Διεθνές Πανεπιστήμιο Ελλάδος
Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

WebText2SQL: Εφαρμογή ιστού για την αυτόματη παραγωγή και εκτέλεση κώδικά SQL από κείμενο με χρήση μεγάλου γλωσσικού μοντέλου

WebText2SQL

Φοιτητής:
Ηρακλής Κόνσουλας
Αριθμός Μητρώου: 185200

Επιβλέπων:
Στέφανος Ουγιάρογλου
Επίκουρος Καθηγητής

Σεπτέμβριος 2025

WebText2SQL: Εφαρμογή ιστού για την αυτόματη παραγωγή και εκτέλεση κώδικά SQL από κείμενο με χρήση μεγάλου γλωσσικού μοντέλου

Κωδικός Δ.Ε.: 25199

Ονοματεπώνυμο φοιτητή: Ηρακλής Κόνσουλας

Ονοματεπώνυμο εισηγητή: Στέφανος Ουγιάρογλου

Ημερομηνία ανάληψης: 23-03-2025

Ημερομηνία περάτωσης: 12-09-2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Αφιέρωση

Θα ήθελα να ευχαριστήσω τον επιβλέποντα της διπλωματικής μου, Επίκουρο Καθηγητή κ. Στέφανο Ουγιάρογλου, για την καθοδήγηση και την υποστήριξή του σε όλα τα στάδια της εργασίας.

Ευχαριστώ επίσης την οικογένειά μου για τη διαρκή στήριξη όλα αυτά τα χρόνια.

Ιδιαίτερες ευχαριστίες στους φίλους μου, Στέφανο Μ., Στέφανο Γ., Θανάση, Σάκη, Θάνο, Γιάννη και Ειρήνη, για την παρέα, τη βοήθεια και την ψυχραιμία που μου μετέδωσαν όταν τη χρειαζόμουν.

Abstract

The proliferation of data has underscored the necessity for accessible database interaction, yet the complexity of SQL remains a significant barrier for non-technical users. This thesis confronts this challenge through the design, implementation, and presentation of WebText2SQL, a comprehensive web application that empowers users to query databases using natural language. The project is motivated by the principle of data democratization, aiming to provide an intuitive, stateful, and dialect-aware tool that abstracts away SQL's intricacies.

The primary contribution is the WebText2SQL application, a robust platform built on a modern Python-based stack featuring FastAPI for the backend API, Chainlit for the interactive chat interface, and SQLModel for data persistence. The system's architecture is designed for modularity and extensibility, with distinct controllers for user authentication, connection management, and AI interaction. A key innovation is its factory pattern for database connections, enabling native support for both PostgreSQL and MySQL dialects and ensuring the generation of syntactically correct and optimized queries. Secure connections are facilitated via direct TCP/IP or SSH tunneling.

Functionally, WebText2SQL provides a complete user workflow, from secure registration and login to the management of multiple saved database connections. Users interact with the system through a conversational UI where they can select a database schema and pose questions in natural language. The application leverages the GPT-4o-mini large language model, providing it with rich context, including database schema and conversation history, to translate user intent into precise SQL queries. The results are presented in a user-friendly tabular format with an option to download the full dataset as a CSV file.

This thesis documents the entire lifecycle of the project, from the definition of functional requirements and architectural design to the detailed presentation of the final application. It concludes by summarizing the project's success in creating a practical and extensible tool for natural language database querying and outlines potential future enhancements, including support for more database systems and integrated data visualization capabilities.

Περίληψη

Η διάδοση των δεδομένων έχει αναδείξει την ανάγκη για προσβάσιμη αλληλεπίδραση με βάσεις δεδομένων, ωστόσο η πολυπλοκότητα της SQL παραμένει ένα σημαντικό εμπόδιο για τους μη τεχνικούς χρήστες. Η παρούσα διπλωματική εργασία αντιμετωπίζει αυτή την πρόκληση μέσω του σχεδιασμού, της υλοποίησης και της παρουσίασης του WebText2SQL, μιας ολοκληρωμένης διαδικτυακής εφαρμογής που δίνει τη δυνατότητα στους χρήστες να υποβάλλουν ερωτήματα σε βάσεις δεδομένων χρησιμοποιώντας φυσική γλώσσα. Το έργο ωθείται από την αρχή του εκδημοκρατισμού των δεδομένων, με στόχο την παροχή ενός διαισθητικού, stateful εργαλείου, ενήμερου για τις SQL διαλέκτους, που αφαιρεί την περιπλοκότητα της SQL.

Η κύρια συνεισφορά είναι η εφαρμογή WebText2SQL, μια πλήρης πλατφόρμα χτισμένη σε μια σύγχρονη στοίβα τεχνολογιών βασισμένη στην Python, που περιλαμβάνει το FastAPI για το backend API, το Chainlit για τη διαδραστική διεπαφή συνομιλίας και το SQLModel για την αποθήκευση δεδομένων. Η αρχιτεκτονική του συστήματος είναι σχεδιασμένη για modular δομή και επεκτασιμότητα, με ξεχωριστούς ελεγκτές για την αυθεντικοποίηση χρηστών, τη διαχείριση συνδέσεων και την αλληλεπίδραση με την Τεχνητή Νοημοσύνη. Μια βασική καινοτομία είναι το πρότυπο factory για τις συνδέσεις βάσεων δεδομένων, που επιτρέπει την εγγενή υποστήριξη για τις διαλέκτους PostgreSQL και MySQL και διασφαλίζει την παραγωγή συντακτικά ορθών και βελτιστοποιημένων ερωτημάτων. Οι συνδέσεις διευκολύνονται μέσω απευθείας TCP/IP ή ασφαλούς SSH tunneling.

Λειτουργικά, το WebText2SQL παρέχει μια πλήρη ροή εργασίας για τον χρήστη, από την ασφαλή εγγραφή και σύνδεση έως τη διαχείριση πολλαπλών αποθηκευμένων συνδέσεων βάσεων δεδομένων. Οι χρήστες αλληλεπιδρούν με το σύστημα μέσω μιας συνομιλιακής διεπαφής όπου μπορούν να επιλέξουν ένα σχήμα βάσης δεδομένων και να θέσουν ερωτήσεις σε φυσική γλώσσα. Η εφαρμογή αξιοποιεί το μεγάλο γλωσσικό μοντέλο GPT-4o-mini, παρέχοντάς του πλούσιο πλαίσιο, συμπεριλαμβανομένου του σχήματος της βάσης δεδομένων και του ιστορικού της συνομιλίας, για να μεταφράσει την πρόθεση του χρήστη σε ακριβή ερωτήματα SQL. Τα αποτελέσματα παρουσιάζονται σε μια φιλική προς τον χρήστη μορφή πίνακα με δυνατότητα λήψης του πλήρους συνόλου δεδομένων ως αρχείο CSV.

Αυτή η διπλωματική εργασία τεκμηριώνει ολόκληρο τον κύκλο ζωής του έργου, από τον ορισμό των λειτουργικών απαιτήσεων και τον αρχιτεκτονικό σχεδιασμό έως τη λεπτομερή παρουσίαση της τελικής εφαρμογής. Ολοκληρώνεται συνοψίζοντας την επιτυχία του έργου στη δημιουργία ενός πρακτικού και επεκτάσιμου εργαλείου για την υποβολή ερωτημάτων σε βάσεις δεδομένων με φυσική γλώσσα και περιγράφει πιθανές μελλοντικές βελτιώσεις, συμπεριλαμβανομένης της υποστήριξης για περισσότερα συστήματα βάσεων δεδομένων και ενσωματωμένων δυνατοτήτων οπτικοποίησης δεδομένων.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Η γλώσσα SQL

Η Structured Query Language (SQL) αποτελεί τη γλώσσα αναφοράς για τη διαχείριση σχεσιακών βάσεων δεδομένων, έχοντας καθιερωθεί ως το πιο ευρέως χρησιμοποιούμενο εργαλείο για την ανάκτηση, τον χειρισμό και τη διαχείριση δεδομένων σε επιχειρηματικά και ακαδημαϊκά περιβάλλοντα. Η ιστορική της εξέλιξη ξεκινά από τη δεκαετία του 1970 στα ερευνητικά εργαστήρια της IBM, όπου αναπτύχθηκε αρχικά ως SEQUEL (Structured English Query Language)[1]. Αυτή η ανάπτυξη στηρίχθηκε στις θεμελιώδεις αρχές του σχεσιακού μοντέλου που είχε διατυπώσει ο Edgar F. Codd[2], οι οποίες επανακαθόρισαν τον τρόπο οργάνωσης και πρόσβασης στα δεδομένα.
Το βασικό πλεονέκτημα της SQL έγκειται στη δηλωτική (declarative) φύση της, η οποία επιτρέπει στους χρήστες να καθορίζουν το επιθυμητό αποτέλεσμα χωρίς να χρειάζεται να περιγράφουν τον ακριβή αλγόριθμο για την επίτευξή του. Αυτή η προσέγγιση καθιστά την SQL ταυτόχρονα ισχυρή και προσβάσιμη, καθώς αφαιρεί την πολυπλοκότητα των διαδικαστικών λεπτομερειών και επιτρέπει στους χρήστες να εστιάσουν στη λογική των δεδομένων που επιθυμούν να ανακτήσουν ή να επεξεργαστούν.
Η δομή της SQL οργανώνεται γύρω από τέσσερις κύριες κατηγορίες εντολών που καλύπτουν το πλήρες φάσμα των λειτουργιών βάσεων δεδομένων. Αυτές περιλαμβάνουν τις εντολές Data Manipulation Language (DML) για την εισαγωγή, ενημέρωση, διαγραφή και ανάκτηση δεδομένων (π.χ., INSERT, UPDATE, DELETE, SELECT), Data Definition Language (DDL) για τον ορισμό και τη διαχείριση της δομής των βάσεων δεδομένων (π.χ., CREATE, ALTER, DROP), Data Control Language (DCL) για τη διαχείριση δικαιωμάτων πρόσβασης (π.χ., GRANT, REVOKE) και Transaction Control Language (TCL) για τη διαχείριση συναλλαγών (π.χ., COMMIT, ROLLBACK).
Η τυποποίηση της SQL από τον ANSI το 1982 και αργότερα από τον ISO το 1987 αποτέλεσε σημαντικό ορόσημο, καθώς κατέστησε τη γλώσσα ανεξάρτητη από συγκεκριμένους προμηθευτές και διασφάλισε βασικό επίπεδο συμβατότητας μεταξύ διαφορετικών συστημάτων. Οι μεταγενέστερες εκδόσεις, συμπεριλαμβανομένης της SQL:1999[3], εισήγαγαν προηγμένα χαρακτηριστικά όπως τα αναδρομικά ερωτήματα, στοιχεία αντικειμενοστρεφούς προγραμματισμού και νέα κατηγορήματα (predicates) που βελτίωσαν περαιτέρω τη λειτουργικότητα της SQL (π.χ., DISTINCT, UNIQUE).
Παρά την ύπαρξη των προτύπων, η πραγματικότητα των σύγχρονων συστημάτων διαχείρισης βάσεων δεδομένων χαρακτηρίζεται από την ποικιλομορφία των διαλέκτων SQL[4]. Συστήματα όπως τα MySQL/MariaDB, PostgreSQL, Oracle Database και Microsoft SQL Server έχουν αναπτύξει τις δικές τους επεκτάσεις και παραλλαγές, προσθέτοντας εξειδικευμένες συναρτήσεις, τύπους δε-

δομένων και συντακτικά στοιχεία που βελτιστοποιούν την απόδοση ή προσφέρουν νέες δυνατότητες. Αυτή η διαφοροποίηση, αν και ευνοεί την καινοτομία, δημιουργεί σημαντικές προκλήσεις στη φορητότητα του κώδικα και την ανάπτυξη εργαλείων που πρέπει να υποστηρίζουν πολλαπλά συστήματα.
Η πρόκληση της συμβατότητας διαλέκτων αποκτά ιδιαίτερη σημασία στο πλαίσιο των σύγχρονων εργαλείων μετατροπής φυσικής γλώσσας σε SQL. Ένα αποτελεσματικό σύστημα πρέπει να είναι ικανό να παράγει όχι μόνο συντακτικά σωστά, αλλά και βελτιστοποιημένα ερωτήματα για το συγκεκριμένο σύστημα διαχείρισης βάσεων δεδομένων που χρησιμοποιείται.

1.2 Μετατροπή κειμένου σε κώδικα SQL

Η μετατροπή φυσικής γλώσσας σε SQL ερωτήματα (Text-to-SQL) αποτελεί ένα από τα πιο δύσκολα προβλήματα στη τομή της επεξεργασίας φυσικής γλώσσας και των συστημάτων βάσεων δεδομένων. Ο στόχος αυτής της διαδικασίας είναι η αυτόματη μετάφραση ερωτήσεων που διατυπώνονται σε φυσική γλώσσα σε εκτελέσιμα SQL ερωτήματα, επιτρέποντας έτσι σε χρήστες χωρίς εξειδικευμένες τεχνικές γνώσεις να αλληλεπιδρούν άμεσα με βάσεις δεδομένων μέσω Διεπαφών Φυσικής Γλώσσας προς Βάσεις Δεδομένων (Natural Language Interfaces to Databases - NLIDBs)[5][6].
Η αρχιτεκτονική ενός Text-to-SQL συστήματος περιλαμβάνει τρία κύρια συστατικά που πρέπει να λειτουργούν συγχρονισμένα[7]. Το πρώτο είναι η κατανόηση φυσικής γλώσσας (Natural Language Understanding), που περιλαμβάνει την ανάλυση της συντακτικής δομής και την εξαγωγή της σημασιολογικής πρόθεσης. Το δεύτερο αφορά στη σύνδεση σχήματος (Schema Linking), η οποία αντιστοιχίζει τους όρους της φυσικής γλώσσας με τα στοιχεία του σχήματος της βάσης δεδομένων. Το τρίτο είναι η παραγωγή SQL ερωτημάτων (SQL Query Generation), που συνθέτει τις πληροφορίες για να δημιουργήσει συντακτικά σωστά και σημασιολογικά ακριβή ερωτήματα.
Η πολυπλοκότητα του προβλήματος εντείνεται όταν εξετάζουμε τους διαφορετικούς τύπους ερωτημάτων που πρέπει να υποστηρίζει ένα Text-to-SQL σύστημα. Ενώ τα απλά ερωτήματα SELECT αποτελούν την πιο στοιχειώδη περίπτωση, τα πραγματικά σενάρια συχνά απαιτούν πολύπλοκες ζευξεις μεταξύ πολλαπλών πινάκων και εμφωλευμένα ερωτήματα με υπο-ερωτήματα σε διαφορετικά επίπεδα βάθους. Οι αμφισημίες που προκύπτουν από τη φύση της φυσικής γλώσσας αποτελούν επιπλέον σημαντικό παράγοντα πολυπλοκότητας, καθώς μια ερώτηση μπορεί να ερμηνευθεί με πολλαπλούς τρόπους.
Η εξέλιξη των Text-to-SQL συστημάτων έχει διανύσει μια ενδιαφέρουσα πορεία από τις πρώτες προσεγγίσεις που βασίζονταν σε κανόνες έως τις μεθόδους που βασίζονται σε δεδομένα[8] και πλέον τις σύγχρονες τεχνικές βαθιάς μάθησης. Οι αρχικές μέθοδοι, αν και αποτελεσματικές σε περιορισμένα πεδία, αντιμετώπιζαν σοβαρά προβλήματα κλιμάκωσης και γενίκευσης. Η εμφάνιση των μεθόδων μηχανικής μάθησης[9] και των προηγμένων τεχνικών σημασιολογικής ανάλυσης[10] αποτέλεσε σημαντικό βήμα προς την αυτοματοποιημένη κατανόηση και μετατροπή φυσικής γλώσσας.

1.3 Μεγάλα Γλωσσικά Μοντέλα και Μετατροπή κειμένου σε κώδικα SQL

Η εμφάνιση των Μεγάλων Γλωσσικών Μοντέλων (Large Language Models - LLMs) έχει φέρει μια επαναστατική αλλαγή στο τοπίο των συστημάτων Text-to-SQL, σηματοδοτώντας μια θεμελιώδη μετατόπιση από την εκπαίδευση εξειδικευμένων μοντέλων από το μηδέν σε προσεγγίσεις που αξιοποιούν ισχυρά προεκπαιδευμένα μοντέλα. Αυτά τα μοντέλα, που βασίζονται στην αρχιτεκτονική του Transformer, έχουν εκπαιδευτεί σε τεράστιες ποσότητες κειμένου από το διαδίκτυο, αποκτώντας εκτεταμένη γνώση για τη σύνταξη SQL, τις έννοιες βάσεων δεδομένων και τις σχέσεις μεταξύ φυσικής γλώσσας και τυπικών γλωσσών ερωτημάτων.

Το κομβικό σημείο αυτής της εξέλιξης ήρθε με την εισαγωγή του GPT-3 και την ανακάλυψη της ικανότητας των μεγάλων γλωσσικών μοντέλων για few-shot learning[11]. Αυτή η ικανότητα επέτρεψε στα μοντέλα να επιδείξουν εντυπωσιακή απόδοση σε Text-to-SQL εργασίες χωρίς εξειδικευμένη εκπαίδευση μέσω προσεκτικά σχεδιασμένων prompts που περιλαμβάνουν παραδείγματα και οδηγίες. Αυτή η προσέγγιση αντιπροσωπεύει μια ριζική αλλαγή από τις παραδοσιακές μεθόδους που απαιτούσαν εκτεταμένα σύνολα δεδομένων εκπαίδευσης και επαναπροσαρμογή για κάθε νέα βάση δεδομένων.

Τα σύγχρονα LLMs όπως το GPT-4[12] έχουν αποδείξει ότι μπορούν να χειρίζονται αποτελεσματικά πολλές από τις κλασικές προκλήσεις του Text-to-SQL προβλήματος. Η πλούσια γλωσσική τους κατανόηση επιτρέπει την επίλυση αμφισημιών, ενώ η εκτεταμένη γνώση τους για SQL σύνταξη διευκολύνει την παραγωγή συντακτικά σωστών ερωτημάτων. Ιδιαίτερα σημαντική είναι η ικανότητά τους να προσαρμόζονται σε διαφορετικές SQL διαλέκτους[4] και σχήματα βάσεων δεδομένων χωρίς την ανάγκη επαναπροσαρμογής, καθιστώντας τα ιδανικά για εφαρμογές που πρέπει να υποστηρίζουν πολλαπλά συστήματα διαχείρισης βάσεων δεδομένων.

Οι βασικές τεχνικές που έχουν αναδειχθεί για την αξιοποίηση των LLMs[13][14] στο Text-to-SQL περιλαμβάνουν το zero-shot prompting, όπου το μοντέλο παράγει SQL ερωτήματα βασισμένο μόνο στο σχήμα της βάσης και την ερώτηση του χρήστη, το few-shot prompting[11] που ενσωματώνει παραδείγματα για καθοδήγηση, και το chain-of-thought reasoning[15] που ενθαρρύνει το μοντέλο να αναλύει τα βήματα που απαιτούνται για την επίλυση πολύπλοκων ερωτημάτων[16][17]. Αυτές οι τεχνικές έχουν αποδειχθεί ιδιαίτερα αποτελεσματικές στην αντιμετώπιση ερωτημάτων που περιλαμβάνουν πολύπλοκες ζεύξεις και εμφωλευμένη λογική.

Η επίδραση των LLMs στον τομέα έχει επιβεβαιωθεί μέσω εκτεταμένων αξιολογήσεων[18] που δείχνουν ότι τα σύγχρονα μοντέλα μπορούν να επιτύχουν απόδοση συγκρίσιμη ή ακόμα και ανώτερη από εξειδικευμένα νευρωνικά δίκτυα σε πολλά benchmarks. Ωστόσο, παραμένουν σημαντικές προκλήσεις[19] όπως η τάση για "παραισθήσεις"[20], το υψηλό κόστος υπολογισμού, και οι περιορισμοί του παραθύρου συμφραζομένων[21] που μπορεί να περιορίζουν την αποτελεσματικότητα σε πολύ μεγάλα σχήματα βάσεων δεδομένων.

Παρά αυτούς τους περιορισμούς, η ευελιξία και η προσαρμοστικότητα των LLMs τα καθιστούν την επικρατούσα προσέγγιση για σύγχρονα Text-to-SQL συστήματα. Η δυνατότητά τους να λειτουργούν αποτελεσματικά χωρίς εξειδικευμένη εκπαίδευση, σε συνδυασμό με την ικανότητά τους να κατανοούν και να παράγουν κώδικα σε διαφορετικές SQL διαλέκτους, τα καθιστά ιδανικά για την ανάπτυξη πρακτικών εφαρμογών που στοχεύουν στη διευκόλυνση της πρόσβασης σε δεδομένα. Αυτή η εξέλιξη έχει δημιουργήσει νέες ευκαιρίες για την ανάπτυξη εργαλείων όπως το WebText2SQL, που μπορούν να αξιοποιήσουν τη δύναμη των LLMs για να προσφέρουν εύχρηστες διεπαφές για

την αλληλεπίδραση με βάσεις δεδομένων σε πραγματικές συνθήκες χρήσης.

1.4 Κίνητρο

Η ανάγκη για άμεση πρόσβαση σε δεδομένα από βάσεις δεδομένων και η ανάλυση τους έχει οδηγήσει στην ανάπτυξη εργαλείων που διευκολύνουν τη μετατροπή φυσικής γλώσσας σε SQL ερωτήματα. Ένα βασικό πρόβλημα είναι η ποικιλομορφία των διαλέκτων SQL[22][4]. Πολλά εργαλεία είναι σχεδιασμένα για γενική χρήση ή είναι εκπαιδευμένα σε συγκεκριμένες διαλέκτους, όπως SQLite, χωρίς να λαμβάνουν υπόψη τις συντακτικές λεπτομέρειες ή τις βελτιστοποιημένες συναρτήσεις που προσφέρουν άλλα ΣΔΒΔ, όπως MySQL/MariaDB και PostgreSQL[23]. Αυτή η έλλειψη προσαρμοστικότητας μπορεί να οδηγήσει παραγωγή μη βέλτιστων ερωτημάτων ή ακόμα και σε σφάλματα κατά την εκτέλεση, όταν ένα τέτοιο εργαλείο χρησιμοποιείται σε διαφορετικό ΣΔΒΔ από αυτό για το οποίο έχει εκπαιδευτεί.
Επιπλέον, πολλές τέτοιες πλατφόρμες μετατροπής φυσικού κειμένου σε SQL παραμένουν δυσπρόσιτες για το ευρύ κοινό και τους μη τεχνικούς χρήστες. Η εγκατάσταση και η χρήση τους απαιτεί συχνά τεχνικές γνώσεις, διότι οι περισσότερες από αυτές είναι σχεδιασμένες για χρήση σε τερματικό ή περίπλοκα περιβάλλοντα ανάπτυξης. Αυτό καθιστά εμπόδιο για την ευρεία υιοθέτηση αυτών των εργαλείων από αναλυτές δεδομένων, επιχειρηματίες ή ερευνητές που, παρ' ότι δεν έχουν προγραμματιστικές γνώσεις, θα μπορούσαν να επωφεληθούν από τη δυνατότητα άμεσης αλληλεπίδρασης με τα δεδομένα τους μέσω φυσικής γλώσσας.
Ένας βασικός στόχος των Διεπαφών Φυσικής Γλώσσας προς Βασεις Δεδομένων (NLIDBs) [5] ήταν πάντα να ενισχύσουν την προσβασιμότητα των δεδομένων. Αυτός ο στόχος συμβαδίζει άμεσα με την μοντέρνα τάση τού "εκδημοκρατισμού των δεδομένων" [24][25], όπου ο στόχος είναι να καταστήσει τα δεδομένα προσβάσιμα σε ένα ευρύτερο κοινό, και όχι μόνο σε πληροφορικούς ή αναλυτές δεδομένων. Με την αφαίρεση της πολυπλοκότητας της σύνταξης SQL και την παροχή ενός φιλικού προς το χρήστη περιβάλλοντος, οι χρήστες μπορούν να επικεντρωθούν στην πληροφορία που θέλουν να εξάγουν από τα δεδομένα τους, αντί να ανησυχούν για τις λεπτομέρειες της σύνταξης SQL.
Τέλος, για να είναι πραγματικά χρήσιμο ένα Text-to-SQL σύστημα, πρέπει να ξεπεράσει τα όρια ενός απλού μεταφραστή για ερωτήματα SQL. Στον πραγματικό κόσμο, η ανάλυση δεδομένων συχνά είναι μία διαδικασία επαναληπτική, όπου οι χρήστες εξερευνούν τα δεδομένα τους και αλληλεπιδρούν με πολλές πηγές δεδομένων[26]. Πολλά από τα υπάρχοντα εργαλεία δεν είναι χτισμένα με αρχιτεκτονικές που υποστηρίζουν αλλαγή κατάστασης, δεν διαχειρίζονται συνεδρίες ερωτημάτων ή δεν παρέχουν δυνατότητες αποθήκευσης και συνδέσεων σε βάσεις δεδομένων, αντιθέτως είναι σχεδιασμένα για να εκτελούν μεμονωμένα ερωτήματα, συχνά, πάνω σε μία μόνο βάση δεδομένων. Αυτό δημιουργεί ένα σημαντικό κενό στην χρησιμότητα τους. Ένα μεγάλο κίνητρο για αυτό το έργο ήταν να δημιουργηθεί ένα πλήρες εργαλείο που παρέχει ένα περιβάλλον όπου η κατάσταση των ερωτημάτων και των συνδέσεων σε βάσεις δεδομένων διατηρείται, μετατρέποντας έτσι τη μετατροπή φυσικής γλώσσας σε SQL από ένα απλό εργαλείο σε μια πλήρης πλατφόρμα για εξερεύνηση και ανάλυση δεδομένων.

1.5 Συνεισφορά

Η κύρια συνεισφορά αυτής της εργασίας είναι η σχεδίαση, υλοποίηση και παρουσίαση της εφαρμογής WebText2SQL, η οποία είναι μια διαδικτυακή εφαρμογή που διευκολύνει την αλληλεπίδραση με βάσεις δεδομένων για μη τεχνικούς χρήστες, επιτρέποντας την μετατροπή φυσικής γλώσσας σε SQL ερωτήματα. Η εργασία αυτή προχωράει ένα βήμα παραπάνω από πολλά από τα υπάρχοντα ακαδημαϊκά εργαλεία Text-to-SQL, παραθέτοντας ένα πρακτικό, έτοιμο προς χρήση εργαλείο που λύνει βασικά προβλήματα χρησιμότητας και προσβασιμότητας άλλων αντίστοιχων εργαλείων. Η συνεισφορά της βασίζεται σε τρεις βασικούς άξονες:

- Πρώτον, το WebText2SQL παρέχει αρχιτεκτονική υποστήριξη για δύο από τις πιο δημοφιλείς διαλέκτους ανοιχτού κώδικα SQL, MySQL/MariaDB και PostgreSQL. Σε αντίθεση με άλλα εργαλεία που επικεντρώνονται σε μία μόνο διάλεκτο (συνήθως SQLite), το WebText2SQL είναι σχεδιασμένο για να είναι ενήμερο της διαλέκτου που χρησιμοποιείται στην εκάστοτε σύνδεση βάσης δεδομένων, και να παράγει ερωτήματα SQL που είναι συμβατά με αυτήν. Το back end της εφαρμογής είναι σχεδιασμένο για να περνάει όλες τις απαραίτητες πληροφορίες για την διάλεκτο SQL και την δομή της βάσης δεδομένων, στην οποία συνδέεται ο χρήστης, στο μεγάλο γλωσσικό μοντέλο (LLM) που χρησιμοποιείται για την μετατροπή φυσικής γλώσσας σε SQL. Αυτό επιτρέπει στο LLM να παράγει ερωτήματα SQL που είναι, όχι μόνο συντακτικά σωστά, αλλά και βελτιστοποιημένα για την συγκεκριμένη διάλεκτο και βάση δεδομένων.

