****

**ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

**Σύστημα πληροφοριών φοίτησης με φυσικούς διαλόγους**

**Student information system with natural dialogues**

**Ζωγραφούλα Αμαλία Κ. Κουρελή**

|  |  |
| --- | --- |
| **Επιβλέπων (ή Επιβλέπουσα ή Επιβλέποντες):** | **Ελευθέριος Μωυσιάδης,** Αναπληρωτής Καθηγητής |

**ΚΑΒΑΛΑ**

**ΟΚΤΩΒΡΙΟΣ 2024**

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

Σύστημα πληροφοριών φοίτησης με φυσικούς διαλόγους

Student information system with natural dialogues

**Ζωγραφούλα Αμαλία Κ. Κουρελή**

**Α.Μ.:** 5372

|  |  |
| --- | --- |
| **ΕΠΙΒΛΕΠΟΝΤΕΣ:** | **Ελευθέριος Μωυσιάδης,** Αναπληρωτής Καθηγητής |

**Copyright@** 2024 Τμήμα **Πληροφορικής, Δημοκρίτειο Πανεπιστήμιο Θράκης,** <https://www.cs.duth.gr/>

Το περιεχόμενο της συγκεκριμένης Πτυχιακής Εργασίας αποτελεί πνευματική ιδιοκτησία του/των συγγραφέα/ων, του/της επιβλέποντα/ουσας καθηγητή/τριας και του Τμήματος Πληροφορικής του ΔΠΘ και προστατεύεται από το νόμο περί πνευματικής ιδιοκτησίας (Νόμος 2121/1993 και κανόνες Διεθνούς Δικαίου που ισχύουν στην Ελλάδα).

**Δήλωση τήρησης ακαδημαϊκής δεοντολογίας**

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπόγραφα ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται με λεπτομέρεια στην πτυχιακή εργασία. Έχω αναφέρει πλήρως και με σαφείς αναφορές όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία, είτε βάσει επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

**ΕΥΧΑΡΙΣΤΙΕΣ**

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, κύριο Ελευθέριο Μωυσιάδη, αναπληρωτή καθηγητή, για την καθοδήγηση που μου προσέφερε και το χρόνο που διέθεσε δίνοντάς μου χρήσιμες συμβουλές και οδηγίες για την ολοκλήρωση της πτυχιακής μου εργασίας. Επίσης, θα ήθελα να ευχαριστήσω όλους τους καθηγητές του Τμήματος Πληροφορικής του Δημοκρίτειου Πανεπιστήμιου Θράκης για τη συμβολή τους στην επιστημονική και τεχνολογική μου συγκρότηση στα χρόνια της φοίτησής μου στο Τμήμα. Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου και τους φίλους μου για την υποστήριξη που μου προσέφεραν όλα τα χρόνια των σπουδών μου αλλά και κατά την διάρκεια εκπόνησης της εργασίας μου.

# **Π****ΕΡΙΛΗΨΗ**

Αντικείμενο της παρούσας εργασίας αποτελεί η κατανόηση και ανάπτυξη ενός συστήματος ικανού να απαντά με φυσικό λόγο (Chatbot) σε απορίες φοιτητών του τμήματος Πληροφορικής του Δημοκρίτειου Πανεπιστήμιου Θράκης και να τους παρέχει χρήσιμες πληροφορίες σχετικά με το τμήμα και τις σπουδές τους. Οι διάλογοι βασίζονται σε τεχνολογία GPT- 4, το πιο πρόσφατο μοντέλο που έχει αναπτύξει η OpenAI, το οποίο με τη βοήθεια της Διεπαφής Προγραμματισμού Εφαρμογών (APIs) μπορεί να αποκτήσει πρόσβαση στα δεδομένα σε πραγματικό χρόνο, σαρώνοντας τις ιστοσελίδες της σχολής. Bασίζεται στη Μάθηση σε Βάθος (Deep Learning) που αποτελεί μέρος της Μηχανικής Μάθησης (Machine Learning) στην Τεχνητή Νοημοσύνη (Artificial Intelligence), και εμπνέεται από τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου.

Η ανάπτυξη του προγράμματος έγινε στο PyCharm, με χρήση της γλώσσας προγραμματισμού Python. Χρησιμοποιήθηκε ακόμη, το FastAPI, το οποίο είναι ένα σύγχρονο και γρήγορο διαδικτυακό πλαίσιο (Web Framework) για τη δημιουργία APIs με Python, και προσφέρει μια ισχυρή βάση για τη δημιουργία και διαχείριση μιας διαδικτυακής εφαρμογής. Για τη διεπαφή χρήστη (User Interface), δημιουργήθηκε μια ιστοσελίδα με τη χρήση κώδικα HTML, CSS και JavaScript. Η ιστοσελίδα προσφέρει μια φιλική και διαδραστική εμπειρία στους χρήστες, επιτρέποντάς τους να επικοινωνούν με το chatbot εύκολα και γρήγορα.

**ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ:** Πληροφορική, **Τεχνητή Νοημοσύνη,** **Επεξεργασία Φυσικής Γλώσσας**

**ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ:** Σύστημα Διαλόγου, Επεξεργασία Φυσικής Γλώσσας, Μηχανική Μάθηση, Σάρωση Ιστοσελίδων, Τεχνητή Νοημοσύνη

# **ABSTRACT**

The object of the current project is the understanding and development of a system capable of answering with natural speech (Chatbot) to student’s questions of the Department of Computer Science of the Democritus University of Thrace and providing them with useful information about the department and their studies. The dialogues are based on GPT-4 technology, the latest model developed by OpenAI, which with the help of Application Programming Interfaces (APIs) can access data in real-time, by scanning the department's web pages. It is based on Deep Learning, which is part of Machine Learning in Artificial Intelligence, and is inspired by the structure and function of the human brain.

The project was developed in PyCharm, using the Python programming language. Furthermore, FastAPI was used, which is a synchronous and high-speed Web Framework for creating APIs with Python, and offers a powerful basis for creating and managing a web application. For the user interface, a web page was created using HTML, CSS and JavaScript code. The website provides a friendly and interactive experience for users, allowing them to communicate with the chatbot easily and quickly.

**SCIENTIFIC AREA:** Computer Science, Artificial Intelligence, Natural Language Processing

**KEYWORDS:** Chatbot, Natural Language Processing (NLP), Machine Learning (ML), Web Scraping, Artificial Intelligence (AI)

# **Περιεχόμενα**

[ΠΕΡΙΛΗΨΗ ix](#_Toc178406475)

[ABSTRACT xi](#_Toc178406476)

[Περιεχόμενα xiii](#_Toc178406477)

[Κατάλογος εικόνων xvi](#_Toc178406478)

[ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ xx](#_Toc178406479)

[ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ xxi](#_Toc178406480)

[1ο Κεφάλαιο: Εισαγωγή 1](#_Toc178406481)

[1.1 Ερευνητική Ανασκόπηση 1](#_Toc178406482)

[1.2 Θεωρητικές έννοιες και τεχνολογίες 3](#_Toc178406483)

[1.3 Δομή εργασίας 3](#_Toc178406484)

[2ο Κεφάλαιο: Θεωρητικό Υπόβαθρο 4](#_Toc178406485)

[2.1 Τεχνητή Νοημοσύνη 4](#_Toc178406486)

[2.2 Επεξεργασία Φυσικής Γλώσσας 5](#_Toc178406487)

[2.3 Μηχανική Μάθηση 7](#_Toc178406488)

[2.4 Αλληλεπίδραση Ανθρώπου-Υπολογιστή 8](#_Toc178406489)

[2.5 Τεχνολογία Λογισμικού 9](#_Toc178406490)

[2.6 Βάσεις Δεδομένων 10](#_Toc178406491)

[2.7 Σάρωση Ιστοσελίδων (Web Scraping) 11](#_Toc178406492)

[2.8 Chatbot 12](#_Toc178406493)

[2.8.1 Ορισμός 12](#_Toc178406494)

[2.8.2 Ιστορική αναδρομή στα Chatbot 12](#_Toc178406495)

[2.8.3 Τύποι των Chatbot 16](#_Toc178406496)

[2.8.4 Χρησιμότητα των Chatbots 17](#_Toc178406497)

[3ο Κεφάλαιο: Εργαλεία και Βιβλιοθήκες 19](#_Toc178406498)

