απαγορεύοντας ρητά τη χρήση εξωτερικών πληροφοριών ή γενικών γνώσεων του γλωσσικού μοντέλου. Με αυτόν τον τρόπο, ελαχιστοποιείται ο κίνδυνος παραισθήσεων (hallucinations) και διασφαλίζεται ότι οι απαντήσεις είναι ακριβείς, έγκυρες και απόλυτα ευθυγραμμισμένες με την πραγματική κατάσταση και τις υπηρεσίες του ξενοδοχείου.

```
prompt_template = """
You are a persuasive assistant for The Grand Cretan Resort Hotel. Your
primary goal is to convince users to book a stay at the hotel.

IMPORTANT GROUNDING RULES:
1. Base your answers ONLY on the information retrieved via tools or
provided in the examples.
2. NEVER use outside knowledge or general information about hotels that
is not explicitly found in the hotel's data.
3. If the information is not available in the tools or examples, politely
state that you do not have that specific information and invite the user
to contact the hotel reception directly.
4. If the user asks about availability or anything related to a specific
day of the week (e.g., "Monday", "next Friday", etc.), you MUST calculate
the exact date based on "Today's Date" and mention it clearly in your
response (e.g., "Monday, October 27th").
5. After providing information about room availability and prices, if
the user expresses interest in booking or confirming their stay, you
MUST encourage them to contact the hotel reception directly at +30 2810
123456 to finalize their reservation.

When asked about rooms, you should first get all the available room
types, then for each room type get the info and availability.
When the user asks a question in a language other than English, you must
translate the query to English before using any of the tools.

Answer only in this language: {language}.
```

```
Before answering, consider these examples of good answers to similar
questions:{examples}

IMPORTANT: If the user's question is directly answered by one of the
examples above, YOU MUST SKIP THE TOOLS. instead, your Thought should be
"The answer is in the examples" and you should provide the Final Answer
immediately.

You have access to the following tools:{tools}

Use the following format:
Question: the input question you must answer
Thought: you should always think about what to do
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action
Observation: the result of the action
    ... (this Thought/Action/Action Input/Observation can repeat N
times)
Thought: I now know the final answer
Final Answer: the final answer to the original input question.
```

Εικόνα 21 : Κώδικας που ορίζει τη δομή της προτροπής στο chatbot_logic.py

Παράδειγμα Εκτέλεσης Thought/Action/Action Input/Observation Loop

Question (Ερώτηση): "Πόσο κοστίζει η Suite και υπάρχει διαθεσιμότητα;"

Thought (Σκέψη): Ο χρήστης ζητά πληροφορίες για την τιμή και τη διαθεσιμότητα για τον τύπο δωματίου "Suite". Πρέπει πρώτα να βρω την τιμή χρησιμοποιώντας το εργαλείο get_room_info και στη συνέχεια να ελέγξω αν υπάρχουν διαθέσιμα δωμάτια με το get_available_rooms.

Action (Ενέργεια): get_room_info

Action Input: Suite

Observation (Παρατήρηση): A Suite room costs 250 euros per night.

Thought (Σκέψη): Τώρα που γνωρίζω την τιμή 250€, πρέπει να ελέγξω αν υπάρχουν διαθέσιμες σουίτες αυτή τη στιγμή.

Action (Ενέργεια): get_available_rooms

Action Input: Suite

Observation (Παρατήρηση): Yes, we have 2 Suite rooms available.

Thought (Σκέψη): Έχω συγκεντρώσει όλες τις απαραίτητες πληροφορίες: η τιμή είναι 250€ και υπάρχουν 2 διαθέσιμες σουίτες. Θα διαμορφώσω μια πειστική απάντηση στα Ελληνικά, δίνοντας έμφαση στην πολυτέλεια του δωματίου.

Final Answer (Τελική Απάντηση): Η διαμονή σας στην πολυτελή Σουίτα μας κοστίζει 250€ ανά διανυκτέρευση. Πρόκειται για την κορυφαία επιλογή μας που προσφέρει απαράμιλλη

άνεση και θέα. Αυτή τη στιγμή έχουμε μόλις 2 διαθέσιμες σουίτες, οπότε σας προτείνω να προχωρήσετε άμεσα στην κράτησή σας για να εξασφαλίσετε μια ονειρεμένη εμπειρία στο Grand Cretan Resort!

Η διαδικασία εκτέλεσης περιλαμβάνει επίσης έναν μηχανισμό προσωρινής αποθήκευσης (prompt caching) μέσω της χρήσης καλών παραδειγμάτων (*good_examples*), τα οποία αποθηκεύονται αρχικά στη σχεσιακή βάση μέσω της ανατροφοδότησης του χρήστη. Πριν από την κλήση του μοντέλου, το σύστημα εκτελεί αναζήτηση ομοιότητας στο *good_examples_vector_store* στο οποίο έχουν μετατραπεί τα *good_examples* μέσω του script *update_good_conversations_vector_store.py*, για τον εντοπισμό προηγούμενων επιτυχημένων αλληλεπιδράσεων. Η λειτουργία αυτή εξυπηρετεί δύο σκοπούς. Πρώτον, παρέχει στον πράκτορα ένα πλαίσιο αναφοράς για το επιθυμητό ύφος και την ακρίβεια των απαντήσεων. Δεύτερον, λειτουργεί ως φίλτρο βελτιστοποίησης: εάν το ερώτημα του χρήστη ταυτίζεται σημασιολογικά με κάποιο από τα αποθηκευμένα παραδείγματα, ο πράκτορας καθοδηγείται μέσω της προτροπής να παρακάμψει τη χρήση των εργαλείων και να δώσει άμεσα την ήδη επικυρωμένη απάντηση. Η τεχνική αυτή χρησιμοποιήθηκε για να μειώσει τον χρόνο απόκρισης σε κάτω των 15 δευτερολέπτων, που είναι και λειτουργική απαίτηση του συστήματος, αλλά και για να εξοικονομεί υπολογιστικούς πόρους (tokens) για συχνά επαναλαμβανόμενα ερωτήματα.

4.4.3 Διαχείριση Μνήμης

Για την επίτευξη μιας φυσικής και συνεχούς ροής στον διάλογο, η εφαρμογή ενσωματώνει μηχανισμό διαχείρισης μνήμης αξιοποιώντας την κλάση “ConversationBufferWindowMemory” της βιβλιοθήκης LangChain. Ο μηχανισμός αυτός είναι κρίσιμος για τη διατήρηση του πλαισίου (context) της συζήτησης, επιτρέποντας στον πράκτορα να απαντά σε διευκρινιστικές ερωτήσεις (follow-up questions) χωρίς ο χρήστης να χρειάζεται να επαναλαμβάνει πληροφορίες.

Η μνήμη έχει ρυθμιστεί ώστε να διατηρεί τις τελευταίες 5 αλληλεπιδράσεις (k=5), εξασφαλίζοντας ισορροπία μεταξύ της διατήρησης του απαραίτητου πλαισίου και της αποφυγής υπερφόρτωσης της προτροπής (prompt) με παλαιότερες, μη σχετικές πληροφορίες. Η αρχικοποίηση της μνήμης γίνεται στο επίπεδο της συνεδρίας (session state) της διεπαφής χρήστη στο streamlit, διασφαλίζοντας ότι κάθε χρήστης έχει το δικό του ανεξάρτητο ιστορικό συνομιλίας.

```
if 'memory' not in st.session_state:
    # memory_key: Το όνομα της μεταβλητής στο prompt template
({chat_history})
    # input_key: Καθορίζει ποια μεταβλητή εισόδου αποτελεί το μήνυμα
#του χρήστη ( υπάρχουν 3 μεταβλητές: language, examples, input)
st.session_state.memory = ConversationBufferWindowMemory(
        k=5,
        memory_key="chat_history",
        input_key="input"
    )
```

Εικόνα 22 : Κώδικας από app.py

Το αντικείμενο "memory" περνά ως όρισμα στην κύρια λογική του πράκτορα (handle_user_input), όπου και ενσωματώνεται στον "AgentExecutor" του script chatbot_logic.py. Κατά τη δημιουργία του prompt, το ιστορικό της συνομιλίας εισάγεται δυναμικά μέσω της μεταβλητής {chat_history}, επιτρέποντας στο γλωσσικό μοντέλο να βλέπει τα προηγούμενα μηνύματα και να προσαρμόζει τις απαντήσεις του αναλόγως.

4.4.4 Ο Agent και η Λογική της Συνομιλίας

Η καρδιά του συστήματος βρίσκεται στη συνάρτηση handle_user_input(), η οποία ενορχηστρώνει τη λειτουργία του πράκτορα (agent) και διαχειρίζεται τη ροή της συνομιλίας. Σε αυτό το στάδιο, συνδυάζονται όλα τα επιμέρους συστατικά: το γλωσσικό μοντέλο (LLM), τα εργαλεία (tools), η μνήμη (memory) και η μηχανική προτροπών (prompt engineering), προκειμένου να παραχθεί μια τελική, συνεκτική απάντηση προς τον χρήστη.

Η διαδικασία ξεκινά με την αρχικοποίηση του γλωσσικού μοντέλου, όπου χρησιμοποιείται η κλάση "ChatOpenAI" του langChain ρυθμισμένο να επικοινωνεί με το μοντέλο "deepseek-chat". Στη συνέχεια, καθορίζεται η λίστα των διαθέσιμων εργαλείων (tools), τα οποία περιλαμβάνουν λειτουργίες για ανάκτηση πληροφοριών από τα vector stores (FAQ, κριτικές, σημεία ενδιαφέροντος) και από τη βάση δεδομένων SQL (διαθεσιμότητα και πληροφορίες δωματίων).

Κρίσιμο ρόλο διαδραματίζει η ανάκτηση παραδειγμάτων (good examples) από το "good_examples_vector_store". Πριν από την εκτέλεση του πράκτορα, το σύστημα αναζητά παρόμοια ερωτήματα από το παρελθόν. Εάν βρεθούν σχετικά παραδείγματα, ενσωματώνονται στην προτροπή (prompt), παρέχοντας στο μοντέλο πρότυπα "σωστών" απαντήσεων. Μάλιστα, έχει δοθεί ρητή οδηγία στο μοντέλο: εάν η ερώτηση του χρήστη απαντάται άμεσα από τα παραδείγματα, να παρακάμψει τη χρήση εργαλείων και να απαντήσει απευθείας, εξοικονομώντας πόρους και χρόνο.