- Δεύτερον, μια μεγάλη συνεισφορά της εφαρμογής είναι η αρχιτεκτονική της, η οποία επιτρέπει την διατήρηση της κατάστασης των ερωτημάτων και των συνδέσεων σε βάσεις δεδομένων. Αναγνωρίζοντας ότι η ανάλυση δεδομένων είναι συχνά μια επαναληπτική διαδικασία, το WebText2SQL επιτρέπει στους χρήστες να διατηρούν την κατάσταση των ερωτημάτων τους και να μπορούν να συνεχίζουν την ανάλυση τους σε διαφορετικές συνεδρίες. Παρέχει ασφαλή αυθεντικοποίηση και διαχείριση συνεδριών, επιτρέποντας στους χρήστες να αποθηκεύουν και να ανακτούν τις συνδέσεις τους σε βάσεις δεδομένων, καθώς και τα ερωτήματα που έχουν εκτελέσει. Αυτό σημαίνει ότι οι χρήστες μπορούν να επιστρέψουν σε προηγούμενα ερωτήματα και να συνεχίσουν την ανάλυση τους χωρίς να χρειάζεται να επαναλάβουν τη διαδικασία σύνδεσης και εισαγωγής διαπιστευτηρίων σύνδεσης.

- Τέλος, αυτή η εργασία βοηθάει έχοντας μια οικεία διεπαφή χρήστη, η οποία είναι σχεδιασμένη για να είναι φιλική προς το χρήστη και προσβάσιμη σε μη τεχνικούς χρήστες. Η διεπαφή είναι εμπνευσμένη από δημοφιλείς εφαρμογές, όπως το ChatGPT, και παρέχει ένα περιβάλλον όπου οι χρήστες μπορούν να εισάγουν φυσική γλώσσα και να λαμβάνουν SQL ερωτήματα ως απάντηση.

1.6 Οργάνωση της Διπλωματικής

Αυτή η διπλωματική εργασία είναι οργανωμένη σε οκτώ κεφάλαια, συμπεριλαμβανομένου αυτού της εισαγωγής.

Το **Κεφάλαιο 1** εισάγει το θέμα της μετατροπής φυσικής γλώσσας σε SQL, εξηγεί τη σημασία της γλώσσας SQL και την ανάγκη για εργαλεία που διευκολύνουν την αλληλεπίδραση με βάσεις δεδο-

μένων μέσω φυσικής γλώσσας. Επίσης, αναλύει τα προβλήματα που αντιμετωπίζουν τα υπάρχοντα εργαλεία και τονίζει την ανάγκη για ένα εργαλείο που να είναι προσβάσιμο σε μη τεχνικούς χρήστες.

Το **Κεφάλαιο 2** παρέχει μια ανασκόπηση της σχετικής βιβλιογραφίας, εστιάζοντας σε προηγούμενες εργασίες που έχουν γίνει στον τομέα της μετατροπής φυσικής γλώσσας σε SQL και των μεγάλων γλωσσικών μοντέλων. Εξερευνεί τις προκλήσεις που άντιμετωπίστηκαν στο παρελθόν μέχρι και την τρέχουσα κατάσταση με την μοντέρνα χρήση βαθείας μάθησης και μεγάλων γλωσσικών μοντέλων για την επίλυση αυτών των προβλημάτων και τα εργαλεία αξιολόγησης που χρησιμοποιούνται.

Το **Κεφάλαιο 3** εστιάζει στις τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη του WebText2SQL, συμπεριλαμβανομένων των γλωσσών προγραμματισμού, των βιβλιοθηκών και των εργαλείων που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής. Επίσης, αναλύει τις αρχιτεκτονικές που χρησιμοποιήθηκαν για την υποστήριξη της μετατροπής φυσικής γλώσσας σε SQL και της διαχείρισης των συνδέσεων σε βάσεις δεδομένων.

Το **Κεφάλαιο 4** περιγράφει τη σχεδίαση και την υλοποίηση της εφαρμογής WebText2SQL, εστιάζοντας στην αρχιτεκτονική της εφαρμογής, τις βασικές λειτουργίες και τις προκλήσεις που αντιμετωπίστηκαν κατά την ανάπτυξη. Επίσης, αναλύει τις αποφάσεις σχεδίασης που ελήφθησαν για να διασφαλιστεί η ευχρηστία και η απόδοση της εφαρμογής.

Το **Κεφάλαιο 5** παρουσιάζει την εφαρμογή WebText2SQL, εστιάζοντας στη διεπαφή χρήστη και τις βασικές λειτουργίες της εφαρμογής. Παρέχει μια λεπτομερή περιγραφή του πώς οι χρήστες μπορούν να αλληλεπιδρούν με την εφαρμογή, να εισάγουν φυσική γλώσσα και να λαμβάνουν SQL ερωτήματα ως απάντηση.

Το **Κεφάλαιο 6** TODO??? εστιάζει στην αξιολόγηση της εφαρμογής WebText2SQL, περιγράφοντας τις μεθόδους που χρησιμοποιήθηκαν για την αξιολόγηση της απόδοσης και της χρησιμότητας της εφαρμογής. Αναλύει τα αποτελέσματα των δοκιμών και τις προκλήσεις που αντιμετωπίστηκαν κατά την αξιολόγηση της εφαρμογής.

Το **Κεφάλαιο 7** εστιάζει στην εμπειρία χρήστη, περιγράφοντας τις δοκιμές που πραγματοποιήθηκαν για να αξιολογηθεί η χρηστικότητα και η ευχρηστία της εφαρμογής από τεχνικούς και μη χρήστες. Αναλύει και αξιολογεί την εμπειρία των χρηστών κατά τη χρήση της εφαρμογής μετά από την υποβολή απαντήσεων τους σε φόρμα ανατροφοδότησης.

Το **Κεφάλαιο 8** συνοψίζει τα συμπεράσματα της εργασίας και προτείνει μελλοντικές επεκτάσεις για την εφαρμογή WebText2SQL. Αναλύει τις δυνατότητες βελτίωσης της εφαρμογής και τις προοπτικές για μελλοντική έρευνα στον τομέα της μετατροπής φυσικής γλώσσας σε SQL και της αλληλεπίδρασης με βάσεις δεδομένων.

Κεφάλαιο 2

Ανασκόπηση της βιβλιογραφίας

2.1 Πρώιμα Συστήματα Διεπαφής Φυσικής Γλώσσας (NLIDBs)

Τα πρώτα συστήματα Διεπαφής Φυσικής Γλώσσας προς Βάσεις Δεδομένων (Natural Language Interfaces to Databases - NLIDBs) αναπτύχθηκαν ήδη από τη δεκαετία του 1970 και περιλάμβαναν πρωτοποριακά έργα όπως το LUNAR[27], που απαντούσε σε ερωτήσεις για σεληνιακά πετρώματα χρησιμοποιώντας συντακτικές γραμματικές, το Chat-80[28], που χειριζόταν γεωγραφικά δεδομένα μέσω Prolog, και το LIFER[29], που εισήγαγε καινοτόμες μεθόδους διαχείρισης σφαλμάτων και ανάκαμψης.
Η τεχνολογική βάση αυτών των συστημάτων στηριζόταν κυρίως σε ανθρώπινα κατασκευασμένους γλωσσικούς κανόνες (hand-crafted linguistic rules) και σημασιολογικές γραμματικές (semantic grammars), απαιτώντας εκτενή προγραμματισμό κανόνων για την ανάλυση σύνταξης, τη σημασιολογική ερμηνεία και τη μετατροπή σε ερωτήματα βάσης δεδομένων.
Όπως αναφέρει η ανασκόπηση του κυρίου Ανδρουτσόπουλου[5], τα πρώιμα NLIDBs παρουσίαζαν σημαντικούς περιορισμούς:

- Πρώτον, ήταν εξαιρετικά εύθραυστα (brittle) και δεν μπορούσαν να χειριστούν ερωτήσεις που αποκλίνουν από τους προκαθορισμένους κανόνες.
- Δεύτερον, είχαν στενή σύζευξη (tight coupling) με συγκεκριμένα σχήματα βάσεων δεδομένων, γεγονός που καθιστούσε δύσκολη την προσαρμογή τους σε νέες βάσεις δεδομένων χωρίς σημαντική εργασία από ειδικούς.
- Τέλος, η ανάπτυξη και η συντήρηση αυτών των συστημάτων απαιτούσε σημαντική προσπάθεια και εξειδίκευση, καθώς οι κανόνες έπρεπε να γραφούν και να συντηρηθούν χειροκίνητα. Οι ειδικοί έπρεπε να έχουν εκτεταμένες γνώσεις τόσο της γλωσσολογίας όσο και της βάσης δεδομένων για να διασφαλίσουν ότι το σύστημα θα μπορούσε να κατανοήσει και να απαντήσει σε ερωτήσεις με ακρίβεια.

Αυτοί οι περιορισμοί οδήγησαν στην αναζήτηση νέων προσεγγίσεων που θα ξεπερνούσαν τις θεμελιώδεις αδυναμίες των πρώιμων NLIDBs, ανοίγοντας το δρόμο για τις μεθόδους βαθιάς μάθησης που θα ακολουθούσαν.

2.2 Η Επανάσταση της Βαθιάς Μάθησης

Η επανάσταση της βαθιάς μάθησης άλλαξε ριζικά το τοπίο των συστημάτων Text-to-SQL, μετατοπίζοντας την εστίαση από τα ανθρώπινα κατασκευασμένα γλωσσικά μοντέλα σε μοντέλα που βασίζονται σε δεδομένα. Αυτή η μετάβαση σηματοδότησε την κίνηση προς συστήματα που μπορούσαν να μάθουν αυτόματα τον πολύπλοκο χάρτη από τη φυσική γλώσσα στη SQL, καθιστώντας τα πιο ανθεκτικά και προσαρμοστικά σε διάφορους τομείς.

2.2.1 Η Εμφάνιση των Σύνολων Δεδομένων Μεγάλης Κλίμακας

Καταλυτική για την επανάσταση της βαθιάς μάθησης ήταν η δημιουργία μεγάλων, δημόσια διαθέσιμων συνόλων δεδομένων προς εκπαίδευση Τεχνητής Νοημοσύνης. Αυτα τα σύνολα δεδομένων αποτέλεσαν τη βάση για την επιστημονική σύγκριση και την αξιολόγηση των νέων αρχιτεκτονικών, κάνοντας δυνατή την αντικειμενική μέτρηση της προόδου στον τομέα.
Το **WikiSQL**[30] αποτέλεσε ένα από τα πρώτα και πιο επιδραστικά σύνολα δεδομένων μεγάλης κλίμακας για την μετατροπή Text-to-SQL. Περιλαμβάνει πάνω από 80.000 ερωτήσεις σε φυσική γλώσσα που αντιστοιχούν σε SQL ερωτήματα. Διακρίνεται για τον μεγάλο όγκο απλών ερωτήσεων που αφορούν μόνο έναν πίνακα (single-table queries). Αυτός ο περιορισμός το κάνει ιδανικό για την αρχική αξιολόγηση και ανάπτυξη βασικών μοντέλων, αλλά ανεπαρκές για την αντιμετώπιση των πολύπλοκων σεναρίων του πραγματικού κόσμου.
Το **Spider**[31] εμφανίστηκε το 2018 ως απάντηση στους περιορισμούς του WikiSQL και γρήγορα έγινε πρότυπο για την αξιολόγηση σύνθετων, διατομεακών ερωτημάτων (cross-domain queries). Περιλαμβάνει πάνω από 10.000 ερωτήσεις, κατανεμημένες σε 200 διαφορετικές βάσεις δεδομένων από διάφορους τομείς, και υποστηρίζει πολύπλοκα ερωτήματα που περιλαμβάνουν ζεύξεις (joins), ομαδοποιήσεις (grouping), υποερωτήματα (subqueries) και εμφωλευμένη λογική (nested logic). Η φύση του συνόλου τοπ καθιστά ιδιαίτερη πρόκληση, επειδή τα μοντέλα πρέπει να γενικεύσουν σε βάσεις δεδομένων που δεν έχουν δει κατά τη διάρκεια της εκπαίδευσης.
Αναγνωρίζοντας την ανάγκη για συστήματα που να μπορούν να χειρίζονται αλληλεπιδράσεις συνομιλίας, αναπτυχθηκάν τα σύνολα δεδομένων **CoSQL**[32] και **SParC**[33]. Το SParC (Semantic Parsing in Context) εστιάζει στην ανάλυση πολλαπλών, εξαρτημένων ερωτημάτων σε μια συνεχή συνομιλία, όπου κάθε ερώτηση μπορεί να αναφέρεται σε προηγούμενες ερωτήσεις ή αποτελέσματα. Το CoSQL επεκτείνει αυτή την ιδέα παρέχοντας ένα σύνολο δεδομένων για συνομιλίες όπου οι χρήστες μπορούν να κάνουν ερωτήσεις που χτίζονται η μία πάνω στην άλλη, απαιτώντας από τα μοντέλα να διατηρούν το πλαίσιο και να κατανοούν τις σχέσεις μεταξύ των ερωτήσεων.
Η διαθεσιμότητα αυτών των συνόλων επέτρεπε την ανάπτυξη και την αυστηρή σύγκριση νέων αρχιτεκτονικών νευρωνικών δικτύων, δημιουργώντας ένα κοινό σημείο αναφοράς για την πρόοδο στον τομέα του Text-to-SQL και ενισχύοντας τον ανταγωνισμό για την ανάπτυξη πιο ακριβών και γενικευμένων μοντέλων. Επίσης, η ποικιλία των συνόλων δεδομένων, από απλά single-table ερωτήματα μέχρι πολύπλοκα conversational interactions, επέτρεψε στους ερευνητές να εξερευνήσουν διαφορετικές πτυχές της μετατροπής Text-to-SQL και να αναπτύξουν μοντέλα που μπορούν να χειριστούν ένα ευρύ φάσμα σεναρίων.

2.2.2 Αρχιτεκτονικές Βασισμένες σε Νευρωνικά Δίκτυα

Η ανάπτυξη των νευρωνικών δικτύων επέτρεψε στα συστήματα Text-to-SQL να μαθαίνουν αυτόματα την αντιστοίχιση από φυσική γλώσσα σε SQL μέσω εκπαίδευσης σε μεγάλα σύνολα δεδομένων, αντικαθιστώντας τους ανθρώπινα κατασκευασμένους κανόνες.
Οι πρώτες προσεγγίσεις χρησιμοποίησαν μοντέλα **ακολουθίας-προς-ακολουθία**[34] (Sequence-to-Sequence) με αρχιτεκτονική κωδικοποιητή-αποκωδικοποιητή (encoder-decoder), όπου ο κωδικοποιητής επεξεργάζεται την ερώτηση φυσικής γλώσσας και ο αποκωδικοποιητής παράγει το SQL ερώτημα. Η εισαγωγή των **μηχανισμών προσοχής**[35] (attention mechanisms) βελτίωσε σημαντικά την απόδοση, επιτρέποντας στο μοντέλο να εστιάζει δυναμικά σε διαφορετικά μέρη της εισόδου κατά την παραγωγή κάθε στοιχείου της εξόδου.
Ωστόσο, η αντιμετώπιση της SQL ως απλής ακολουθίας κειμένου αγνοούσε τη δομική της οργάνωση. Το **SQLNet**[36] έλυσε αυτό το πρόβλημα εισάγοντας μια εξειδικευμένη αρχιτεκτονική που αναγνωρίζει τα διακριτά συστατικά της SQL (`SELECT`, `WHERE`, κλπ.) και προβλέπει ξεχωριστά κάθε συστατικό, εγγυώμενη συντακτικά σωστά ερωτήματα.
Η επόμενη εξέλιξη ήρθε με το **RAT-SQL**[37] (Relation-Aware Transformer), που επεκτείνει την αρχιτεκτονική του Transformer για να ενσωματώνει σχεσιακές πληροφορίες του σχήματος της βάσης δεδομένων. Αυτή η σχεσιακή επίγνωση επιτρέπει στο μοντέλο να κατανοεί τις σχέσεις μεταξύ πινάκων και στηλών, βελτιώνοντας την ικανότητά του να δημιουργεί σωστά `JOIN` ερωτήματα (ζεύξεις) και να χειρίζεται πολύπλοκα σχήματα βάσεων δεδομένων.
Αυτές οι αρχιτεκτονικές νευρωνικών δικτύων όχι μόνο βελτίωσαν δραστικά την ακρίβεια των συστημάτων Text-to-SQL, αλλά και επέτρεψαν την ανάπτυξη μοντέλων που μπορούν να γενικεύσουν καλύτερα σε νέα σχήματα βάσεων δεδομένων χωρίς εκτεταμένη επαναπροσαρμογή. Η εξέλιξη από απλά seq2seq μοντέλα σε δομημένες αρχιτεκτονικές όπως το SQLNet και τελικά σε σχεσιακά επίγνωστα μοντέλα όπως το RAT-SQL, αντιπροσωπεύει μια συστηματική πρόοδο προς συστήματα που κατανοούν τόσο τη φυσική γλώσσα όσο και τη δομή των βάσεων δεδομένων.

2.3 Η Σύγχρονη Εποχή των Μεγάλων Γλωσσικών Μοντέλων (LLMs)

Πλέον, η επανάσταση της βαθιάς μάθησης έχει οδηγήσει στην εμφάνιση των Μεγάλων Γλωσσικών Μοντέλων (LLMs), όπως το GPT-4[12], που δίνουν νέα διάσταση στην προσέγγιση του Text-to-SQL. Αυτά τα μοντέλα έχουν εκπαιδευτεί σε τεράστιες ποσότητες κειμένου και έχουν αποδείξει την ικανότητά τους να κατανοούν και να παράγουν φυσική γλώσσα με εντυπωσιακή ακρίβεια.

2.3.1 Prompt Engineering και In-Context Learning

Αντί για την εκπαίδευση εξειδικευμένων μοντέλων από το μηδέν, η σύγχρονη προσέγγιση επικεντρώνεται στο **Prompt Engineering**[13] και το **In-Context Learning**[14].
Το **zero-shot prompting** αποτελεί την απλούστερη μέθοδο, όπου το μοντέλο παράγει ένα SQL ερώτημα βασισμένο μόνο στο σχήμα της βάσης δεδομένων και την ερώτηση του χρήστη, χωρίς προηγούμενα παραδείγματα. Παρά την απλότητά του, αυτή η προσέγγιση μπορεί να έχει εκπληκτικά αποτελέσματα για βασικά ερωτήματά, αξιοποιώντας τις γνώσεις που έχει αποκτήσει το μοντέλο κατά την εκπαίδευσή του σε γενικά κείμενα.

Το **few-shot prompting** ή **in-context learning** επεκτείνει αυτή την ιδέα, προσθέτοντας μερικά παραδείγματα ζευγών ερώτησης-ερωτήματος στο prompt για να καθοδηγήσει την έξοδο του μοντέλου. Αυτή η τεχνική επιτρέπει στο μοντέλο να κατανοήσει καλύτερα τη μορφή και τη δομή των ερωτημάτων που αναμένονται, βελτιώνοντας την ακρίβεια των παραγόμενων SQL ερωτημάτων[11].
Μια από τις πιο καινοτόμες προσεγγίσεις είναι το **Chain-of-Thought Prompting**[15], όπου το μοντέλο ενθαρρύνεται να παράγει ενδιάμεσες σκέψεις ή λογικές ακολουθίες πριν καταλήξει στο τελικό SQL ερώτημα. Αυτό επιτρέπει στο μοντέλο να αναλύει τα βήματα που απαιτούνται για την επίλυση του ερωτήματος, βελτιώνοντας την ακρίβεια και την αποτελεσματικότητα των παραγόμενων ερωτημάτων. Είναι ιδιαίτερα χρήσιμη για πολύπλοκα ερωτήματα που απαιτούν ζεύξεις πινάκων (joins) ή υποερωτήματα (subqueries) και εμφωλευμένη λογική.
Οι τεχνικές αυτές επιτρέπουν την ταχεία ανάπτυξη εφαρμογών Text-to-SQL χωρίς την ανάγκη για εκτεταμένη επαναπροσαρμογή μοντέλων ή συνόλου δεδομένων. Επίσης, η ευελιξία του prompt engineering επιτρέπει την προσαρμογή του συστήματος σε διαφορετικές διαλέκτους SQL[4] και απαιτήσεις της βάσης δεδομένων, απλά αλλάζοντας το περιεχόμενο του prompt χωρίς να απαιτείται τεχνική εξειδίκευση ή ανάπτυξη νέων μοντέλων.

2.3.2 Προκλήσεις και Περιορισμοί των LLMs

Παρά την εντυπωσιακή τους απόδοση, τα Μεγάλα Γλωσσικά Μοντέλα (LLMs) αντιμετωπίζουν σημαντικούς περιορισμούς που επηρεάζουν την αποτελεσματικότητά τους στην παραγωγή SQL ερωτημάτων και καθιστούν απαραίτητη την ύπαρξη εφαρμογών όπως το WebText2SQL.
Μια από τις κύριες προκλήσεις είναι η **τάση των μοντέλων για "παραισθήσεις" (hallucinations)**[20], όπου τα μοντέλα παράγουν εσφαλμένα ή μη εκτελέσιμα SQL ερωτήματα που φαίνονται συντακτικά σωστά αλλά περιέχουν λογικά λάθη ή αναφορές σε ανύπαρκτα πεδία και πίνακες. Αυτό το πρόβλημα είναι ιδιαίτερα επικίνδυνο στο πλαίσιο των βάσεων δεδομένων, καθώς μπορεί να οδηγήσει σε εσφαλμένα αποτελέσματα χωρίς να το αντιληφθεί ο χρήστης.
Ένας άλλος σημαντικός περιορισμός είναι το **υψηλό κόστος υπολογισμού**[21] και **καθυστέρησης απόκρισης των API κλήσεων**. Τα LLMs απαιτούν σημαντικούς πόρους, και οι συχνές κλήσεις σε εμπορικά API μπορούν να γίνουν δαπανηρές, για εφαρμογές με μεγάλο όγκο χρηστών.
Κρίσιμο εμπόδιο είναι επίσης το **περιορισμένο παράθυρο συμφραζομένων (context window)** των μοντέλων. Παρότι τα συγχρονα LLMs υποστηρίζουν μεγαλύτερα παράθυρα συμφραζομένων (άνω των 128k tokens), πολύ μεγάλα ή πολύπλοκα σχήματα βάσεων δεδομένων μπορεί να υπερβαίνουν αυτό το όριο. Αυτό καθιστά δύσκολη την αποτελεσματική ενσωμάτωση όλων των σχεσιακών πληροφοριών και των απαιτούμενων ερωτήσεων σε ένα μόνο prompt, περιορίζοντας την ικανότητα του μοντέλου να κατανοεί πλήρως τη δομή της βάσης δεδομένων.
Επιπλέον, όπως επισημαίνουν οι ερευνητές[38], τα LLMs είναι ουσιαστικά "στοχαστικοί παπαγάλοι" (stochastic parrots) που αναπαράγουν μοτίβα από τα δεδομένα εκπαίδευσής τους χωρίς πραγματική κατανόηση. Αυτό σημαίνει ότι η επιτυχία τους στην παραγωγή SQL ερωτημάτων εξαρτάται σε μεγάλο βαθμό από την ποιότητα και την ποικιλία των δεδομένων εκπαίδευσης, και μπορεί να αποτύχουν σε σενάρια που δεν έχουν επαρκή κάλυψη στα δεδομένα αυτά.
Τέλος, μια συνολική ανασκόπηση των περιορισμών των LLMs[19] δείχνει επιπλέον προκλήσεις όπως η δυσκολία χειρισμού πολύ εκειδικευμένων ή σπάνιων SQL λειτουργιών, η έλλειψη συνέπειας στις απαντήσεις για παρόμοια ερωτήματα, και η περιορισμένη ικανότητα εκμάθησης από λάθη χωρίς ρητή επανατροφοδότηση.
Αυτοί οι περιορισμοί υπογραμμίζουν τη σημασία της ανάπτυξης καλά σχεδιασμένων εφαρμογών,

που μπορούν να συρρικνώσουν αυτές τις αδυναμίες μέσω έξυπνων στρατηγικών prompt engineering, επικύρωσης ερωτημάτων, και βελτιστοποίησης της διαχείρισης του παραθύρου συμφραζομένων.

2.4 Μέθοδοι Αξιολόγησης

Για την αξιολόγηση των συστημάτων Text-to-SQL έχουν καθιερωθεί δύο κύριες μετρικές που προσφέρουν διάφορες οπτικές επιτυχίας[31].

- **Ακριβής Ταύτιση Συνόλου (Exact Set Match)**: Αυτή η μετρική απαιτεί το παραγόμενο SQL ερώτημα να είναι συντακτικά ταυτόσημο με το ground-truth ερώτημα. Παρότι παρέχει ακριβή μέτρηση της συντακτικής ορθότητας, μπορεί να είναι υπερβολικά αυστηρή, καθώς πολλά, συντακτικά διαφορετικά, ερωτήματα μπορεί να παράγουν το ίδιο σωστό αποτέλεσμα.
- **Ακρίβεια Εκτέλεσης (Execution Accuracy)**: Αξιολογεί μόνο αν το αποτέλεσμα της εκτέλεσης του παραγόμενου SQL ερωτήματος είναι ταυτόσημο με το αποτέλεσμα του ground-truth ερωτήματος. Αυτή η μετρική είναι πιο πρακτική, καθώς αναγνωρίζει ότι πολλαπλά ερωτήματα μπορεί να είναι σημασιολογικά ισοδύναμα. Παρόλα αυτά, ο κίνδυνος είναι ότι μπορεί να επιτρέπει ερωτήματα που δεν είναι βέλτιστα ή καθυστερούν την απόδοση, αρκεί να παράγουν το σωστό αποτέλεσμα.

Πρόσφατες έρευνες έχουν προτείνει πιο εξελιγμένες μεθόδους αξιολόγησης[39] που χρησιμοποιούν "απομονωμένα σύνολα δοκιμών" (distilled test suites) για πιο αξιόπιστες συγκρίσεις, ενώ νέες μελέτες[18] επεκτείνουν την αξιολόγηση σε ρεαλιστικά σενάρια μεγαλύτερης κλίμακας με τη χρήση LLMs.
Αυτές οι μετρικές παρέχουν ένα ολοκληρωμένο πλαίσιο για την κατανόηση και σύγκριση της ποιότητας των συστημάτων Text-to-SQL, από τα πρώιμα συστήματα βασισμένα σε κανόνες μέχρι τα σύγχρονα LLMs.

Κεφάλαιο 3

Τεχνολογίες

Αυτή η ενότητα περιγράφει τις τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη του έργου. Οι τεχνολογίες αυτές περιλαμβάνουν γλώσσες προγραμματισμού, βιβλιοθήκες, εργαλεία και πλατφόρμες που χρησιμοποιήθηκαν για την υλοποίηση των λειτουργιών της εφαρμογής.