[3.1 PyCharm 19](#_Toc178406499)

[3.2 FastAPI 20](#_Toc178406500)

[3.3 Uvicorn 21](#_Toc178406501)

[3.4 OpenAI 22](#_Toc178406502)

[3.5 GitHub 23](#_Toc178406503)

[4ο Κεφάλαιο: Περιγραφή Λογισμικού 24](#_Toc178406504)

[4.1 Γλώσσες προγραμματισμού 24](#_Toc178406505)

[4.1.1 Python 24](#_Toc178406506)

[4.1.2 HTML 25](#_Toc178406507)

[4.1.3 CSS 26](#_Toc178406508)

[4.1.4 JavaScript 27](#_Toc178406509)

[4.2 Αρχιτεκτονική συστήματος 28](#_Toc178406510)

[4.2.1 Χρήστες 28](#_Toc178406511)

[4.2.2 Λειτουργικές Απαιτήσεις 29](#_Toc178406512)

[4.2.3 Μη Λειτουργικές Απαιτήσεις 29](#_Toc178406513)

[4.3 Διαγράμματα Συστήματος 30](#_Toc178406514)

[4.3.1 UML Διάγραμμα Περιπτώσεων Χρήσης (Use Case Diagram) 30](#_Toc178406515)

[4.3.2 UML Διάγραμμα Κλάσεων (Class Diagrams) 31](#_Toc178406516)

[4.3.3 UML Διάγραμμα Ακολουθίας (Sequence Diagram) 32](#_Toc178406517)

[4.4 Χαρακτηριστικά τμήματα κώδικα 34](#_Toc178406518)

[4.4.1 Web Scraper 34](#_Toc178406519)

[4.4.2 Διαχείριση Επικοινωνίας Χρήστη-Chatbot. 35](#_Toc178406520)

[4.5 Περιγραφή Παραθύρου Εφαρμογής 36](#_Toc178406521)

[5ο Κεφάλαιο: Αποτελέσματα Χρήσης 38](#_Toc178406522)

[5.1 Έλεγχος Ποιότητας Απαντήσεων του Chatbot 39](#_Toc178406523)

[5.2 Έλεγχος Χρόνων Απόκρισης του Chatbot 42](#_Toc178406524)

[5.3 Ανάλυση Πειραματικών Αποτελεσμάτων 44](#_Toc178406525)

[6ο Κεφάλαιο: Συμπεράσματα, Προβλήματα και Μελλοντικές προτάσεις 45](#_Toc178406526)

[6.1 Προβλήματα που παρουσιάστηκαν 45](#_Toc178406527)

[6.2 Συμπεράσματα 48](#_Toc178406528)

[6.3 Προοπτικές βελτίωσης 49](#_Toc178406529)

[Αναφορές 51](#_Toc178406530)

[Παράρτημα I 54](#_Toc178406531)

[Παράρτημα II. 60](#_Toc178406532)

# **Κατάλογος εικόνων**

[Εικόνα 1: Alan Turing 4](#_Toc178434549)

[Εικόνα 2: Ορισμός της Επεξεργασίας Φυσικής Γλώσσας 6](#_Toc178434550)

[Εικόνα 3: Εφαρμογές της NLP στην καθημερινότητα 6](#_Toc178434551)

[Εικόνα 4: Παράδειγμα Επιβλεπόμενης Μάθησης 7](#_Toc178434552)

[Εικόνα 5: Παράδειγμα Μη Επιβλεπόμενης Μάθησης 8](#_Toc178434553)

[Εικόνα 6: Παράδειγμα Ενισχυτικής Μάθησης 8](#_Toc178434554)

[Εικόνα 7: Stages of the Software Development Life Cycle 10](#_Toc178434555)

[Εικόνα 8: Συνομιλία με το ELIZA 13](#_Toc178434556)

[Εικόνα 9: Συνομιλία με το ALICE 14](#_Toc178434557)

[Εικόνα 10: Συνομιλία με το ChatGPT 15](#_Toc178434558)

[Εικόνα 11: Παράδειγμα χρήσης Chatbot από τράπεζα 17](#_Toc178434559)

[Εικόνα 12: Αξιοποίηση των Chatbots σε διάφορους κλάδους 19](#_Toc178434560)

[Εικόνα 13: Το project σε περιβάλλον PyCharm 20](#_Toc178434561)

[Εικόνα 14: Σύγκριση του FastAPI με άλλα Web Frameworks 21](#_Toc178434562)

[Εικόνα 15: Αρχιτεκτονική Web Εφαρμογών 22](#_Toc178434563)

[Εικόνα 16: Το Project σε περιβάλλον GitHub 24](#_Toc178434564)

[Εικόνα 17: Παραδείγματα χρήσης της Python 25](#_Toc178434565)

[Εικόνα 18: Παράδειγμα χρήσης CSS κώδικα 27](#_Toc178434566)

[Εικόνα 19: UML Διάγραμμα Περιπτώσεων Χρήσης 31](#_Toc178434567)

[Εικόνα 20: UML Διάγραμμα Κλάσεων 32](#_Toc178434568)

[Εικόνα 21: UML Διάγραμμα Ακολουθίας 34](#_Toc178434569)

[Εικόνα 22: Η συνάρτηση scrape\_website με εσωτερική τεκμηρίωση 35](#_Toc178434570)

[Εικόνα 23: Η συνάρτηση chat() με εσωτερική τεκμηρίωση 36](#_Toc178434571)

[Εικόνα 24: Αρχική Σελίδα της Εφαρμογής 37](#_Toc178434572)

[Εικόνα 25: Παράδειγμα συνομιλίας με τον Ακαδημαϊκό Βοηθό 37](#_Toc178434573)

[Εικόνα 26: Παράδειγμα συνομιλίας (1) 40](#_Toc178434574)

[Εικόνα 27: Παράδειγμα συνομιλίας (2) 40](#_Toc178434575)

[Εικόνα 28: Παράδειγμα συνομιλίας (3) 40](#_Toc178434576)

[Εικόνα 29: Παράδειγμα συνομιλίας (4) 41](#_Toc178434577)

[Εικόνα 30: Διάγραμμα Ποσοστών Επιτυχίας-Αποτυχίας 42](#_Toc178434578)

[Εικόνα 31: Διάγραμμα Χρόνου Απόκρισης ανά Ερώτηση 43](#_Toc178434579)

[Εικόνα 32: Συνάρτηση μορφοποίησης στοιχείων επικοινωνίας 46](#_Toc178434580)

[Εικόνα 33: Συνάρτηση μορφοποίησης μαθημάτων 46](#_Toc178434581)

[Εικόνα 34: Συνάρτηση αφαίρεσης περιττών σημείων στίξης 46](#_Toc178434582)

[Εικόνα 35: Συνάρτηση κλήσης των συναρτήσεων μορφοποίησης 46](#_Toc178434583)

[Εικόνα 36: Δομή απάντησης πριν την μορφοποίηση (1) 47](#_Toc178434584)

[Εικόνα 37: Δομή απάντησης μετά την μορφοποίηση (1) 47](#_Toc178434585)

[Εικόνα 38: Δομή απάντησης πριν την μορφοποίηση (2) 47](#_Toc178434586)

[Εικόνα 39: Δομή απάντησης μετά την μορφοποίηση (2) 48](#_Toc178434587)

[Εικόνα 40: Εγκατάσταση Python σε Windows 54](#_Toc178434588)

[Εικόνα 41: Εγκατάσταση Pycharm σε Windows (βήμα 1 και 2) 55](#_Toc178434589)

[Εικόνα 42:Εγκατάσταση Pycharm σε Windows (βήμα 3 και 4) 55](#_Toc178434590)

[Εικόνα 43:Εγκατάσταση Pycharm σε Windows (βήμα 4 και 5) 55](#_Toc178434591)

[Εικόνα 44: Απόκτηση API Key (βήμα 1) 56](#_Toc178434592)

[Εικόνα 45 :Απόκτηση API Key (βήμα 2) 56](#_Toc178434593)

[Εικόνα 46: Απόκτηση API Key (Βήμα 3) 57](#_Toc178434594)

[Εικόνα 47:Δημιουργία Νέου Project στο PyCharm (βήμα 1). 57](#_Toc178434595)