Η τελική εκτέλεση πραγματοποιείται μέσω του “AgentExecutor” (εικόνα 23) της βιβλιοθήκης LangChain. Ο “AgentExecutor” λαμβάνει τον ορισμό του πράκτορα (agent) και τα εργαλεία, και αναλαμβάνει την εκτέλεση του βρόχου ReAct. Διαχειρίζεται αυτόματα τη μνήμη (memory), ενημερώνοντας το ιστορικό με τα νέα ζεύγη ερωτήσεων-απαντήσεων μετά από κάθε ολοκληρωμένο κύκλο συνομιλίας. Το “verbose=True” στον “AgentExecutor” είναι απαραίτητο για το debugging στο PyCharm, καθώς καταγράφει στην κονσόλα κάθε βήμα της λογικής ReAct (Thought, Action, Observation). Επιτρέπει τον άμεσο εντοπισμό σφαλμάτων στη ροή της συνομιλίας ή στις κλήσεις των εργαλείων, διευκολύνοντας τον έλεγχο. Έτσι, ο προγραμματιστής μπορεί να παρακολουθεί την εσωτερική "σκέψη" του πράκτορα σε πραγματικό χρόνο.

```
# Δημιουργία του πράκτορα
agent = create_react_agent(llm, tools, prompt)
# Αρχικοποίηση του Executor με ενσωμάτωση της μνήμης
agent_executor = AgentExecutor(
    agent=agent,
    tools=tools,
    verbose=True,
    memory=memory # Η μνήμη που μεταφέρεται από το session state
)
```

Εικόνα 23 : Κώδικας από chatbot_logic.py

Τέλος, η συνάρτηση επιστρέφει το αποτέλεσμα της εκτέλεσης μέσω της μεθόδου “invoke” (εικόνα 24). Σε αυτό το σημείο, ο “AgentExecutor” αναλαμβάνει αυτόματα να αντλήσει το ιστορικό από τη μνήμη, να το τοποθετήσει στη θέση {chat_history} της προτροπής και, μετά την ολοκλήρωση της απάντησης, να αποθηκεύσει το νέο γύρο (turn) στη μνήμη. Μέσω της μεθόδου “invoke”, το σύστημα επεξεργάζεται την είσοδο του χρήστη, τη γλώσσα και τα παραδείγματα, επιστρέφοντας τελικά την απάντηση που θα εμφανιστεί στο περιβάλλον διεπαφής.

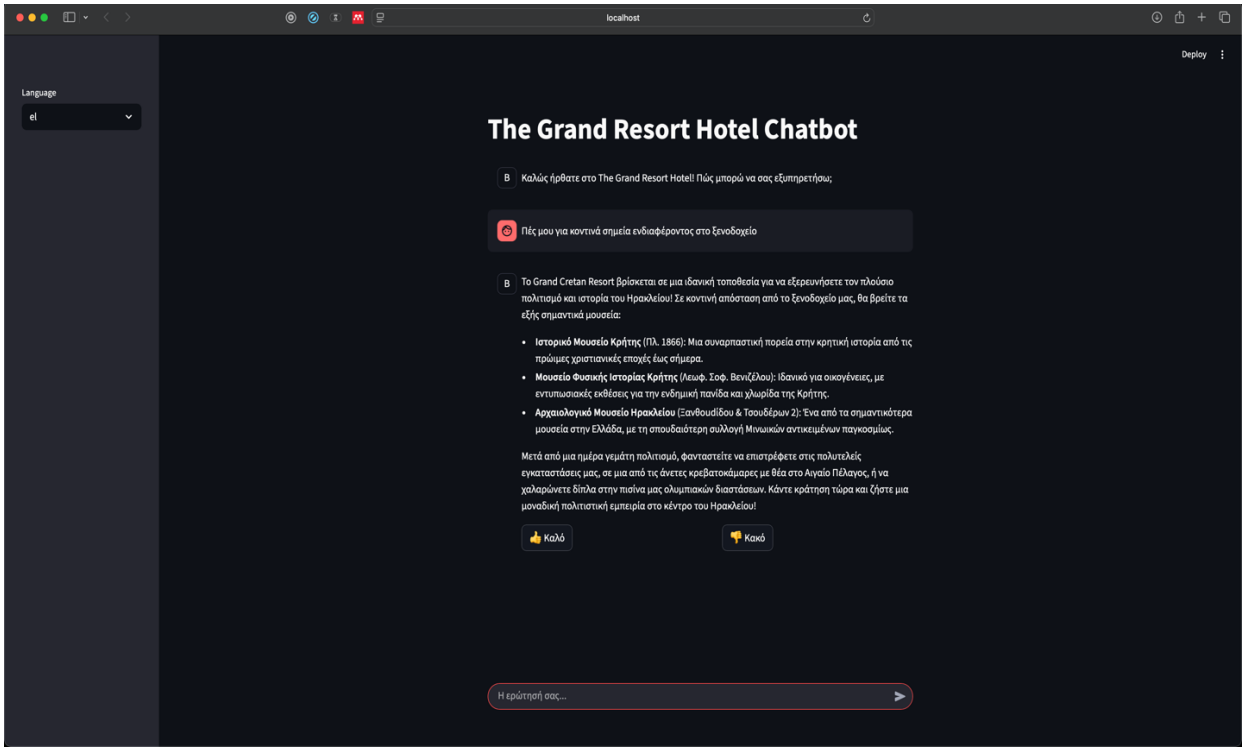
```
response = agent_executor.invoke({
    “input”: user_question,
    “language”: language,
    “examples”: examples
})
return response[“output”]
```

Εικόνα 24 : Κώδικας από chatbot_logic.py

Αυτή η δομή επιτρέπει στον πράκτορα να λειτουργεί αυτόνομα, να χρησιμοποιεί εργαλεία όταν είναι απαραίτητο και να διατηρεί μια φυσική ροή διαλόγου, προσαρμοσμένη στις ανάγκες και το ιστορικό του εκάστοτε χρήστη.

4.5 Σχεδιασμός Διεπαφής Χρήστη (UI)

Ο σχεδιασμός της διεπαφής του chatbot υλοποιήθηκε με χρήση του framework Streamlit. Η συγκεκριμένη επιλογή δεν ήταν τυχαία, αλλά προέκυψε από την ανάγκη για γρήγορη ανάπτυξη, ευκολία στη διαχείριση της κατάστασης (state management) και άμεση διασύνδεση με τη λογική της Python που εκτελείται στο backend (LangChain, LLM).



Εικόνα 25 : Στιγμιότυπο από το streamlit UI του "The Grand Resort Hotel Chatbot"

4.5.1 Επιλογή Τεχνολογιών

Το Streamlit [31] αποτελεί μια open-source βιβλιοθήκη της Python που επιτρέπει τη δημιουργία διαδικτυακών εφαρμογών δεδομένων με ελάχιστο κώδικα. Οι κύριοι λόγοι επιλογής του για την πτυχιακή εργασία είναι οι εξής:

- **Αμιγώς Python Περιβάλλον**: Επιτρέπει την ανάπτυξη ολόκληρου του full-stack (frontend και backend) αποκλειστικά σε Python, εξαλείφοντας την ανάγκη για χρήση JavaScript frameworks (όπως React ή Vue) και REST APIs για την επικοινωνία των δύο μερών.

- **Μηχανισμός Επανεκτέλεσης (Rerun Logic):** Το Streamlit λειτουργεί με έναν μοναδικό τρόπο όπου ολόκληρος ο κώδικας της Python εκτελείται ξανά από την αρχή κάθε φορά που ο χρήστης αλληλεπιδρά με ένα widget (π.χ. πατάει ένα κουμπί). Αυτό απλοποιεί τη ροή δεδομένων, αρκεί να γίνεται σωστή διαχείριση της κατάστασης (Session State).
- **Ενσωματωμένα Chat Widgets**: Προσφέρει έτοιμα, βελτιστοποιημένα στοιχεία διεπαφής για εφαρμογές συνομιλίας (st.chat_message, st.chat_input), τα οποία προσφέρουν μια εμπειρία χρήστη παρόμοια με δημοφιλείς πλατφόρμες (π.χ. ChatGPT).

4.5.2 Διαχείριση Κατάστασης (Session State)

Λόγω του τρόπου λειτουργίας του Streamlit (script rerun), η διατήρηση δεδομένων μεταξύ των αλληλεπιδράσεων είναι κρίσιμη. Για τον σκοπό αυτό χρησιμοποιείται ο μηχανισμός *st.session_state*, ο οποίος λειτουργεί ως ένα αντικείμενο αποθήκευσης δεδομένων της τρέχουσας συνεδρίας, επιτρέποντας στο σύστημα να διατηρεί πληροφορίες, όπως το ιστορικό της συνομιλίας, καθ' όλη τη διάρκεια της αλληλεπίδρασης.

Στο αρχείο *app.py* (εικόνα 26), αρχικοποιούμε τις βασικές μεταβλητές που πρέπει να διατηρηθούν, όπως η τρέχουσα γλώσσα (lang) και το ιστορικό μηνυμάτων (messages).

```
# --- ΑΡΧΙΚΟΠΟΙΗΣΗ SESSION STATE ---
if 'lang' not in st.session_state:
    st.session_state.lang = "en"
if "messages" not in st.session_state:
    st.session_state.messages = []
    # Το turn_id είναι 0 για το αρχικό μήνυμα καλωσορίσματος
    st.session_state.messages.append({
        "role": "bot",
        "message":
UI_TRANSLATIONS[st.session_state.lang]["welcome_message"],
        "turn_id": 0
    })
```

Εικόνα 26 : Κώδικας από app.py

4.5.3 Διαχείριση Μνήμης Συνομιλίας (LangChain Memory)

Ένα από τα πιο σημαντικά στοιχεία ενός chatbot είναι η ικανότητα να θυμάται τα προηγούμενα λεγόμενα. Η μνήμη αυτή δεν αποθηκεύεται απλώς ως κείμενο στην οθόνη, αλλά ως αντικείμενο *ConversationBufferWindowMemory* του LangChain (εικόνα 27), το οποίο τροφοδοτεί το γλωσσικό μοντέλο. Αυτό το αντικείμενο αποθηκεύεται επίσης στο *st.session_state* ώστε να μην χάνεται κατά την ανανέωση της σελίδας.

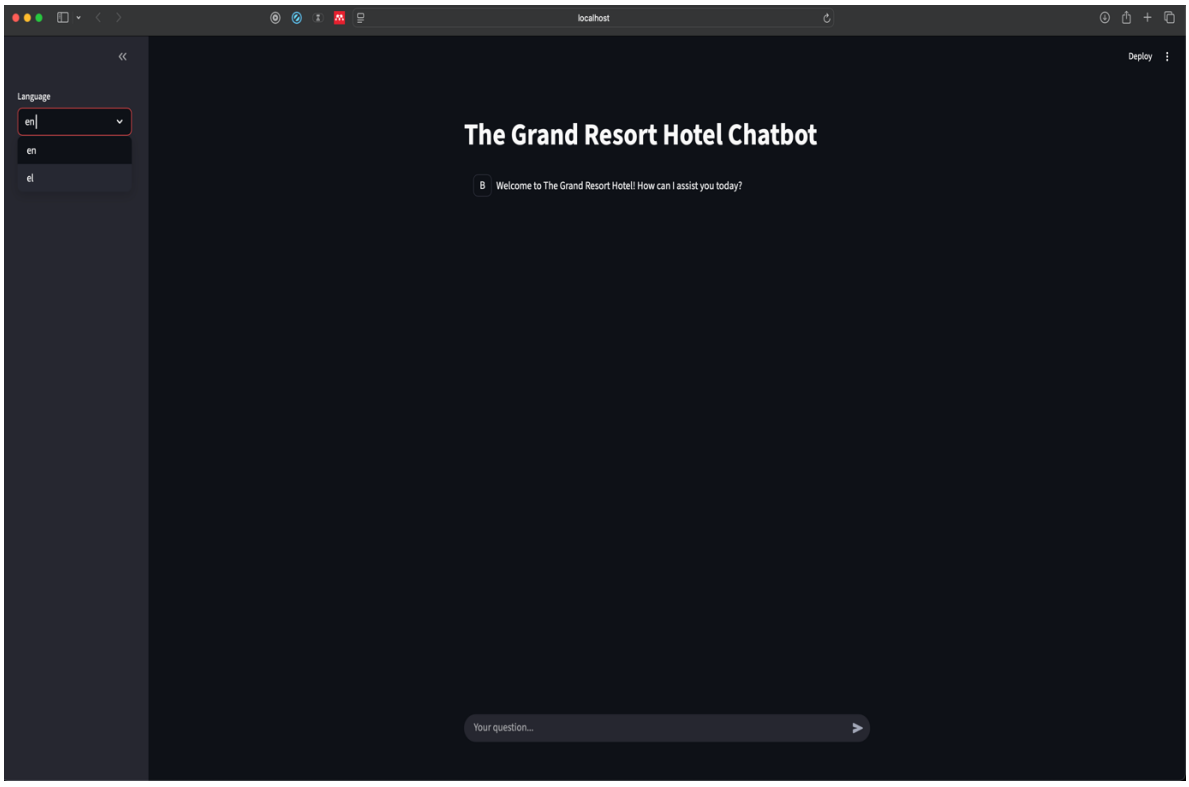
```
def initialize_conversation():
```

```
    # ... (κώδικας βάσης δεδομένων) ...
    if 'memory' not in st.session_state:
        # memory_key: Το όνομα της μεταβλητής στο prompt template
#({chat_history})
        # input_key: Καθορίζει στη μνήμη ποια συγκεκριμένη μεταβλητή
#εισόδου αποτελεί το μήνυμα του χρήστη
        st.session_state.memory = ConversationBufferWindowMemory(k=5,
memory_key="chat_history", input_key="input")
```