3.1 Python

Η γλώσσα προγραμματισμού Python χρησιμοποιήθηκε για την ανάπτυξη του backend αλλά και του frontend της εφαρμογής. Η Python είναι μια δημοφιλής γλώσσα προγραμματισμού που προσφέρει ευκολία στη χρήση, πλούσια οικοσυστήματα βιβλιοθηκών και εργαλεία για την ανάπτυξη εφαρμογών. Είναι μια γλώσσα υψηλού επιπέδου που υποστηρίζει πολλαπλά προγραμματιστικά παραδείγματα, όπως αντικειμενοστραφή, λειτουργική και διαδικαστική προγραμματιστική προσέγγιση. Έχει ευρεία υποστήριξη για βιβλιοθήκες και εργαλεία που διευκολύνουν την ανάπτυξη εφαρμογών, όπως το FastAPI για τη δημιουργία RESTful APIs, το SQLmodel για την αλληλεπίδραση με βάσεις δεδομένων και το Chainlit για τη δημιουργία εφαρμογών με διαδραστικό περιβάλλον συζήτησης. Η Python 3.13 είναι η έκδοση που χρησιμοποιήθηκε για την ανάπτυξη του έργου, παρέχοντας βελτιώσεις στην απόδοση και νέες δυνατότητες.
Παρακάτω παρατίθεται μια λίστα με τις κύριες βιβλιοθήκες και εργαλεία που χρησιμοποιήθηκαν στην ανάπτυξη του έργου, μαζί με μια σύντομη περιγραφή της κάθε μίας.

Βιβλιοθήκες

- **aiohttp** - Μια βιβλιοθήκη για την ασύγχρονη HTTP επικοινωνία, που χρησιμοποιείται για την αποστολή και λήψη αιτημάτων HTTP από τον εξυπηρετητή. Χρησιμοποιείται κυρίως από το **Chainlit** για την αλληλεπίδραση του frontend με το backend.

- **asyncpg** - Μια βιβλιοθήκη για την ασύγχρονη σύνδεση και αλληλεπίδραση με βάσεις δεδομένων PostgreSQL από Python, που επιτρέπει την εκτέλεση πολλαπλών αιτήσεων ταυτόχρονα χωρίς να μπλοκάρει την εκτέλεση του προγράμματος.

- **boto3** - Μια βιβλιοθήκη για την αλληλεπίδραση με τις υπηρεσίες AWS (Amazon Web Services), που χρησιμοποιείται για την αποθήκευση και ανάκτηση δεδομένων από το S3 (Amazon Simple Storage Service), καθώς και για τη διαχείριση άλλων υπηρεσιών του AWS.

- **cachetools** - Μια βιβλιοθήκη για τη διαχείριση της κρυφής μνήμης (cache) σε εφαρμογές Python, που χρησιμοποιείται για την αποθήκευση προσωρινών δεδομένων και την επιτάχυνση της απόδοσης. [1]

- **Chainlit** - Ένα εργαλείο για τη δημιουργία διαδραστικών εφαρμογών με Python, που επιτρέπει την εύκολη ενσωμάτωση με APIs και την αλληλεπίδραση με χρήστες.

- **FastAPI**[40] - Ένα σύγχρονο, γρήγορο (high-performance) web framework για τη δημιουργία APIs με Python 3.7+ βασισμένο σε τύπους Python.

- **openai** - Μια βιβλιοθήκη για την αλληλεπίδραση με το OpenAI API, που χρησιμοποιείται για την αποστολή αιτημάτων και την λήψη απαντήσεων από μοντέλα γλώσσας όπως το GPT-4o-mini[41] που χρησιμοποιείται στην εφαρμογή.

- **Pandas** - Μια βιβλιοθήκη για την ανάλυση δεδομένων που παρέχει δομές δεδομένων και εργαλεία για την επεξεργασία δεδομένων.

- **passlib[bcrypt]** - Μια βιβλιοθήκη για την ασφαλή αποθήκευση και επαλήθευση κωδικών πρόσβασης. Συγκεκριμένα, χρησιμοποιεί το bcrypt για την κρυπτογράφηση των κωδικών πρόσβασης, παρέχοντας υψηλό επίπεδο ασφάλειας. [2]

- **psycopg** & **psycopg2** - Βιβλιοθήκες για την αλληλεπίδραση με βάσεις δεδομένων PostgreSQL από Python. Το **psycopg** είναι η νεότερη έκδοση (v3), ενώ το **psycopg2** είναι η παλαιότερη έκδοση που χρησιμοποιείται ευρέως, και κυρίως στην βιβλιοθήκη **SQLmodel**.

- **pymysql** - Μια βιβλιοθήκη και οδηγός για την αλληλεπίδραση με βάσεις δεδομένων MySQL/MariaDB από Python, γραμμένος πλήρως σε Python. [3]

- **python-jose** - Μια βιβλιοθήκη για την επεξεργασία JSON δεδομένων, που χρησιμοποιείται συχνά για την ανταλλαγή δεδομένων μεταξύ πελάτη και διακομιστή, καθώς και για την υποστήριξη JWT (JSON Web Tokens) για την αυθεντικοποίηση και εξουσιοδότηση χρηστών.

- **python-multipart** - Μια βιβλιοθήκη για την επεξεργασία multipart/form-data αιτήσεων, που χρησιμοποιείται συχνά για την αποστολή αρχείων μέσω HTTP, απαραίτητη για την υποστήριξη του Chainlit.

- **SQLmodel** - Μια βιβλιοθήκη που συνδυάζει SQLAlchemy και Pydantic για τη δημιουργία μοντέλων δεδομένων και τη διαχείριση βάσεων δεδομένων.

[1] Προτιμήθηκε κυρίως για την υποστήριξη ttlcache, που είναι ένας τύπος κρυφής μνήμης που αποθηκεύει τα αποτελέσματα των συναρτήσεων με βάση τα ορίσματά τους και τα διαγράφει αυτόματα μετά από ένα καθορισμένο χρονικό διάστημα.

[2] Η επιλογή του bcrypt ως αλγορίθμου κρυπτογράφησης παρέχει υψηλό επίπεδο ασφάλειας, καθώς είναι σχεδιασμένος για να είναι αργός και να απαιτεί περισσότερους πόρους, καθιστώντας δύσκολη την εκτέλεση επιθέσεων brute-force.

[3] Η χρήση του **mysql-connector-python**, που είναι ο επίσημος οδηγός της MySQL από την Oracle, δεν ήταν δυνατή λόγω γνωστών προβλημάτων με την υποστήριξη ssh tunneling, που είναι απαραίτητο για την απομακρυσμένη σύνδεση σε βάσεις δεδομένων.

- **sshtunnel** - Μια βιβλιοθήκη που επιτρέπει τη δημιουργία SSH tunnels για ασφαλή σύνδεση σε απομακρυσμένους εξυπηρετητές. Εν προκείμενου, χρησιμοποιείται για την ασφαλή σύνδεση σε βάσεις δεδομένων που βρίσκονται, πιθανώς, πίσω από ένα firewall ή σε απομακρυσμένους εξυπηρετητές.
- **tabulate** - Μια βιβλιοθήκη για την εύκολη μορφοποίηση και εμφάνιση πινάκων σε κείμενο. Χρησιμοποιείται κυρίως απο την βιβλιοθήκη **Pandas** για την μετατροπή των dataframes σε μορφή markdown πίνακα, που είναι συμβατή με το **Chainlit**.

Εργαλεία

- **ruff** - Ένα εργαλείο για τη στατική ανάλυση κώδικα Python, που χρησιμοποιείται για την εύρεση και διόρθωση σφαλμάτων, την τήρηση των προτύπων κωδικοποίησης και τη βελτίωση της ποιότητας του κώδικα.
- **uv** - Ένα εργαλείο για την εκτέλεση και διαχείριση εφαρμογών Python, που παρέχει μια απλή διεπαφή για την εκτέλεση εντολών και τη διαχείριση περιβαλλόντων εκτέλεσης. Παράλληλα, λειτουργεί και σαν διαχειριστής πακέτων, επιτρέποντας την γρήγορη εγκατάσταση και διαχείριση εξαρτήσεων για έργα Python διατηρώντας διασυμβατότητα μεταξύ διαφορετικών συστημάτων και περιβαλλόντων.

Παρακάτω παρατίθεται μια πιο αναλυτική περιγραφή των σημαντικότερων βιβλιοθηκών και πλαισίων που χρησιμοποιήθηκαν στην ανάπτυξη του έργου, καθώς και των λειτουργιών που παρέχουν.

3.1.1 Chainlit

Το **Chainlit** είναι ένα πλαίσιο για την ανάπτυξη διαδραστικών εφαρμογών συνομιλίας (chat applications) με Python. Είναι συμβατό με πολλές δημοφιλείς βιβλιοθήκες και εργαλεία της Python, όπως το FastAPI και προσφέρει ενσωματωμένη υποστήριξη για την αλληλεπίδραση με το OpenAI API, Mistral AI, LangChain, και άλλες βιβλιοθήκες τεχνητής νοημοσύνης. Επιτρέπει στους προγραμματιστές να δημιουργούν εφαρμογές συνομιλίας που μπορούν να αλληλεπιδρούν με χρήστες μέσω ενός διαδραστικού περιβάλλοντος συζήτησης, παρέχοντας δυνατότητες όπως η αποστολή και λήψη μηνυμάτων, η επεξεργασία γεγονότων (events), η αποθήκευση και ανάκτηση δεδομένων, και η ενσωμάτωση με APIs. Ερχεται με μια προεπιλεγμένη διεπαφή χρήστη (UI) που μπορεί να προσαρμοστεί, καθώς και με ενσωματωμένη υποστήριξη για την αποθήκευση δεδομένων σε βάσεις δεδομένων, όπως PostgreSQL, MySQL, SQLite και άλλες. Τα χαρακτηριστικά που χρειάστηκε να ενεργοποιηθούν και να ρυθμιστούν για την εφαρμογή είναι τα εξής:

- **Αποθήκευση δεδομένων**: Από προεπιλογή, το Chainlit δεν αποθηκεύει τα δεδομένα και τις συνομιλίες που παράγονται κατά την εκτέλεση της εφαρμογής. Ωστόσο, παρέχει τη δυνατότητα ενεργοποίησης της αποθήκευσης δεδομένων σε μια βάση δεδομένων, όπως PostgreSQL, MySQL, SQLite και άλλες. Η υλοποίηση του επιπέδου αποθήκευσης δεδομένων δεν είναι υποχρεωτική, καθώς υπάρχει εσωτερική υποστήριξη για βάσεις PostgreSQL εφόσον η κατάλληλη μεταβλητή περιβάλλοντος (`DATABASE_URL`) έχει οριστεί και η βιβλιοθήκη **asyncpg** έχει εγκατασταθεί. Τα δεδομένα αποθηκεύονται σε πίνακες που πρέπει να δημιουργηθούν χειροκίνητα, καθώς το Chainlit δεν παρέχει μηχανισμό αυτόματης δημιουργίας πινάκων βάσης δεδομένων. Η αποθήκευση των δεδομένων επιτρέπει την αποθήκευση των συνομιλιών,

των μηνυμάτων και πιθανών επισυνάψεων (attachments) που αποστέλλονται και λαμβάνονται κατά την εκτέλεση της εφαρμογής, καθώς και την αποθήκευση των ρυθμίσεων και των παραμέτρων της εφαρμογής.

- **Αυθεντικοποίηση**: Η εφαρμογή υποστηρίζει την αυθεντικοποίηση χρηστών με 3 τρόπους:
  - **μέσω κωδικού πρόσβασης**: Υπάρχει μία φόρμα σύνδεσης χρηστών, που απαιτεί την εισαγωγή ενός ονόματος χρήστη και κωδικού πρόσβασης. Η υλοποίηση της αυθεντικοποίησης αφήνεται στον προγραμματιστή, ο οποίος μπορεί να χρησιμοποιήσει οποιαδήποτε μέθοδο επιθυμεί, όπως η ανακτηση των χρηστών από μια βάση δεδομένων, η χρήση ενός αρχείου κειμένου. [4]
  - **μέσω OAuth2**: Υπάρχει η δυνατότητα σύνδεσης χρηστών μέσω OAuth2, που επιτρέπει την αυθεντικοποίηση χρηστών μέσω εξωτερικών παρόχων, όπως το Google, το GitHub, το Facebook και άλλοι. Αυτό επιτρέπει στους χρήστες να συνδέονται στην εφαρμογή χρησιμοποιώντας τους λογαριασμούς τους σε αυτές τις υπηρεσίες, χωρίς να χρειάζεται να δημιουργήσουν νέο λογαριασμό. [5]
  - **μέσω Header Authentication**: Υπάρχει η δυνατότητα αυθεντικοποίησης χρηστών μέσω ενός HTTP header, που περιέχει ένα token που αποστέλλεται με κάθε αίτημα. Αυτό συνήθως χρησιμοποιείται όταν η εφαρμογή είναι ενσωματωμένη σε άλλο σύστημα (όπως στην περίπτωση αυτής της εφαρμογής) και το σύστημα αυτό παρέχει ένα token για την αυθεντικοποίηση των χρηστών. [6]

  Προαπαιτούμενο για την αποθήκευση των συνομιλιών των χρηστών είναι η ενεργοποίηση της αποθήκευσης δεδομένων και της αυθεντικοποίησης χρηστών, καθώς χωρίς αυτά τα χαρακτηριστικά, οι συνομιλίες δεν θα αποθηκεύονται και οι χρήστες δεν θα μπορούν να συνδεθούν στην εφαρμογή.

- **Παραμετροποίηση Διεπαφής Χρήστη**: Το Chainlit παρέχει τη δυνατότητα παραμετροποίησης της διεπαφής χρήστη, επιτρέποντας στους προγραμματιστές να προσαρμόσουν την εμφάνιση και τη λειτουργικότητα της εφαρμογής τους. Αυτό περιλαμβάνει την προσαρμογή του θέματος, μεταφράσεων του περιβάλλοντος χρήστη και την προσθήκη λογοτύπου και favicon.

3.1.2 FastAPI

Το **FastAPI**[40] είναι ένα σύγχρονο, γρήγορο (high-performance) πλάισιο για αναπτύξη διαδικτυακών εφαρμογων (web framework) για τη δημιουργία APIs με Python 3.7+ βασισμένο σε τύπους Python. Προσφέρει αυτόματη δημιουργία τεκμηρίωσης API (OpenAPI[7]) και υποστηρίζει ασύγχρονη λειτουργία, επιτρέποντας την αποτελεσματική διαχείριση πολλαπλών αιτήσεων ταυτόχρονα.

[4] Το κύριο πρόβλημα με αυτή την μέθοδο είναι ότι δεν υποστηρίζει την εγγραφή νέων χρηστών. Υπάρχει πλάνο για την προσθήκη αυτής της δυνατότητας στο μέλλον, αλλά προς το παρόν, οι χρήστες πρέπει να δημιουργηθούν εξωτερικά και να εισαχθούν στην βάση δεδομένων χειροκίνητα.

[5] Η υλοποίηση της αυθεντικοποίησης OAuth2 αποφεύχθηκε, διότι η εφαρμογή προοορίζεται να τρέχει στο περιβάλλον της σχολής και δεν υπήρχε λόγος να χρησιμοποιηθεί.

[6] Η υλοποίηση της αυθεντικοποίησης μέσω Header Authentication προτιμήθηκε, καθώς ήταν η πιο απλή και γρήγορη μέθοδος για την εφαρμογή, ώστε να δημιουργηθεί μια διεπαφή χρήστη για την σύνδεση και εγγραφή χρηστών.

[7] Πρώην Swagger UI

Είναι ιδανικό για την ανάπτυξη RESTful APIs και γιατί χρησιμοποιεί εσωτερικά το **Starlette**, το οποίο τρέχει εντός ενός **uvicorn** εξυπηρετητή εφαρμογών, για τη διαχείριση αιτήσεων και απαντήσεων, καθώς και την **Pydantic** για την επικύρωση και μετατροπή δεδομένων. Είναι συμβατό με το ASGI (Asynchronous Server Gateway Interface), που είναι ένα πρότυπο για τη δημιουργία ασύγχρονων εφαρμογών web με Python.
Παρακάτω παρατίθεται μια λίστα με τις κύριες βιβλιοθήκες και εργαλεία που κατεβαίνουν αυτόματα μαζί με το FastAPI, καθώς και μια σύντομη περιγραφή της κάθε μίας:

uvicorn

Το **uvicorn**[42] είναι ένας εξυπηρετητής εφαρμογών ASGI (Asynchronous Server Gateway Interface) που χρησιμοποιείται κυρίως για την εκτέλεση εφαρμογών FastAPI. Είναι γνωστός για την ταχύτητά του και την υποστήριξή του για ασύγχρονη λειτουργία, επιτρέποντας την αποτελεσματική διαχείριση πολλαπλών αιτήσεων ταυτόχρονα.

Jinja2

Το **Jinja2** είναι μια ισχυρή μηχανή προτύπων (template engine) για Python, που χρησιμοποιείται κυρίως για την απόδοση HTML σε εφαρμογές web. Παρέχει απλή σύνταξη για τη δημιουργία προτύπων, που μπορούν να χρησιμοποιηθούν για την απόδοση δυναμικού περιεχομένου σε ιστοσελίδες.

ItsDangerous

Το **ItsDangerous** είναι μια βιβλιοθήκη για την ασφαλή δημιουργία και επαλήθευση υπογεγραμμένων δεδομένων (signed data) σε εφαρμογές Python. Χρησιμοποιείται κυρίως για την υπογραφή και επαλήθευση cookies, tokens και άλλων δεδομένων που απαιτούν ασφάλεια και ακεραιότητα.

3.1.3 Pandas

Η **Pandas**[43] είναι μια βιβλιοθήκη για την ανάλυση δεδομένων που παρέχει δομές δεδομένων και εργαλεία για την επεξεργασία δεδομένων. Είναι ιδανική για την ανάλυση και επεξεργασία μεγάλων συνόλων δεδομένων, παρέχοντας λειτουργίες όπως φιλτράρισμα, ομαδοποίηση, συγχώνευση και μετασχηματισμό δεδομένων. Χρησιμοποιείται ευρέως σε εφαρμογές επιστήμης δεδομένων και μηχανικής μάθησης.
Στο προκείμενο έργο, χρησιμοποιείται για την γρήγορη επεξεργασία των δεδομένων που λαμβάνονται από τις βάσεις δεδομένων, καθώς και για την μετατροπή των δεδομένων σε μορφή πίνακα markdown, μέσω της βιβλιοθήκης **tabulate**, και CSV (comma-separated values), μέσω εσωτερικής συνάρτησης, που είναι συμβατες με την διεπαφή χρήστη που προσφέρει το **Chainlit**.

3.1.4 SQLmodel

Το **SQLmodel**[44] είναι μια βιβλιοθήκη που συνδυάζει τις δυνατότητες του SQLAlchemy και του Pydantic για τη δημιουργία μοντέλων δεδομένων και τη διαχείριση βάσεων δεδομένων. Παρέχει

έναν απλό και ευανάγνωστο τρόπο για τον ορισμό μοντέλων δεδομένων, που μπορούν να χρησιμοποιηθούν για την αλληλεπίδραση με βάσεις δεδομένων, καθώς και για την επικύρωση και μετατροπή δεδομένων. Είναι ιδανικό για την ανάπτυξη εφαρμογών που απαιτούν αλληλεπίδραση με βάσεις δεδομένων, καθώς παρέχει μια απλή διεπαφή για τη δημιουργία, ανάγνωση, ενημέρωση και διαγραφή (CRUD) δεδομένων. Για την υλοποίηση αυτών χρησιμοποιεί σύνταξη παρόμοια με την SQL γλώσσα, που βοηθά στην κατανόηση και τη συντήρηση του κώδικα.

```
# Παράδειγμα Python SELECT με SQLmodel
users = session.exec(select(User).where(User.name=="Iraklis")).fetchall()

-- Παράδειγμα SQL
SELECT * FROM users WHERE name = 'Iraklis';

```

Listing 3.1: Ομοιότητα γλώσσας SQL και σύνταξης ενός ερωτήματος SQLmodel

SQLAlchemy

Η **SQLAlchemy**[45] είναι μια δημοφιλής βιβλιοθήκη για την αλληλεπίδραση με βάσεις δεδομένων από Python, που παρέχει ένα ORM (Object-Relational Mapping) σύστημα για τη διαχείριση δεδομένων. Επιτρέπει στους προγραμματιστές να εργάζονται με βάσεις δεδομένων χρησιμοποιώντας αντικειμενοστραφή προγραμματισμό, παρέχοντας έναν απλό και ευανάγνωστο τρόπο για τον ορισμό μοντέλων δεδομένων και τη διαχείριση βάσεων δεδομένων.

Pydantic

Η **Pydantic**[46] είναι μια βιβλιοθήκη για την επικύρωση και μετατροπή δεδομένων (data validation and parsing). Βοηθάει τους προγραμματιστές να ορίζουν μοντέλα δεδομένων με τύπους δεδομένων της Python και να επικυρώνουν τα δεδομένα που λαμβάνονται από εξωτερικές πηγές, όπως APIs ή βάσεις δεδομένων. Αυτό επιτρέπει την εύκολη ανίχνευση σφαλμάτων και την εξασφάλιση της ακεραιότητας των δεδομένων στην εφαρμογή κατά την εκτέλεση της.

3.2 HTML

Η γλώσσα HTML (HyperText Markup Language) είναι η βασική γλώσσα σήμανσης για τη δημιουργία ιστοσελίδων και εφαρμογών ιστού. Χρησιμοποιείται για τη δομή και την παρουσίαση του περιεχομένου στο διαδίκτυο, επιτρέποντας στους προγραμματιστές να ορίζουν στοιχεία όπως τίτλους, παραγράφους, συνδέσμους, εικόνες και άλλα στοιχεία περιεχομένου. Είναι μια γλώσσα σήμανσης που χρησιμοποιεί ετικέτες (tags) για να καθορίσει τη δομή του περιεχομένου και να παρέχει πληροφορίες σχετικά με το πώς πρέπει να εμφανίζεται στο πρόγραμμα περιήγησης.
Επιπλέον, η HTML συνεργάζεται στενά με άλλες τεχνολογίες όπως το CSS (Cascading Style Sheets) και τη JavaScript για τη δημιουργία δυναμικών και διαδραστικών ιστοσελίδων. Η HTML5 είναι η τελευταία έκδοση της γλώσσας, που προσφέρει νέες δυνατότητες και βελτιώσεις, όπως υποστήριξη για πολυμέσα (ήχος και βίντεο), νέα στοιχεία για τη δομή του περιεχομένου (όπως το `<article>`, `<section>`, `<nav>`, κ.λπ.), και βελτιωμένη υποστήριξη για εφαρμογές ιστού (web applications). Η HTML5 είναι συμβατή με τα περισσότερα σύγχρονα προγράμματα περιήγησης και παρέχει μια ισχυρή βάση για την ανάπτυξη σύγχρονων εφαρμογών ιστού.

Στο εργο αυτό, η HTML είναι κυρίως αυτόματα παραγόμενη από το **Chainlit**, που χρησιμοποιεί την HTML για να δημιουργήσει τη διεπαφή χρήστη της εφαρμογής. Παρ' όλα αυτά, υπάρχει και κώδικας HTML που έχει γραφτεί για την Σύνδεση και Εγγραφή χρηστών.

3.3 JavaScript

Η γλώσσα προγραμματισμού JavaScript είναι μια δυναμική, αντικειμενοστραφής γλώσσα που χρησιμοποιείται κυρίως για την ανάπτυξη διαδραστικών στοιχείων σε ιστοσελίδες και εφαρμογές ιστού. Είναι μια γλώσσα υψηλού επιπέδου που υποστηρίζει συναρτησιακό και αντικειμενοστραφή προγραμματισμό, και είναι ευρέως χρησιμοποιούμενη για την ανάπτυξη frontend εφαρμογών, καθώς και για backend εφαρμογές μέσω του πλαισίου Node.js.
Στο frontend, η JavaScript μπορεί να χρησιμοποιηθεί για την αλληλεπίδραση με το DOM (Document Object Model), την επεξεργασία γεγονότων (events), την επικοινωνία με εξυπηρετητές μέσω AJAX (Asynchronous JavaScript and XML) και την δημιουργία διαδραστικών στοιχείων όπως φόρμες, κουμπιά και άλλα στοιχεία διεπαφής χρήστη. Αυτό επιτυγχάνεται με 4 τρόπους:

- Εσωτερική (internal), δηλαδή ενσωματωμένη μέσα σε HTML έγγραφα.
- Εξωτερική (external), δηλαδή σε ξεχωριστό αρχείο JavaScript που καλείται από το HTML έγγραφο.
- Ενσωματωμένη (inline), δηλαδή μέσα σε HTML ετικέτες.
- Προεπεξεργασμένη (preprocessed) ή transpiled, δηλαδή γραμμένη σε γλώσσα υψηλού επιπέδου όπως TypeScript ή CoffeeScript, μεταγλωττισμένη σε JavaScript και έπειτα ενσωματωμένη σε HTML έγγραφα.

Στο συγκεκριμένο έργο, η JavaScript χρησιμοποιείται κυρίως μέσω του **Chainlit**, που χρησιμοποιεί το πλαίσιο React[47] για την ανάπτυξη της διεπαφής χρήστη. Το ίδιο το Chainlit όμως προσφέρει και την δυνατότητα χρήσης εξωτερικών επιπρόσθετων αρχειών JavaScript, που μπορούν να χρησιμοποιηθούν για την επέκταση της λειτουργικότητας της εφαρμογής. Τέτοια αρχεία δημιουργήθηκαν για την υποστήριξη της ανακατεύθυνσης χρηστών στη σελίδα σύνδεσης και εγγραφής, όταν λαμβάνεται ένα γεγονός (event) "κλικ" σε ένα κουμπί σύνδεσης ή το λογότυπο της εφαρμογής.

3.4 CSS

Η γλώσσα CSS (Cascading Style Sheets) είναι μια γλώσσα στυλ που χρησιμοποιείται για την περιγραφή της εμφάνισης και της μορφοποίησης εγγράφων HTML και XML (συμπεριλαμβανομένων των SVG και XHTML). Επιτρέπει στους προγραμματιστές να ορίζουν στυλ για στοιχεία HTML, όπως χρώματα, γραμματοσειρές, περιθώρια, διαστάσεις, διάταξη και άλλα οπτικά χαρακτηριστικά. Αυτό επιτρέπει την διαχωρισμό της δομής του περιεχομένου (HTML) από την παρουσίαση του (CSS), διευκολύνοντας τη συντήρηση και την αναβάθμιση των ιστοσελίδων. Υποστηρίζει επίσης την αντίδραση σε γεγονότα (events) και την δημιουργία διαδραστικών στοιχείων, όπως hover effects, transitions και animations. Χρησιμοποιείται ευρέως για την ανάπτυξη responsive σχεδίων, που προσαρμόζονται σε διαφορετικά μεγέθη οθονών και συσκευών.
Η CSS μπορεί να ενσωματωθεί σε HTML έγγραφα με τρεις τρόπους, όπως και η **JavaScript**:

- Εσωτερική (internal), δηλαδή ενσωματωμένη μέσα σε HTML έγγραφα, συνήθως μέσα σε ένα `<style>` στοιχείο στο `<head>` τμήμα του εγγράφου.
- Εξωτερική (external), δηλαδή σε ξεχωριστό αρχείο CSS που καλείται από το HTML έγγραφο.
- Ενσωματωμένη (inline), δηλαδή μέσα σε HTML ετικέτες, χρησιμοποιώντας το `style` χαρακτηριστικό.

Στο συγκεκριμένο έργο, η CSS χρησιμοποιείται κυρίως μέσω του **Chainlit**, που παρέχει ένα προεπιλεγμένο στυλ για την διεπαφή χρήστη της εφαρμογής. Παρ' όλα αυτά, υπάρχει και κώδικας CSS που έχει γραφτεί χειροκίνητα για την Σύνδεση και Εγγραφή χρηστών, καθώς και για την προσαρμογή του στυλ ολόκληρης της εφαρμογής.