[Εικόνα 48:Δημιουργία Νέου Project στο PyCharm (βήμα 2) 58](#_Toc178434596)

[Εικόνα 49: Δομή των αρχείων Project 58](#_Toc178434597)

[Εικόνα 50: Απαραίτητες Βιβλιοθήκες για το Project 59](#_Toc178434598)

[Εικόνα 51: Εκτέλεση της εφαρμογής 59](#_Toc178434599)

***Κατάλογος Πινάκων***

[Πίνακας 1: Συγκεντρωτικός Πίνακας ερωτήσεων που απαντάει το Chatbot 37](#_Toc178434734)

[Πίνακας 2: Καταστάσεις Λειτουργίας 38](#_Toc178434735)

[Πίνακας 3: Χρόνοι απόκρισης του συστήματος 43](#_Toc178434736)

# **ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ**

|  |  |
| --- | --- |
| **Ξενόγλωσσος Όρος** | **Ελληνικός Όρος** |
| Chatbot | Σύστημα Ανάπτυξης Διαλόγων |
| Application Programming Interface | Διεπαφή Προγραμματισμού Εφαρμογών |
| Machine Learning | Μηχανική Μάθηση |
| Deep Learning | Βαθιά Μάθηση |
| Artificial Intelligence | Τεχνητή Νοημοσύνη |
| Natural Language Processing | Επεξεργασία Φυσικής Γλώσσας |
| Web Framework | Διαδικτυακό Πλαίσιο |
| Web Scraping | Σάρωση Ιστοσελίδων |
| Natural Language Understanding | Κατανόηση Φυσικής Γλώσσας |
| Database | Βάση Δεδομένων |
| Neural Networks | Νευρωνικά Δίκτυα |
| Server | Διακομιστής |
| Large Language Models | Μοντέλα Μεγάλων Γλωσσών |
| User Interface | Διεπαφή Χρήστη |
| Computer Vision | Τεχνητή Όραση |
| Supervised Learning | Επιβλεπόμενη Μάθηση |
| Unsupervised Learning | Μη Επιβλεπόμενη Μάθηση |
| Reinforcement Learning | Ενισχυτική Μάθηση |
| Human-Computer Interaction | Αλληλεπίδραση Ανθρώπου-Υπολογιστή |
| Software Engineering | Τεχνολογία Λογισμικού |
| Software Development Process Models | Μοντέλα διαδικασιών ανάπτυξης λογισμικού |
| World Wide Web | Παγκόσμιος Ιστός |
| Internet of Things | Διαδίκτυο των Πραγμάτων |
| Integrated Development Environment | Ολοκληρωμένο Περιβάλλον Ανάπτυξης |
| Hypertext Transfer Protocol | Πρωτόκολλο Μεταφοράς Υπερκειμένου |
| Cascading Style Sheets | Διαδοχικά Φύλλα Στυλ |
| HyperText Markup Language | Γλώσσα Σήμανσης Υπερκειμένου |
| Artificial General Intelligence | Γενική Τεχνητή Νοημοσύνη |

# **ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ**

|  |  |
| --- | --- |
| ΤΝ | Τεχνητή Νοημοσύνη |
| ΤΟ | Τεχνητή Όραση |
| ΣΔΒΔ | Σύστημα Διαχείρισης Βάσεων Δεδομένων |
| GPT-4 | Generative Pre-trained Transformer 4 |
| API | Application Programming Interface |
| ML | Machine Learning |
| DL | Deep Learning |
| AI | Artificial Intelligence |
| NLP | Natural Language Processing |
| NLU | Natural Language Understanding |
| GPS | Global Positioning System |
| HCI | Human-Computer Interaction |
| WWW | World Wide Web |
| MIT | Massachusetts Institute of Technology |
| ALICE | Artificial Linguistic Internet Computer Entity |
| AIML | Artificial Intelligence Markup Language |
| IoT | Internet of Things |
| LLM | Large Language Model |
| IDE | Integrated Development Environment |
| HTTP | Hypertext Transfer Protocol |
| HTML | HyperText Markup Language |
| CSS | Cascading Style Sheets |
| ASGI | Asynchronous Server Gateway Interface |
| AGI | Artificial General Intelligence |

# 1ο Κεφάλαιο: Εισαγωγή

Αναμφίβολα η Τεχνητή Νοημοσύνη, αλλάζει πλέον τον κόσμο και δεν θεωρείται τεχνολογία του μέλλοντος, καθώς έχουμε ήδη αρχίσει να βιώνουμε τις επιδράσεις της στην καθημερινή μας ζωή. Η Τεχνητή Νοημοσύνη, έχει θέσει τα θεμέλια για τη δημιουργία ευφυών πρακτόρων που γνωρίζουμε και ως Chatbots. Οι ευφυείς πράκτορες ορίζονται ως συστήματα που χρησιμοποιούν Τεχνητή Νοημοσύνη για να αλληλοεπιδρούν με χρήστες μέσω γραπτού ή προφορικού λόγου, φέρνοντας επανάσταση στην διαδικασία της επικοινωνίας του ανθρώπου με τον υπολογιστή. Στη συγκεκριμένη εργασία υλοποιείται ένα Chatbot που λειτουργεί ως βοηθός φοιτητών, στόχος του οποίου είναι να πραγματοποιεί συνομιλίες με τους χρήστες του, παρέχοντας πληροφορίες και απαντώντας σε ερωτήσεις σχετικά με το τμήμα και τις σπουδές τους.

## Ερευνητική Ανασκόπηση

Η αναζήτηση της βιβλιογραφίας επικεντρώθηκε σε παρόμοια συστήματα διαλογικών πρακτόρων, στους αλγορίθμους και στις τεχνικές που εφαρμόζονται σε αυτά, μέσα από ερευνητικά άρθρα και επιστημονικές εργασίες.

Θεωρητικά, τα Chatbots είναι ευέλικτα εργαλεία που μπορούν να χρησιμοποιηθούν σε πολλούς κλάδους με διάφορους τρόπους και να κάνουν πιο εύκολη και ευχάριστη την επικοινωνία του ανθρώπου με τον υπολογιστή [1]. Κατά κύριο λόγο, τα Chatbots χρησιμοποιούν μια συγκεκριμένη αρχιτεκτονική για τον σχεδιασμό τους, η οποία σύμφωνα με το άρθρο [2], αποτελείται από τέσσερις ενότητες. Συγκεκριμένα, αυτές είναι το Front End, το Back End, η ενότητα Knowledge Base και η ενότητα E-learning BOT. Η πρώτη ενότητα αντιπροσωπεύει το επίπεδο παρουσίασης μέσω της παροχής μιας φιλικής προς τον χρήστη διεπαφής η οποία είναι εύχρηστη και συμβατή από διάφορα είδη συσκευών όπως tablets, smartphones και υπολογιστές. Το Back End χρησιμοποιείται για τη διαχείριση λειτουργιών που δε γίνονται αντιληπτές από τον χρήστη του Chatbot. Αυτή η ενότητα λειτουργεί στο παρασκήνιο για την αποτελεσματικότερη ικανοποίηση των χρηστών και επιμελείται τη λογική και την αποθήκευση δεδομένων στη βάση. Η Ενότητα της Knowledge Base είναι ένας ειδικός τύπος βάσης δεδομένων, όπου τα δεδομένα επεξεργάζονται από έναν διακομιστή (Server), για τη διαχείριση γνώσεων και πληροφοριών. Τέλος, η ενότητα E-learning BOT αποτελεί το βασικότερο κομμάτι του Chatbot το οποίο διαχειρίζεται τους διαλόγους μεταξύ ανθρώπου και υπολογιστικού συστήματος και επομένως είναι υπεύθυνο για την διεξοδική επικοινωνία μεταξύ τους.