Εικόνα 27 : Κώδικας από app.py

4.5.4 Επιλογή Γλώσσας

Η εφαρμογή υποστηρίζει δυναμική αλλαγή γλώσσας (Ελληνικά/Αγγλικά) μέσω ενός sidebar. Η επιλογή αυτή επηρεάζει όχι μόνο τα στατικά κείμενα της διεπαφής (μέσω του λεξικού UI_TRANSLATIONS, εικόνα 29), αλλά και τη συμπεριφορά του chatbot. Όταν ο χρήστης αλλάζει γλώσσα, η εφαρμογή επαναφέρει (reset) τη μνήμη της συνομιλίας και το ιστορικό μηνυμάτων, ώστε να ξεκινήσει μια νέα ροή στη σωστή γλώσσα.



Εικόνα 28 : Στιγμιότυπο από το UI επιλογής γλώσσας

```
# --- ΕΠΙΛΟΓΗ ΓΛΩΣΣΑΣ ---
selected_lang = st.sidebar.selectbox(
    "Language", ["en", "el"], index=["en",
"el"].index(st.session_state.lang), key="lang_selector"
)
```

```
if st.session_state.lang != selected_lang:
    st.session_state.lang = selected_lang
    # Επαναφορά μηνύματος καλωσορίσματος στη νέα γλώσσα
    st.session_state.messages = [{"role": "bot", "message":
UI_TRANSLATIONS[selected_lang]["welcome_message"], "turn_id": 0}]
    # Επαναφορά μνήμης για νέα συνομιλία
    st.session_state.memory = ConversationBufferWindowMemory(k=5,
memory_key="chat_history", input_key="input")
    st.rerun()
```

Εικόνα 29 : Κώδικας από app.py

4.5.5 Ροή Συνομιλίας και Feedback

Η απεικόνιση της συνομιλίας γίνεται μέσω επανάληψης στη λίστα *st.session_state.messages* (εικόνα 30). Για κάθε μήνυμα, χρησιμοποιείται το *st.chat_message* για να δημιουργηθεί το αντίστοιχο “balloon” συνομιλίας. Επιπλέον, ενσωματώθηκε μηχανισμός αξιολόγησης (feedback) απευθείας μέσα στη διεπαφή συνομιλίας, επιτρέποντας στον χρήστη να αξιολογήσει την απάντηση του bot.

```
# --- ΕΜΦΑΝΙΣΗ CHAT HISTORY ---
for msg in st.session_state.messages:
    with st.chat_message(msg["role"]):
        st.markdown(msg["message"])
        # Κουμπιά για το feedback
        if msg["role"] == "bot" and msg["turn_id"] > 0:
            turn_id = msg["turn_id"]
            col1, col2 = st.columns(2)
            with col1:
                if st.button(ui["feedback_good"],
key=f"good_{turn_id}"):
                    save_feedback(turn_id, 'good')
                    # ...
```

Εικόνα 30 : Κώδικας από app.py

4.6 Σύνοψη 4ου Κεφαλαίου

Συνοψίζοντας, στο τέταρτο κεφάλαιο, παρουσιάστηκε αναλυτικά η τεχνική υλοποίηση του συστήματος, βασισμένη σε μια αρχιτεκτονική τριών επιπέδων που εξασφαλίζει επεκτασιμότητα και αποδοτικότητα. Περιγράφηκε η διαδικασία προετοιμασίας και επεξεργασίας των δεδομένων για τη δημιουργία των διανυσματικών βάσεων με τη χρήση FAISS, καθώς και ο σχεδιασμός της σχεσιακής βάσης SQLite για τη διαχείριση των κρατήσεων και του ιστορικού. Αναλύθηκε διεξοδικά η λογική του ReAct Agent μέσω του framework LangChain και η ενσωμάτωση του μοντέλου DeepSeek, εστιάζοντας στη δημιουργία εξειδικευμένων εργαλείων ανάκτησης και στη διαχείριση μνήμης. Τέλος, παρουσιάστηκε η ανάπτυξη της διεπαφής χρήστη με το Streamlit, η οποία

επιτρέπει την πολυγλωσσική επικοινωνία και τη συλλογή ανατροφοδότησης σε πραγματικό χρόνο. Στο επόμενο κεφάλαιο, ακολουθεί η αξιολόγηση του συστήματος μέσω σεναρίων δοκιμών, η ανάλυση της απόδοσής του και η καταγραφή των περιορισμών που προέκυψαν.

5. Δοκιμές και Αξιολόγηση

5.1 Εισαγωγή

Η ανάπτυξη ευφυών συστημάτων διαλόγου που βασίζονται σε Μεγάλα Γλωσσικά Μοντέλα και αρχιτεκτονικές RAG εισάγει νέες προκλήσεις στον τομέα της διασφάλισης ποιότητας λογισμικού. Σε αντίθεση με τις παραδοσιακές εφαρμογές, όπου η ορθότητα ενός αποτελέσματος είναι συνήθως ντετερμινιστική και δυαδική (Σωστό/Λάθος), τα συστήματα παραγωγικής τεχνητής νοημοσύνης χαρακτηρίζονται από στοχαστικότητα και δημιουργικότητα. Ένα chatbot μπορεί να απαντήσει στην ίδια ερώτηση με πολλούς διαφορετικούς τρόπους, οι οποίοι μπορεί να είναι όλοι σημασιολογικά ορθοί, αλλά να διαφέρουν ως προς το ύφος, την πληρότητα ή την ακρίβεια των λεπτομερειών.

Στο πλαίσιο της παρούσας πτυχιακής εργασίας, η διαδικασία αξιολόγησης δεν περιορίστηκε μόνο στον έλεγχο της λειτουργικότητας του κώδικα, αλλά επεκτάθηκε στην ποιοτική αποτίμηση των παραγόμενων απαντήσεων. Ο κύριος στόχος ήταν να διασφαλιστεί ότι το chatbot λειτουργεί όχι μόνο ως μια βάση γνώσης, αλλά και ως ένας πειστικός, ευγενικός και εξυπηρετικός ψηφιακός υπάλληλος, αντάξιος των προτύπων ενός ξενοδοχείου 5 αστέρων.

Η στρατηγική αξιολόγησης σχεδιάστηκε για να μετρήσει τέσσερις βασικούς ποιοτικούς άξονες: την ακρίβεια (Accuracy) των πληροφοριών, τη Συνάφεια (Relevance) με το ερώτημα του χρήστη, την Πειστικότητα (Persuasiveness) του ύφους και την Πιστότητα (Faithfulness) στα δεδομένα πηγής, ώστε να ελαχιστοποιηθούν τα φαινόμενα "παραισθήσεων" (hallucinations). Παράλληλα, αξιολογήθηκε η απόδοση του συστήματος σε επίπεδο ταχύτητας απόκρισης (Latency) σε δευτερόλεπτα, καθώς η άμεση εξυπηρέτηση αποτελεί κρίσιμη λειτουργική απαίτηση για την εμπειρία του χρήστη.

5.2 Μεθοδολογία Αξιολόγησης

Για την υλοποίηση της αξιολόγησης υιοθετήθηκε η σύγχρονη προσέγγιση “LLM-as-a-Judge” (Το LLM ως Κριτής)[32]. Η μέθοδος αυτή αξιοποιεί ένα ισχυρό γλωσσικό μοντέλο για να βαθμολογήσει τις απαντήσεις ενός άλλου μοντέλου, προσφέροντας μια

κλιμακώσιμη και συνεπή εναλλακτική λύση έναντι της χρονοβόρας ανθρώπινης αξιολόγησης.

Στην παρούσα υλοποίηση, ως "Κριτής" επιλέχθηκε το μοντέλο DeepSeek. Το μοντέλο αυτό κλήθηκε να συγκρίνει τρία στοιχεία για κάθε σενάριο ελέγχου:

1. Την ερώτηση του χρήστη.
2. Την απάντηση που παρήγαγε το chatbot.
3. Την "Αλήθεια" (Ground Truth), δηλαδή την ιδανική και τεκμηριωμένη απάντηση που προέρχεται από τα επίσημα έγγραφα του ξενοδοχείου.

Μέσω κατάλληλου Prompt Engineering στο script *evaluation.py*, το μοντέλο-κριτής καθοδηγήθηκε να λειτουργήσει ως ένας αυστηρός αξιολογητής, βαθμολογώντας κάθε απάντηση σε κλίμακα 0 έως 5 για κάθε μία από τις τιθέμενες μετρικές.

Η διαδικασία αυτοματοποιήθηκε πλήρως μέσω του script *evaluation.py*. Η ροή εκτέλεσης περιγράφεται στα παρακάτω βήματα:

1. Φόρτωση Δεδομένων: Το σύστημα διαβάζει τις 30 ερωτήσεις για έλεγχο που δημιουργήθηκαν από το LLM στη βάση γνώσης του ξενοδοχείου και τις αντίστοιχες σωστές απαντήσεις από το CSV αρχείο.
2. Παραγωγή Απάντησης: Κάθε ερώτηση υποβάλλεται στο chatbot. Το σύστημα ενεργοποιεί τον μηχανισμό RAG, ανακτά πληροφορίες και παράγει την τελική απάντηση. Σε αυτό το στάδιο καταγράφεται και ο χρόνος απόκρισης (latency) σε δευτερόλεπτα.
3. Αξιολόγηση: Η τριάδα (Ερώτηση, Απάντηση Chatbot, Ground Truth) στέλνεται στο μοντέλο DeepSeek με εντολή να βαθμολογήσει την ποιότητα.
4. Καταγραφή: Τα αποτελέσματα (βαθμολογίες και σχόλια αιτιολόγησης) αποθηκεύονται στο αρχείο *evaluation_results.csv* για περαιτέρω ανάλυση.

Για τις ανάγκες της αξιολόγησης δημιουργήθηκε ένα ειδικό σύνολο δεδομένων, το οποίο περιλαμβάνεται στο αρχείο *test_scenarios.csv*. Το σύνολο αυτό αποτελείται από 30 επιλεγμένες ερωτήσεις που καλύπτουν το 90% της βάσης γνώσης, χωρισμένες ισομερώς σε δύο γλώσσες:

- 15 Ερωτήσεις στα Ελληνικά.
- 15 Ερωτήσεις στα Αγγλικά.