3.4.1 Tailwind CSS

Εσωτερικά, το Chainlit χρησιμοποιεί την βιβλιοθήκη Tailwind CSS για την διαχείριση του στυλ της διεπαφής χρήστη. Το Tailwind CSS είναι ένα utility-first CSS framework που επιτρέπει στους προγραμματιστές να δημιουργούν προσαρμοσμένα στυλ χρησιμοποιώντας μικρές, επαναχρησιμοποιήσιμες κλάσεις CSS. Αντί να γράφουν μεγάλες, σύνθετες κλάσεις CSS, οι προγραμματιστές μπορούν να χρησιμοποιούν μικρές κλάσεις για να συνδυάσουν στυλ και να δημιουργήσουν προσαρμοσμένα σχέδια. Αυτό επιτρέπει την γρήγορη ανάπτυξη και την ευκολία συντήρησης του κώδικα CSS, καθώς οι κλάσεις είναι ευανάγνωστες και κατανοητές.
Στο συγκεκριμένο έργο, η Tailwind δεν χρησιμοποιήθηκε άμεσα, αλλά μέσω ενός JSON αρχείου ρυθμίσεων που παρέχεται από το Chainlit, το οποίο περιέχει τις εκτιθημένες κλάσεις CSS που χρησιμοποιούνται στην διεπαφή χρήστη (π.χ. χρώματα, γραμματοσειρές, περιθώρια, κ.λπ.) για το εκάστοτε θέμα (theme) της εφαρμογής (φωτεινό ή σκοτεινό).

3.5 Docker

Το **Docker**[48] είναι μια πλατφόρμα για την ανάπτυξη, μεταφορά και εκτέλεση εφαρμογών σε εικονικοποιημένα περιβάλλοντα (virtualized environments), γνωστά ως containers. Τα containers επιτρέπουν στους προγραμματιστές να συσκευάζουν μια εφαρμογή και όλες τις εξαρτήσεις της σε ένα ενιαίο πακέτο, που μπορεί να εκτελείται σε οποιοδήποτε περιβάλλον που υποστηρίζει Docker, ανεξαρτήτα από το λειτουργικό σύστημα ή την υποδομή. Αυτό διευκολύνει την ανάπτυξη και τη συντήρηση εφαρμογών, καθώς οι προγραμματιστές μπορούν να διασφαλίσουν ότι η εφαρμογή θα εκτελείται με τον ίδιο τρόπο σε οποιοδήποτε περιβάλλον, χωρίς να ανησυχούν για διαφορές στις ρυθμίσεις ή τις εξαρτήσεις. Το Docker χρησιμοποιεί την τεχνολογία εικονικοποίησης (virtualization) για να δημιουργήσει απομονωμένα περιβάλλοντα εκτέλεσης, που επιτρέπουν την ταυτόχρονη εκτέλεση πολλών εφαρμογών σε έναν μόνο διακομιστή (server) χωρίς να επηρεάζουν η μία την άλλη.
Στο συγκεκριμένο έργο, το Docker χρησιμοποιήθηκε για τη συντήρηση 2 κύριων υπηρεσιών:

- Την υπηρεσία του **PostgreSQL** για την αποθήκευση των δεδομένων της εφαρμογής, που τρέχει σε ένα container PostgreSQL.
- Την υπηρεσία του **LocalStack**, που παρέχει ένα τοπικό περιβάλλον για την ανάπτυξη και δοκιμή εφαρμογών που χρησιμοποιούν υπηρεσίες AWS, όπως το S3 (Simple Storage Service) για την αποθήκευση αρχείων.

3.5.1 LocalStack

Το **LocalStack** είναι ένα εργαλείο που παρέχει ένα τοπικό περιβάλλον για την ανάπτυξη και δοκιμή εφαρμογών που χρησιμοποιούν υπηρεσίες AWS (Amazon Web Services). Επιτρέπει στους προγραμματιστές να σχεδιάζουν εφαρμογές που χρησιμοποιούν υπηρεσίες όπως το S3 (Simple Storage Service), το DynamoDB, το Lambda και άλλες υπηρεσίες της AWS, χωρίς να χρειάζεται να συνδεθούν σε πραγματικές υπηρεσίες AWS. Αυτό διευκολύνει την ανάπτυξη και τη συντήρηση εφαρμογών, καθώς οι προγραμματιστές μπορούν να δοκιμάσουν τις εφαρμογές τους τοπικά, χωρίς να ανησυχούν για το κόστος ή την πολυπλοκότητα της χρήσης πραγματικών υπηρεσιών AWS. Παρέχει μια πλήρη προσομοίωση των υπηρεσιών AWS, επιτρέποντας στους προγραμματιστές να αναπτύσουν εφαρμογές που χρησιμοποιούν τις υπηρεσίες αυτές, χωρίς να χρειάζεται να ανησυχούν για τις διαφορές μεταξύ των τοπικών και των πραγματικών υπηρεσιών AWS.
Στο συγκεκριμένο έργο, το LocalStack χρησιμοποιήθηκε για την προσομοίωση των S3 bucket υπηρεσιών της AWS, που χρησιμοποιούνται για την αποθήκευση και ανάκτηση αρχείων που ανεβάζουν οι χρήστες της εφαρμογής.

3.5.2 PostgreSQL

Η **PostgreSQL**[49] είναι ένα ισχυρό, ανοιχτού κώδικα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (ΣΔΒΔ ή αλλίως RDBMS) που υποστηρίζει ένα ευρύ φάσμα χαρακτηριστικών και λειτουργιών. Είναι γνωστή για την αξιοπιστία, την ευελιξία και την υποστήριξη για πολύπλοκες ερωτήσεις και συναλλαγές. Η PostgreSQL υποστηρίζει πλήρως το πρότυπο SQL (Structured Query Language) και παρέχει επιπλέον δυνατότητες όπως υποστήριξη για JSON δεδομένα, γεωχωρικά δεδομένα, και επεκτάσεις για την προσθήκη νέων λειτουργιών. Είναι ιδανική για εφαρμογές που απαιτούν υψηλή απόδοση, ασφάλεια και επεκτασιμότητα, όπως επιχειρηματικές εφαρμογές, εφαρμογές ιστού και εφαρμογές επιστήμης δεδομένων.
Στο συγκεκριμένο έργο, η PostgreSQL χρησιμοποιείται ως κύρια βάση δεδομένων για την αποθήκευση των δεδομένων της εφαρμογής, όπως οι χρήστες, το ιστορικό των συνομιλιών και η αποθήκευση συνδέσμων για αρχεία που κατεβάζονται από τους χρήστες.

3.6 OpenAI API

Το **OpenAI API** [50] είναι μια διεπαφή προγραμματισμού εφαρμογών (API) που παρέχεται από την OpenAI, επιτρέποντας στους προγραμματιστές να αλληλεπιδρούν με τα μοντέλα γλώσσας της OpenAI, όπως το GPT-4o-mini[41]. Το API επιτρέπει την αποστολή αιτημάτων και την λήψη απαντήσεων από τα μοντέλα γλώσσας, επιτρέποντας την ενσωμάτωση των λειτουργιών τους σε εφαρμογές και υπηρεσίες. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν το API για να δημιουργήσουν εφαρμογές που χρησιμοποιούν τα μοντέλα γλώσσας για διάφορες λειτουργίες, όπως η δημιουργία κειμένου, η απάντηση σε ερωτήσεις, η μετάφραση κειμένου, η ανάλυση συναισθήματος και άλλες λειτουργίες που σχετίζονται με την επεξεργασία φυσικής γλώσσας (NLP).

3.6.1 GPT-4o-mini

Στο συγκεκριμένο έργο, χρησιμοποιείται το μοντέλο **GPT-4o-mini**[41], που είναι μια έκδοση του GPT-4[12] που έχει σχεδιαστεί για να είναι πιο αποδοτική και γρήγορη, διατηρώντας παράλληλα

υψηλή ποιότητα απαντήσεων. Το μοντέλο αυτό έχει εκπαιδευτεί σε ένα ευρύ φάσμα δεδομένων και είναι ικανό να κατανοεί και να παράγει κείμενο σε πολλές γλώσσες, συμπεριλαμβανομένων των ελληνικών και αγγλικών.

3.7 GitHub

Το **GitHub** είναι μια πλατφόρμα φιλοξενίας κώδικα που χρησιμοποιεί το σύστημα ελέγχου εκδόσεων Git. Παρέχει εργαλεία για τη διαχείριση έργων λογισμικού, την συνεργασία μεταξύ προγραμματιστών και την παρακολούθηση αλλαγών στον κώδικα. Οι προγραμματιστές μπορούν να δημιουργούν αποθετήρια (repositories) για τα έργα τους, να διαχειρίζονται εκδόσεις του κώδικα, να συνεργάζονται με άλλους προγραμματιστές μέσω pull requests και να παρακολουθούν ζητήματα (issues) και αιτήματα χαρακτηριστικών (feature requests). Στο συγκεκριμένο έργο, το GitHub χρησιμοποιήθηκε για την αποθήκευση και διαχείριση του κώδικα της εφαρμογής, καθώς και για την οργάνωση της ανάπτυξης και την παρακολούθηση των αλλαγών στον κώδικα. Ο κώδικας του έργου είναι διαθέσιμος στη διεύθυνση: `https://github.com/dyka3773/WebText2SQL` με άδεια χρήσης MIT και καλείται ο οποιοδήποτε να τον χρησιμοποιήσει, να τον τροποποιήσει και να τον διανείμει ελεύθερα.

Κεφάλαιο 4

Σχεδίαση και Υλοποίηση του WebText2SQL

4.1 Λειτουργικές Απαιτήσεις

Πίνακας 4.1: Διαχείριση Χρηστών και Αυθεντικοποίηση

Κωδικός	**Τίτλος**	**Περιγραφή Απαίτησης**
FR1	Εγγραφή Χρήστη	Το σύστημα θα επιτρέπει στους χρήστες να εγγραφούν με μία διεύθυνση ηλεκτρονικού ταχυδρομείου και έναν κωδικό πρόσβασης.
FR2	Σύνδεση Χρήστη	Το σύστημα θα επιτρέπει στους χρήστες να συνδέονται με τη διεύθυνση ηλεκτρονικού ταχυδρομείου και τον κωδικό πρόσβασής τους.
FR3	Διαχείριση Συνεδριών (Session Management)	Μετά από μια επιτυχημένη σύνδεση, το σύστημα θα δημιουργεί μια συνεδρία για τον χρήστη, αποθηκεύοντας ένα ασφαλές αναγνωριστικό συνεδρίας (session cookie) στον περιηγητή του χρήστη.
FR4	Ασφάλεια Κωδικού Πρόσβασης	Το σύστημα θα αποθηκεύει τους κωδικούς πρόσβασης με ασφάλεια, χρησιμοποιώντας αλγορίθμους κατακερματισμού (hashing) προτού τους αποθηκεύσει στη βάση δεδομένων.
FR5	Μοναδικότητα Χρήστη	Το σύστημα θα διασφαλίζει ότι κάθε διεύθυνση ηλεκτρονικού ταχυδρομείου μπορεί να χρησιμοποιηθεί μόνο για έναν λογαριασμό χρήστη.
FR6	Αποσύνδεση Χρήστη	Το σύστημα θα παρέχει μηχανισμό αποσύνδεσης στον χρήστη, ο οποίος θα τερματίζει τη συνεδρία και θα διαγράφει το αναγνωριστικό συνεδρίας από τον περιηγητή.
FR7	Αποθήκευση Προφίλ Χρήστη	Το σύστημα θα αποθηκεύει πληροφορίες προφίλ χρήστη, όπως ένα μοναδικό αναγνωριστικό χρήστη, διεύθυνση ηλεκτρονικού ταχυδρομείου και τον κρυπτογραφημένο κωδικό πρόσβασης σε ένα πίνακα χρηστών (APP_USERS) στη βάση δεδομένων.

Πίνακας 4.2: Διαχείριση και Σύνδεση Βάσεων Δεδομένων

Κωδικός	**Τίτλος**	**Περιγραφή Απαίτησης**
FR8	Υποστήριξη Πολλαπλών Μοντέλων Βάσεων Δεδομένων	Το σύστημα θα υποστηρίζει τουλάχιστον 2 μοντέλα βάσεων δεδομένων, συγκεκριμένα MySQL και PostgreSQL
FR9	Επιλογές Ασφαλούς Σύνδεσης	Το σύστημα θα επιτρέπει στους χρήστες να συνδέονται σε βάσεις δεδομένων χρησιμοποιώντας είτε άμεσες TCP/IP συνδέσεις είτε μέσω SSH tunneling για αυξημένη ασφάλεια.
FR10	Παραμετροποίηση Συνδέσεων	Το σύστημα θα συλλέγει όλες τις απαραίτητες πληροφορίες σύνδεσης, όπως όνομα εξυπηρετητή (Host), θύρα (Port), όνομα χρήστη (Username), κωδικό πρόσβασης (Password) και, για την PostgreSQL, το όνομα της βάσης δεδομένων (Database Name).
FR11	Αποθήκευση Συνδέσεων	Το σύστημα θα αποθηκεύει τις πληροφορίες σύνδεσης σε έναν πίνακα βάσης δεδομένων (USER_CONNECTIONS) για την εύκολη πρόσβαση σε αυτές στο μέλλον.
FR12	Ονοματοδοσία Συνδέσεων	Οι χρήστες θα μπορούν να δίνουν όνομα στις συνδέσεις τους για εύκολη αναγνώριση, το οποίο θα αποθηκεύεται και αυτό μαζί με τις υπόλοιπες πληροφορίες σύνδεσης.
FR13	Επαλήθευση Συνδέσεων	Το σύστημα θα επαληθεύει τις πληροφορίες σύνδεσης κατά την αποθήκευση, διασφαλίζοντας ότι οι χρήστες μπορούν να συνδεθούν επιτυχώς στις βάσεις δεδομένων τους.
FR14	Επιλογή Συνδέσεων	Οι χρήστες θα μπορούν να επιλέγουν από τις αποθηκευμένες συνδέσεις τους για να συνδεθούν σε μια βάση δεδομένων, διευκολύνοντας την πρόσβαση σε πολλαπλές βάσεις δεδομένων.
FR15	Διαγραφή Συνδέσεων	Οι χρήστες θα μπορούν να διαγράφουν τις αποθηκευμένες συνδέσεις τους, εάν δεν τις χρειάζονται πλέον. Η διαγραφή θα περιλαμβάνει και την αφαίρεση προηγούμενων συνομιλιών (chat conversation/thread histories) που σχετίζονται με αυτές τις συνδέσεις.
FR16	Έυρεση Σχημάτων	Μετά την επιτυχή σύνδεση σε μια βάση δεδομένων, το σύστημα θα ανακτά και θα εμφανίζει τα διαθέσιμα προς τον χρήστη σχήματα (schemas) της βάσης δεδομένων, στην οποία ο χρήστης έχει συνδεθεί.
FR17	Επιλογή Σχήματος	Το σύστημα θα απαιτεί από τον χρήστη να επιλέξει ένα σχήμα (schema) για να εκτελέσει ερωτήματα, διασφαλίζοντας ότι οι ερωτήσεις θα εκτελούνται στο σωστό πλαίσιο της βάσης δεδομένων.

Πίνακας 4.3: Φυσική Γλώσσα και Ερωτήματα SQL

Κωδικός	**Τίτλος**	**Περιγραφή Απαίτησης**
FR18	Υποστήριξη Φυσικής Γλώσσας	Η διεπαφή χρήστη του συστήματος θα επιτρέπει στους χρήστες να εισάγουν ερωτήματα σε φυσική γλώσσα, και σε οποιαδήποτε γλώσσα, όπως Αγγλικά, Ελληνικά, Γαλλικά, κ.λπ.
FR19	Παραγωγή Ερωτημάτων SQL με Χρήση AI	Το σύστημα θα χρησιμοποιεί ένα μεγάλο γλωσσικό μοντέλο (LLM) για να μετατρέπει τα ερωτήματα φυσικής γλώσσας σε ερωτήματα SQL.
FR20	Παραγωγή Ερωτήματος στο Πλαίσιο της Βάσης Δεδομένων	Για πιο ακριβή αποτελέσματα, το σύστημα θα παρέχει στο LLM το πλήρες σχήμα της βάσης δεδομένων, συμπεριλαμβανομένων των πινάκων, των στηλών και των σχέσεων μεταξύ τους, κατά τη διαδικασία παραγωγής του ερωτήματος SQL.
FR21	Παραγωγή Ερωτήματος με το Πλαίσιο της Συνομιλίας	Το σύστημα θα έχει τη δυνατότητα να παρέχει στο LLM το πλαίσιο της συνομιλίας (chat context) των τελευταίων 10 μηνυμάτων, για να διασφαλίσει ότι το παραγόμενο ερώτημα SQL είναι σχετικό με την τρέχουσα συνομιλία. Η λειτουργία αυτή θα μπορεί να ενεργοποιείται ή να απενεργοποιείται ανά χρήστη.
FR22	Βελτιστοποίηση Εισόδου για το LLM	Το σύστημα θα βελτιστοποιεί την περιγραφή του σχήματος της βάσης δεδομένων και το πλαίσιο της συνομιλίας πριν την αποστολή τους στο LLM, για να διασφαλίσει ότι η είσοδος είναι όσο το δυνατόν πιο συμπαγής για την μείωση του κόστους χρήσης του LLM.
FR23	Εμφάνιση Παραγόμενου Ερωτήματος SQL	Το σύστημα θα εμφανίζει το παραγόμενο ερώτημα SQL στον χρήστη πριν την εκτέλεσή του, επιτρέποντας στον χρήστη να το επανεξετάσει και να το τροποποιήσει εάν χρειάζεται.
FR24	Εκτέλεση Ερωτήματος SQL	Το σύστημα θα εκτελεί το παραγόμενο ερώτημα SQL στη βάση δεδομένων και θα επιστρέφει τα αποτελέσματα στον χρήστη.

Πίνακας 4.4: Παρουσίαση Δεδομένων και Διεπαφή Χρήστη

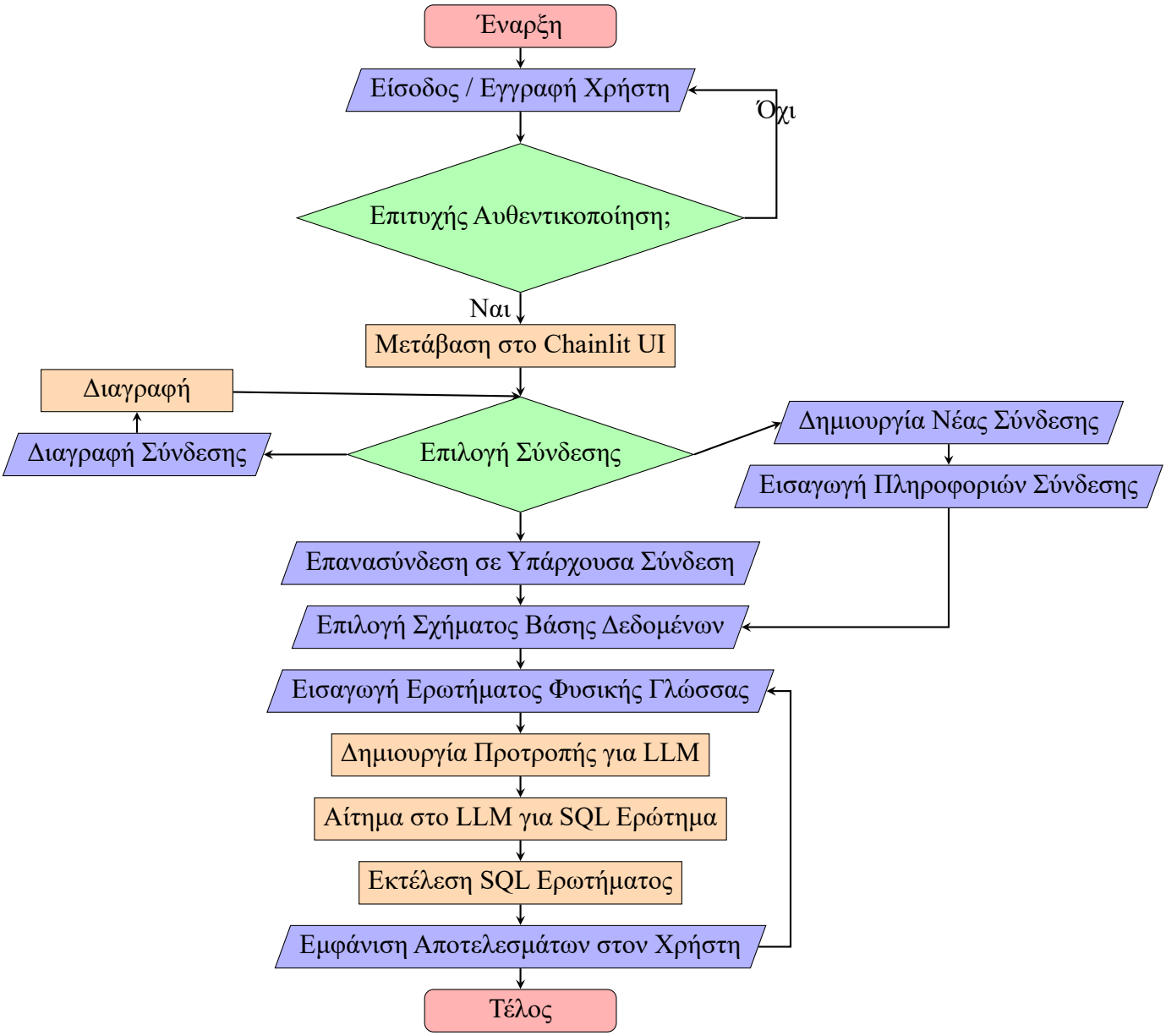
Κωδικός	**Τίτλος**	**Περιγραφή Απαίτησης**
FR25	Διεπαφή Συνομιλίας	Όλες οι αλληλεπιδράσεις του χρήστη με το σύστημα θα πραγματοποιούνται μέσω μιας διεπαφής συνομιλίας (chat interface), όπου ο χρήστης μπορεί να εισάγει ερωτήματα και να λαμβάνει απαντήσεις.
FR26	Παρουσίαση SQL Ερωτηματός	Το σύστημα θα εμφανίζει το παραγόμενο ερώτημα SQL στον χρήστη μαζί με τα αποτελέσματα του.
FR27	Παρουσίαση Αποτελεσμάτων	Τα αποτελέσματα των εκτελεσμένων ερωτημάτων SQL θα εμφανίζονται στον χρήστη σε ευανάγνωστη μορφή.
FR28	Σελιδοποίηση Αποτελεσμάτων	Για την διατήρηση της διεπαφής ευανάγνωστη, το σύστημα θα εμφανίζει μία προεπισκόπηση των πρώτων 10 αποτελεσμάτων από κάθε ερώτημα SQL.
FR29	Κατέβασμα Πλήρους Αποτελέσματος	Ο χρήστης θα έχει τη δυνατότητα να κατεβάσει το πλήρες αποτέλεσμα του ερωτήματος SQL σε μορφή CSV.
FR30	Ανατροφοδότηση "No results found"	Εάν το ερώτημα SQL δεν επιστρέφει αποτελέσματα, το σύστημα θα εμφανίζει ένα μήνυμα "No results found" στον χρήστη, ενημερώνοντάς τον ότι δεν υπάρχουν αποτελέσματα για το συγκεκριμένο ερώτημα.
FR31	Διαχείριση Συνομιλιών	Το σύστημα θα οργανώνει τις συνομιλίες του χρήστη σε νήματα (threads), ονομάζοντας κάθε νήμα αυτόματα με βάση την ενεργή σύνδεση της βάσης δεδομένων και το επιλεγμένο σχήμα, για να διευκολύνει την αναγνώριση και την πλοήγηση.

Πίνακας 4.5: Σύστημα και Διαχειριστικά

Κωδικός	**Τίτλος**	**Περιγραφή Απαίτησης**
FR32	Προσωρινή Μνήμη (Cache)	Το σύστημα θα χρησιμοποιεί προσωρινή μνήμη (cache) για την αποθήκευση των αποτελεσμάτων των ερωτημάτων SQL, των μεταδεδομένων της βάσης δεδομένων και των σχημάτων, για να βελτιώσει τον χρόνο απόκρισης του.
FR33	Παραμετροποίηση της Εφαρμογής	Το σύστημα θα επιτρέπει την παραμετροποίηση των βασικών ρυθμίσεων της εφαρμογής, όπως το URL της βάσης δεδομένων της, το κλειδί API του LLM και το κρυφό κλειδί για την κρυπτογράφηση των συνδέσεων, μέσω μεταβλητών περιβάλλοντος (environment variables).
FR34	Καταγραφή Και Παρακολούθηση (Logging)	Το σύστημα θα καταγράφει όλες τις σημαντικές ενέργειες και σφάλματα σε αρχεία καταγραφής (log files), για να διευκολύνει την παρακολούθηση και την επίλυση προβλημάτων.
FR35	Δομή Εφαρμογής Ιστού	Το σύστημα θα χτιστεί με δομή εφαρμογής ιστού (web application structure), για να είναι διαθέσιμο και λειτουργικό μέσω ενός περιηγητή ιστού (web browser).