Με παρόμοια λογική, περιγράφεται ο σχεδιασμός ενός γενικού διαλογικού πράκτορα από τον Haung [3], ο οποίος αποτελείται από πέντε κύρια μέρη, τα οποία είναι τα εξής: η διεπαφή χρήστη, η Κατανόησης Φυσικής Γλώσσας (Natural Language Understanding, NLU), διαχείριση των διαλόγων, το Back End και η παραγωγή απαντήσεων. Όσον αφορά τη διεπαφή χρήστη, είναι το κομμάτι του λογισμικού που επιτρέπει στους χρήστες να επικοινωνούν και να αλληλοεπιδρούν με ένα Chatbot μέσω κάποιων εφαρμογών messenger. Η πιο γνωστή εφαρμογή messenger στις μέρες μας είναι το Facebook Messenger αφού, όλο και περισσότερες επιχειρήσεις το εντάσσουν για την καλύτερη εξυπηρέτηση των πελατών τους. Σχετικά με το κομμάτι της Κατανόησης Φυσικής Γλώσσας, όταν το σύστημα λάβει κάποιο αίτημα από τον χρήστη, το χρησιμοποιεί για να εξάγει πληροφορίες από την είσοδο. Στη συνέχεια, η ενότητα διαχείρισης διαλόγου καθοδηγεί και διατηρεί τη συνοχή της συνομιλίας με έναν χρήστη. Αυτή η λειτουργία είναι σημαντική για την επιτυχία της αλληλεπίδρασης, καθώς διασφαλίζει ότι ο πράκτορας κατανοεί και ανταποκρίνεται σωστά στα αιτήματα και τις ερωτήσεις του χρήστη. Το κομμάτι του Backend, είναι υπεύθυνο για τη διαχείριση και την επεξεργασία των δεδομένων και την αλληλεπίδραση με εξωτερικές υπηρεσίες ή βάσεις δεδομένων γνώσεων (Knowledge Base). Τέλος, ο Huang αναφέρεται στην ενότητα παραγωγής απαντήσεων, στην οποία μόλις ληφθεί το αίτημα του χρήστη, το βήμα που ακολουθεί για το σύστημα διαλόγου είναι να καθορίσει το περιεχόμενο της απάντησης και τον καλύτερο τρόπο έκφρασής της.

Με το ίδιο σκεπτικό, αναζητήθηκαν λογισμικά που αναπτύχθηκαν με παρόμοια αρχιτεκτονική για την υποστήριξη των φοιτητών από τα Πανεπιστημιακά Ιδρύματα. Έτσι, σε πρακτικό επίπεδο, βρέθηκαν πτυχιακές και διπλωματικές εργασίες με σχετικό περιεχόμενο, που υλοποιούν ένα Chatbot για χρήση από ακαδημαϊκά ιδρύματα. Αναπόφευκτα, οι εργασίες διαθέτουν κάποια κοινά χαρακτηριστικά μεταξύ τους όσον αφορά τη μεθοδολογία και τα εργαλεία που χρησιμοποιούν, ωστόσο κάθε μια είναι ξεχωριστή.

Για παράδειγμα, στη διπλωματική εργασία του Μάριου Αναστασίου [4], γίνεται χρήση της γλώσσας προγραμματισμού Python και συγκεκριμένα αξιοποιείται η ανοιχτού κώδικα βιβλιοθήκη «ChatterBot», η οποία παράγει αυτοματοποιημένες απαντήσεις στις ερωτήσεις των χρηστών. Επίσης, προτιμήθηκε το ανοιχτού κώδικα Framework της Rasa, το οποίο βασίζεται στη Μηχανική Μάθηση και παράγει αυτοματοποιημένες συζητήσεις. Επιπλέον, για τους διαλόγους χρησιμοποιήθηκε το DialogFlow, το οποίο είναι μια από τις δημοφιλέστερες πλατφόρμες κατανόησης φυσικής γλώσσας και απαιτεί ελάχιστο κώδικα για τον προγραμματισμό του Chatbot. Τέλος, αξιοποιείται το ανοιχτού λογισμικού Framework «Pandorabots», το οποίο επιτρέπει την ανάπτυξη και δημοσίευση των Chatbot στο διαδίκτυο αλλά και σε εφαρμογές έξυπνων κινητών συσκευών.

Επίσης, μια παρόμοια εργασία, η οποία δημιουργεί έναν διαλογικό πράκτορα είναι του Αλέν Μπασιά [5], και χρησιμοποιεί τη βιβλιοθήκη «ChatterBot» της Python. Ωστόσο για τη διεπαφή χρήστη χρησιμοποιεί τις γλώσσες προγραμματισμού HTML, CSS και JavaScript. Επιπροσθέτως, για τη διαχείριση των δεδομένων του Chatbot δημιουργεί έναν server (API) με τη χρήση της βιβλιοθήκης Flask που είναι ένα δημοφιλές και ελαφρύ Framework για την ανάπτυξη Web εφαρμογών σε Python.

Ένα επιπλέον παράδειγμα, αποτελεί η παρακάτω διπλωματική εργασία της Ελένης Ακριτίδου [6], η οποία χρησιμοποιεί το Language Studio της Azure, το οποίο είναι ένα σύνολο εργαλείων που βασίζονται στη διεπαφή χρήστη. Στη συγκεκριμένη διπλωματική εργασία, επιλέχθηκε το εργαλείο "Understanding questions and Conversational language", προκειμένου να χρησιμοποιεί φυσική γλώσσα κατά την εισαγωγή του αιτήματος του χρήστη και να ανταποκρίνεται ανάλογα. Έπειτα, για την υλοποίηση γίνεται χρήση της Azure, μιας δημόσιας πλατφόρμας υπολογιστών Cloud της Microsoft. Για τη διεπαφή χρήστη, επιλέχθηκε το Bot Framework Composer, είναι ένα ιδανικό εργαλείο για τον σχεδιασμό και την κατασκευή Chatbots, ιδίως για χρήστες που δεν έχουν εκτεταμένη γνώση κωδικοποίησης. Τέλος, κάνει χρήση της Bot Framework Emualator, μιας εφαρμογής που δίνει τη δυνατότητα στους προγραμματιστές Chatbot να δοκιμάσουν και αν χρειάζεται να διορθώσουν τα προγράμματά τους κατάλληλα, πριν από την τελική ανάπτυξη.

Τέλος, μια ακόμη εργασία με παρόμοια χαρακτηριστικά είναι των Τσιρίδης & Σταύρου [7], η οποία αξιοποιεί την Angular, ένα ισχυρό Framework ανοιχτού κώδικα βασισμένο στη γλώσσα TypeScript, που προτιμάται για την ανάπτυξη εφαρμογών σε πολλαπλές πλατφόρμες λόγω των υψηλών ταχυτήτων που προσφέρει. Επιπλέον, χρησιμοποιεί το Firebase, μια πλατφόρμα ανάπτυξης εφαρμογών ιστού και κινητών, παρέχοντας πολλαπλά εργαλεία. Στην εργασία αυτή, έγινε χρήση της βάσης δεδομένων (Database) και του Back End σε πραγματικό χρόνο του Firebase. Ακόμη, αξιοποιήθηκε το Dialogflow, μια υπηρεσία της Google που βασίζεται σε φυσικές συνομιλίες για την αλληλεπίδραση ανθρώπου-υπολογιστή. Για τη διεπαφή χρήστη επιλέχθηκε το Bootstrap, ένα σύνολο εργαλείων ανοιχτού κώδικα που περιλαμβάνει HTML και CSS για τη διαμόρφωση της τυπογραφίας, των κουμπιών πλοήγησης και άλλων στοιχείων της διεπαφής, με προαιρετικές προσθήκες σε JavaScript. Τέλος, για τον σχεδιασμό του Dashboard χρησιμοποιήθηκε το Material Design framework, και πιο συγκεκριμένα η έκδοση Angular Material UI, η οποία εφαρμόστηκε σε φόρμες όπως το Login και σε διάφορα Card-Views στις σελίδες.

## Θεωρητικές έννοιες και τεχνολογίες

Η ανάπτυξη της Μηχανικής Μάθησης και συγκεκριμένα της Τεχνητής Νοημοσύνης, μέσω της εξέλιξης των Νευρωνικών Δικτύων (Neural Networks) και της επεξεργασίας φυσικών γλωσσών, έχει δώσει τη δυνατότητα στον άνθρωπο να δημιουργήσει μοντέλα επεξεργασίας φυσικής γλώσσας (Chatbots) τα οποία μπορούμε να προσαρμόσουμε στις απαιτήσεις μας, ώστε η επικοινωνία ανθρώπου-υπολογιστή να γίνει πιο εύκολη και αποτελεσματική σε οποιονδήποτε κλάδο επιθυμούμε.