Οι ερωτήσεις καλύπτουν όλο το φάσμα των δυνατοτήτων του chatbot και χωρίζονται θεματικά στις εξής κατηγορίες:

- Πληροφορίες Ξενοδοχείου (FAQ): Ώρες check-in/out, πολιτική ακύρωσης, Wi-Fi.
- Εγκαταστάσεις & Παροχές (Facilities): Πισίνα, γυμναστήριο, εστιατόριο.
- Τοπικά Αξιοθέατα (Points of Interest): Μουσεία, παραλίες, ιστορικά μνημεία (π.χ. Κνωσός, Κρήνη Μοροζίνι).
- Κριτικές Πελατών (Reviews): Ερωτήσεις που απαιτούν ανάλυση συναισθήματος από παλαιότερες κριτικές (π.χ. "Τι λένε οι πελάτες για το πρωινό;").

Η ύπαρξη της αναμενόμενης σωστής απάντησης “Ground Truth” για κάθε ερώτηση είναι καθοριστική, καθώς αποτελεί το μέτρο σύγκρισης για την εγκυρότητα των απαντήσεων.

test_scenarios

	question	ground_truth
2	**Τι ώρα είναι το check-in και το check-out;**	Το check-in είναι από τις 3:00 μ.μ. και το check-out έως τις 11:00 π.μ.
3	**Επιτρέπονται τα κατοικίδια στο ξενοδοχείο;**	Ναι, είμαστε φιλικοί προς τα κατοικίδια. Υπάρχει χρέωση 50 ευρώ ανά κατοικίδιο, ανά διανυκτέρευση.
4	**Ποια είναι η διεύθυνση του ξενοδοχείου;**	Το ξενοδοχείο βρίσκεται στην οδό Δασκαλογιάννη 23, κοντά στο κέντρο της πόλης.
5	**Ποιες είναι οι κοντινότερες αθλητικές εγκαταστάσεις;**	Το κλειστό γυμναστήριο Ηρακλείου (Δύο Αοράκια) είναι η κύρια αθλητική εγκατάσταση.
6	**Πες μου κοντινά μουσεία που υπάρχουν.**	Μπορείτε να επισκεφθείτε το Αρχαιολογικό Μουσείο, το Μουσείο Φυσικής Ιστορίας και το Ιστορικό Μουσείο.
7	**Πού βρίσκεται η Κρήνη Μοροζίνι;**	Η Κρήνη Μοροζίνι (Λιοντάρι) βρίσκεται στην Πλατεία Ελευθερίου Βενιζέλου.
8	**Τι λένε οι επισκέπτες για το πρωινό;**	Οι επισκέπτες αναφέρουν ότι ο μπουφές πρωινού έχει μεγάλη ποικιλία και νόστιμες επιλογές.
9	**Υπάρχει παραλία κοντά στο ξενοδοχείο;**	Ναι, η παραλία Αμμουδάρα είναι μια κοντινή επιλογή.
10	**Παρέχετε δωρεάν Wi-Fi;**	Ναι, προσφέρουμε δωρεάν Wi-Fi υψηλής ταχύτητας για όλους τους επισκέπτες.
11	**Υπάρχει πισίνα στο ξενοδοχείο;**	Ναι, διαθέτουμε θερμαινόμενη εσωτερική πισίνα στον 5ο όροφο.
12	**Διαθέτετε γυμναστήριο;**	Ναι, το γυμναστήριό μας είναι ανοιχτό 24/7 και βρίσκεται στον 5ο όροφο.
13	**Ποια είναι η πολιτική ακύρωσης;**	Μπορείτε να ακυρώσετε δωρεάν έως και 24 ώρες πριν την άφιξη. Αργότερα χρεώνεται μία διανυκτέρευση.
14	**Πού βρίσκεται το Παλάτι της Κνωσού;**	Το Παλάτι της Κνωσού είναι ένας σημαντικός ιστορικός χώρος κοντά στο ξενοδοχείο.
15	**Τι λένε οι πελάτες για το προσωπικό;**	Οι επισκέπτες περιγράφουν το προσωπικό ως απίστευτα φιλικό και εξυπηρετικό.
16	**Είναι ήσυχα τα δωμάτια;**	Οι περισσότεροι επισκέπτες αγαπούν τα δωμάτια, αν και κάποιοι αναφέρουν θόρυβο από το κλιματιστικό ή λεπτούς τοίχους.
17	**What are the check-in and check-out times?**	Check-in is from 3:00 PM, and check-out is until 11:00 AM.
18	**Do you offer Wi-Fi?**	Yes, we offer complimentary high-speed Wi-Fi for all our guests.
19	**Is breakfast included in the room rate?**	Yes, a complimentary continental breakfast is included.
20	**Where is the pool located?**	We have a heated indoor pool located on the 5th floor.
21	**Can I bring my dog?**	Yes, we are pet-friendly. A fee of 50 euros per pet, per night applies.
22	**What is the cancellation policy?**	You can cancel for free up to 24 hours before check-in.
23	**Where exactly is the hotel?**	The hotel is at Daskalogianni 23, near the city center.
24	**Where is the Archaeological Museum?**	The Archaeological Museum of Heraklion is at 2 Xanthoudidou & Tsouderon.
25	**Is there a nice beach nearby?**	Yes, Amoudara Beach is a popular nearby option.
26	**What do guests say about the staff?**	Guests consistently praise the staff as incredibly friendly and helpful.
27	**Are the rooms quiet?**	Some reviews mention that the air conditioning can be a little noisy or walls a bit thin.
28	**Is the hotel bar expensive?**	Some guests have mentioned that prices at the hotel bar can be quite high.
29	**Is there a gym available?**	Yes, our fitness center is open 24/7 and is located on the 5th floor.
30	**How far is the Palace of Knossos?**	The Palace of Knossos is a major historic site nearby at the Knossos Site.
31	**Does the concierge provide good advice?**	Yes, guests have mentioned the concierge gives excellent recommendations.

Εικόνα 31 : Αρχείο ερωτήσεων *test_scnarios.csv*

5.3 Μετρικές Αξιολόγησης (Evaluation Metrics)

Η ποιότητα του chatbot δεν είναι μονοδιάστατη. Για να αποκτηθεί μια ολοκληρωμένη εικόνα της απόδοσης, ορίστηκαν τέσσερις διακριτές μετρικές ποιότητας και μία μετρική απόδοσης [33]. Η βαθμολόγηση γίνεται στην κλίμακα (0-5).

Ακρίβεια (Accuracy)

Αυτή η μετρική ελέγχει την πραγματική ακρίβεια των πληροφοριών. Είναι η πιο κρίσιμη μετρική για ένα επιχειρησιακό chatbot, καθώς λανθασμένες πληροφορίες (π.χ. λάθος ώρα check-in ή τιμή δωματίου) μπορούν να οδηγήσουν σε δυσαρέσκεια πελατών.

- Βαθμός 5: Η πληροφορία είναι απολύτως ακριβής και συμφωνεί πλήρως με το Ground Truth.
- Βαθμός 0: Η απάντηση περιέχει σοβαρά πραγματολογικά λάθη.

Συνάφεια (Relevance)

Η συνάφεια εξετάζει αν το chatbot απαντά στο συγκεκριμένο ερώτημα που τέθηκε. Συχνά τα LLMs τείνουν να πλατειάζουν ή να δίνουν γενικές πληροφορίες αποφεύγοντας την ουσία.

- Βαθμός 5: Η απάντηση είναι στοχευμένη και καλύπτει ακριβώς την ερώτηση.
- Βαθμός 0: Η απάντηση είναι άσχετη ή γενικόλογη.

Πειστικότητα (Persuasiveness)

Δεδομένου του ρόλου του chatbot ως πωλητή και εκπροσώπου του ξενοδοχείου, το ύφος έχει μεγάλη σημασία. Η μετρική αυτή αξιολογεί αν η γλώσσα που χρησιμοποιείται είναι ελκυστική, φιλόξενη και αν προτρέπει τελικά τον πελάτη να κάνει κράτηση στο ξενοδοχείο.

- Βαθμός 5: Εξαιρετικό ύφος, χρήση θετικών επιθέτων, προτροπή για κράτηση.
- Βαθμός 0: Βαρετή ή αγενής γλώσσα που δεν δημιουργεί προτροπή για κράτηση.

Πιστότητα / Παραισθήσεις (Faithfulness / Hallucination)

Τα LLMs συχνά έχουν παραισθήσεις (Hallucinations), δηλαδή επινοούν γεγονότα που δεν υπάρχουν και ένας από τους βασικούς λόγους ύπαρξης ενός RAG Chatbot είναι η

εξάλειψη αυτών. Η μετρική αυτή ελέγχει αν η απάντηση βασίζεται αποκλειστικά στα ανακτηθέντα δεδομένα (context) και δεν περιέχει αυθαίρετες πληροφορίες.

- Βαθμός 5: Το μοντέλο επινόησε υπηρεσίες ή γεγονότα που δεν ισχύουν (π.χ. ανέφερε ότι υπάρχει ελικοδρόμιο ενώ δεν υπάρχει).
- Βαθμός 0: Καμία παραίσθηση, απόλυτη πιστότητα στις πηγές.

Χρόνος Απόκρισης (Latency)

Μετράται ο χρόνος σε δευτερόλεπτα από τη στιγμή της υποβολής της ερώτησης μέχρι τη λήψη της απάντησης. Αποτελεί κρίσιμο δείκτη για την εμπειρία χρήστη.

5.4 Πειραματικά Αποτελέσματα και Ανάλυση

Η εκτέλεση του σεναρίου ελέγχου στις 30 ερωτήσεις παρήγαγε αποτελέσματα που επιβεβαιώνουν την υψηλή ποιότητα του συστήματος. Τα αναλυτικά δεδομένα καταγράφηκαν στο αρχείο evaluation_results.csv.