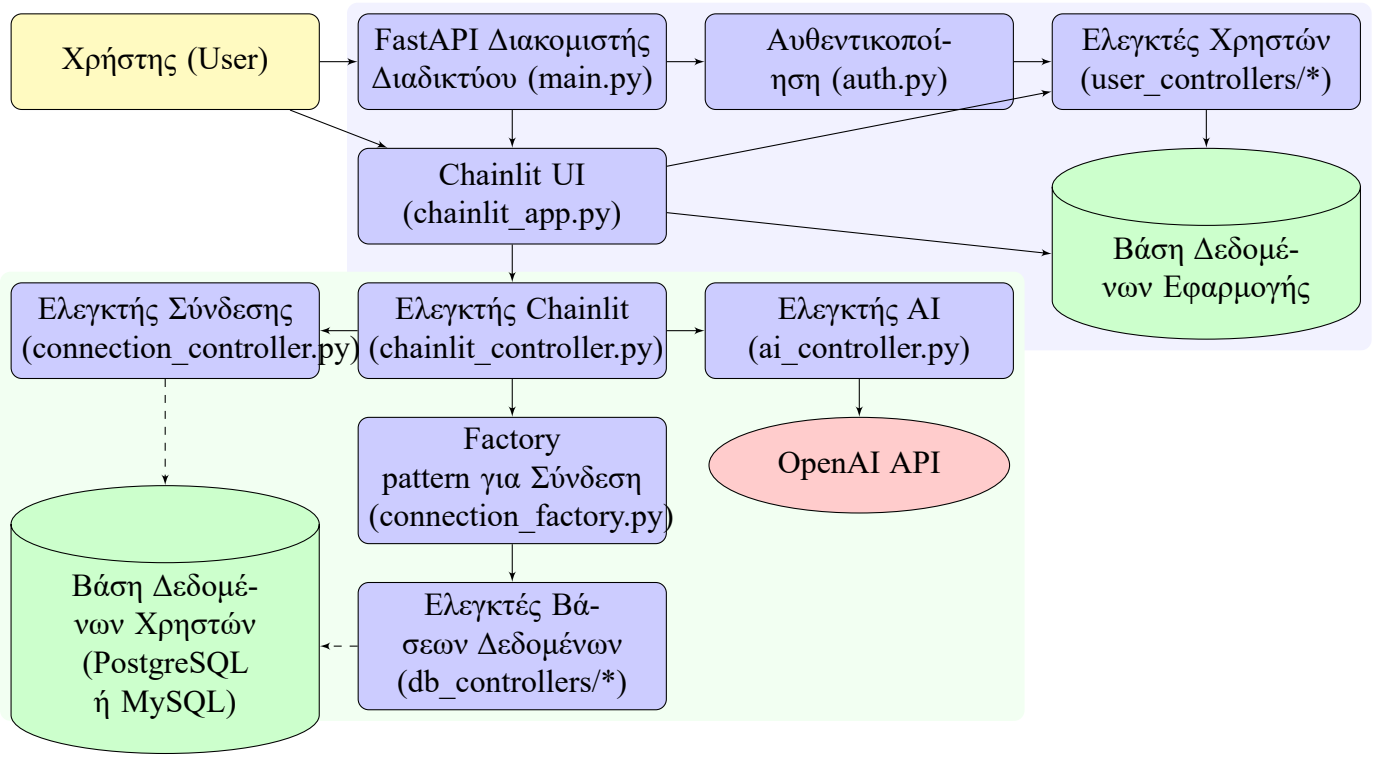
4.2 Αρχιτεκτονική της Εφαρμογής

4.2.1 Διάγραμμα Ροής



Σχήμα 4.1: Διάγραμμα Ροής Αλληλεπίδρασης Χρήστη

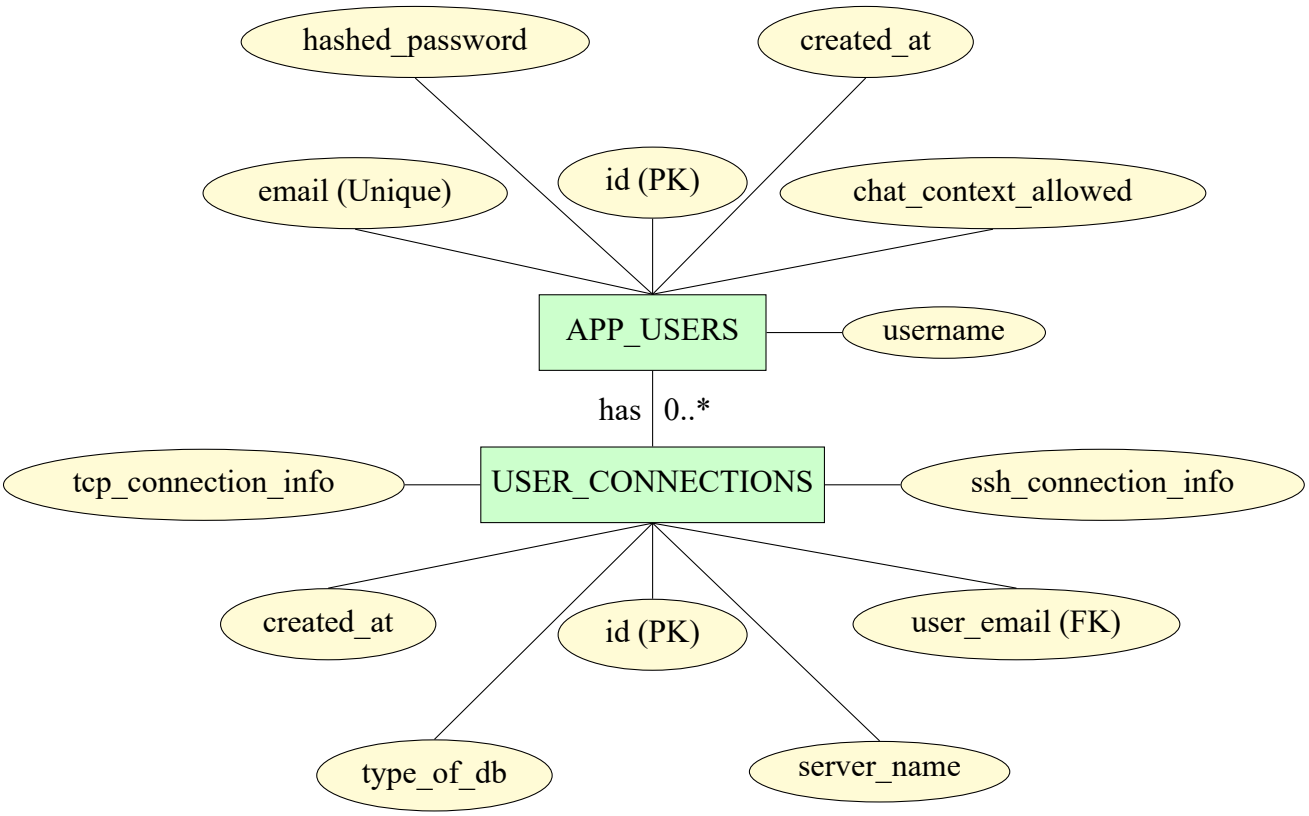
4.2.2 Διάγραμμα Αρχιτεκτονικής



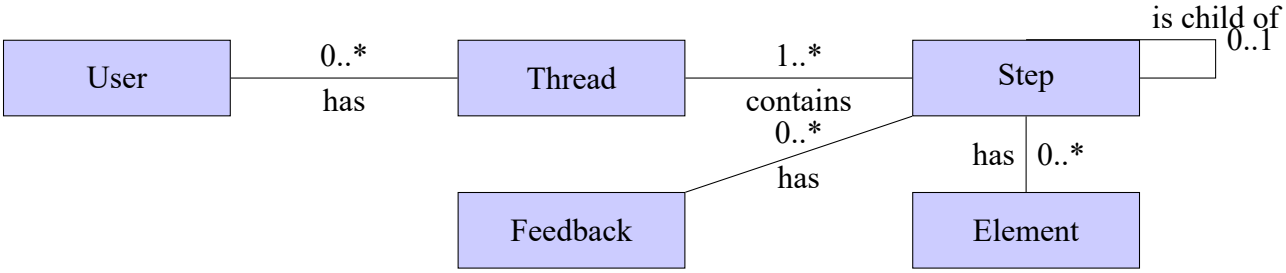
Σχήμα 4.2: Αρχιτεκτονική WebText2SQL

4.2.3 Διαγράμματα Σχέσεων Οντοτήτων (ERD)

Παρακάτω παρατίθενται 2 διαγράμματα σχέσεων οντοτήτων (ERD) που απεικονίζουν την δομή της βάσης δεδομένων του WebText2SQL. Το πρώτο διάγραμμα αφορά την αποθήκευση των χρηστών και των συνδέσεών τους με τις βάσεις δεδομένων, ενώ το δεύτερο διάγραμμα αφορά την αποθήκευση των συνομιλιών και των ερωτημάτων που εκτελούνται.



Σχήμα 4.3: Διάγραμμα Σχέσεων Οντοτήτων (ERD) της Βάσης Δεδομένων για το WebText2SQL



Σχήμα 4.4: Διάγραμμα Σχέσεων Οντοτήτων (ERD) του Chainlit

4.3 Υλοποίηση Back End

Το Back End της εφαρμογής WebText2SQL είναι υπεύθυνο για την οργάνωση και διαχείριση της αυθεντικοποίησης των χρηστών, την διαχείριση των βάσεων δεδομένων, την μετατροπή ερωτημάτων φυσικής γλώσσας σε SQL ερωτήματα μέσω τεχνητής νοημοσύνης, και την εκτέλεση αυτών των ερωτημάτων στις συνδεδεμένες βάσεις δεδομένων. Η υλοποίηση του βασίζεται σε μοντέρνες τεχνολογίες και αρχιτεκτονικές προσεγγίσεις της Python, που επιτρέπουν την ασύγχρονη λειτουργία, το διαχωρισμό των ευθυνών (separation of concerns) και την εύκολη συντήρηση του κώδικα μέσω δομοστοιχείων (modularization).

4.3.1 Τεχνολογίες και Αρχιτεκτονική

Το Back End της εφαρμογής σχεδιάστηκε ως μία μονολιθική (monolithic) υπηρεσία, που χρησιμοποιεί το πλάισιο FastAPI. Αυτή η επιλογή παρέχει γερά θεμέλια για την ανάπτυξη, και βασικών RESTful APIs, αλλά και επικοινωνιών WebSocket για την διαχείριση της διεπαφής συνομιλίας (chat interface) του Chainlit.
Οι βασικές τεχνολογίες που χρησιμοποιούνται είναι:

- **FastAPI**: Ένα μοντέρνο, γρήγορο (high-performance) πλαίσιο για την ανάπτυξη APIs με Python. Χρησιμοποιήθηκε για την διαχείριση της σύνδεσης και εγγραφής χρηστών και ώστε να σερβίρει την εσωτερική εφαρμογή του Chainlit.

- **SQLModel**: Μία βιβλιοθήκη γαι την αλληλεπίδραση με SQL βάσεις δεδομένων, με κώδικα Python, μέσω κλάσεων (classes) και αντικειμενοστραφούς προγραμματισμού (OOP). Χρησιμοποιείται για αντικειμενο-σχεσιακή απεικόνιση (Object-Relational Mapping - ORM), ορίζοντας τα μοντέλα δεδομένων `AppUser` και `UserConnection` και διαχειρίζεται την σύνδεση με την βάση δεδομένων της εφαρμογής.

- **Psycopg 3 & PyMySQL**: Οδηγοί βάσεων δεδομένων για την σύνδεση με PostgreSQL και MySQL αντίστοιχα. Αυτές οι βιβλιοθήκες χρησιμοποιούνται για την χαμηλού επιπέδου (low-level) αλληλεπίδραση με τις βάσεις δεδομένων, επιτρέποντας την εκτέλεση ερωτημάτων SQL και την ανάκτηση αποτελεσμάτων.

- **OpenAI API**: Η επίσημη βιβλιοθήκη της OpenAI στην Python, που χρησιμοποιείται για την αλληλεπίδραση με τα μεγάλα γλωσσικά μοντέλα (LLMs) της OpenAI. Χρησιμοποιείται για την αποστολή ερωτημάτων και τη λήψη απαντήσεων από το μοντέλο, που μετατρέπει τα ερωτήματα φυσικής γλώσσας σε SQL ερωτήματα, μέσω του μοντέλου `gpt-4o-mini`.

- **SSHTunnel**: Μία βιβλιοθήκη για την δημιουργία και διαχείριση SSH tunnels, που επιτρέπει την ασφαλή σύνδεση σε βάσεις δεδομένων που δεν είναι δημόσια προσβάσιμες.

- **Passlib**: Μία εκτενής βιβλιοθήκη κρυπτογράφησης, που χρησιμοποιήθηκε για την ασφαλή κρυπτογράφηση και απόκρυπτογράφηση των κωδικών πρόσβασης των χρηστών.

- **ItsDangerous**: Μία βιβλιοθήκη για την υπογραφή δεδομένων, ώστε να εξασφαλιστεί η ακεραιότητα και η αυθεντικότητα τους. Χρησιμοποιείται για την δημιουργία και επαλήθευση των αναγνωριστικών συνεδρίας (session identifiers) που αποθηκεύονται στα cookies του περιηγητή του χρήστη.

4.3.2 Βασικά Στοιχεία

Η λογική του Back End είναι οργανωμένη σε ξεχωριστά αρχεία και φακέλους, που διαχωρίζουν τις ευθύνες και διευκολύνουν την συντήρηση του κώδικα.

Web Server και Αυθεντικοποίηση (*main.py*)

Αυτό το αρχείο χρησιμοποιείται ως η πρώτη είσοδος της εφαρμογής.

- Ορίζει τα `/login` και `/register` endpoints χρησιμοποιώντας το FastAPI. Αυτά τα endpoints λαμβάνουν δεδομένα σε μορφή φόρμας (form data) από το Front End, χρησιμοποιούν το δομοστοιχείο `auth.py` για την αυθεντικοποίηση και κρυπτογράφηση των κωδικών πρόσβασης και αλληλεπιδρούν με τους `user_controllers` για την διαχείριση των εγγραφών των χρηστών στην βάση δεδομένων.
- Επειτα από μία επιτυχημένη αυθεντικοποίηση, χρησιμοποιεί το `itsdangerous` για να δημιουργήσει ένα υπογεγραμμένο χρονικά περιορισμένο αναγνωριστικό συνεδρίας (session token), το οποίο αποθηκεύεται στα cookies του περιηγητή του χρήστη.
- Φορτώνει την εφαρμογή του Chainlit στο `/chainlit` endpoint, παραδίδοντας την ευθύνθηνη της διαχείρισης της διεπαφής συνομιλίας (chat interface) και των αλληλεπιδράσεων του χρήστη όταν ο χρήστης έχει αυθεντικοποιηθεί. Η αυθεντικοποίηση για τις συνδέσεις WebSocket του Chainlit γίνεται μέσω της συνάρτησης `@cl.header_auth_callback`, στο αρχείο `chainlit_app.py`.

Διαχείριση Χρηστών και Συνδέσεων (*/user_controllers*)

Το πακέτο αυτό είναι υπεύθυνο για όλες τις low-level αλληλεπιδράσεις με την βάση δεδομένων της εφαρμογής (και όχι με τις συνδεδεμένες βάσεις δεδομένων των χρηστών).

- **Μοντέλα (*/models*)**: Ορίζει τα μοντέλα δεδομένων `AppUser` και `UserConnection` χρησιμοποιώντας το SQLmodel. Αυτά τα μοντέλα αντιπροσωπεύουν τους πίνακες, της βάσης δεδομένων της εφαρμογής, `APP_USERS` και `USER_CONNECTIONS` αντίστοιχα.
- **Ελεγκτές (Controllers)**: Τα αρχεία `app_users.py` και `user_connections.py` παρέχουν ένα καθαρό επίπεδο αφαίρεσης (Abstraction Layer) για την αλληλεπίδραση με την βάση δεδομένων. Αυτά τα αρχεία περιέχουν συναρτήσεις για την δημιουργία, ανάκτηση και διαγραφή χρηστών και συνδέσεων (CRUD operations) π.χ. `get_app_user_by_email`, `insert_user_connection` κ.α.

Διαχείριση Συνδέσεων Βάσεων Δεδομένων (*connection_controller.py*, *connection_factory.py*, */db_controllers*)

Αυτό είναι ένα από τα πιο περιπλοκα μέρη της εφαρμογής, καθώς είναι υπεύθυνο για την διαχείριση των συνδέσεων στις βάσεις δεδομένων των χρηστών, και την εκτέλεση ερωτημάτων SQL σε αυτές.

- **Connection Factory (*connection_factory.py*)**: Ένα απλό μοντέλο εργοστασίου (factory pattern)[51] που επιστρέφει ένα στιγμιότυπο (instance) του κατάλληλου ελεγκτή σύνδεσης (connection

controller) για την βάση δεδομένων του χρήστη (`MySQLController` ή `PostgresController`), με βάση μία συμβολοσειρά ("mysql" ή "postgres"). Αυτό επιτρέπει την εύκολη προσθήκη υποστήριξης για νέες βάσεις δεδομένων στο μέλλον και διαχωρίζει την λογική της εφαρμογής από τις λεπτομέρειες της σύνδεσης με τις βάσεις δεδομένων.

- **Ελεγκτές Συνδέσεων (*/db_controllers*)**:
  - Μία αφηρημένη κλάση `BaseDBController` ορίζει τις κοινές λειτουργίες που πρέπει να υποστηρίζουν όλοι οι ελεγκτές βάσεων δεδομένων, όπως:
    * `get_available_dbs`: Επιστρέφει τις διαθέσιμες βάσεις δεδομένων ή σχήματα του χρήστη.
    * `get_db_metadata`: Επιστρέφει τα μεταδεδομένα της βάσης δεδομένων, όπως πίνακες, στήλες και σχέσεις.
    * `execute_query`: Εκτελεί ένα ερώτημα SQL και επιστρέφει τα αποτελέσματα.
  - Οι κλάσεις `MySQLController` και `PostgresController` κληρονομούν από την `BaseDBController` και υλοποιούν όποιες λειτουργίες απαιτούν εξειδικευμένη λογική για τις αντίστοιχες βάσεις δεδομένων. Αυτές οι κλάσεις χρησιμοποιούν τους οδηγούς `PyMySQL` και `Psycopg 3`, αντίστοιχα, για την σύνδεση και εκτέλεση ερωτημάτων στις βάσεις δεδομένων.

- **Διαχείριση Συνδέσεων (*connection_controller.py*)**: Αυτό το module είναι υπεύθυνο για την αρχική σύνδεση στις βάσεις δεδομένων των χρηστών.

  Η συνάρτηση του, `try_establish_connection`, διαχειρίζεται την λογική σύνδεσης, συμπεριλαμβανομένης της επαλήθευσης των πληροφοριών σύνδεσης, και της δημιουργίας ενός SSH tunnel όπου είναι απαραίτητο, χρησιμοποιώντας έναν `SSHTunnelForwarder` μέσω της βιβλιοθήκης `SSHTunnel`.

Ενσωμάτωση της Τεχνητής Νοημοσύνης (*ai_controller.py*)

Αυτό το module είναι ένα απλό περιτύλιγμα γύρω από το API της OpenAI.

- Η συνάρτηση `get_ai_response` λαμβάνει μορφοποιημένο κείμενο, το στέλνει στο μοντέλο `gpt-4o-mini` της OpenAI μέσω ενός ασύγχρονου αιτήματος (asynchronous request) και επιστρέφει την απάντηση του μοντέλου.

- Είναι παραμετροποιημένο με τις ρυθμίσεις του μοντέλου και το API key της OpenAI, που φορτώνται με ασφαλή τρόπο από τις μεταβλητές περιβάλλοντος (environment variables).

4.3.3 Ροή Διαχείρισης Ερωτημάτων και Λογικής

Η ροή διαχείρισης ερωτημάτων και της λογικής, οργάνωνεται μέσα από το αρχείο `chainlit_app.py`, το οποίο λειτουργεί ως ο κεντρικός διαχειριστής των συνομιλιών της εφαρμογής.

1. **Αυθεντικοποίηση**: Ο διαχειριστής γεγονότων (event handler) `@cl.header_auth_callback` λαμβάνει κάθε αίτημα σύνδεσης στην εφαρμογή Chainlit. Διαβάζει το session cookie του

χρήστη, επαληθεύει την αυθεντικότητα και την ημερομηνία λήξης του, μέσω της βιβλιοθήκης `itsdangerous`. Εάν είναι έγκυρο, ανακτά τον χρήστη από τη βάση δεδομένων και δημιουργεί ένα αντικείμενο `cl.User` το οποίο αποθηκεύεται στη συνεδρία (session).

2. **Εγκαθίδρυση Συνδεσης**: Ο διαχειριστής γεγονότων `@cl.on_chat_start` πυροδοτεί μία ελεγχόμενη ροή, όπου το back end ρωτά τον χρήστη να επιλέξει μία αποθηκευμένη σύνδεση βάσης δεδομένων μέσω του front end. Το module `chainlit_controller.py` διαχειρίζεται αυτή την διάδραση, βήμα προς βήμα, συλλέγοντας τις εισόδους του χρήστη και καλώντας τον `connection_controller.py` για να επαληθεύσει την σύνδεση. Εάν η σύνδεση είναι επιτυχής, το σχήμα της βάσης δεδομένων ανακτάται και αποθηκεύεται στην συνεδρία.

3. **Εκτέλεση Ερωτημάτων Φυσικής Γλώσσας**: Ο διαχειριστής γεγονότων `@cl.on_message` αποτελεί τον πυρήνα της λειτουργικότητας της εφαρμογής. Όταν ένας χρήστης στέλνει ένα μήνυμα, εκτελούνται τα εξής βήματα:

   (α′) Ανακτά τις πληροφορίες της τρέχουσας σύνδεσης και του σχήματος από την συνεδρία.

   (β′) Δημιουργεί έναν Ελεγκτή Σύνδεσης (Connection Controller) μέσω του `connection_factory.py` με βάση τον τύπο της βάσης δεδομένων.

   (γ′) Καλεί την συνάρτηση `get_db_metadata` του ελεγκτή για να ανακτήσει τα μεταδεδομένα της βάσης δεδομένων, όπως πίνακες και στήλες. Αυτά τα μεταδεδομένα επίσης αποθηκεύονται στην προσωρινή μνήμη (cache) για μεγαλύτερη απόδοση σε μελλοντικά αιτήματα.

   (δ′) Δημιουργεί ένα λεπτομερές μήνυμα (prompt) για το μοντέλο AI, που περιλαμβάνει:

   - Το σχήμα και την δομή της βάσης δεδομένων.
   - Το πλαίσιο της συνομιλίας (αν είναι ενεργοποιημένο).
   - Το ερώτημα φυσικής γλώσσας του χρήστη.
   - Τον τύπο της βάσης δεδομένων, ωστέ να διασφαλιστεί ότι το παραγόμενο ερώτημα SQL θα είναι συμβατό με την εκάστοτε διάλεκτο (SQL dialect)[4].

   (ε′) Στέλνει το μήνυμα στο μοντέλο AI μέσω του `ai_controller.py` και λαμβάνει την απάντηση.

   (ς′) Το module `str_manipulation.py` χρησιμοποιείται για να καθαρίσει και να μορφοποιήσει το παραγόμενο ερώτημα SQL, αφαιρώντας περιττά κενά και σχόλια.

   (ζ′) Το παραγόμενο ερώτημα SQL εκτελείται στην βάση δεδομένων μέσω του ελεγκτή σύνδεσης, μέσω της συνάρτησης `execute_query`.

   (η′) Τα αποτελέσματα της εκτέλεσης του ερωτήματος SQL μορφοποιούνται και επιστρέφονται στον χρήστη, σε μορφή πίνακα και σε μορφή CSV για λήψη, μέσω του front end.

Αυτή η αρχιτεκτονική σιγουρεύει ότι η εφαρμογή είναι επεκτάσιμη, συντηρήσιμη και εύκολη στην κατανόηση. Κάθε μέρος της λογικής έχει τον δικό του ρόλο και ευθύνες, επιτρέποντας την εύκολη τροποποίηση και προσθήκη νέων λειτουργιών στο μέλλον.

4.4 Υλοποίηση Front End

Η αρχιτεκτονική του Front End της εφαρμογής WebText2SQL είναι σχεδιασμένη με στόχο την λειτουργικότητα και εύκολη ανάπτυξη της εφαρμογής. Χρησιμοποιεί ένα υβριδικό μοντέλο που συνδυάζει την χρήση του FastAPI και των Jinja2 προτύπων (templates) για την αρχική πλατφόρμα αυθεντικοποίησης και διαχείρισης χρηστών, και ύστερα ενσωματώνει το πλαίσιο Chainlit για να παρέχει την βασική, διαδραστική διεπαφή χρήστη (UI) με την μορφή συνομιλίας (chat interface). Αυτή η προσέγγιση διαχωρίζει την στατική και δυναμική πλευρά της εφαρμογής, επιτρέποντας την ευκολότερη συντήρηση και ανάπτυξη.

4.4.1 Τεχνολογίες

Τα στοιχεία της εφαρμογής που αφορούν τον χρήστη είναι χτισμένα με βάση ένα συγκεκριμένο σύνολο τεχνολογιών, που επιτρέπουν την εύκολη ανάπτυξη και συντήρηση της διεπαφής χρήστη, ενώ παράλληλα παρέχουν την ευελιξία να ασχολήθει ο προγραμματιστής με την λογική της εφαρμογής και την αλληλεπίδραση με το back end. Οι τεχνολογίες που χρησιμοποιούνται είναι:

- **FastAPI & Jinja2**: Ο εξυπηρετητής διαδικτύου (web server) FastAPI είναι υπεύθυνος για να σερβίρει τις αρχικές `login.html` και `register.html` σελίδες. Χρησιμοποιεί την μηχανή προτύπων Jinja2 για να αποδώσει αυτές τις απλές φορμές, που διαχειρίζονται την αυθεντικοποίηση και την εγγραφή χρηστών.

- **Chainlit**: Το Chainlit είναι το κύριο πλαίσιο για την βασική διεπαφή χρήστη (UI) της εφαρμογής. Αφού ο χρήστης έχει αυθεντικοποιηθεί, ανακατευθύνεται στο περιβάλλον του Chainlit, το οποίο διαχειρίζεται ολόκληρη την συνεδρία συνομιλίας (chat session) μέσω μίας σύνδεσης WebSocket. Παρέχει μία πλήρη, έτοιμη προς χρήση διεπαφή συνομιλίας, και μία σουίτα από διαδραστικά στοιχεία UI που είναι απαραίτητα για την ροή της εφαρμογής.

- **HTML, CSS, JavaScript**: Αυτές οι βασικές τεχνολογίες του διαδικτύου χρησιμοποιούνται για την δημιουργία των στατικών σελίδων και την διαχείριση της διεπαφής χρήστη. Προσαρμοσμένο CSS εφαρμόζεται μέσω του αρχείου `src/static/style.min.css` για να δώσει στην εφαρμογή ένα συγχρονο και ελκυστικό στυλ. Η επωνυμία της εφαρμογής, συμπεριλαμβανομένου του λογότυπου και των χρωμάτων, βρίσκονται στα αρχεία `logo_light.png`, `logo_dark.png`, `favicon.png` και `theme.json` στον φάκελο `src/public/` και χρησιμοποιούνται για να προσαρμόσουν την εμφάνιση της εφαρμογής.

4.4.2 Βασικά Στοιχεία

Το Front End είναι χωρισμένο σε δύο βασικά κομμάτια: την πλατφόρμα αυθεντικοποίησης και διαχείρισης χρηστών, και την διεπαφή συνομιλίας (chat interface).

Πλατφόρμα Αυθεντικοποίησης και Διαχείρισης Χρηστών

Αυτή είναι η αρχική διεπαφή, και είσοδος των χρήστων στην εφαρμογή. Αποτελείται από δύο βασικές σελίδες:

- **Login Page** (`templates/login.html`): Μία απλή HTML φόρμα που λαμβάνει την διεύθυνση ηλεκτρονικού ταχυδρομείου και τον κωδικό πρόσβασης του χρήστη. Όταν ο χρήστης υποβάλει την φόρμα, ένα αίτημα POST αποστέλλεται στον εξυπηρετητή FastAPI για να επαληθεύσει τα στοιχεία του χρήστη, στην υποδιεύθυνση `/login` η οποία διαχειρίζεται στο αρχείο `src/main.py`.
- **Registration Page** (`templates/register.html`): Μία παρόμοια φόρμα που επιτρέπει στους νέους χρήστες να εγγραφούν στην εφαρμογή. Αυτή τη φορά, όμως, το αίτημα POST αποστέλλεται στην υποδιεύθυνση `/register`.

Αυτές οι σελίδες είναι, επίτηδες, απλές και δεν περιέχουν δυναμικά στοιχεία, καθώς αποσκοπούν στην γρήγορη και εύκολη αυθεντικοποίηση των χρηστών και στην μετάβαση τους στην κεντρική διεπαφή συνομιλίας (chat interface).