Έχοντας, λοιπόν, υπόψιν όλα τα παραπάνω, υλοποιήθηκε η παρούσα εργασία στο PyCharm κάνοντας χρήση της γλώσσας προγραμματισμού Python. Στην οποία οι διάλογοι βασίζονται σε τεχνολογία GPT- 4, το πιο πρόσφατο μοντέλο μεγάλων γλωσσών (Large Language Model, LLM) που αναπτύχθηκε μέχρι σήμερα από την OpenAI. Χρησιμοποιήθηκε επίσης, το Web Framework της FastAPI για τη δημιουργία APIs, το οποίο προσφέρει μια ισχυρή βάση για τη δημιουργία και διαχείριση διαδικτυακών εφαρμογών. Για τη συλλογή πληροφοριών επιλέχθηκε η χρήση μιας συνάρτησης Σάρωσης Ιστοσελίδων (Web Scraping), η οποία επιτρέπει η εξαγωγή κειμένου από τις επιλεγμένες ιστοσελίδες και συμβάλει στην παραγωγή απαντήσεων. Για τη διεπαφή χρήστη, δημιουργήθηκε μια ιστοσελίδα φιλική προς τους χρήστες με τη χρήση HTML, CSS και JavaScript κώδικα.

## Δομή εργασίας

Η παρούσα εργασία, περιλαμβάνει συνολικά 6 κεφάλαια, συμπεριλαμβανομένου και του παρόντος εισαγωγικού.

Στο Κεφάλαιο 2, γίνεται ανάλυση των θεωρητικών εννοιών του πεδίου και παρέχεται το βασικό θεωρητικό υπόβαθρο το οποίο χρειάζεται να γνωρίζει ο αναγνώστης προκειμένου να είναι σε θέση να κατανοήσει κάποιες έννοιες και τεχνολογίες που αναφέρονται κατά τη διάρκεια των κεφαλαίων.

Στο Κεφάλαιο 3, περιγράφονται αναλυτικά τα εργαλεία και οι βιβλιοθήκες που χρησιμοποιήθηκαν για την υλοποίηση του λογισμικού.

Στο Κεφάλαιο 4, γίνεται η περιγραφή του λογισμικού που χρησιμοποιήθηκε συμπεριλαμβανομένων των γλωσσών προγραμματισμού, της τεχνολογίας λογισμικού, της παρουσίαση μικρών χαρακτηριστικών τμημάτων κώδικα και επεξήγηση της λειτουργίας τους.

Στο Κεφάλαιο 5, παρουσιάζονται παραδείγματα ελέγχου καλής λειτουργίας και χρήσης από την εφαρμογή λογισμικού με φωτογραφίες. Επίσης, παρουσιάζονται διάφορες μετρήσεις και πειραματικά αποτελέσματα, όπως μετρήσεις χρόνων απόκρισης για διάφορα μεγέθη κειμένου με την χρήση διαγραμμάτων.

Στο Κεφάλαιο 6, γίνεται ανάλυση των συμπερασμάτων από την εκπόνηση της πτυχιακής εργασίας. Περιγράφονται επίσης μελλοντικές προτάσεις για βελτιώσεις, αλλαγές και προσθήκες που μπορούν να γίνουν προκειμένου να εξελίξουν την εργασία.

# **2ο Κεφάλαιο: Θεωρητικό Υπόβαθρο**

## 2.1 Τεχνητή Νοημοσύνη

Ο όρος Τεχνητή Νοημοσύνη (Artificial Intelligence, AI), αναφέρεται στον τομέα της Επιστήμης των Υπολογιστών ο οποίος ασχολείται με τη σχεδίαση και την υλοποίηση [υπολογιστικών συστημάτων](https://el.wikipedia.org/wiki/%CE%A5%CF%80%CE%BF%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CF%84%CE%B9%CE%BA%CF%8C_%CF%83%CF%8D%CF%83%CF%84%CE%B7%CE%BC%CE%B1) τα οποία έχουν την ικανότητα να μιμούνται γνωστικές ικανότητες των ανθρώπων και εμφανίζουν χαρακτηριστικά τα που φέρουν στοιχεία της ανθρώπινης συμπεριφοράς και υπονοούν έστω και στοιχειώδη ευφυΐα. Παραδείγματα αποτελούν η [μάθηση](https://el.wikipedia.org/wiki/%CE%9C%CE%AC%CE%B8%CE%B7%CF%83%CE%B7), η εξαγωγή συμπερασμάτων, η αντίληψη και κατανόηση εικόνων, η κατανόηση φυσικής γλώσσας κ.α.

Η ιστορία της Τεχνητής Νοημοσύνης (ΤΝ), ξεκινάει πριν αιώνες, όταν μεγάλοι αρχαίοι Έλληνες, Κινέζοι και Ινδοί, φιλόσοφοι και μαθηματικοί, διερεύνησαν σημαντικά το πεδίο της συλλογιστικής και έθεσαν τις βάσεις για μεταγενέστερες έρευνες. Σημαντικός κόμβος μέσα στα χρόνια, αποτέλεσε το Turing Test του Alan Turing το 1950, ο οποίος και θεωρείται πατέρας της ΤΝ. Επίσης, αξίζει να αναφέρουμε ότι το έτος 1956 έλαβε χώρα η επίσημη ίδρυση του πεδίου, σε συνέδριο στο Dartmouth College, στο οποίο συμμετείχαν αξιόλογοι επιστήμονες του κλάδου, οι οποίοι πρωτοστάτησαν στο πεδίο της έρευνας της Τεχνητής Νοημοσύνης. Έκτοτε, η ΤΝ έχει λάβει ανοδική πορεία και εξελίσσεται καθημερινά κάνοντας την καθημερινότητά μας πιο εύκολη και ενδιαφέρουσα [8].

**Εικόνα 1: Alan Turing**

Πηγή: <https://medium.com/@DrChuckle/the-unknown-heroes-alan-turing-0c6305602de0>

Κατά διαστήματα έχουν διατυπωθεί διάφοροι ορισμοί για την Τεχνητή Νοημοσύνη, αφού κάποιοι επιστήμονες επικεντρώνονται στη διαδικασία της σκέψης και κάποιοι στην συμπεριφορά των συστημάτων. Οι Russell & Norvig ταξινόμησαν τους ορισμούς της ΤΝ σε τέσσερις βασικές κατηγορίες, από τις οποίες καθεμιά προσεγγίζει το στόχο της από διαφορετική σκοπιά [9].

Η πρώτη κατηγορία, ορίζει ως στόχο την ανάπτυξη συστημάτων που σκέφτονται όπως ο ανθρώπινος εγκέφαλος. Ένα παράδειγμα ορισμού που ανήκει σε αυτή την κατηγορία είναι ο ορισμός από τον John Haugeland που διατυπώθηκε το 1985 στο βιβλίο του με τίτλο "Artificial Intelligence: The Very Idea", ο οποίος υποστήριζε ότι "ΤΝ είναι η προσπάθεια να κατασκευάσουμε υπολογιστές με διανοητική ικανότητα με την πλήρη και κυριολεκτική έννοια του όρου".

Η δεύτερη κατηγορία, θέτει ως στόχο την ανάπτυξη συστημάτων που σκέφτονται λογικά. Ένα χαρακτηριστικό παράδειγμα τέτοιου ορισμού δίνεται από τον Patrick Henry Winston, έναν από τους γνωστούς ερευνητές στον τομέα της Τεχνητής Νοημοσύνης που παρουσίασε αυτόν τον ορισμό στο βιβλίο του "Artificial Intelligence" (1977), ο οποίος θεωρεί πως "ΤΝ είναι η μελέτη των υπολογισμών που καθιστούν εφικτή την αντίληψη, τη λογική σκέψη και την αντίδραση".