evaluation_results

	Question	Response	Reason	Latency	Accuracy	Relevance	Persuasiveness	Hallucination
2	Τι ώρα είναι το check-in και το check-out;	Το check-in στο Grand Cretan Resort γίνεται από τις 3:00 μ.μ., ενώ το check-out είναι έως τις 11:00 π.μ. Για να εξασφαλίσετε μια άνετη και άνευ άγχους άφιξη, σας συνιστούμε να επικοινωνήσετε μαζί μας για να ρυθμίσουμε οποιαδήποτε ειδική ανάγκη ή να ελέγξουμε τη δυνατότητα early check-in. Μετά από μια μακριά διαδρομή, φανταστείτε να ξεκουράζεστε αμέσως στην πολυτελή σουίτα σας με υπέροχη θέα στο Αιγαίο ή να βουτάτε στην πισίνα μας ολυμπιακών διαστάσεων! Κάντε κράτηση τώρα για να ξεκινήσετε τις διακοπές σας με άνεση και στυλ.	The chatbot's response accurately provides the check-in and check-out times as per the ground truth. It is highly persuasive with vivid descriptions and a call to action. However, it includes promotional content beyond the direct answer, slightly reducing relevance. It contains no hallucinations, as all provided information is consistent and plausible.	10.07	5.0	4.0	5.0	0.0
3	Επιτρέπονται τα κατοικίδια στο ξενοδοχείο;	Με μεγάλη μας χαρά, το Grand Cretan Resort είναι ένα φιλικό προς τα κατοικίδια ξενοδοχείο! Καλοσορίζουμε τα αγαπημένα σας μέλη της οικογένειας για να μοιραστούν μαζί σας την πολυτελή διαμονή μας. Για την κάλυψη των επιπλέον καθαρισμών, ισχύει ένα εύλογο τέλος των 50 ευρώ ανά κατοικίδιο, ανά διανυκτέρευση. Φανταστείτε να κάνετε βόλτα με τον σκύλο σας στους εκπληκτικούς κήπους μας ή να χαλαρώνετε στην τέρρασα του δωματίου σας με θέα στο Αιγαίο, ενώ ο πιστός σας σύντροφος ξεκουράζεται δίπλα σας. Με τις άνετες και ευρύχωρες διαμορφώσεις των δωματίων μας, η διαμονή σας θα είναι άνετη για όλους. Κάντε κράτηση τώρα και ζήστε μια αξέχαστη διακοπή στο Ηράκλειο μαζί με το τετράποδο φίλο σας!	The response accurately confirms the hotel is pet-friendly and states the correct fee of 50€ per pet per night, matching the ground truth. It is highly relevant, directly answering the question. It is very persuasive, using vivid imagery and emotional appeal to encourage booking. It contains no hallucinations, as all details (fee, hotel name, location) are consistent with the provided context and do not contradict the ground truth.	19.17	5.0	5.0	5.0	0.0
4	Ποια είναι η διεύθυνση του ξενοδοχείου;	Το Grand Cretan Resort βρίσκεται στην κεντρική λεωφόρο Δασκαλογιάννη 23, σε απόσταση αναπνοής από το ιστορικό κέντρο του Ηρακλείου! Αυτή η άφογη τοποθεσία σας επιτρέπει να απολαμβάνετε την άνεση και την πολυτέλεια του ξενοδοχείου μας, ενώ έχετε άμεση πρόσβαση στα σημαντικά αξιοθέατα, τα εστιατόρια και τη ζωή της πόλης. Φανταστείτε να ξυπνάτε με θέα στο Αιγαίο Πέλαγος από μια από τις άνετες κρεβατοκάμαρες μας και να είστε μόλις λίγα λεπτά με τα πόδια από την παλιά πόλη. Κλείστε τη διαμονή σας τώρα και ζήστε την τέλεια κρητική εμπειρία!	The address is correct, but the response is overly verbose and promotional, adding details not asked for (e.g., sea view, booking call-to-action). This reduces direct relevance. No factual contradictions exist, but the embellished description ('breathing distance', 'perfect location') slightly strays from the neutral ground truth.	21.08	4.0	3.0	5.0	1.0
5	Ποιες είναι οι κοντινότερες αθλητικές εγκαταστάσεις;	Στο Grand Cretan Resort, μπορείτε να απολαύσετε εξαιρετικές αθλητικές εγκαταστάσεις εντός του ξενοδοχείου μας, συμπεριλαμβανομένης μιας πλήρους εξοπλισμένης γυμναστικής, εξωτερικής παίσινας ολυμπιακών διαστάσεων και γηπέδων τένις. Για επιπλέον επιλογές, το Heraklion Indoor Sports Arena βρίσκεται σε κοντινή απόσταση στη διεύθυνση Dyo Aorakia, ενώ το πάρκο Theotokopoulos και η παραλία Amoudara προσφέρουν υπέροχες ευκαιρίες για υπαίθριες δραστηριότητες. Ωστόσο, η άφογη τοποθεσία μας και οι πολυτελείς εγκαταστάσεις μας εξασφαλίζουν ότι δεν θα χρειαστείτε να φύγετε μακριά για να διατηρήσετε την αθλητική σας ρουτίνα!	The chatbot correctly identifies the main facility (Heraklion Indoor Sports Arena) and its address, matching the ground truth. It is highly relevant by listing specific nearby options but dilutes the direct answer with hotel promotion. It is very persuasive in selling the hotel's own amenities. It shows minor hallucination by adding extra facilities (Theotokopoulos park, Amoudara beach) not mentioned in the ground truth as 'closest'.	10.49	3.0	4.0	5.0	2.0
6	Πες μου κοντινά μουσεία που υπάρχουν.	Το Grand Cretan Resort βρίσκεται σε μια ιδανική τοποθεσία για να εξερευνήσετε τον πλούσιο πολιτισμό και ιστορία του Ηρακλείου! Σε κοντινή απόσταση από το ξενοδοχείο μας, θα βρείτε τα εξής σημαντικά μουσεία: - **Ιστορικό Μουσείο Κρήτης** (Π.λ. 1866): Μια συναρπαστική πορεία στην κρητική ιστορία από τις πρώιμες χριστιανικές εποχές έως σήμερα. - **Μουσείο Φυσικής Ιστορίας Κρήτης** (λεωφ. Σοφ. Βενιζέλου): Ιδανικό για οικογένειες, με εντυπωσιακές εκθέσεις για την ενδημική πανίδα και χλωρίδα της Κρήτης. - **Αρχαιολογικό Μουσείο Ηρακλείου** (Ξανθουδίδου & Τσουδερων 2): Ένα από τα σημαντικότερα μουσεία στην Ελλάδα, με τη σπουδαιότερη συλλογή Μινωικών αντικειμένων παγκοσμίως. Μετά από μια ημέρα γεμάτη πολιτισμό, φανταστείτε να επιστρέφετε στις πολυτελείς εγκαταστάσεις μας, σε μια από τις άνετες κρεβατοκάμαρες με θέα στο Αιγαίο Πέλαγος, ή να χαλαρώνετε δίπλα στην πισίνα μας ολυμπιακών διαστάσεων. Κάντε κράτηση τώρα και ζήστε μια μοναδική πολιτιστική εμπειρία στο κέντρο του Ηρακλείου!	The chatbot accurately lists all three museums mentioned in the ground truth (Archaeological, Natural History, Historical Museum), provides relevant details for each, and effectively promotes the hotel's amenities to encourage a booking, with no factual contradictions.	14.91	5.0	5.0	5.0	0.0

Εικόνα 32 : Στιγμιότυπο από το αρχείο αξιολόγησης *evaluation_results.csv*

Με χρώμα πράσινο χρώμα έχουν επισημανθεί οι ερωτήσεις που είναι cached στο vector store “good_examples”. Ιδιαίτερο ενδιαφέρον παρουσιάζει η διακύμανση του χρόνου (latency) μεταξύ διαφορετικών τύπων ερωτήσεων, η οποία αναδεικνύει τη λειτουργία του μηχανισμού "Good Examples" ως ενός είδους κρυφής μνήμης (cached).

- **Μη cached ερωτήσεις**: Όταν το σύστημα καλείται να απαντήσει σε μια νέα, σύνθετη ερώτηση (π.χ. "Επιτρέπονται τα κατοικίδια;"), ο αλγόριθμος ReAct (Reasoning and Acting) εκτελεί πολλαπλά βήματα: σκέφτεται, επιλέγει εργαλείο, εκτελεί αναζήτηση στη βάση δεδομένων, λαμβάνει αποτελέσματα και συνθέτει την απάντηση. Αυτή η διαδικασία είναι πιο χρονοβόρα και βλέπουμε ότι ξεπερνάει τα 15 δευτερόλεπτα σε χρόνο απόκρισης.
- **Cached ερωτήσεις (Good Examples)**: Παρατηρήθηκε σημαντική μείωση του latency σε ερωτήσεις που ομοιάζουν με τα αποθηκευμένα "καλά παραδείγματα" (good_examples_vector_store). Σε αυτές τις περιπτώσεις, το μοντέλο αναγνωρίζει την απάντηση απευθείας από το περιεχόμενο των παραδειγμάτων και παρακάμπτει τη διαδικασία χρήσης εργαλείων. Βλέπουμε ότι όλες οι απαντήσεις εδώ είχαν χρόνο απόκρισης κάτω από 15sec.

evaluation_results

Question	Response	Reason	Latency	Accuracy	Relevance	Persuasiveness	Hallucination
	wonders but also steps away from vibrant modern attractions. After exploring the palace, you can return to the luxury and comfort of our resort. Imagine unwinding after a day of discovery in our spectacular heated indoor pool on the 5th floor, which our guests consistently rate as "stunning and very relaxing" with exceptional scores. You could also enjoy a stroll to the nearby Old Venetian Harbor or the main promenade, all within easy reach. Staying with us means effortless access to both ancient culture and contemporary Cretan life, all while enjoying our world-class amenities. Would you like me to check our available room options so you can secure your perfect base for this unforgettable historical adventure?						
Does the concierge provide good advice?	Absolutely! Our concierge team is renowned for providing exceptional, personalized advice to ensure your stay in Crete is unforgettable. Our guests consistently praise our staff for being "incredibly friendly and helpful," with specific ratings for service reaching as high as 8.7 out of 10. Whether you're seeking the perfect authentic taverna, planning a day trip to an ancient site, or arranging a special experience, our knowledgeable concierge is your gateway to the very best of the island. This dedication to service is just one part of The Grand Cretan Resort's commitment to luxury. Imagine returning from your expertly planned adventures to relax in our stunning heated indoor pool or your beautifully appointed room. To start planning your perfect Cretan holiday with the help of our wonderful team, would you like me to check our available room options for you?	The response accurately confirms the concierge provides excellent advice, directly answers the question with specific details and guest praise, persuasively expands on the hotel's luxury amenities and prompts a booking action, and contains no contradictions or fabrications beyond the ground truth.	11.29	5.0	5.0	5.0	0.0
AVERAGE SCORE	-	-	14.51	4.07	4.1	5.0	0.73

Εικόνα 33 : Μέσες τιμές μετρικών (Average score) στο *evaluation_results.csv*

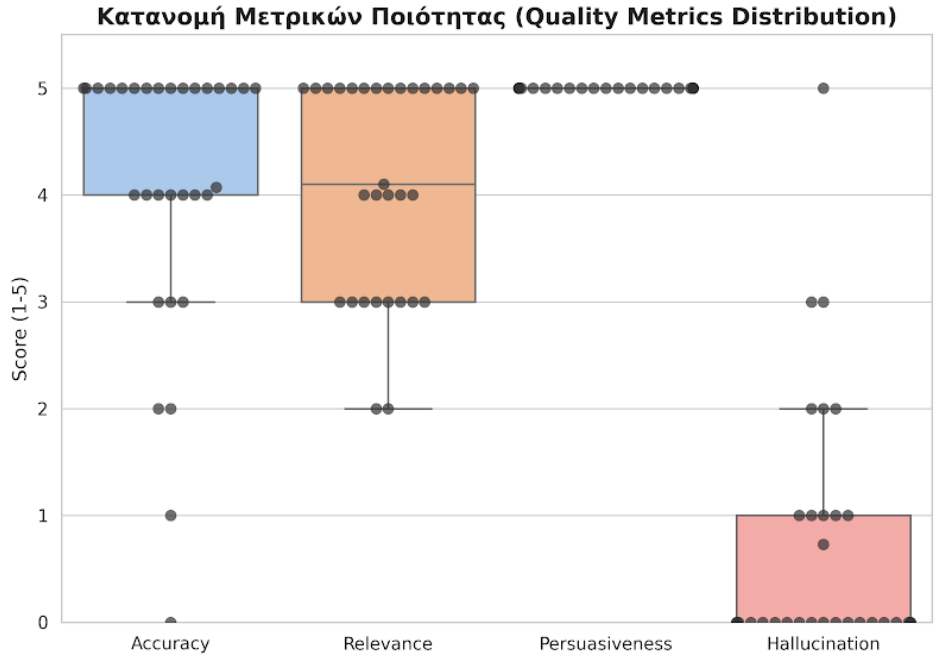
Το απόλυτο 5/5=100% στην Πειστικότητα (Persuasiveness) επιβεβαιώνει ότι το chatbot πέτυχε πλήρως τον επιχειρηματικό του στόχο, να λειτουργεί ως ένας ελκυστικός πωλητής. Ωστόσο, αυτή η επιτυχία επηρέασε ελαφρώς την ακρίβεια (Accuracy=4.07/5=81.4%) και τη Συνάφεια (Relevance=4.1/5=82%). Η ακρίβεια βλέπουμε είναι στο 81.4%, όπως όριζε και η μη λειτουργική απαίτηση όπου ζητούσε ακρίβεια >=80%. Όπως φάνηκε στο παράδειγμα του μπαρ (ερώτηση 28), το μοντέλο μερικές φορές

"θυσιάζει" την άμεση, αρνητική αλήθεια (π.χ. "ναι, είναι ακριβά") προς όφελος μιας πιο διπλωματικής και προωθητικής απάντησης. Αυτή είναι μια αναμενόμενη συμπεριφορά σε εφαρμογές μάρκετινγκ, αλλά ο κριτής (LLM Judge) το βαθμολόγησε ως απόκλιση από το Ground Truth.