Διεπαφή Συνομιλίας (Chat Interface)

Έπειτα από μία επιτυχημένη αυθεντικοποίηση, οι χρήστες ανακατευθύνονται στην υποδιεύθυνση `/chainlit`, της οποίας η λογική της εφαρμογής υπάρχει στο αρχείο `src/chainlit_app.py`. Ολόκληρη η διεπαφή συνομιλίας οργανώνεται μέσω στοιχείων του Chainlit:

- **Αρχική Επαφή**: Όταν ξεκινάει μία καινούργια συνομιλία (`@cl.on_chat_start`), εμφανίζονται στον χρήστη, μέσω ενός `cl.AskActionMessage`, 3 κουμπιά `cl.Action`:
  - **"Connect to a new database"**: Για να συνδεθεί σε μία νέα βάση δεδομένων
  - **"Reconnect to a previously connected database"**: Για να επανασυνδεθεί σε μία βάση δεδομένων στην οποία έχει συνδεθεί στο παρελθόν
  - **"Delete an existing connection"**: Για να διαγράψει μία αποθηκευμένη σύνδεση βάσης δεδομένων
- **Συλλογή Πληροφοριών Σύνδεσης**: Το σύστημα βασίζεται σε μία σειρά από παρόρμησεις (prompts) για να συλλέξει τις απαραίτητες πληροφορίες σύνδεσης βάσης δεδομένων από τον χρήστη. Ενδεικτικά:
  - `cl.AskUserMessage`: Χρησιμοποιείται συνεχόμενα για να ζητήσει από τον χρήστη να εισάγει πληροφορίες όπως το όνομα της βάσης δεδομένων, το όνομα χρήστη, τον κωδικό πρόσβασης, κ.λπ.
  - `cl.AskActionMessage`: Χρησιμοποιείται για εμφανίσει κουμπιά επιλογής, όπως η επιλογή του σχήματος της βάσης δεδομένων, η επιλογή της αποθηκευμένης σύνδεσης βάσης δεδομένων ή ο τύπος της βάσης δεδομένων (MySQL ή PostgreSQL).
- **Εκτέλεση Ερωτημάτων SQL και Αποτελέσματα**: Η κύρια λειτουργία της διεπαφής συνομιλίας είναι η μετατροπή ερωτημάτων φυσικής γλώσσας σε SQL ερωτήματα και η εκτέλεση τους.
  - Σε κάθε μήνυμα του χρήστη, ενεργοποιείται ο διαχειριστής γεγονότων (event handler) `@cl.on_message`.

- Μετά την ολοκλήρωση της διαδικασίας στο Back End, μία απάντηση δημιουργείται και αποστέλλεται πίσω στον χρήστη μέσω της συνάρτησης `cl.Message`. Αυτή περιέχει το παραγόμενο ερώτημα SQL (μορφοποιημένο σε ένα κουτί κώδικα) και μία σύνοψη των αποτελεσμάτων.
- Τα αποτελέσματα εμφανίζονται με 2 τρόπους:
  * Έναν πίνακα αποτελεσμάτων, ο οποίος εμφανίζει τα πρώτα 10 αποτελέσματα του ερωτήματος SQL.
  * Ένα κουμπί για την λήψη του πλήρους αποτελέσματος σε μορφή CSV.

  Αυτή η προσέγγιση επιτρέπει, και μία γρήγορη ανατροφοδότηση στον χρήστη, αλλα και τα πλήρη αποτελέσματα του ερωτήματος SQL, χωρίς να χρειάζεται να φορτώσει ολόκληρη την σελίδα.

4.4.3 Ροή Αλληλεπίδρασης Χρήστη

Η ροή αλληλεπίδρασης του χρήστη με την εφαρμογή WebText2SQL είναι μία διαδικασία που περιλαμβάνει τα εξής βήματα:

1. **Αυθεντικοποίηση Χρήστη**: Ο χρήστης συνδέεται ή εγγράφεται στην εφαρμογή μέσω των σελίδων που παρέχονται από το FastAPI.

2. **Ανακατεύθυνση στο Chainlit**: Μετά την επιτυχή αυθεντικοποίηση, ο χρήστης ανακατευθύνεται στην διεπαφή συνομιλίας του Chainlit.

3. **Επιλογή Σύνδεσης Βάσης Δεδομένων**: Ο χρήστης επιλέγει αν θα συνδεθεί σε μία νέα βάση δεδομένων, θα επανασυνδεθεί σε μία αποθηκευμένη σύνδεση ή θα διαγράψει μία υπάρχουσα σύνδεση.

4. **Συλλογή Πληροφοριών Σύνδεσης**: Εάν ο χρήστης επιλέξει να συνδεθεί σε μία νέα βάση δεδομένων, το σύστημα θα ζητήσει τις απαραίτητες πληροφορίες σύνδεσης, όπως το όνομα της βάσης δεδομένων, το όνομα χρήστη, τον κωδικό πρόσβασης, κ.λπ.

   Σε περίπτωση που επιλέξει να συνδεθεί σε μία νέα βάση δεδομένων, το σύστημα θα του παρουσιάσει όλες τις διαθέσιμες επιλογές σύνδεσης και θα κληθεί να επιλέξει μία από αυτές. Εάν δεν υπάρχουν αποθηκευμένες συνδέσεις, θα του ζητηθεί να εισάγει τις πληροφορίες σύνδεσης σαν να συνδέεται σε μία νέα βάση δεδομένων.

5. **Επιλογή Σχήματος**: Στον χρήστη παρουσιάζονται τα διαθέσιμα σχήματα της βάσης δεδομένων, και θα πρέπει να επιλέξει ένα από αυτά για να εκτελέσει ερωτήματα.

6. **Αποστολή Ερωτήματος Φυσικής Γλώσσας**: Ο χρήστης πλέον μπορεί να εισάγει ερωτήματα φυσικής γλώσσας στην διεπαφή συνομιλίας συσχετιζόμενα με την επιλεγμένη βάση δεδομένων και το σχήμα της.

7. **Λήψη Αποτελεσμάτων**: Το σύστημα θα εμφανίσει το παραγόμενο ερώτημα SQL και τα αποτελέσματα του στην διεπαφή συνομιλίας, περιμένοντας την ανατροφοδότηση ή την επόμενη ερώτηση του χρήστη.

Αυτή η στρατηγική υλοποίησης, βασισμένη κυρίως στο Chainlit, επιτρέπει στην εργασία να παρέχει μία διαδραστική και οικεία διεπαφή χρήστη, που είναι εύκολη στην χρήση και κατανοητή για τους περισσότερους χρήστες, χωρίς να απαιτεί παραπάνω πόρους για την ανάπτυξη και συντήρηση ενός πλήρως προσαρμοσμένου front end από την αρχή.

Κεφάλαιο 5

Παρουσίαση του WebText2SQL

5.1 Εισαγωγή

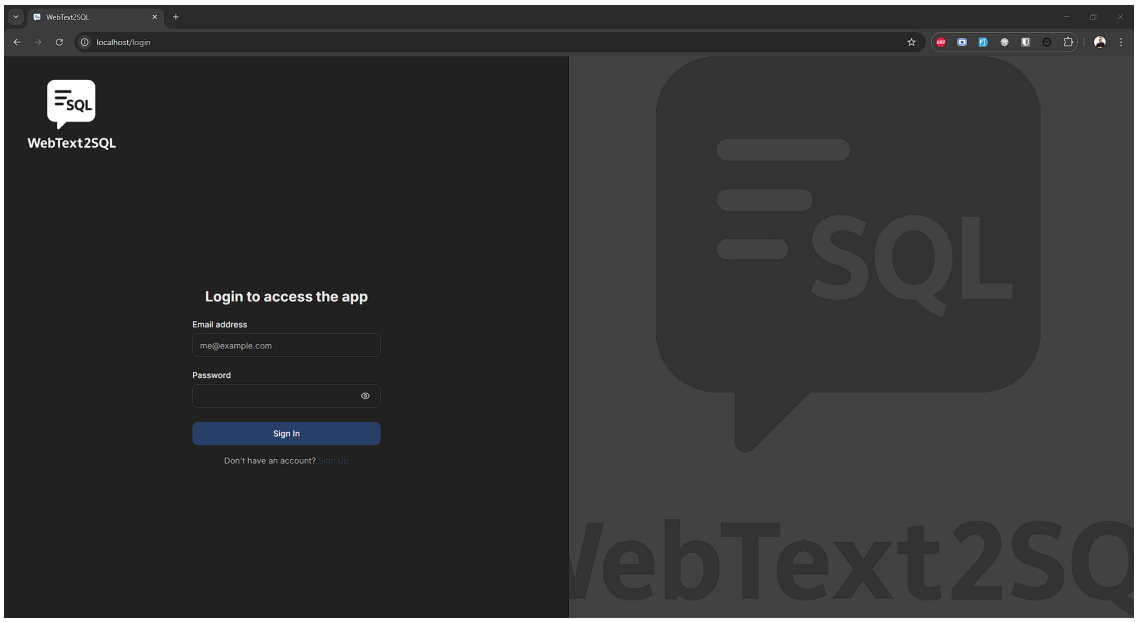
Στο κεφάλαιο αυτό, πραγματοποιείται μια αναλυτική παρουσίαση της εφαρμογής WebText2SQL. Σκοπός είναι η λεπτομερής, βήμα προς βήμα, οπτική καθοδήγηση του αναγνώστη στη διεπαφή χρήστη (UI) της εφαρμογής και στις λειτουργίες της. Η παρουσίαση ακολουθεί τη φυσική ροή πλοήγησης ενός τυπικού χρήστη, ξεκινώντας από την αυθεντικοποίηση, προχωρώντας στη διαχείριση και επιλογή συνδέσεων βάσεων δεδομένων, και καταλήγοντας στη δημιουργία και εκτέλεση ερωτημάτων SQL.

5.2 Αυθεντικοποίηση Χρήστη

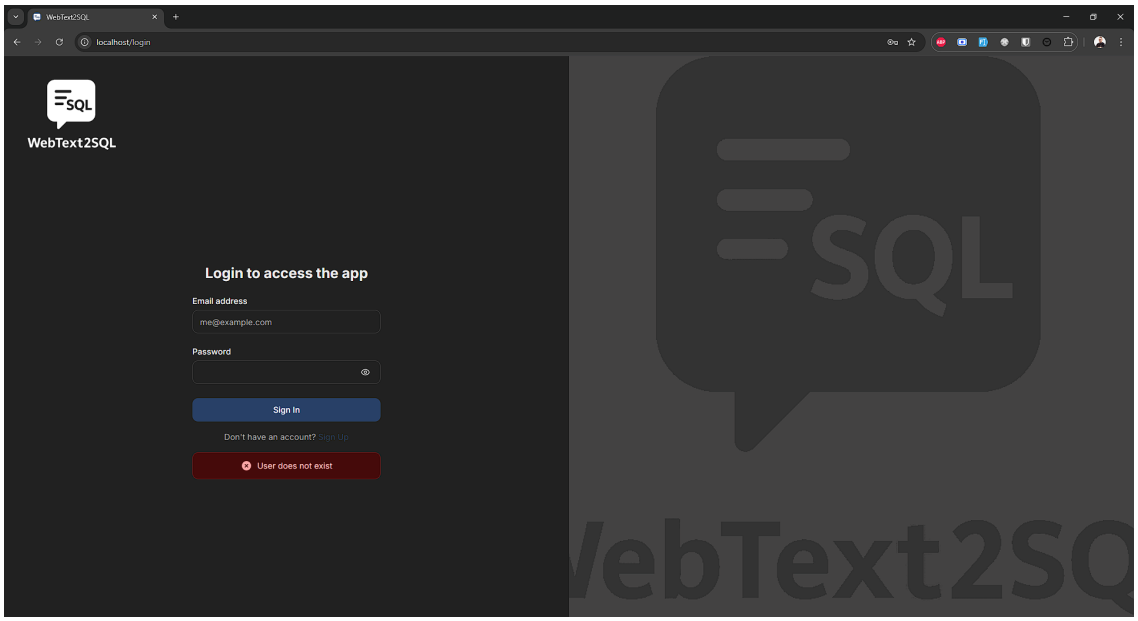
Η πρόσβαση στην εφαρμογή προϋποθέτει την ταυτοποίηση του χρήστη μέσω ενός απλού συστήματος αυθεντικοποίησης, το οποίο διασφαλίζει ότι μόνο εξουσιοδοτημένοι χρήστες μπορούν να χρησιμοποιήσουν την εφαρμογή και την εξατομικευμένη εμπειρία της, δηλαδή ο κάθε χρήστης έχει πρόσβαση μόνο στις δικές του συνδέσεις βάσεων δεδομένων και ερωτήματα.

5.2.1 Είσοδος/Σύνδεση Χρήστη

Οι εγγεγραμμένοι χρήστες μπορούν να εισέλθουν στο σύστημα από την αρχική σελίδα της εφαρμογής ή την σελίδα `/login` (βλ. σχήμα 5.1). Αφού εισάγουν τα διαπιστευτήριά τους, το backend, υλοποιημένο σε FastAPI, παραλαμβάνει τα στοιχεία μέσω ενός POST αιτήματος. Το σύστημα ελέγχει την εγκυρότητα των διαπιστευτηρίων συγκρίνοντάς το hash του κωδικού πρόσβασης με αυτό που είναι αποθηκευμένο στη βάση δεδομένων.
Σε περίπτωση επιτυχούς αυθεντικοποίησης, δημιουργείται ένα ασφαλές session token, το οποίο αποστέλλεται στον πελάτη και αποθηκεύεται ως cookie, και ακολούθως ο χρήστης ανακατευθύνεται στην κύρια σελίδα της εφαρμογής.
Σε περίπτωση αποτυχίας, εμφανίζεται ένα μήνυμα σφάλματος και ο χρήστης καλείται να προσπαθήσει ξανά (βλ. σχήμα 5.2).



Σχήμα 5.1: Σελίδα εισόδου χρήστη στην εφαρμογή

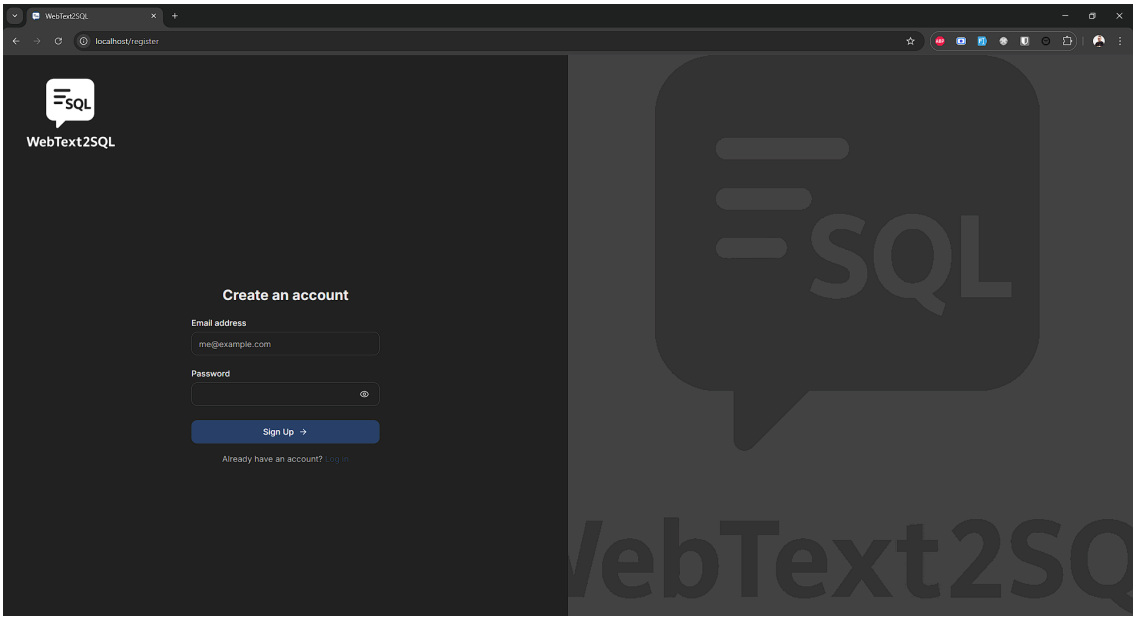


Σχήμα 5.2: Αποτυχία σύνδεσης χρήστη στην εφαρμογή

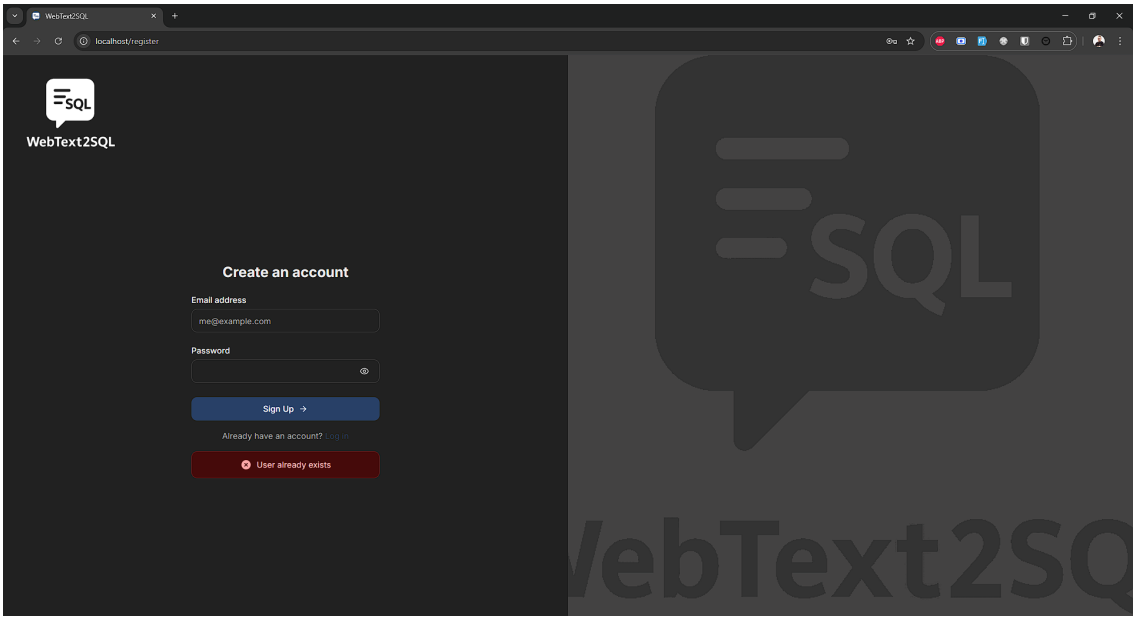
5.2.2 Εγγραφή Νέου Χρήστη

Οι νέοι χρήστες μπορούν να εγγραφούν μέσω της σελίδας `/register`. Η διαδικασία εγγραφής περιλαμβάνει την εισαγωγή ενός μοναδικού ονόματος χρήστη υπό την μορφή email και ενός κωδικού πρόσβασης (βλ. σχήμα 5.3). Με την υποβολή της φόρμας, το backend ελέγχει αν το όνομα χρήστη είναι ήδη σε χρήση.

Αν είναι διαθέσιμο, δημιουργείται ένας νέος χρήστης στη βάση δεδομένων, ο κωδικός πρόσβασης κρυπτογραφείται και αποθηκεύεται με ασφάλεια.
Εάν δεν είναι διαθέσιμο, εμφανίζεται ένα μήνυμα σφάλματος και ο χρήστης καλείται να επιλέξει ένα διαφορετικό όνομα χρήστη (βλ. σχήμα 5.4).



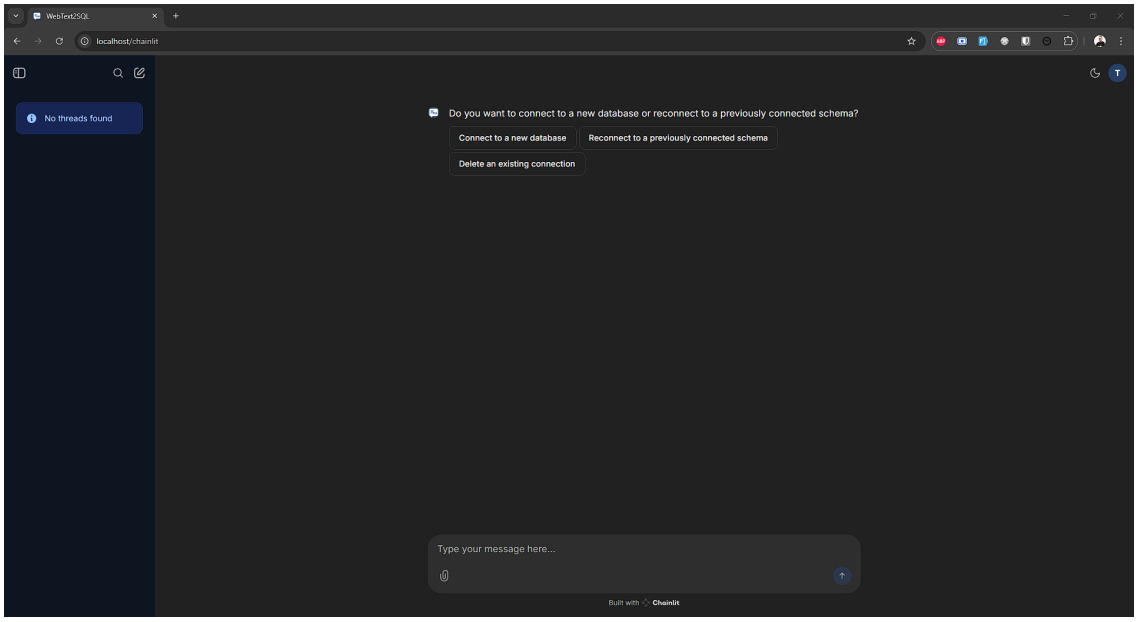
Σχήμα 5.3: Σελίδα εγγραφής νέου χρήστη στην εφαρμογή



Σχήμα 5.4: Αποτυχία εγγραφής νέου χρήστη στην εφαρμογή

5.3 Κεντική Διεπαφή και Διαχείριση Συνδέσεων

Μετά την επιτυχή είσοδο, ο χρήστης μεταφέρεται στην κεντρική οθόνη του chainlit (βλ. σχήμα 5.5). Σε αυτό το σημείο, δεν έχει επιλεγεί ακόμη καμία σύνδεση βάσης δεδομένων. Η εφαρμογή παρουσιάζει στον χρήστη τρεις βασικές επιλογέςγια την διαχείριση των συνδέσεών του (όπως αποτυπώνεται και στο διάγραμμα του σχήματος 4.1)



Σχήμα 5.5: Κεντρική οθόνη της εφαρμογής πριν την επιλογή σύνδεσης

5.3.1 Δημιουργία Νέας Σύνδεσης

Επιλέγοντας την δημιουργία νέας σύνδεσης, η εφαρμογή καθοδηγεί τον χρήστη μέσα από μια σειρά διαδοχικών ερωτήσεων για τη συλλόγη των απαραίτητων πληροφοριών σύνδεσης:

1. **Επιλογή Τύπου Σύνδεσης**: Ο χρήστης επιλέγει μεταξύ μιας απλής TCP/IP σύνδεσης ή μιας σύνδεσης μέσω SSH tunnelling (βλ. σχήμα 5.6).

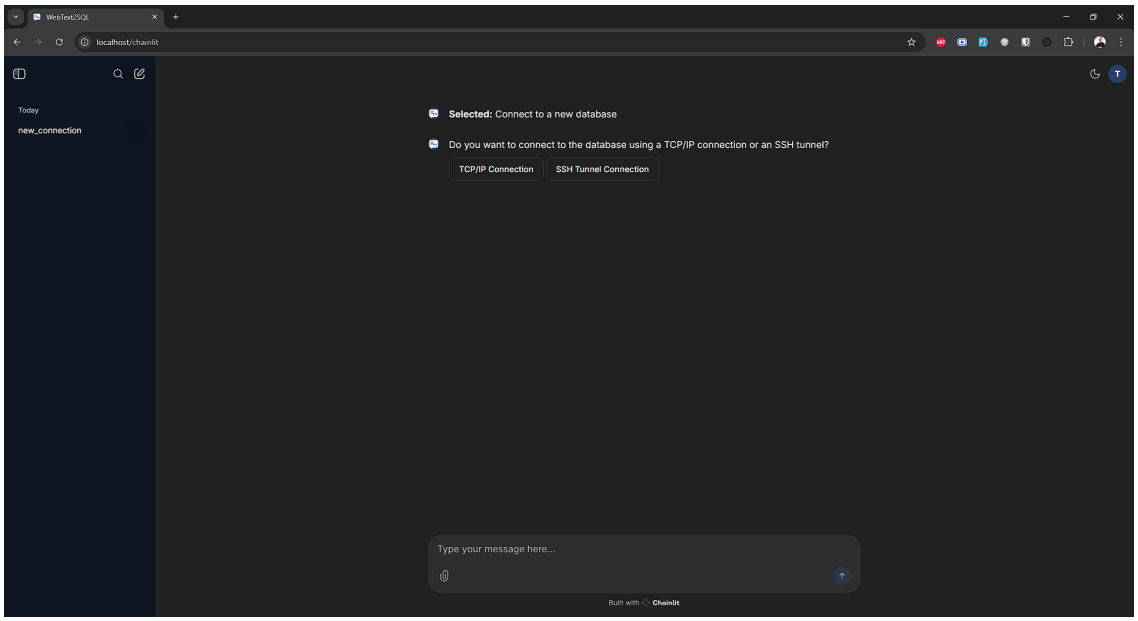
2. **Εισαγωγή Πληροφοριών Σύνδεσης**: Ανάλογα με την επιλογή του, ο χρήστης εισάγει:

   - Για TCP/IP σύνδεση (βλ. επίσης σχήμα 5.7):
     - Διεύθυνση IP ή όνομα κεντρικού υπολογιστή της βάσης δεδομένων.
     - Θύρα σύνδεσης (συνήθως 3306 για MySQL).
     - Όνομα χρήστη και κωδικό πρόσβασης.
     - Τύπο βάσης δεδομένων (MySQL ή PostgreSQL).
     - (Σε περίπτωση PostgreSQL) Το όνομα της βάσης δεδομένων στην οποία θα συνδεθεί (βλ. δεύτερη εικόνα στο σχήμα 5.8)
   - Για SSH σύνδεση (βλ. επίσης σχήμα 5.8):
     - Διεύθυνση IP ή όνομα κεντρικού υπολογισή της SSH σύνδεσης.

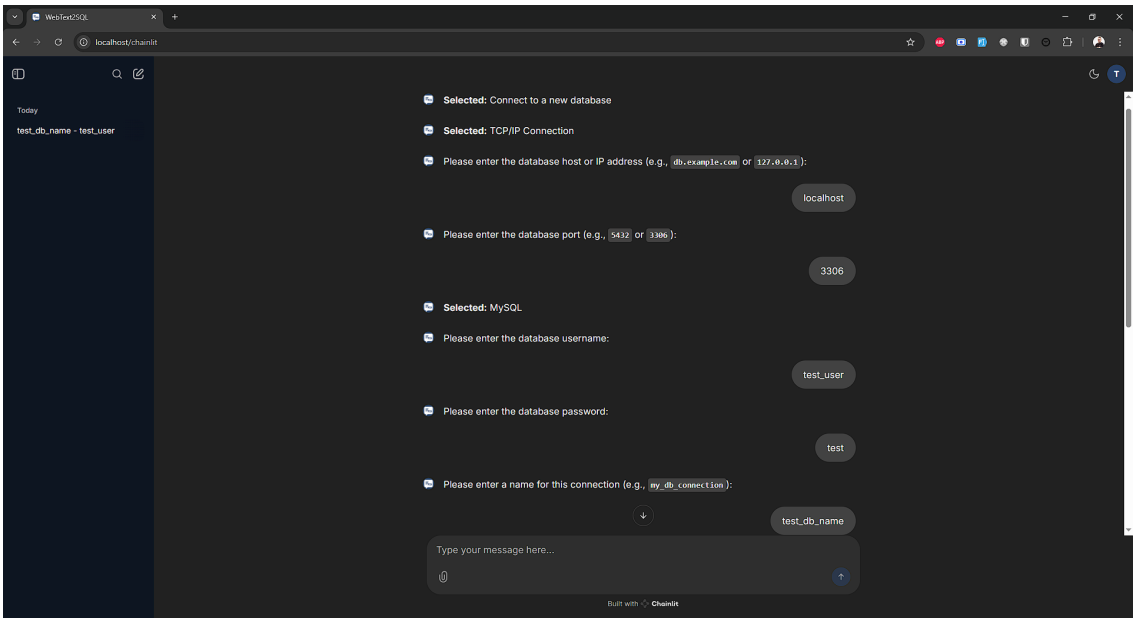
- Την θύρα SSH (συνήθως 22).
- Όνομα χρήστη και κωδικό πρόσβασης για την SSH σύνδεση.
- Τα στοιχεία που περιγράφηκαν παραπάνω για την TCP/IP σύνδεση της βάσης δεδομένων.