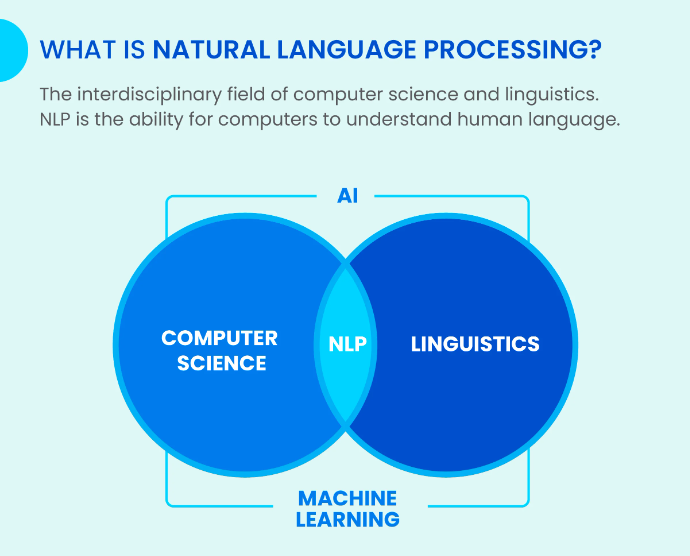
Η τρίτη κατηγορία, εστιάζει σε συστήματα που συμπεριφέρονται όπως οι άνθρωποι. Ο ορισμός των Elaine Rich και Kevin Knight, από το βιβλίο τους "Artificial Intelligence" (1983), αποτελεί παράδειγμα για αυτή την κατηγορία καθώς ορίζει την ΤΝ ως "Η μελέτη του πως να κάνουμε τους υπολογιστές να κάνουν πράγματα στα οποία αυτήν τη στιγμή οι άνθρωποι είναι καλύτεροι".

Η τελευταία κατηγορία, επικεντρώνεται στην ανάπτυξη συστημάτων που αντιδρούν λογικά. Ο ακόλουθος ορισμός του George F. Luger, στο βιβλίο του "Artificial Intelligence: Structures and Strategies for Complex Problem Solving" (1993) αποτελεί χαρακτηριστικό παράδειγμα, καθώς υποστηρίζει ότι "ΤΝ είναι ο τομέας της επιστήμης των υπολογιστών που ασχολείται με την αυτοματοποίηση της ευφυούς συμπεριφοράς".

Ο γενικός ορισμός που δώσαμε παραπάνω, περιλαμβάνει χαρακτηριστικά και από τις τέσσερις κατηγορίες καθώς ο κλάδος της Τεχνητής Νοημοσύνης είναι συνεχώς εξελισσόμενος και προσπαθεί να δώσει λύσεις σε προβλήματα ποικίλων κλάδων. Επομένως, δεν μπορεί να εστιάσει μόνο στον τρόπο σκέψης ή συμπεριφοράς των συστημάτων αλλά και στα δυο ταυτόχρονα για να έχουμε τα επιθυμητά αποτελέσματα. Η έρευνα στην ΤΝ έχει ήδη αποδώσει καρπούς σε ένα ευρύ φάσμα περιοχών όπως είναι η Μηχανική Μάθηση, η Επεξεργασία και Κατανόηση Φυσικής Γλώσσας, οι Ευφυείς Πράκτορες, η Τεχνητή Όραση (Computer Vision), η Ρομποτική κλπ.

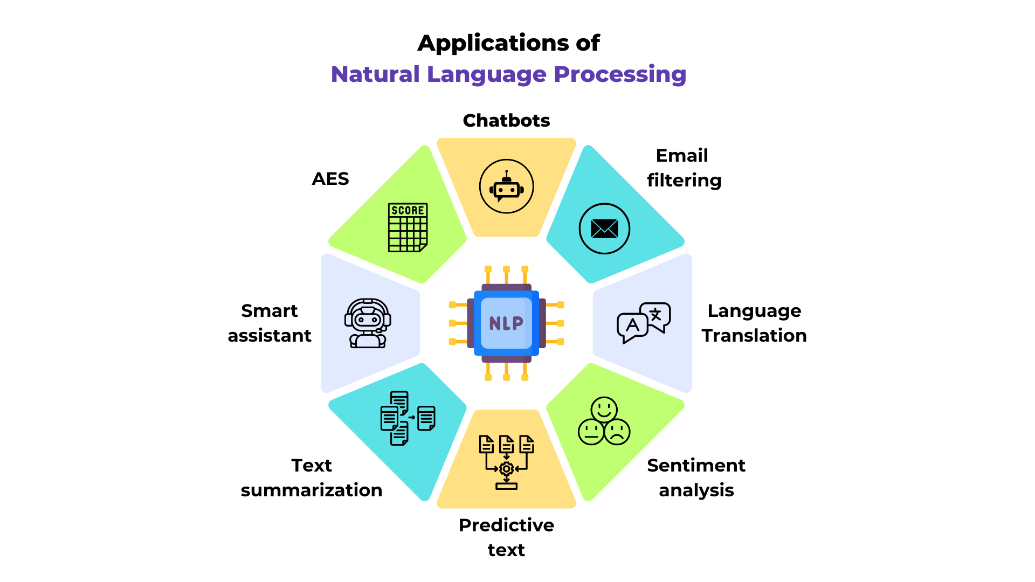
## 2.2 Επεξεργασία Φυσικής Γλώσσας

Η Επεξεργασία Φυσικής Γλώσσας (Natural Language Processing, NLP), είναι ο τομέας που συνδυάζει την επιστήμη της Γλωσσολογίας, της Πληροφορικής και της Τεχνητής Νοημοσύνης. Ερευνά την επικοινωνία μεταξύ υπολογιστών και ανθρώπινων φυσικών γλωσσών, με στόχο να δώσει την δυνατότητα στον υπολογιστή να κατανοεί τη φυσική γλώσσα και να μπορεί εξάγει φυσική γλώσσα από γλωσσικά δεδομένα. Εν ολίγης, ο κύριος στόχος της Επεξεργασίας Φυσικής Γλώσσας είναι ο υπολογιστής να μπορεί να αποκρυπτογραφήσει την ανθρώπινη γλώσσα, όπως την αντιλαμβάνεται ο άνθρωπος, καθώς και τα νοήματα που μεταφέρονται μέσα από ύφος της (σαρκασμός, χιούμορ, ειρωνεία κ.α.), κάτι που είναι δύσκολο να αντιληφθεί μια μηχανή για να μπορεί να τη χρησιμοποιήσει για όφελος της ανθρωπότητας.

**Εικόνα 2: Ορισμός της Επεξεργασίας Φυσικής Γλώσσας**

Πηγή: <https://www.scalenut.com/blogs/nlp-vs-nlu>

Η NLP, αποτελεί εδώ και χρόνια μέρος της καθημερινής μας ζωής, τροφοδοτώντας τις μηχανές αναζήτησης, μεταφράζοντας αυτόματα κείμενα από μια γλώσσα σε μια άλλη, βοηθώντας τα Chatbots στην εξυπηρέτηση πελατών με προφορικές αλλά και γραπτές εντολές, τα συστήματα GPS που λειτουργούν με φωνή και τους ψηφιακούς βοηθούς στα smartphones. Η NLP διαδραματίζει επίσης, αυξανόμενο ρόλο σε επιχειρηματικές λύσεις που συμβάλλουν στον έλεγχο και την αυτοματοποίηση των επιχειρηματικών λειτουργιών, στην αύξηση της παραγωγικότητας των εργαζομένων και στην απλοποίηση κρίσιμων επιχειρηματικών διαδικασιών [10].

**Εικόνα 3: Εφαρμογές της NLP στην καθημερινότητα**

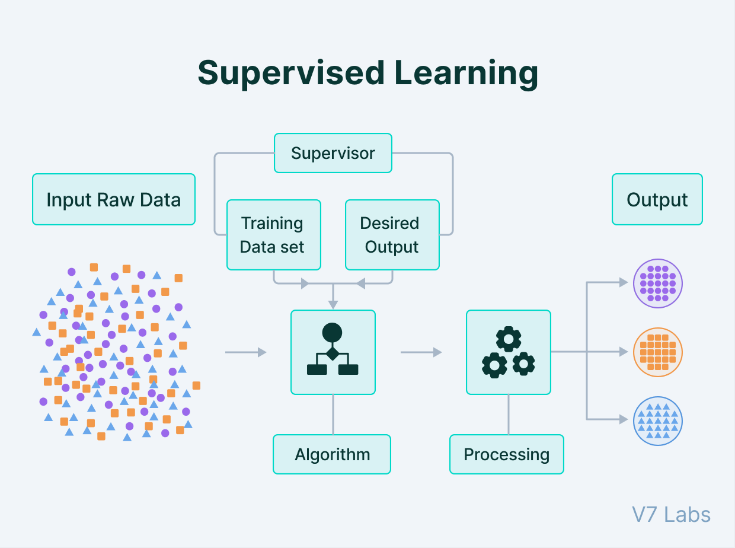
Πηγή: <https://eastgate-software.com/top-8-applications-of-natural-language-processing-nlp/>