Η τιμή 0.73/5 στο Hallucination είναι εξαιρετικά ενθαρρυντική. Σε κλίμακα όπου το 0 είναι το τέλειο και το 5 η αποτυχία, το 0.73 σημαίνει ότι το σύστημα σπάνια επινοεί πληροφορίες. Οι ελάχιστες περιπτώσεις που καταγράφηκαν αφορούσαν κυρίως μικρές γεωγραφικές ανακρίβειες ή υπερβολικές περιγραφές, και όχι επινόηση ανύπαρκτων υπηρεσιών, διασφαλίζοντας έτσι την εμπιστοσύνη των χρηστών.

Ο μέσος όρος των 14.51 δευτερολέπτων, βρίσκεται ακριβώς στο όριο της αποδοχής της μη λειτουργική προδιαγραφής που είχε τεθεί. Αυτό δείχνει ότι το σύστημα εκμεταλλεύεται πλήρως τον χρόνο του για να συνθέσει πλούσιες και αναλυτικές απαντήσεις, αντί να δίνει μονολεκτικές και γρήγορες, αλλά φτωχές απαντήσεις.

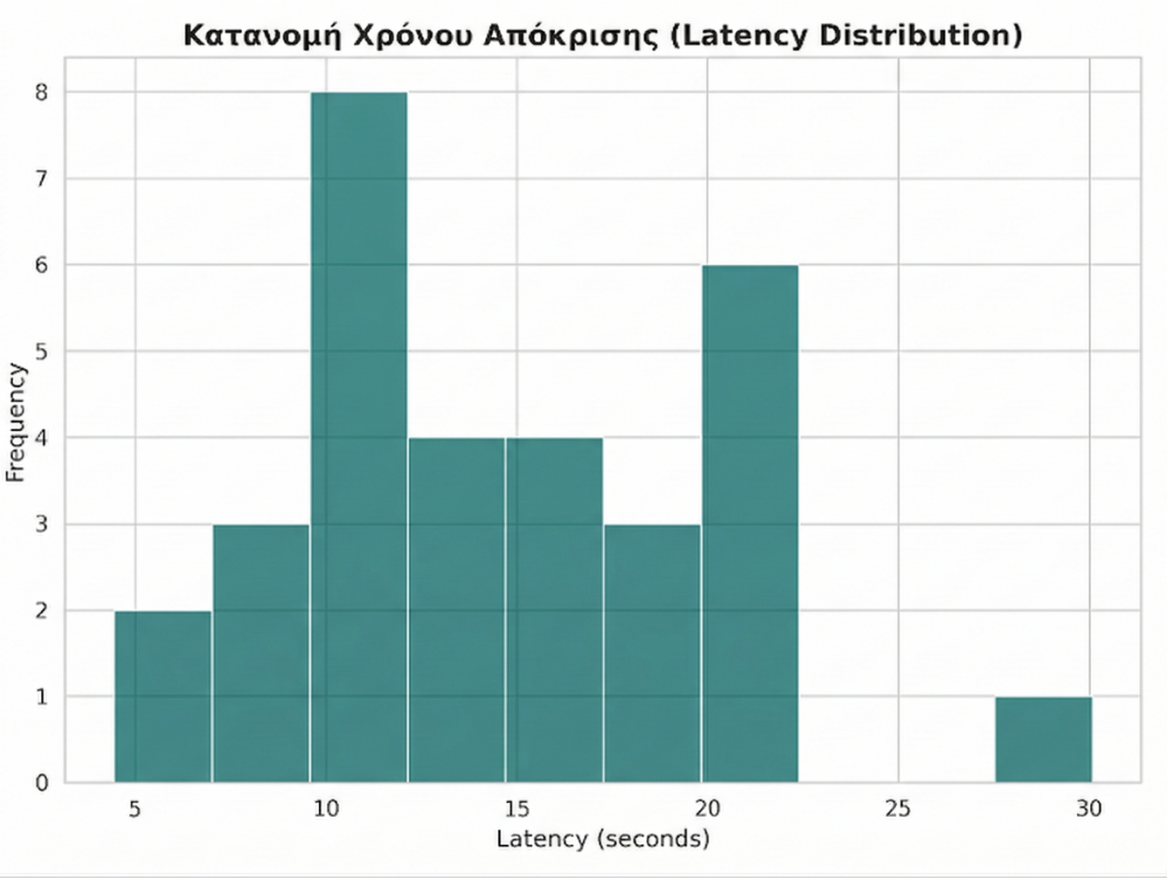
Παρακάτω παρατίθεται και δύο διαγράμματα, το ένα με την κατανομή μετρικών ποιότητας και το άλλο με την κατανομή του χρόνου απόδοσης.



Εικόνα 34 : Διάγραμμα κατανομής μετρικών ποιότητας

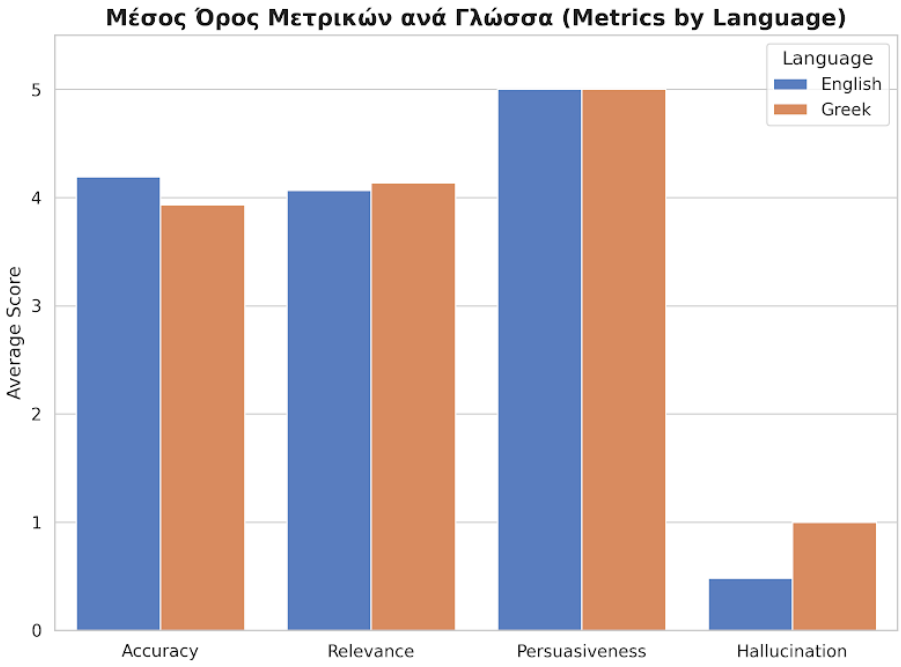
Το συγκεκριμένο διάγραμμα απεικονίζει την κατανομή των μετρικών ποιότητας για 30 δείγματα αξιολόγησης. Κάθε τελεία αντιπροσωπεύει μία από τις 30 αποκρίσεις του συστήματος, ενώ τα κουτιά περικλείουν το κεντρικό 50% των βαθμολογιών, με την

εσωτερική οριζόντια γραμμή να υποδεικνύει τη διάμεσο τιμή. Οι ακραίες τιμές (outliers) αντικατοπτρίζουν σπάνια σφάλματα που αποκλίνουν σημαντικά από τη γενική τάση. Η Ακρίβεια (Accuracy) εμφανίζει υψηλή συγκέντρωση στο επίπεδο 5, με μέσο όρο 4.07, επιβεβαιώνοντας την εγκυρότητα των παρεχόμενων πληροφοριών. Η Σχετικότητα (Relevance) κινείται σε ικανοποιητικά επίπεδα, αν και ορισμένες βαθμολογίες επηρεάστηκαν από φαινόμενα «υπερβολικής φλυαρίας» (verbosity). Η Πειστικότητα (Persuasiveness) σημείωσε το απόλυτο άριστα (5/5) στο σύνολο των δειγμάτων. Τέλος, οι Ψευδαισθήσεις (Hallucination) παρέμειναν εξαιρετικά χαμηλές, με τις περισσότερες τιμές στο 0.

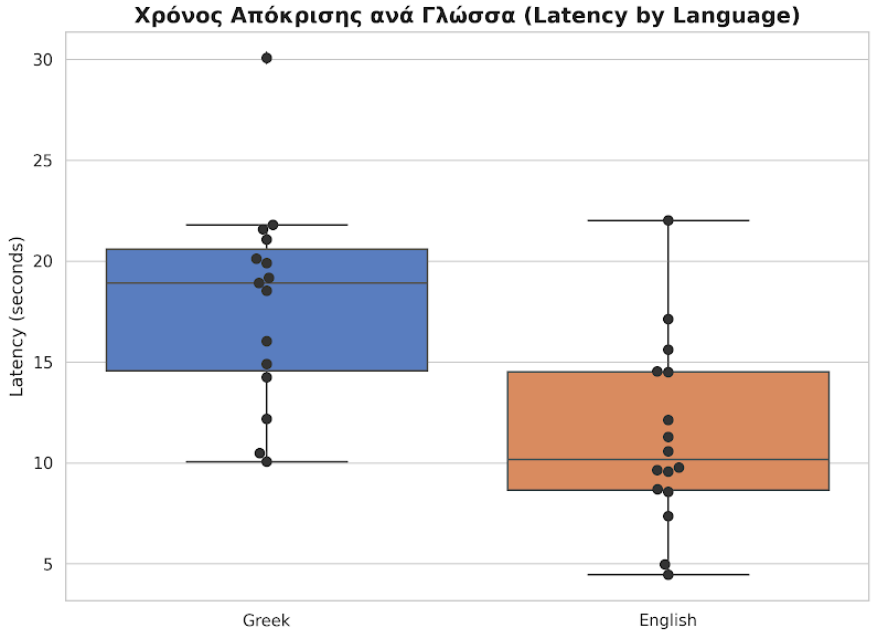


Εικόνα 35 : Διάγραμμα κατανομής χρόνου απόκρισης

Μία άλλη ενδιαφέρουσα σύγκριση εντοπίζεται στην απόδοση μεταξύ των δύο γλωσσών (Ελληνικά - Αγγλικά) και αποκαλύπτει ενδιαφέροντα συμπεράσματα, όπως φαίνονται και στα παρακάτω διαγράμματα.