3. **Επιλογή Ονόματος Σύνδεσης**: Ο χρήστης δίνει ένα μοναδικό όνομα για την σύνδεση, το οποίο θα χρησιμοποιηθεί για την αναγνώριση της σύνδεσης στο μέλλον.

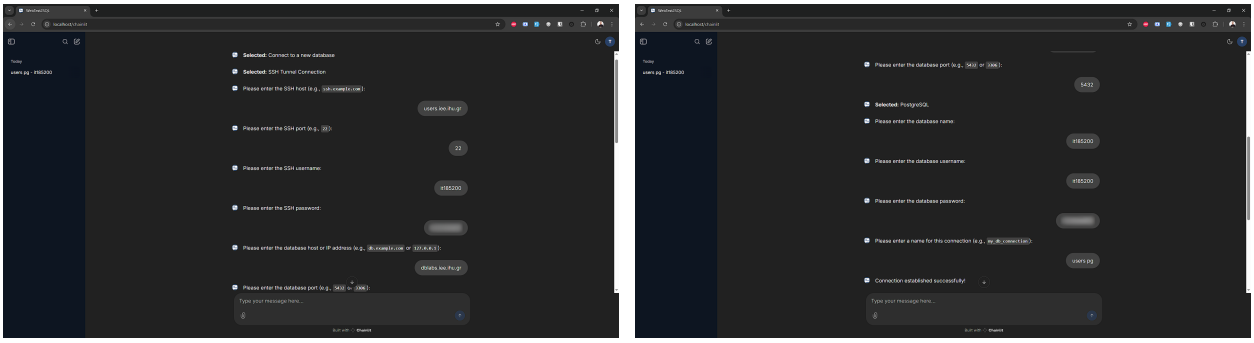
Όλες αυτές οι πληροφορίες αποθηκεύονται στον πίνακα `USER_CONNECTIONS` της βάσης δεδομένων, με το όνομα χρήστη να χρησιμοποιείται ως κλειδί για την συσχέτιση των συνδέσεων με τον χρήστη.



Σχήμα 5.6: Φόρμα δημιουργίας νέας σύνδεσης



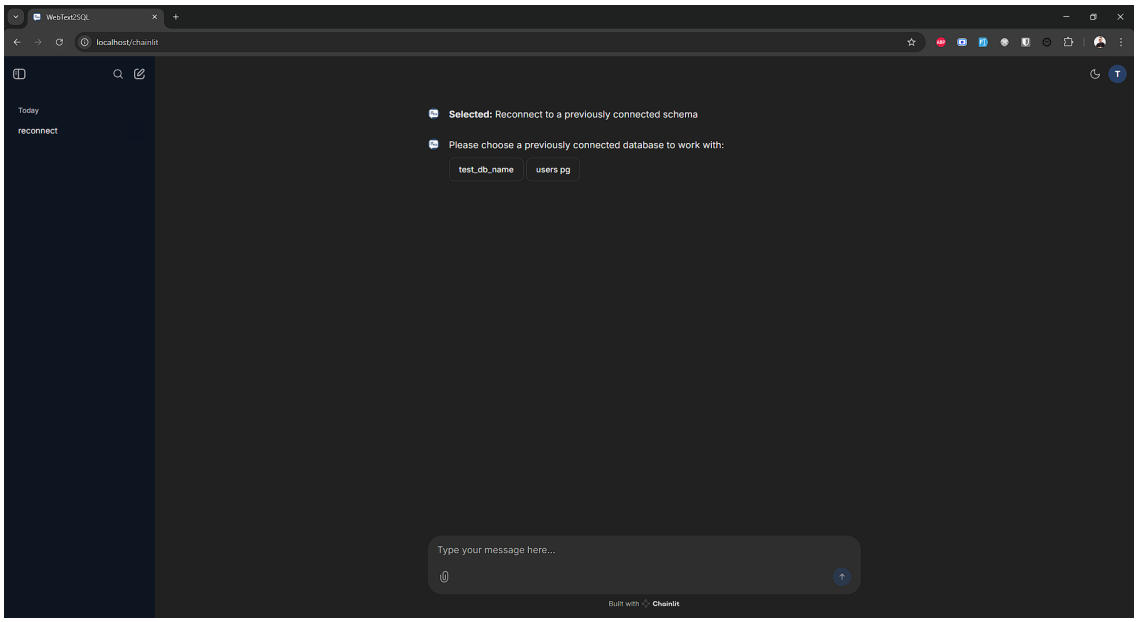
Σχήμα 5.7: Φόρμα δημιουργίας νέας σύνδεσης TCP/IP



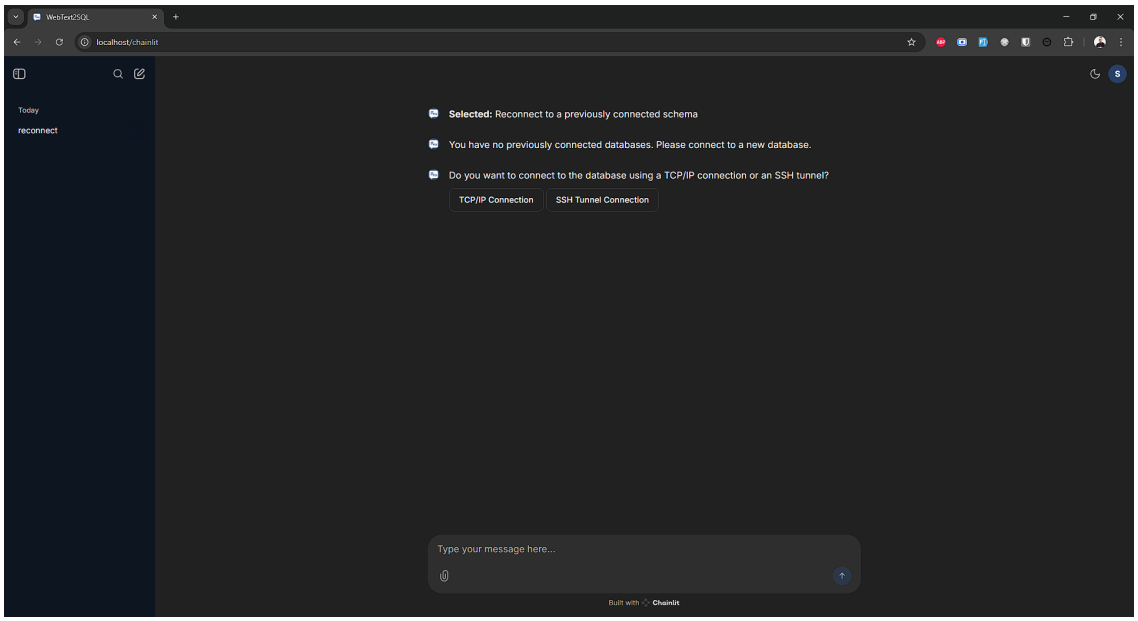
Σχήμα 5.8: Φόρμα δημιουργίας νέας σύνδεσης SSH

5.3.2 Επιλογή Υπάρχουσας Σύνδεσης

Αν ο χρήστης έχει ήδη δημιουργήσει συνδέσεις, μπορεί να επιλέξει μία από αυτές για να ξεκινήσει την αλληλεπίδραση με την βάση δεδομένων (βλ. σχήμα 5.9). Η εφαρμογή ανακτά την λίστα των συνδέσεων από την βάση δεδομένων και τις παρουσιάζει στον χρήστη με την μορφή κουμπιών επιλογής. Με ένα κλικ, η σύνδεση ενεργοποιείται άμεσα.
Σε περίπτωση που ο χρήστης δεν έχει δημιουργήσει καμία σύνδεση, εμφανίζεται ένα μήνυμα που τον καθοδηγεί να δημιουργήσει μια νέα σύνδεση όπως περιγράφηκε παραπάνω (βλ. σχήμα 5.10).



Σχήμα 5.9: Λίστα αποθηκευμένων συνδέσεων

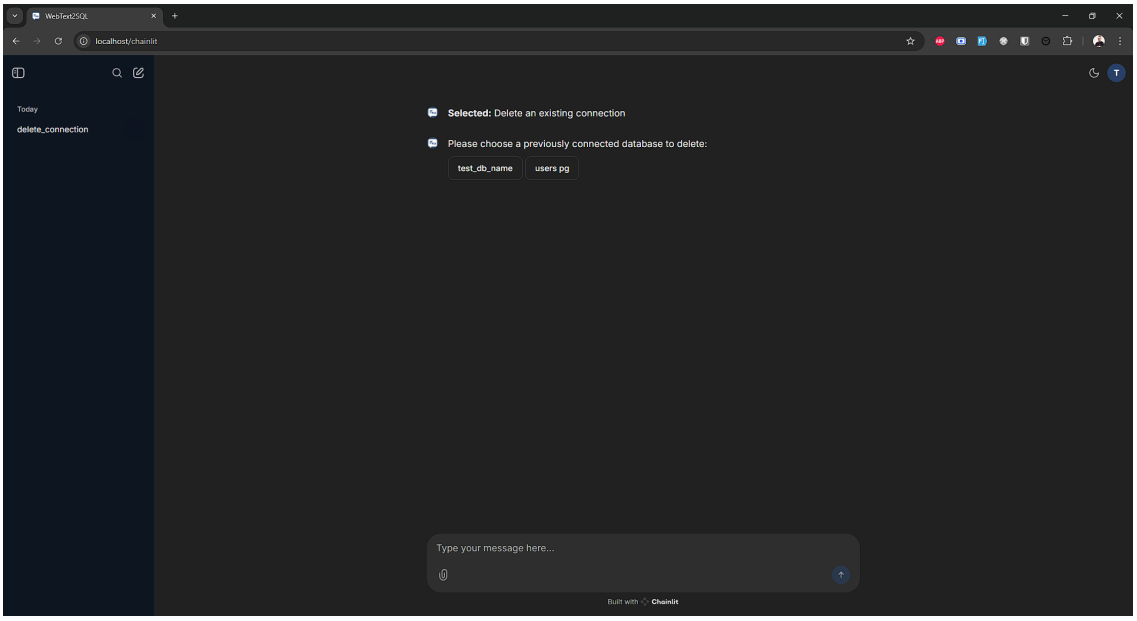


Σχήμα 5.10: Μήνυμα για την έλλειψη αποθηκευμένων συνδέσεων

5.3.3 Διαγραφή Σύνδεσης

Ο χρήστης έχει επίσης τη δυνατότητα να διαγράψει μια αποθηκευμένη σύνδεση. Η διαδικασία είναι παρόμοια με την επανασύνδεση. Ο χρήστης επιλέγει την σύνδεση που θέλει να διαγράψει από την λίστα των αποθηκευμένων συνδέσεων (βλ. σχήμα 5.11). Η αντίστοιχη σύνδεση διαγράφεται από την βάση δεδομένων και μαζί με αυτήν διαγράφεται και όλο το ιστορικό συνομιλιών που σχετίζεται

με αυτήν. Ύστερα από την διαγραφή, ο χρήστης επιστρέφει στην κεντρική οθόνη της εφαρμογής, όπου μπορεί να επιλέξει μια άλλη σύνδεση, να δημιουργήσει μια νέα ή να διαγράψει μια άλλη σύνδεση.

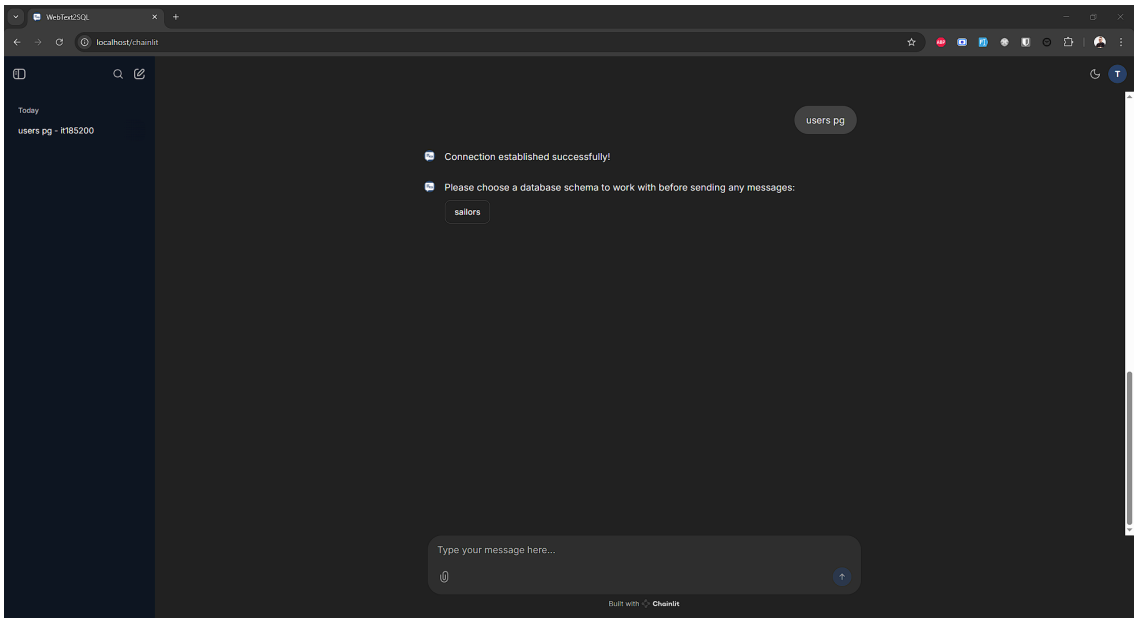


Σχήμα 5.11: Διαγραφή αποθηκευμένης σύνδεσης

5.4 Αλληλεπίδραση με την Βάση Δεδομένων

5.4.1 Επιλογή Σχήματος Βάσης Δεδομένων

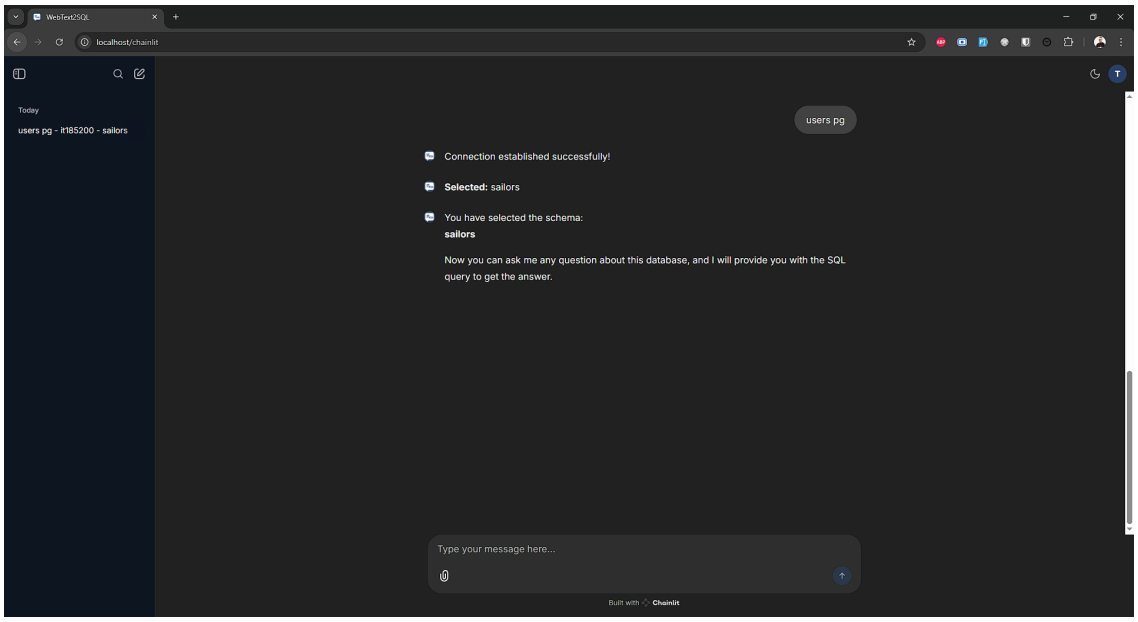
Αφού μια σύνδεση (νέα ή υπάρχουσα) ενεργοποιηθεί, το επόμενο βήμα είναι η επιλογή του συγκεκριμένου σχήματος (database schema) με το οποίο ο χρήστης επιθυμεί να αλληλεπιδράσει. Η εφαρμογή συνδέεται στον απομακρυσμένο server και ανακτά δυναμικά τη λίστα των διαθέσιμων σχημάτων (schema) για τον συγκεκριμένο χρήστη της βάσης, παρουσιάζοντάς τα ως επιλογές μέσω κουμπιών (βλ. σχήμα 5.12).



Σχήμα 5.12: Επιλογή σχήματος βάσης δεδομένων

5.4.2 Διεπαφή Συνομιλίας (Chat Interface)

Αφού ο χρήστης επιλέξει ένα σχήμα, εμφανίζεται το κατάλληλο μήνυμα επιβεβαίωσης και μία προτροπή για την εισαγωγή του πρώτου ερωτήματος (βλ. σχήμα 5.13).



Σχήμα 5.13: Διεπαφή συνομιλίας μετά την επιλογή σχήματος

5.5 Δημιουργία και Εκτέλεση Ερωτημάτων SQL

Αυτή η ενότητα περιγράφει την κεντρική λειτουργία της εφαρμογής μέσα από ένα παράδειγμα χρήσης.

1. **Βήμα 1: Εισαγωγή Ερωτήματος** Ο χρήστης πληκτρολογεί το ερώτημά του σε φυσική γλώσσα, οποιαδήποτε γλώσσα υποστηρίζεται από το μοντέλο, στο πεδίο εισαγωγής κειμένου και πατάει το κουμπί "Αποστολή" ή πατάει το πλήκτρο Enter.

   Για παράδειγμα, ο χρήστης μπορεί να εισάγει το ερώτημα *"Ποιες είναι οι 5 χώρες με τον μεγαλύτερο πληθυσμό;"*

2. **Βήμα 2: Παραγωγή Ερωτήματος SQL** Η εφαρμογή λαμβάνει το ερώτημα. Ανακτά δυναμικά τα μεταδεδομένα της βάσης δεδομένων για το επιλεγμένο σχήμα (ονόματα πινάκων, στήλες, τύποι δεδομένων κ.λπ.), σε περίπτωση που υπάρχουν προηγούμενα μηνύματα και επιτρέπεται αυτή η λειτουργία συλλέγει επίσης και το ιστορικό της συνομιλίας, και τότε τα συνδιάζει με το ερώτημα του χρήστη για να δημιουργήσει ένα λεπτομερές prompt (προτροπή) για το μοντέλο LLM. Το μεγάλο γλωσσικό μοντέλο (LLM) αναλύει το ερώτημα και παράγει ένα αντίστοιχο ερώτημα SQL.

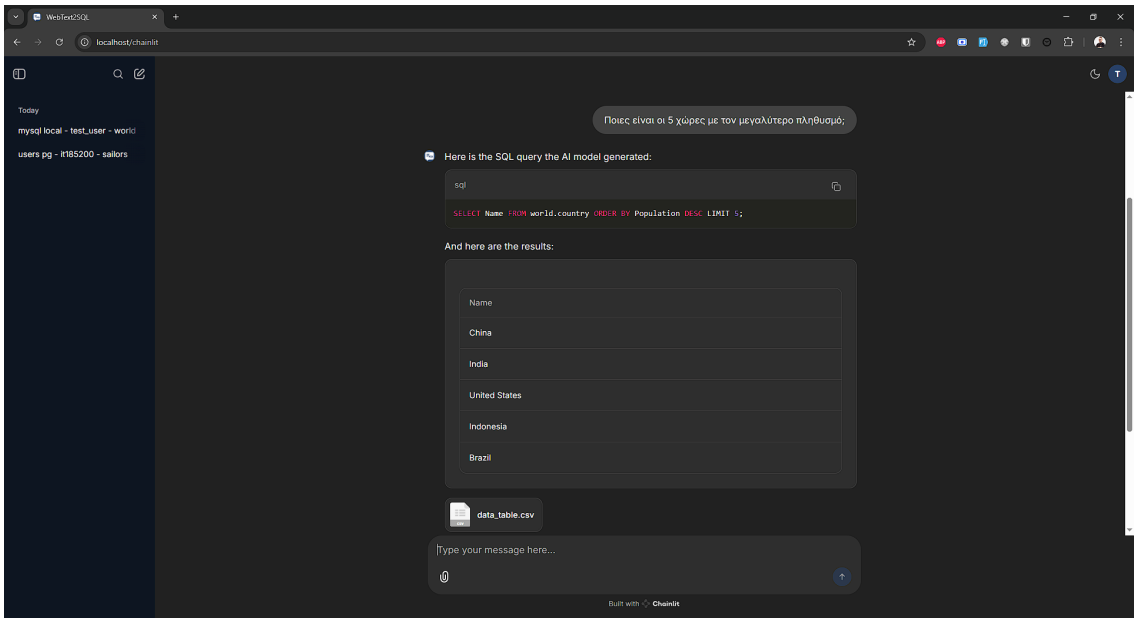
   Για παράδειγμα, το LLM μπορεί να μετατρέψει το ερώτημα σε SQL ως εξής:

```sql
SELECT name, population
FROM countries
ORDER BY population DESC
LIMIT 5;

```

3. **Βήμα 3: Εκτέλεση Ερωτήματος SQL** Το παραγόμενο ερώτημα SQL αποστέλλεται στον διακομιστή βάσης δεδομένων μέσω της ενεργής σύνδεσης. Ο διακομιστής εκτελεί το ερώτημα και επιστρέφει τα αποτελέσματα στην εφαρμογή.

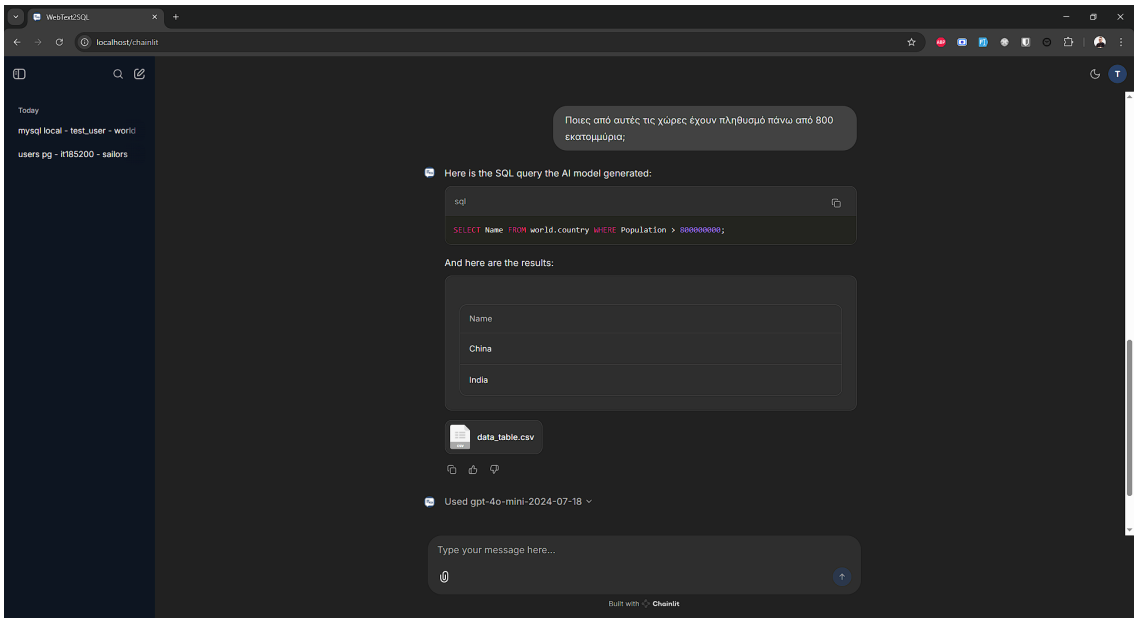
4. **Βήμα 4: Εμφάνιση Αποτελεσμάτων** Τα αποτελέσματα του ερωτήματος μορφοποιούνται σε έναν ευανάγνωστο πίνακα και εμφανίζεται μια προεπισκόπηση των δεδομένων στον χρήστη άμεσα, μαζί με την επιλογή λήψης των αποτελεσμάτων σε μορφή CSV, κάτω από το αρχικό ερώτημα.



Σχήμα 5.14: Παράδειγμα συνομιλίας με ερώτημα και αποτελέσματα

5.5.1 Αξιοποίηση Ιστορικού Συνομιλίας

Η δύναμη της εφαρμογής δεν περιορίζεται σε μεμονωμένα ερωτήματα. Ο χρήστης μπορεί να υποβάλει και ένα επακόλουθο ερώτημα, το οποίο θα χρησιμοποιεί το ιστορικό της συνομιλίας για να παρέχει πιο ακριβείς και σχετικές απαντήσεις.
Για παράδειγμα, αν ο χρήστης υποβάλει το ερώτημα *"Ποιες από αυτές τις χώρες έχουν πληθυσμό πάνω από 800 εκατομμύρια;"*, το LLM θα αναλύσει το ιστορικό συνομιλίας και θα χρησιμοποιήσει τα αποτελέσματα του προηγούμενου ερωτήματος για να δημιουργήσει ένα νέο SQL ερώτημα που θα φιλτράρει τα αποτελέσματα.

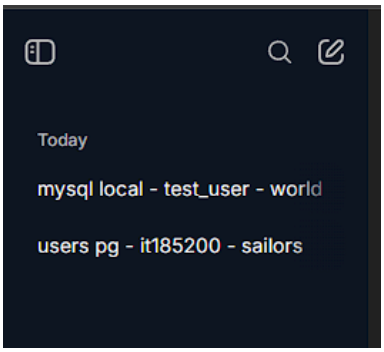


Σχήμα 5.15: Παράδειγμα επακόλουθου ερωτήματος με χρήση ιστορικού συνομιλίας

5.6 Γενικές Λειτουργίες και Ρυθμίσεις

5.6.1 Επαναφορά Προηγούμενης Συνομιλίας

Σε περίπτωση που ο χρήστης επιθυμεί να επαναφέρει μια προηγούμενη συνομιλία, μπορεί να το κάνει μέσω του πλάγιου αριστερού μενού της εφαρμογής. Η εφαρμογή εμφανίζει μια λίστα με όλες τις προηγούμενες συνομιλίες που έχει πραγματοποιήσει ο χρήστης με όλες τις συνδέσεις του. Ο χρήστης μπορεί να επιλέξει οποιαδήποτε συνομιλία για να την επαναφέρει. Αυτό θα φορτώσει την συνομιλία στην τρέχουσα διεπαφή, επιτρέποντας στον χρήστη να δει τα προηγούμενα ερωτήματα και απαντήσεις, καθώς και να συνεχίσει την συνομιλία από το σημείο που την άφησε.