## 2.3 Μηχανική Μάθηση

Η Μηχανική Μάθηση (Machine Learning, ML), αποτελεί υποπεδίο της επιστήμης των υπολογιστών και μπορεί να οριστεί ως "το φαινόμενο κατά το οποίο ένα σύστημα βελτιώνει την απόδοσή του όταν εκτελεί μια συγκεκριμένη εργασία, χωρίς να υπάρχει ανάγκη να προγραμματιστεί εκ νέου". Η Μηχανική Μάθηση, έχει ως σκοπό τη δημιουργία μηχανών που είναι ικανές να μαθαίνουν και επομένως να βελτιώνουν, την απόδοσή τους σε κάποιους τομείς μέσω της αξιοποίησης της γνώσης και εμπειρίας που έχουν ήδη αποκτήσει. Ο Tom M. Mitchell το 1997 πρότεινε έναν πιο επίσημο ορισμό που έγινε ευρέως χρησιμοποιούμενος και υποστηρίζει ότι "Ένα πρόγραμμα υπολογιστή λέμε ότι μαθαίνει από την εμπειρία Ε ως προς κάποια κλάση εργασιών Τ και μέτρο απόδοσης Ρ, αν η απόδοσή του σε εργασίες από το Τ, όπως μετριέται από το Ρ, βελτιώνεται μέσω της εμπειρίας Ε" [11].

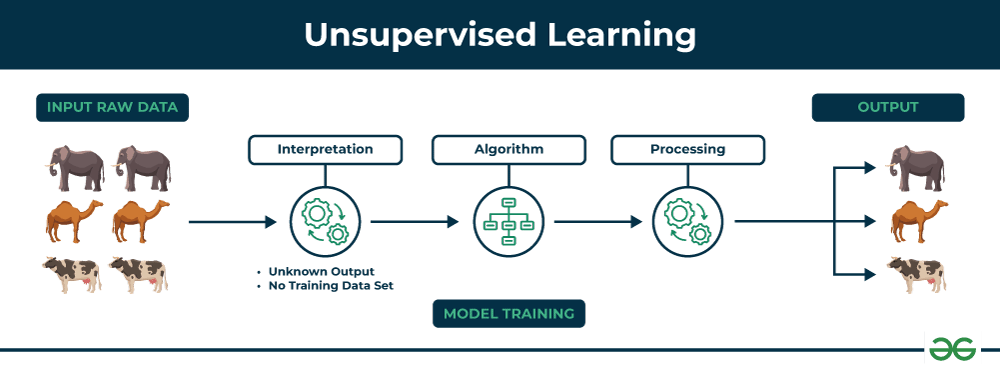
Η Μηχανική Μάθηση αναπτύσσει τρεις τρόπους μάθησης, ανάλογους με τους τρόπους με τους οποίους μαθαίνει και ο άνθρωπος: επιβλεπόμενη μάθηση, μη επιβλεπόμενη μάθηση και ενισχυτική μάθηση.

Στην Επιβλεπόμενη Μάθηση (Supervised Learning), ο αλγόριθμος κατασκευάζει μια συνάρτηση που απεικονίζει δεδομένες εισόδους (σύνολο εκπαίδευσης) σε γνωστές επιθυμητές εξόδους, έχει ως τελικό στόχο τη γενίκευση της συνάρτησης αυτής, για να μπορεί να χρησιμοποιηθεί και σε εισόδους με άγνωστη έξοδο. Χρησιμοποιείται σε προβλήματα όπως είναι η Ταξινόμηση, η Πρόγνωση και η Διερμηνεία [12].

**Εικόνα 4: Παράδειγμα Επιβλεπόμενης Μάθησης**

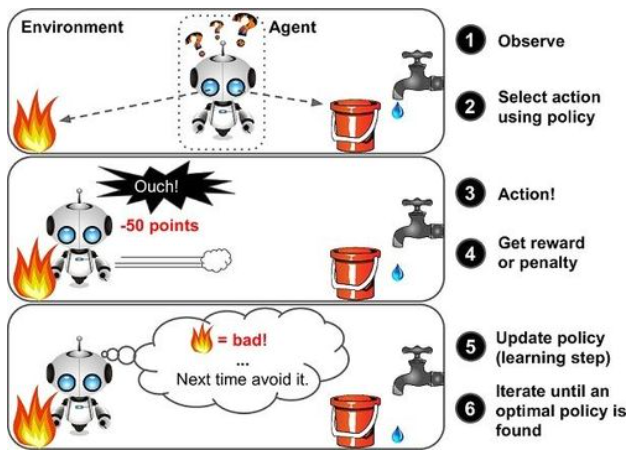
Πηγή: <https://www.v7labs.com/blog/supervised-vs-unsupervised-learning>

Κατά τη Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning), ο αλγόριθμος κατασκευάζει ένα μοντέλο για ένα σύνολο εισόδων υπό μορφή παρατηρήσεων χωρίς να γνωρίζει τις επιθυμητές εξόδους. Πιο απλά, χωρίς να παρέχεται κάποια εμπειρία στον αλγόριθμο μάθησης, πρέπει αυτός να βρει την δομή των δεδομένων εισόδου. Κάποια από τα προβλήματα στα οποία μπορεί να φανεί χρήσιμη είναι η Ανάλυση Συσχετισμών και η Ομαδοποίηση [12].

**Εικόνα 5: Παράδειγμα Μη Επιβλεπόμενης Μάθησης**

Πηγή: <https://www.geeksforgeeks.org/supervised-unsupervised-learning/>

Στην Ενισχυτική Μάθηση (Reinforcement Learning), ο αλγόριθμος μαθαίνει μια στρατηγική ενεργειών μέσω συνεχούς αλληλεπίδρασης με το περιβάλλον με στόχο την μεγιστοποίηση ενός βαθμωτού δείκτη απόδοσης. Η δομή των αλγορίθμων αυτής της κατηγορίας, στηρίζεται στην διαδικασία αναζήτησης με δοκιμή, στην εύρεση σφάλματος καθώς και στην ετεροχρονισμένη επιβράβευση. Χρησιμοποιείται κατά κύριο λόγο σε προβλήματα Σχεδιασμού, όπως για παράδειγμα ο έλεγχος κίνησης ρομπότ και η βελτιστοποίηση εργασιών σε εργοστασιακούς χώρους [4].