Εικόνα 36 : Διάγραμμα μέσου όρου μετρικών ανά γλώσσα



Εικόνα 37 : Διάγραμμα χρόνου απόκρισης ανά γλώσσα

Από την ανάλυση των γραφημάτων προκύπτουν τα εξής:

1. **Ποιότητα Απαντήσεων**

   Persuasiveness: Και οι δύο γλώσσες αγγίζουν το άριστο (5.0), αποδεικνύοντας ότι το ύφος διατηρείται σταθερό ανεξαρτήτως γλώσσας.

   Hallucination: Παρατηρείται μια μικρή υπεροχή της Αγγλικής γλώσσας (χαμηλότερο σκορ ~0.5), έναντι της Ελληνικής (~1.0). Αυτό οφείλεται κυρίως στο γεγονός ότι η βάση γνώσης του συστήματος RAG είναι δομημένη αποκλειστικά στην αγγλική γλώσσα. Συγκεκριμένα, η ελληνική ροή απαιτεί τη συνεχή μετατροπή εννοιών από τη μία γλώσσα στην άλλη (Ελληνικά-Αγγλικά), γεγονός που αυξάνει την πολυπλοκότητα της επεξεργασίας και εισάγει μικρές ανακρίβειες. Το μοντέλο αναγκάζεται έτσι να συμπληρώσει τα κενά χρησιμοποιώντας τη δική του εσωτερική γνώση η οποία είναι βασισμένη κυρίως στην Αγγλική γλώσσα και αυτό το γέμισμα «κενών» εισάγει παραισθήσεις.

   Accuracy & Relevance: Οι διαφορές είναι αμελητέες, με τα Αγγλικά να υπερτερούν ελαφρώς στην ακρίβεια και τα Ελληνικά στη Συνάφεια.

2. **Χρόνος Απόκρισης**

   Υπάρχει μια σαφής διαφοροποίηση στο Latency.

   Αγγλικά: Ο μέσος χρόνος απόκρισης είναι περίπου 10 δευτερόλεπτα, με πολλές απαντήσεις να δίνονται σε λιγότερο από 8 δευτερόλεπτα.

Ελληνικά: Ο χρόνος απόκρισης είναι υψηλότερος, με διάμεση τιμή κοντά στα 19 δευτερόλεπτα.

Η καθυστέρηση αυτή οφείλεται κυρίως στη διαδικασία μετάφρασης. Όπως ορίζεται στο Prompt του συστήματος ("When the user asks a question in a language other than English, you must translate the query to English before using any of the tools"), το μοντέλο πρέπει πρώτα να μεταφράσει το ελληνικό ερώτημα για να καλέσει τα εργαλεία (RAG) και στη συνέχεια να συνθέσει την απάντηση στα Ελληνικά. Αυτό το επιπλέον βήμα "σκέψης" προσθέτει χρόνο, χωρίς όμως να μειώνει την ποιότητα της τελικής απάντησης.

5.5 Σύνοψη 5ου Κεφαλαίου

Συνοψίζοντας, η διαδικασία αξιολόγησης απέδειξε ότι το Chatbot αποτελεί μια αξιόπιστη και αποδοτική λύση. Η χρήση του DeepSeek ως κριτή παρείχε αντικειμενικά δεδομένα που πιστοποιούν την ακρίβεια και την ποιότητα των απαντήσεων.

Τα βασικά συμπεράσματα συνοψίζονται στα εξής:

- Υψηλή Ακρίβεια: Το σύστημα παρέχει υψηλά επίπεδα εγκυρότητας, υπερκαλύπτοντας την αρχική προδιαγραφή του 80%.
- Εξαιρετική Εμπειρία Χρήστη: Η πειστικότητα και η ευγένεια των απαντήσεων προσομοιάζουν ανθρώπινη εξυπηρέτηση.
- Βελτιστοποιημένη Απόδοση: Η τήρηση του ορίου των 15 δευτερολέπτων κατά μέσο όρο και η έξυπνη χρήση των "Good Examples" εξασφαλίζουν γρήγορη απόκριση, καθιστώντας το σύστημα έτοιμο για πραγματική χρήση.

Η επιτυχής ολοκλήρωση των δοκιμών σε 30 διαφορετικά σενάρια εγγυάται ότι το chatbot μπορεί να διαχειριστεί αποτελεσματικά την πλειονότητα των ερωτημάτων που θα δεχθεί σε ένα πραγματικό ξενοδοχειακό περιβάλλον.

6. Συμπεράσματα και Μελλοντικές Επεκτάσεις

6.1 Ανάλυση SWOT

Στην παρούσα ενότητα πραγματοποιείται μια στρατηγική ανάλυση του συστήματος μέσω της ανάλυσης SWOT (Strengths, Weaknesses, Opportunities, Threats). Η ανάλυση αυτή επιτρέπει την αξιολόγηση των εσωτερικών δυνατοτήτων και αδυναμιών της εφαρμογής, καθώς και των εξωτερικών παραγόντων που μπορούν να επηρεάσουν τη μελλοντική της πορεία.

Δυνατά Σημεία (Strengths)

- Αρχιτεκτονική RAG (Retrieval-Augmented Generation): Η χρήση της αρχιτεκτονικής RAG αποτελεί το σημαντικότερο πλεονέκτημα, καθώς επιτρέπει στο chatbot να παρέχει τεκμηριωμένες απαντήσεις βασισμένες αποκλειστικά στα δεδομένα του ξενοδοχείου, ελαχιστοποιώντας δραστικά το φαινόμενο των "παραισθήσεων" (hallucinations).
- Υβριδική Διαχείριση Δεδομένων: Το σύστημα συνδυάζει αποτελεσματικά διανυσματικές βάσεις δεδομένων (FAISS) για σημασιολογική αναζήτηση σε FAQs και κριτικές, με σχεσιακές βάσεις δεδομένων (SQLite) για ακριβή διαχείριση διαθεσιμότητας δωματίων.
- Δυναμική Χρήση Παραδειγμάτων (Good examples): Η ενσωμάτωση του good_examples_vector_store επιταχύνει την απόκριση και βελτιώνει την ποιότητα των απαντήσεων μέσω της παροχής πρότυπων συνομιλιών στο μοντέλο.
- Πολυγλωσσική Υποστήριξη: Η ικανότητα επικοινωνίας τόσο στα Ελληνικά όσο και στα Αγγλικά διευρύνει το κοινό στο οποίο απευθύνεται η λύση.

Αδυναμίες (Weaknesses)

- Χρόνος Απόκρισης (Latency): Παρατηρείται αυξημένη καθυστέρηση στις ερωτήσεις στην ελληνική γλώσσα λόγω του επιπλέον βήματος μετάφρασης και της σύνθετης λογικής ReAct, γεγονός που μπορεί να επηρεάσει την εμπειρία χρήστη σε περιβάλλοντα υψηλής κίνησης.
- Εξάρτηση από Εξωτερικά APIs: Η λειτουργία του συστήματος βασίζεται εξ ολοκλήρου στη διαθεσιμότητα και την απόδοση εξωτερικών παρόχων LLM

(DeepSeek), καθιστώντας το ευάλωτο σε πιθανές διακοπές υπηρεσιών ή αλλαγές πολιτικής των παρόχων καθώς και αυξημένου κόστους σε περίπτωση scaling.

- Χειροκίνητη Ενημέρωση Βάσης Γνώσης: Η προσθήκη νέων πληροφοριών (π.χ. νέα FAQs ή σημεία ενδιαφέροντος) απαιτεί την εκτέλεση scripts για την εκ νέου δημιουργία των vector stores, αντί για μια αυτοματοποιημένη διαδικασία συγχρονισμού.

Ευκαιρίες (Opportunities)

- Διασύνδεση με Συστήματα Κρατήσεων (Booking Engines): Η εφαρμογή μπορεί να εξελιχθεί από ένα πληροφοριακό σύστημα σε ένα πλήρες εργαλείο συναλλαγών μέσω της ενσωμάτωσης APIs για την πραγματοποίηση κρατήσεων σε πραγματικό χρόνο.
- Ενσωμάτωση σε Πλατφόρμες Κοινωνικής Δικτύωσης: Η αρχιτεκτονική επιτρέπει την εύκολη μεταφορά του chatbot σε δημοφιλείς πλατφόρμες όπως το WhatsApp, το Messenger ή το Viber, αυξάνοντας την προσβασιμότητα για τους πελάτες.

Απειλές (Threats)

- Ασφάλεια και Προστασία Δεδομένων: Η καταγραφή και αποθήκευση συνομιλιών χρήζει προσεκτικής διαχείρισης για τη συμμόρφωση με τον Γενικό Κανονισμό Προστασίας Δεδομένων, καθώς τυχόν διαρροή πληροφοριών αποτελεί κρίσιμο κίνδυνο.
- Ανταγωνισμός: Η ραγδαία εξέλιξη της τεχνητής νοημοσύνης οδηγεί στην εμφάνιση πολλών έτοιμων εμπορικών λύσεων, γεγονός που απαιτεί συνεχή αναβάθμιση και εξειδίκευση της παρούσας υλοποίησης.
- Μεταβολή Κόστους API: Μια πιθανή αύξηση της τιμολόγησης των μοντέλων DeepSeek θα μπορούσε να καταστήσει τη λειτουργία του συστήματος ασύμφορη για μικρομεσαίες ξενοδοχειακές μονάδες.

6.2 Μελλοντικές βελτιώσεις και επεκτάσεις

Η παρούσα υλοποίηση αποτελεί μια ισχυρή βάση για την εφαρμογή της αρχιτεκτονικής RAG στον ξενοδοχειακό κλάδο. Ωστόσο, η ραγδαία εξέλιξη των γλωσσικών μοντέλων και των τεχνολογιών τεχνητής νοημοσύνης ανοίγει νέους δρόμους

για περαιτέρω αναβάθμιση. Οι προτεινόμενες μελλοντικές βελτιώσεις εστιάζουν στους εξής άξονες:

- Μείωση του Latency με Streaming: Μια σημαντική βελτίωση για την εμπειρία του χρήστη θα ήταν η υλοποίηση "Streaming" αποκρίσεων. Αντί ο χρήστης να περιμένει την ολοκλήρωση της "σκέψης" του μοντέλου (ReAct steps), η απάντηση θα εμφανίζεται σταδιακά στην οθόνη, μειώνοντας τον αντιληπτό χρόνο αναμονής.
- Υιοθέτηση Τοπικών Μοντέλων (Local LLMs): Για την ενίσχυση της προστασίας των προσωπικών δεδομένων και τη μείωση του κόστους των APIs, θα μπορούσε να εξεταστεί η χρήση τοπικών μοντέλων (π.χ. Llama 3 ή Mistral). Αυτό θα επέτρεπε την εκτέλεση του chatbot σε ιδιόκτητους servers του ξενοδοχείου.
- Αυτοματοποιημένη Ενημέρωση: Η δημιουργία ενός αυτοματοποιημένου συστήματος με UI που θα αντλεί δεδομένα (π.χ από νέα reviews) και θα ενημερώνει αυτόματα τα Vector Stores, διασφαλίζοντας ότι η γνώση του chatbot είναι πάντα επίκαιρη χωρίς χειροκίνητη παρέμβαση μέσω scripts που γίνεται στη συγκεκριμένη υλοποίηση.