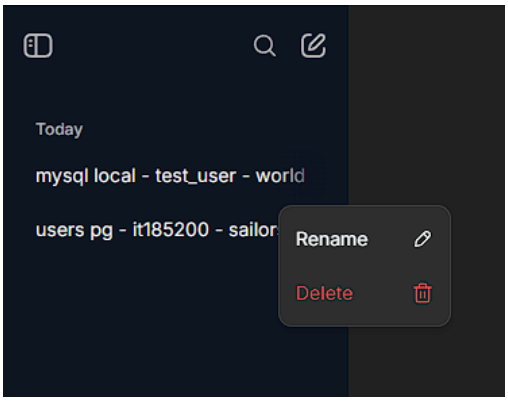
Σχήμα 5.16: Λίστα προηγούμενων συνομιλιών

5.6.2 Διαγραφή Προηγούμενης Συνομιλίας

Ο χρήστης έχει επίσης τη δυνατότητα να διαγράψει μια προηγούμενη συνομιλία (βλ. σχήμα 5.17). Αυτό μπορεί να γίνει είτε άμεσα από την λίστα των συνομιλιών, όπου υπάρχει η επιλογή διαγραφής δίπλα σε κάθε συνομιλία, είτε έμμεσα διαγράφοντας την σύνδεση βάσης δεδομένων, η οποία θα διαγράψει και όλες τις συνομιλίες που σχετίζονται με αυτήν. Η διαγραφή μιας συνομιλίας είναι μόνιμη και δεν μπορεί να αναιρεθεί, επομένως ο χρήστης θα πρέπει να είναι προσεκτικός κατά την επιλογή αυτής της λειτουργίας.

5.6.3 Μετονομασία Συνομιλίας

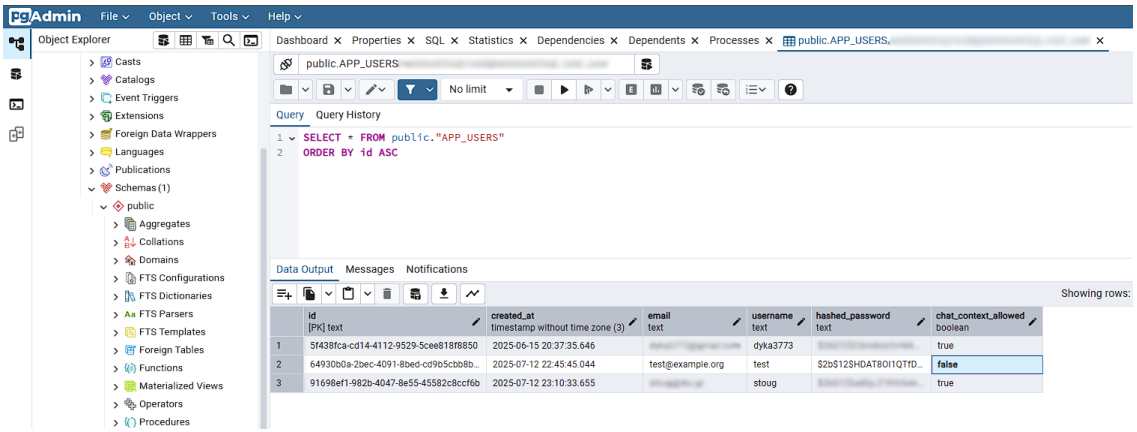
Κάθε συνομιλία έχει ένα προεπιλεγμένο όνομα που βασίζεται στο όνομα της σύνδεσης βάσης δεδομένων, το σχήμα που έχει επελεγεί και τον χρήστη της βάσης δεδομένων (βλ. σχήμα 5.17). Ο χρήστης έχει τη δυνατότητα να μετονομάσει μια συνομιλία για να την κάνει πιο κατανοητή και εύκολα αναγνωρίσιμη. Αυτό μπορεί να γίνει από την λίστα των συνομιλιών, όπου υπάρχει η επιλογή μετονομασίας δίπλα σε κάθε συνομιλία. Η μετονομασία της συνομιλίας δεν επηρεάζει τα δεδομένα της συνομιλίας, αλλά απλώς αλλάζει το όνομα που εμφανίζεται στην λίστα των συνομιλιών.



Σχήμα 5.17: Διαγραφή ή μετονομασία συνομιλίας

5.6.4 Αφαίρεση Δικαιώματος Πρόσβασης στο Ιστορικό Συνομιλίας

Η δυνατότητα χρήσης του ιστορικού συνομιλίας για τη δημιουργία επακόλουθων ερωτημάτων μπορεί να ενεργοποιηθεί ή να απενεργοποιηθεί για κάθε χρήστη. Αυτή η ρύθμιση γίνεται απευθείας στη βάση δεδομένων της εφαρμογής από έναν διαχειριστή. Στον πίνακα `APP_USERS`, η στήλη `chat_context_allowed` ελέχει αυτή τη λειτουργία. Το Σχήμα 5.18 δείχνει ένα παράδειγμα όπου η τιμή έχει τεθεί σε `False`, απενεργοποιώντας τη δυνατότητα για τον συγκεκριμένο χρήστη. Αυτό σημαίνει ότι ο χρήστης δεν θα μπορεί να χρησιμοποιήσει το ιστορικό συνομιλίας για τη δημιουργία επακόλουθων ερωτημάτων, και κάθε νέο ερώτημα θα αντιμετωπίζεται ως ανεξάρτητο από τα προηγούμενα.
Δεν υπάρχει γραφικό περιβάλλον για αυτήν τη ρύθμιση, καθώς θεωρείται διαχειριστική ενέργεια.



Σχήμα 5.18: Απενεργοποίηση δικαιώματος πρόσβασης στο ιστορικό συνομιλίας μέσω της βάσης δεδομένων

Κεφάλαιο 6

Αξιολόγηση Εμπειρίας Χρηστών

6.1 Το σύστημα SUS

Η χρηστικότητα και η εμπειρία χρήστη της εφαρμογής αξιολογήθηκε μέσω της μεθόδου System Usability Scale (SUS) [52]. Η μέθοδος αυτή είναι μια απλή, γρήγορη και αξιόπιστη μέθοδος για την αξιολόγηση της εμπειρίας χρήστη. Αποτελείται από ένα ερωτηματολόγιο 10 ερωτήσεων, όπου οι συμμετέχοντες καλούνται να βαθμολογήσουν κάθε ερώτηση σε μια κλίμακα από 1 έως 5, όπου 1 σημαίνει "διαφωνώ έντονα" και 5 σημαίνει "συμφωνώ έντονα". Η πλήρης λίστα των ερωτήσεων που χρησιμοποιήθηκαν είναι η εξής:

1. Νομίζω ότι θα ήθελα να χρησιμοποιήσω αυτή την εφαρμογή συχνά.
2. Βρήκα την εφαρμογή περιττά πολύπλοκη.
3. Βρήκα την εφαρμογή εύκολη στη χρήση.
4. Νομίζω ότι θα χρειαστώ βοήθεια από κάποιον ειδικό για να χρησιμοποιήσω αυτή την εφαρμογή.
5. Βρήκα τις διάφορες λειτουργίες της εφαρμογής να είναι καλά ενσωματωμένες.
6. Νομίζω ότι υπήρχαν πάρα πολλές ασυνέπειες στην εφαρμογή.
7. Φαντάζομαι ότι οι περισσότεροι άνθρωποι θα μάθαιναν να χρησιμοποιούν αυτή την εφαρμογή πολύ γρήγορα.
8. Βρήκα την εφαρμογή πολύ δύσχρηστη.
9. Ένιωσα άνετος να χρησιμοποιήσω την εφαρμογή.
10. Έπρεπε να μάθω πολλά πράγματα πριν να μπορέσω να χρησιμοποιήσω την εφαρμογή.

Για τον υπολογισμό της βαθμολογίας SUS, στις απαντήσεις των περιττά αριθμημένων ερωτήσεων (1, 3, 5, 7, 9) αφαιρούνται από το σκορ τους το 1, ενώ στις απαντήσεις των ζυγά αριθμημένων ερωτήσεων (2, 4, 6, 8, 10) αφαιρούνται από το 5. Στη συνέχεια, οι βαθμολογιες αυτές προστίθενται και το άθροισμα τους πολλαπλασιάζεται με το 2,5 για να δώσει μια συνολική βαθμολογία SUS

που κυμαίνεται από 0 έως 100 για κάθε συμμετέχοντα. Η μέση βαθμολογία SUS για όλους τους συμμετέχοντες υπολογίζεται στη συνέχεια ως ο μέσος όρος των ατομικών βαθμολογιών SUS. Η βαθμολογία SUS μπορεί να ερμηνευτεί ως εξής:

- 0-25: Πολύ κακή
- 26-51: Κακή
- 52-73: Μέτρια
- 74-85: Καλή
- 86-100: Εξαιρετική

6.2 Αποτελέσματα Αξιολόγησης

Η εφαρμογή αξιολογήθηκε από 16 συμμετέχοντες, οι οποίοι ήταν φοιτητές, επαγγελματίες στον τομέα της πληροφορικής και άλλοι χρήστες με διάφορα επίπεδα εμπειρίας στη χρήση εφαρμογών. Η τελική βαθμολογία SUS που προέκυψε ήταν 83,1 και δείχνει ότι η εφαρμογή έχει καλή χρηστικότητα και εμπειρία χρήστη.
Στον παρακάτω πίνακα παρουσιάζονται οι απαντήσεις των συμμετεχόντων στο ερωτηματολόγιο SUS:

Συμμετέχων	Βαθμολογία
1	82.5
2	77.5
3	77.5
4	92.5
5	55
6	80
7	50
8	92.5
9	100
10	82.5
11	87.5
12	100
13	90
14	90
15	72.5
16	100
Μέσος Όρος	83,1

Πίνακας 6.1: Η βαθμολογία SUS της εφαρμογής WebText2SQL

Κεφάλαιο 7

Συμπεράσματα και Μελλοντικές Επεκτάσεις

7.1 Συμπεράσματα

Η εργασία αυτή επεδίωξε να αντιμετωπίσει την πρόκληση της γεφύρωσης του χάσματος μεταξύ της φυσικής-ανθρώπινης γλώσσας και των δομιμένων ερωτημάτων σε βάσεις δεδομένων. Ο προτεύων σκοπός ήταν να σχεδιαστεί και να υλοποιηθεί μία εφαρμογή ιστού, το WebText2SQL, που να είναι ικανή να μετατρέπει φυσική γλώσσα σε εκτελέσιμα ερωτήματα SQL. Στοχεύσαμε να δημιουργήσουμε μια φιλική προς τον χρήστη πλατφόρμα που να επιτρέπει στους χρήστες, ανεξαρτήτως τεχνικών γνώσεων, να αλληλεπιδρούν με βάσεις δεδομένων χρησιμοποιώντας φυσική γλώσσα.

Μέσω αυτής της εργασίας, έχουμε αποδείξει ότι είναι εφικτό να αναπτύξουμε ένα τέτοιο λειτουργικό και πρακτικό σύστημα. Η εφαρμογή WebText2SQL παρέχει μια πλήρη λύση, που περιλαμβάνει αυθεντικοποίηση χρηστών, διαχείριση πολλαπλών συνδέσεων σε βάσεις δεδομένων διαφορετικών τύπων, και τη δυνατότητα μετατροπής ερωτημάτων φυσικής γλώσσας σε SQL. Η κύρια ιδιαιτερότητα της εφαρμογής είναι η χρήση ενός μεγάλου γλωσσικού μοντέλου, το GPT-4o-mini, το οποίο είναι ικανό να κατανοεί την πρόθεση του χρήστη και να παράγει ακριβή SQL ερωτήματα, τα οποία εκτελούνται άμεσα σε πραγματικές βάσεις δεδομένων του χρήστη.

Η υλοποίηση της εφαρμογής βασίστηκε σε σύγχρονες τεχνολογίες και πρακτικές, όπως την Python, το FastAPI, και το Chainlit, χρησιμοποιώντας μία πολυεπίπεδη αρχιτεκτονική περιγραφόμενη στο Κεφάλαιο 4. Η παρουσίαση της εφαρμογής στο Κεφάλαιο 5, είναι η απόδειξη της λειτουργικότητάς της και της ευχρηστίας της, καθώς και της ικανότητάς της να συνδιάζει τις παραπάνω τεχνολογίες για την επίτευξη του στόχου της.

Συνολικά, η εργασία είναι μια σημαντική συμβολή στον τομέα της αλληλεπίδρασης ανθρώπου-υπολογιστή, και ειδικότερα στην κατηγορία των συστημάτων που επιτρέπουν την αλληλεπίδραση με βάσεις δεδομένων μέσω φυσικής γλώσσας. Όχι μόνο μειώνει το χάσμα μεταξύ των τεχνικών και μη τεχνικών χρηστών, αλλά και προάγει την προσβασιμότητα και την ευχρηστία των βάσεων δεδομένων.

7.2 Μελλοντικές Επεκτάσεις

Παρόλο που η εφαρμογή WebText2SQL είναι ήδη πλήρως λειτουργική, υπάρχουν πολλές δυνατότητες για μελλοντικές επεκτάσεις και βελτιώσεις. Ορισμένες από αυτές περιλαμβάνουν:

- **Υποστήριξη Περισσότερων Διαλέκτων SQL:** Η εφαρμογή αυτή τη στιγμή είναι σχεδιασμένη για να υποστηρίζει PostgreSQL και MySQL. Ωστόσο, θα μπορούσε να επεκταθεί για να υποστηρίζει και άλλες διαλέκτους SQL, όπως SQLite, Oracle SQL, ή Microsoft SQL Server, ώστε να καλύψει ένα ευρύτερο φάσμα χρηστών και περιβαλλόντων.

- **Οπτικοποίηση Δεδομένων:** Μια ενδιαφέρουσα προσθήκη θα ήταν η δυνατότητα οπτικοποίησης των αποτελεσμάτων των ερωτημάτων. Αυτό θα μπορούσε να περιλαμβάνει γραφήματα, πίνακες και άλλες μορφές παρουσίασης δεδομένων, διευκολύνοντας την κατανόηση των αποτελεσμάτων από τους χρήστες. Αυτό θα μπορούσε να επιτευχθεί με τη χρήση βιβλιοθηκών όπως της Plotly ή της Pyplot που μάλιστα υποστηρίζονται και από την πλατφόρμα του Chainlit ως native components.

- **Υποστήριξη Ειδικών Μοντέλων:** Η εφαρμογή θα μπορούσε να επεκταθεί για να υποστηρίζει ειδικά γλωσσικά μοντέλα που έχουν σχεδιαστεί για μετατροπή Text-to-SQL, για βάσεις δεδομένων που είναι παγιωμένες και πολλοί χρήστες τις χρησιμοποιούν, ώστε να μπορούν τα μοντέλα να εκπαιδευτούν σε συγκεκριμένα σχήματα βάσεων δεδομένων και ερωτήματα. Αυτό θα μπορούσε να βελτιώσει την ακρίβεια και την απόδοση της μετατροπής ερωτημάτων.

- **Υποστήριξη Συνεργασίας και Κοινοποίησης:** Η δυνατότητα δημιουργίας ομάδων χρηστών και κοινοποίησης συνδέσεων βάσεων δεδομένων θα μπορούσε να είναι χρήσιμη για ομάδες εργασίας που συνεργάζονται σε κοινά έργα.

- **Υποστήριξη NoSQL Βάσεων Δεδομένων:** Η επέκταση της εφαρμογής για να υποστηρίζει NoSQL βάσεις δεδομένων, όπως MongoDB ή Cassandra, θα μπορούσε να διευρύνει τη χρησιμότητά της σε περιβάλλοντα που δεν χρησιμοποιούν παραδοσιακές SQL βάσεις δεδομένων. Με βάση και την υπάρχουσα υποδομή, θα μπορούσε να γίνει χρήση του GPT-4o-mini για την παραγωγή ερωτημάτων σε γλώσσες όπως η MongoDB Query Language (MQL).

Αυτές οι επεκτάσεις θα μπορούσαν να κάνουν την εφαρμογή ακόμα πιο ισχυρή και ευέλικτη, καλύπτοντας ένα ευρύτερο φάσμα αναγκών και σεναρίων χρήσης.

Βιβλιογραφία

[1] D. D. Chamberlin και R. F. Boyce, ``SEQUEL: A structured English query language", στο *Proceedings of the 1974 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*, σειρά SIGFIDET '74, Ann Arbor, Michigan: Association for Computing Machinery, 1974, σσ. 249–264, ISBN: 9781450374156. DOI: 10.1145/800296.811515 διεύθν.: https://doi.org/10.1145/800296.811515

[2] E. F. Codd, ``A relational model of data for large shared data banks", *Commun. ACM*, τόμ. 13, αρθμ. 6, σσ. 377–387, Ιούν. 1970, ISSN: 0001-0782. DOI: 10.1145/362384.362685 διεύθν.: https://doi.org/10.1145/362384.362685

[3] A. Eisenberg και J. Melton, ``SQL: 1999, formerly known as SQL3", *SIGMOD Rec.*, τόμ. 28, αρθμ. 1, σσ. 131–138, Μαρ. 1999, ISSN: 0163-5808. DOI: 10.1145/309844.310075 διεύθν.: https://doi.org/10.1145/309844.310075

[4] R. Zmigrod, S. Alamir και X. Liu, ``Translating Between SQL Dialects for Cloud Migration", στο *2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2024, σσ. 189–191. DOI: 10.1145/3639477.3639727

[5] I. Androutsopoulos, G. D. Ritchie και P. Thanisch, *Natural Language Interfaces to Databases - An Introduction*, 1995. arXiv: cmp-lg/9503016 [cmp-lg]. διεύθν.: https://arxiv.org/abs/cmp-lg/9503016

[6] F. Li και H. V. Jagadish, ``Constructing an interactive natural language interface for relational databases", *Proc. VLDB Endow.*, τόμ. 8, αρθμ. 1, σσ. 73–84, Σεπτ. 2014, ISSN: 2150-8097. DOI: 10.14778/2735461.2735468 διεύθν.: https://doi.org/10.14778/2735461.2735468

[7] A.-M. Popescu, O. Etzioni και H. Kautz, ``Towards a theory of natural language interfaces to databases", στο *Proceedings of the 8th International Conference on Intelligent User Interfaces*, σειρά IUI '03, Miami, Florida, USA: Association for Computing Machinery, 2003, σσ. 149–157, ISBN: 1581135866. DOI: 10.1145/604045.604070 διεύθν.: https://doi.org/10.1145/604045.604070

[8] R. J. Kate, Y. W. Wong και R. J. Mooney, ``Learning to transform natural to formal languages", στο *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 3*, σειρά AAAI'05, Pittsburgh, Pennsylvania: AAAI Press, 2005, σσ. 1062–1068, ISBN: 157735236x.

[9] J. M. Zelle και R. J. Mooney, ``Learning to parse database queries using inductive logic programming", στο *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, σειρά AAAI'96, Portland, Oregon: AAAI Press, 1996, σσ. 1050–1055, ISBN: 026251091X.

[10] L. R. Tang και R. J. Mooney, ``Using multiple clause constructors in inductive logic programming for semantic parsing", στο *Proceedings of the 12th European Conference on Machine Learning*, σειρά ECML'01, Freiburg, Germany: Springer-Verlag, 2001, σσ. 466–477, ISBN: 3540425365. DOI: 10.1007/3-540-44795-4_40 διεύθν.: https://doi.org/10.1007/3-540-44795-4_40

[11] T. B. Brown κ.ά., *Language Models are Few-Shot Learners*, 2020. arXiv: 2005.14165 [cs.CL]. διεύθν.: https://arxiv.org/abs/2005.14165

[12] OpenAI κ.ά., *GPT-4 Technical Report*, 2024. arXiv: 2303.08774 [cs.CL]. διεύθν.: https://arxiv.org/abs/2303.08774

[13] *Prompt Engineering Guide*, https://www.promptingguide.ai/, Accessed: 2025-06-01.

[14] Q. Dong κ.ά., *A Survey on In-context Learning*, 2024. arXiv: 2301.00234 [cs.CL]. διεύθν.: https://arxiv.org/abs/2301.00234

[15] J. Wei κ.ά., *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*, 2023. arXiv: 2201.11903 [cs.CL]. διεύθν.: https://arxiv.org/abs/2201.11903

[16] N. Rajkumar, R. Li και D. Bahdanau, *Evaluating the Text-to-SQL Capabilities of Large Language Models*, 2022. arXiv: 2204.00498 [cs.CL]. διεύθν.: https://arxiv.org/abs/2204.00498

[17] M. Pourreza και D. Rafiei, *DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction*, 2023. arXiv: 2304.11015 [cs.CL]. διεύθν.: https://arxiv.org/abs/2304.11015

[18] J. Li κ.ά., *Can LLM Already Serve as A Database Interface? A BIg Bench for Large-Scale Database Grounded Text-to-SQLs*, 2023. arXiv: 2305.03111 [cs.CL]. διεύθν.: https://arxiv.org/abs/2305.03111

[19] A. Matarazzo και R. Torlone, *A Survey on Large Language Models with some Insights on their Capabilities and Limitations*, 2025. arXiv: 2501.04040 [cs.CL]. διεύθν.: https://arxiv.org/abs/2501.04040

[20] Z. Ji κ.ά., ``Survey of Hallucination in Natural Language Generation", *ACM Computing Surveys*, τόμ. 55, αρθμ. 12, σσ. 1–38, Μαρ. 2023, ISSN: 1557-7341. DOI: 10.1145/3571730 διεύθν.: http://dx.doi.org/10.1145/3571730

[21] *OpenAI API Pricing*, https://openai.com/api/pricing/, Accessed: 2025-08-10.

[22] D. Tolpeko και A. Tolpeko, *SQLines SQL Converter*. διεύθν.: https://github.com/dmtolpeko/sqlines

[23] J. Zhang κ.ά., *ExeSQL: Self-Taught Text-to-SQL Models with Execution-Driven Bootstrapping for SQL Dialects*, 2025. arXiv: 2505.17231 [cs.CL]. διεύθν.: https://arxiv.org/abs/2505.17231

[24] H. Lefebvre, C. Legner και M. Fadler, ``Data democratization: toward a deeper understanding", Σεπτ. 2021.

[25] T. Harland, C. Hocken, T. Schröer και V. Stich, ``Towards a Democratization of Data in the Context of Industry 4.0", *Sci*, τόμ. 4, αρθμ. 3, 2022, ISSN: 2413-4155. DOI: 10.3390/sci4030029 διεύθν.: https://www.mdpi.com/2413-4155/4/3/29

[26] A. Halevy, A. Rajaraman και J. Ordille, ``Data integration: the teenage years", στο *Proceedings of the 32nd International Conference on Very Large Data Bases*, σειρά VLDB '06, Seoul, Korea: VLDB Endowment, 2006, σσ. 9–16.

[27] W. A. Woods, ``Progress in natural language understanding: an application to lunar geology", στο *Proceedings of the June 4-8, 1973, National Computer Conference and Exposition*, σειρά AFIPS '73, New York, New York: Association for Computing Machinery, 1973, σσ. 441–450, ISBN: 9781450379168. DOI: 10.1145/1499586.1499695 διεύθν.: https://doi.org/10.1145/1499586.1499695

[28] D. H. Warren και F. C. Pereira, ``An Efficient Easily Adaptable System for Interpreting Natural Language Queries", *American Journal of Computational Linguistics*, τόμ. 8, αρθμ. 3-4, σσ. 110–122, 1982. διεύθν.: https://aclanthology.org/J82-3002/

[29] G. G. Hendrix, ``Lifer: a natural language interface facility", *SIGART Bull.*, αρθμ. 61, σσ. 25–26, Φεβ. 1977, ISSN: 0163-5719. DOI: 10.1145/1045283.1045289 διεύθν.: https://doi.org/10.1145/1045283.1045289

[30] V. Zhong, C. Xiong και R. Socher, *Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning*, 2017. arXiv: 1709.00103 [cs.CL]. διεύθν.: https://arxiv.org/abs/1709.00103

[31] T. Yu κ.ά., *Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task*, 2019. arXiv: 1809.08887 [cs.CL]. διεύθν.: https://arxiv.org/abs/1809.08887

[32] T. Yu κ.ά., *CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases*, 2019. arXiv: 1909.05378 [cs.CL]. διεύθν.: https://arxiv.org/abs/1909.05378

[33] T. Yu κ.ά., *SParC: Cross-Domain Semantic Parsing in Context*, 2019. arXiv: 1906.02285 [cs.CL]. διεύθν.: https://arxiv.org/abs/1906.02285

[34] I. Sutskever, O. Vinyals και Q. V. Le, *Sequence to Sequence Learning with Neural Networks*, 2014. arXiv: 1409.3215 [cs.CL]. διεύθν.: https://arxiv.org/abs/1409.3215

[35] D. Bahdanau, K. Cho και Y. Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate*, 2016. arXiv: 1409.0473 [cs.CL]. διεύθν.: https://arxiv.org/abs/1409.0473

[36] X. Xu, C. Liu και D. Song, *SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning*, 2017. arXiv: 1711.04436 [cs.CL]. διεύθν.: https://arxiv.org/abs/1711.04436

[37] B. Wang, R. Shin, X. Liu, O. Polozov και M. Richardson, ``RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers", στο *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter και J. Tetreault, επιμελητές, Online: Association for Computational Linguistics, Ιούλ. 2020, σσ. 7567–7578. DOI: 10.18653/v1/2020.acl-main.677 διεύθν.: https://aclanthology.org/2020.acl-main.677/

[38] E. M. Bender, T. Gebru, A. McMillan-Major και S. Shmitchell, ``On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? □", στο *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, σειρά FAccT '21, Virtual Event, Canada: Association for Computing Machinery, 2021, σσ. 610–623, ISBN: 9781450383097. DOI: 10.1145/3442188.3445922 διεύθν.: https://doi.org/10.1145/3442188.3445922

[39] R. Zhong, T. Yu και D. Klein, *Semantic Evaluation for Text-to-SQL with Distilled Test Suites*, 2020. arXiv: 2010.02840 [cs.CL]. διεύθν.: https://arxiv.org/abs/2010.02840

[40] S. Ramírez, *FastAPI*. διεύθν.: https://github.com/fastapi/fastapi

[41] OpenAI κ.ά., *GPT-4o System Card*, 2024. arXiv: 2410.21276 [cs.CL]. διεύθν.: https://arxiv.org/abs/2410.21276

[42] M. Trylesinski και T. Christie, *Uvicorn*. διεύθν.: https://github.com/encode/uvicorn

[43] T. pandas development team, *pandas-dev/pandas: Pandas*, έκδ. v2.3.0, Ιούν. 2025. DOI: 10.5281/zenodo.15597513 διεύθν.: https://doi.org/10.5281/zenodo.15597513

[44] S. Ramírez, *SQLModel*. διεύθν.: https://github.com/fastapi/sqlmodel

[45] M. Bayer, ``SQLAlchemy", στο *The Architecture of Open Source Applications Volume II: Structure, Scale, and a Few More Fearless Hacks*, A. Brown και G. Wilson, επιμελητές, aosabook.org, 2012. διεύθν.: http://aosabook.org/en/sqlalchemy.html

[46] S. Colvin κ.ά., *Pydantic*, έκδ. v2.11.7, Ιούν. 2025. διεύθν.: https://github.com/pydantic/pydantic

[47] J. Larsen. Manning, 2021, ISBN: 9781617297632.

[48] D. Merkel, ``Docker: lightweight linux containers for consistent development and deployment", *Linux journal*, τόμ. 2014, αρθμ. 239, σ. 2, 2014.

[49] D. Q. H. Le και M. Diaz. Packt Publishing, 2021, ISBN: 9781838641061. διεύθν.: https://ieeexplore.ieee.org/document/10162802

[50] E. Filipovska κ.ά., ``Benchmarking OpenAI's APIs and other Large Language Models for Repeatable and Efficient Question Answering Across Multiple Documents", στο *2024 19th Conference on Computer Science and Intelligence Systems (FedCSIS)*, Σεπτ. 2024, σσ. 107–117. DOI: 10.15439/2024F3979

[51] E. Gamma, *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995.

[52] J. Brooke, ``SUS: A quick and dirty usability scale", *Usability Eval. Ind.*, τόμ. 189, Νοέ. 1995.