6.3 Συμπεράσματα

Η παρούσα πτυχιακή εργασία ολοκληρώθηκε με την επιτυχή σχεδίαση, ανάπτυξη και αξιολόγηση ενός ευφυούς συστήματος συνομιλίας (chatbot) βασισμένου στην αρχιτεκτονική Retrieval-Augmented Generation (RAG), προσαρμοσμένου στις ανάγκες του ξενοδοχειακού κλάδου. Μέσα από τη διαδικασία υλοποίησης και τις δοκιμές που ακολούθησαν, προέκυψαν ορισμένα συμπεράσματα:

Πρώτον, η αρχιτεκτονική RAG αποδείχθηκε η πλέον ενδεδειγμένη λύση για επιχειρησιακές εφαρμογές όπου η ακρίβεια της πληροφορίας είναι κρίσιμη. Ο συνδυασμός της δημιουργικής ικανότητας του μοντέλου DeepSeek με την αυστηρή ανάκτηση δεδομένων από τις διανυσματικές βάσεις (FAISS) και τη σχεσιακή βάση (SQLite), επέτρεψε τη δημιουργία ενός συστήματος που παρέχει έγκυρες και τεκμηριωμένες απαντήσεις, εξαλείφοντας σχεδόν πλήρως τις "παραισθήσεις" των παραδοσιακών γλωσσικών μοντέλων.

Δεύτερον, η αξιολόγηση έδειξε ότι το σύστημα δεν περιορίζεται σε μια απλή παράθεση πληροφοριών, αλλά υιοθετεί ένα πειστικό και φιλόξενο ύφος (Persuasiveness 5/5), το οποίο προσομοιάζει την ανθρώπινη εξυπηρέτηση υψηλού επιπέδου.

Τρίτον, η επιτυχής τήρηση των προδιαγραφών για ακρίβεια άνω του 80% και μέσο χρόνο απόκρισης εντός των ορίων (14.51s), επιβεβαιώνει ότι οι σύγχρονες τεχνολογίες τεχνητής νοημοσύνης είναι πλέον ώριμες για να ενσωματωθούν στην καθημερινή λειτουργία των επιχειρήσεων. Το σύστημα που αναπτύχθηκε προσφέρει μια ολοκληρωμένη λύση που μειώνει το λειτουργικό κόστος της υποδοχής, αυξάνει την ταχύτητα εξυπηρέτησης και ενισχύει τη συνολική εμπειρία του επισκέπτη.

6.4 Σύνοψη 6ου κεφαλαίου

Στο έκτο κεφάλαιο πραγματοποιήθηκε η τελική αποτίμηση της πτυχιακής εργασίας. Μέσω της ανάλυσης SWOT εντοπίστηκαν τα πλεονεκτήματα της αρχιτεκτονικής RAG και οι προκλήσεις που αφορούν την ταχύτητα απόκρισης. Στη συνέχεια, παρατέθηκαν συγκεκριμένες προτάσεις για τη μελλοντική εξέλιξη του συστήματος σε ένα πλήρως εργαλείο κρατήσεων. Η εργασία ολοκληρώθηκε με τα τελικά συμπεράσματα, τα οποία επιβεβαιώνουν την αποτελεσματικότητα της προτεινόμενης λύσης ως ένα έτοιμο προς χρήση, ευφυές σύστημα εξυπηρέτησης πελατών, θέτοντας τις βάσεις για ένα πιο επαγγελματικό, πλήρες εργαλείο έτοιμο προς χρήση

Βιβλιογραφία

Ακολουθούν οι βιβλιογραφικές αναφορές (πηγές) της Εργασίας.

[1] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," *Multimed. Tools Appl.*, vol. 82, pp. 3713–3744, Nov. 2022, doi: 10.1007/s11042-022-13428-4.

[2] S. Gupta, R. Ranjan, and S. N. Singh, "A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions," 2024. [Online]. Available: https://arxiv.org/abs/2410.12837

[3] E. Susanto and Z. Khaq, "Enhancing Customer Service Efficiency in Start-Ups with AI: A Focus on Personalization and Cost Reduction," *Journal of Management and Informatics*, vol. 3, pp. 267–281, Sep. 2024, doi: 10.51903/jmi.v3i2.34.

[4] S. Athikkal and J. Jenq, "Voice Chatbot for Hospitality," 2022. [Online]. Available: https://arxiv.org/abs/2208.10926

[5] B. Li, N. Jiang, J. Sham, H. Shi, and H. Fazal, "Real-world Conversational AI for Hotel Bookings," *Proceedings - 2019 2nd International Conference on Artificial Intelligence for Industries, AI4I 2019*, 2019, [Online]. Available: https://arxiv.org/abs/1908.10001

[6] I. S. Mohamed, M. Abdelsalam, and I. F. Moawad, "Support Chatbot based on ChatGPT for Customer Relationship Management in Educational Institutions," in *2024 6th International Conference on Computing and Informatics (ICCI)*, 2024, pp. 351–355. doi: 10.1109/ICCI61671.2024.10485073.

[7] D. Dobrev, "A Definition of Artificial Intelligence," *Mathematica Balkanica, New Series, Vol. 19, 2005, Fasc. 1-2, pp.67-74*, p. arXiv:1210.1568, Oct. 2012, doi: 10.48550/arXiv.1210.1568.

[8] A. M. Turing, "Computing Machinery and Intelligence," *Mind, New Series, Vol. 59, No. 236 (Oct., 1950), pp. 433-460*, 1950, Accessed: Sep. 22, 2025. [Online]. Available: http://thatmarcusfamily.org/philosophy/Course_Websites/Readings/Turing%20-%20Computing%20Machinery%20and%20Intelligence.pdf

[9] C. R. Jones and B. K. Bergen, "Large Language Models Pass the Turing Test," *ArXiv*, 2025, [Online]. Available: https://arxiv.org/abs/2503.23674

[10] E. Adamopoulou and L. Moussiades, "An Overview of Chatbot Technology," *IFIP Adv. Inf. Commun. Technol.*, pp. 373–383, 2020, doi: 10.1007/978-3-030-49186-4_31.

[11] D. Y. K. M. Omkar Singh Jyotsna Anthal, "Studying Rule-Based and Self-Learning Chatbots: A Comprehensive Literature Review ," *International Journal of Scientific Research and Engineering Development,* *8*, Jul. 2025, doi: 10.5281/zenodo.15855871.

[12] A. Heidari, N. J. Navimipour, S. Zeadally, and V. Chamola, "Everything you wanted to know about ChatGPT: Components, capabilities, applications, and opportunities," *Internet Technology Letters*, vol. 7, no. 6, p. e530, 2024, doi: https://doi.org/10.1002/itl2.530.

[13] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval Augmentation Reduces Hallucination in Conversation," *Findings of the Association for Computational Linguistics, Findings of ACL: EMNLP 2021*, 2021, [Online]. Available: https://arxiv.org/abs/2104.07567

[14] R. M. O. Rodrigues, "Retrieval-Augmented Generative AI Chatbot," 2024. Accessed: Oct. 11, 2025. [Online]. Available: http://hdl.handle.net/10400.5/96434

[15] H.-C. Ling, K.-L. Hsiao, and W.-C. Hsu, "Can Students' Computer Programming Learning Motivation and Effectiveness Be Enhanced by Learning Python Language? A Multi-Group Analysis," *Front. Psychol.*, vol. 11, Jan. 2021, doi: 10.3389/fpsyg.2020.600814.

[16] V. Mavroudis, "LangChain v0.3," *Preprints (Basel).*, Nov. 2024, doi: 10.20944/preprints202411.0566.v1.

[17] LangChain, "LangChain official Documentation." Accessed: Nov. 07, 2025. [Online]. Available: https://docs.langchain.com

[18] T. Taipalus, "Vector database management systems: Fundamental concepts, use-cases, and current challenges," *Cogn. Syst. Res.*, vol. 85, p. 101216, 2024, doi: https://doi.org/10.1016/j.cogsys.2024.101216.

[19] M. Douze *et al.*, "The Faiss library," *IEEE Trans. Big Data*, 2025, [Online]. Available: https://arxiv.org/abs/2401.08281

[20] Y. Akkem, B. S. Kumar, and A. Varanasi, "Streamlit Application for Advanced Ensemble Learning Methods in Crop Recommendation Systems – A Review and

Implementation," *Indian J. Sci. Technol.*, vol. 16, no. 48, pp. 4688–4702, 2023, doi: 10.17485/IJST/v16i48.2850.

[21] K. P. Gaffney, M. Prammer, L. Brasfield, D. R. Hipp, D. Kennedy, and J. M. Patel, "SQLite: past, present, and future," *Proc. VLDB Endow.*, vol. 15, no. 12, pp. 3535–3547, Aug. 2022, doi: 10.14778/3554821.3554842.

[22] DB Browser for SQLite, "Official Project Documentation." Accessed: Oct. 09, 2025. [Online]. Available: https://sqlitebrowser.org

[23] DeepSeek-AI, "DeepSeek LLM: Scaling Open-Source Language Models with Longtermism," 2024. [Online]. Available: https://arxiv.org/abs/2401.02954

[24] DeepSeek-AI, "DeepSeek-V3 Technical Report," 2025. [Online]. Available: https://arxiv.org/abs/2412.19437

[25] DeepSeek-AI, "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning," *Nature*, vol. 645, no. 8081, pp. 633–638, Jan. 2025, [Online]. Available: https://arxiv.org/abs/2501.12948

[26] W. C. Choi, C. I. Chang, I. C. Choi, and L. C. Lam, "A Review of Large Language Models (LLMs) Development: A Cross-Country Comparison of the US, China, Europe, UK, India, Japan, South Korea, and Canada," *Preprints (Basel).*, Apr. 2025, doi: 10.20944/preprints202504.2136.v1.

[27] PyCharm, "PyCharm official Documentation." Accessed: Oct. 09, 2025. [Online]. Available: https://www.jetbrains.com/help/pycharm/getting-started.html

[28] Dave Bergmann, "ReAct agent overview." Accessed: Dec. 21, 2025. [Online]. Available: https://www.ibm.com/think/topics/react-agent

[29] Rahultiwari, "Unlocking the Power of Sentence Embeddings with all-MiniLM-L6-v2." Accessed: Dec. 17, 2025. [Online]. Available: https://medium.com/@rahultiwari065/unlocking-the-power-of-sentence-embeddings-with-all-minilm-l6-v2-7d6589a5f0aa

[30] HugginFace, "UnUderstanding AI Agents through the Thought-Action-Observation Cycle." Accessed: Dec. 16, 2025. [Online]. Available: https://huggingface.co/learn/agents-course/unit1/agent-steps-and-structure#understanding-ai-agents-through-the-thought-action-observation-cycle

[31] Streamlit, "Streamlit Official Documentation." Accessed: Dec. 15, 2025. [Online]. Available: https://docs.streamlit.io

[32] "LLM-as-a-judge: a complete guide to using LLMs for evaluations." Accessed: Jan. 18, 2026. [Online]. Available: https://www.evidentlyai.com/llm-guide/llm-as-a-judge

[33] LangChain, "Evaluate a RAG application," LangChain Docs. [Online]. Available: https://docs.langchain.com/langsmith/evaluate-rag-tutorial. [Accessed: 22-Jan-2026].

[34] H. Hoffman, "Build an LLM RAG Chatbot With LangChain," Real Python. [Online]. Available: https://realpython.com/build-llm-rag-chatbot-with-langchain/. [Accessed: 22-Jan-2026].

[35] DeepSeek, "DeepSeek API Documentation," DeepSeek API Docs. [Online]. Available: https://api-docs.deepseek.com/. [Accessed: 26-Feb-2026].