**Εικόνα 6: Παράδειγμα Ενισχυτικής Μάθησης**

Πηγή: <https://marutitech.com/businesses-reinforcement-learning/>

## 2.4 Αλληλεπίδραση Ανθρώπου-Υπολογιστή

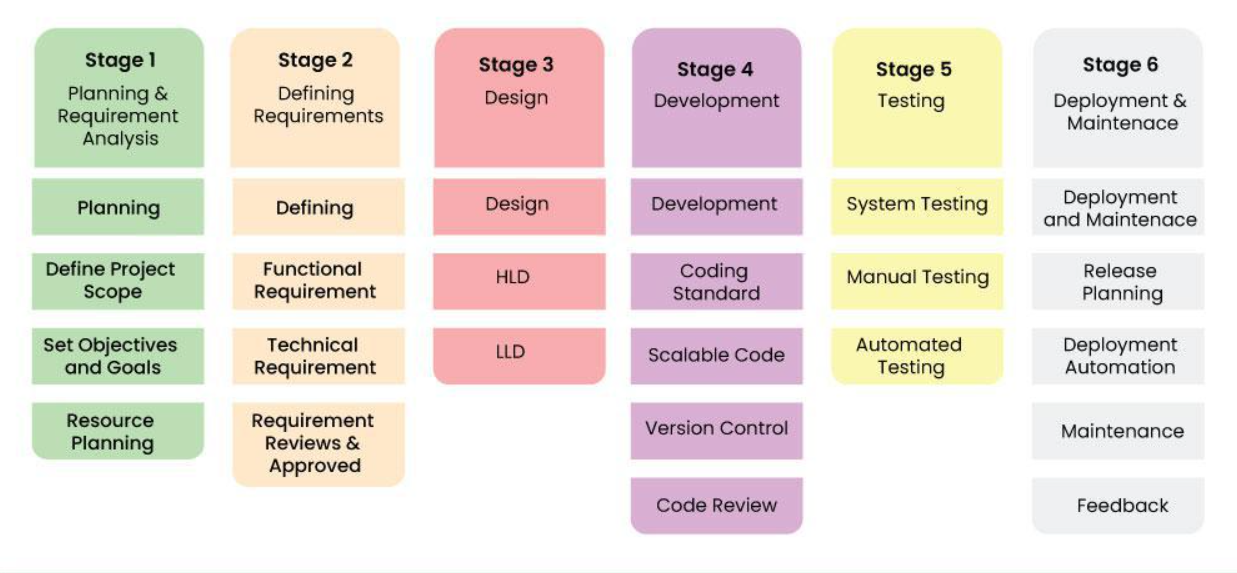
Η Αλληλεπίδραση Ανθρώπου-Υπολογιστή (Human-Computer Interaction, HCI), αποτελεί ένα διεπιστημονικό πεδίο που συνδυάζει στοιχεία από πολλούς τομείς μεταξύ των οποίων είναι η Πληροφορική, η γνωστική και κοινωνική ψυχολογία, η γλωσσολογία, ο βιομηχανικός σχεδιασμός και άλλα σχετικά γνωστικά πεδία. Σκοπός του είναι η ανάλυση, η σχεδίαση, η υλοποίηση και η αξιολόγηση των διεπαφών που χρησιμοποιούνται σε υπολογιστικά συστήματα για να διευκολύνουν την αλληλεπίδραση με τους χρήστες. Η αλληλεπίδραση μεταξύ χρηστών και υπολογιστών πραγματοποιείται μέσω της διεπαφής χρήστη, η οποία περιλαμβάνει το κατάλληλο λογισμικό και υλικό [13].

## 2.5 Τεχνολογία Λογισμικού

Το 1968, ο Fritz Bauer όρισε τη Τεχνολογία Λογισμικού (Software Engineering) ως «την καθιέρωση και χρήση ορθών αρχών μηχανικής προκειμένου να επιτευχθεί οικονομικά ανεπτυγμένο λογισμικό που είναι αξιόπιστο και λειτουργεί αποτελεσματικά σε πραγματικές μηχανές». Αργότερα, ο Stephen Schach στο βιβλίο του "Object-Oriented and Classical Software Engineering" (1999) έδωσε τον δικό του ορισμό για τη Τεχνολογία Λογισμικού ως «Μια επιστήμη της οποίας στόχος είναι η παραγωγή ποιοτικού λογισμικού, λογισμικού που παραδίδεται εγκαίρως, εντός του προϋπολογισμού και ικανοποιεί τις απαιτήσεις του». Οι ορισμοί αυτοί είναι αποδεκτοί από την πλειοψηφία, ωστόσο, λόγω της αύξησης του κόστους συντήρησης του λογισμικού, ο στόχος μετατοπίζεται πλέον στην υλοποίηση καλής ποιότητας λογισμικού που έχει τη δυνατότητα να συντηρηθεί, να παραδοθεί εντός του χρονικού πλαισίου και προϋπολογισμού καθώς και να ικανοποιεί τις απαιτήσεις του πελάτη [14]. Επομένως, ως Τεχνολογία Λογισμικού ορίζουμε τον κλάδο της Πληροφορικής ο οποίος ασχολείται με την μελέτη και την εφαρμογή συστηματικών, μεθοδικών και ποσοτικοποιημένων προσεγγίσεων για την ανάπτυξη, λειτουργία και συντήρηση του λογισμικού [15].

Το λογισμικό ως προϊόν, όπως άλλωστε και όλα τα βιομηχανικά προϊόντα, έχει τον κύκλο ζωής του. Ο κύκλος ζωής μπορεί να οριστεί ως ο χρόνος από την σύλληψη της ιδέας που πρόκειται να αναπτυχθεί μέχρι την απόσυρση του προϊόντος. Ένας κύκλος ανάπτυξης λογισμικού συνήθως περιλαμβάνει τα εξής έξι βασικά στάδια: Ανάλυση απαιτήσεων, Ορισμός Απαιτήσεων, Σχεδιασμός, Υλοποίηση, Δοκιμές και Ανάπτυξη, με την σειρά που αναφέρθηκαν [16]

Υπάρχουν πολλά και διαφορετικά μοντέλα κύκλου ζωής λογισμικού (Software Development Process Models) που ακολουθούνται κατά τη φάση ανάπτυξης λογισμικού. Κάθε μοντέλο ακολουθεί μια σειρά φάσεων που είναι μοναδικές για τον τύπο του, ώστε να διασφαλίζεται η επιτυχία στο στάδιο της ανάπτυξης λογισμικού. Η επιλογή του κατάλληλου μοντέλου είναι πολύ σημαντική για την ανάπτυξη ενός προϊόντος λογισμικού. Οι εταιρείες, ανάλογα με το προϊόν λογισμικού που θέλουν να αναπτύξουν, επιλέγουν τον τύπο μοντέλου ανάπτυξης που είναι κατάλληλος για αυτό. Κάποια από τα πιο γνωστά μοντέλα είναι του καταρράκτη, το επαυξητικό, το σπειροειδές και το επαναληπτικό [15]. Ωστόσο, αυτές τις μέρες το «Agile Methodology» είναι το πιο δημοφιλής στην αγορά.

**Εικόνα 7: Stages of the Software Development Life Cycle**

Πηγή: <https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/>

## 2.6 Βάσεις Δεδομένων

Η ανάπτυξη των επιχειρήσεων και των οργανισμών με την πάροδο των χρόνων οδήγησε στην ανάγκη για αποτελεσματική διαχείριση μεγάλων όγκων δεδομένων και στην απαίτηση για ταχεία και ακριβή ανάκτηση πληροφοριών. Έτσι, παρουσιάστηκε η ανάγκη ανάπτυξης ενός συστήματος που θα μπορούσε να οργανώσει, να αποθηκεύσει και να εξασφαλίσει την ασφάλεια και την ακεραιότητα των δεδομένων. Τα συστήματα αυτά ονομάστηκαν Βάσεις Δεδομένων και επιτρέπουν επίσης πρόσβαση σε δεδομένα από πολλαπλούς χρήστες ταυτόχρονα, διασφαλίζοντας τη συνέπεια και την αποτελεσματικότητα στη λειτουργία ενός οργανισμού.

Ορίζουμε ως Βάση Δεδομένων (Database), μια οργανωμένη συλλογή δεδομένων που σχετίζονται μεταξύ τους με κάποιο λογικό τρόπο και τα έχουν την δυνατότητα ανάκτησης μέσω αναζήτησης κατ' απαίτηση.

Τα πρώτα συστήματα διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ), εμφανίστηκαν την δεκαετία του 1960 και χρησιμοποιήθηκαν σε μεγάλες και δύσκολες επιχειρήσεις, όπως ήταν το έργο προσσελήνωσης Apollo. Τη δεκαετία του 1970, η χρήση των ΣΔΒΔ ήρθε πιο κοντά στην εμπορική πραγματικότητα αφού άρχισαν να χρησιμοποιούνται για την αντιμετώπιση των ολοένα αυξανόμενων και πολύπλοκων δομών δεδομένων. Παράδειγμα αποτελεί ο βιομηχανικός κατάλογος πρώτων υλών, ο οποίος ήταν εξαιρετικά δύσκολος στην διαχείριση με τις παλαιότερες μεθόδους διαχείρισης αρχείων. Αργότερα, κατά την διάρκεια της δεκαετίας του 1980, ο E. F. Codd μαζί με τους συνεργάτες του διέδωσαν το σχεσιακό μοντέλο δεδομένων, με το οποίο τα δεδομένα αναπαρίστανται με την μορφή πινάκων. Με αυτόν τον τρόπο οι Βάσεις Δεδομένων έγιναν προσιτές και σε μη προγραμματιστές, ξεπερνώντας ένα βασικό πρόβλημα των βάσεων πρώτης γενιάς. Την επόμενη δεκαετία, ξεκίνησε μια νέα εποχή για τους υπολογιστές, αρχικά με την αρχιτεκτονική πελάτη/ διακομιστή, και έπειτα με την οργανωμένη αποθήκευση δεδομένων και την αλματώδη ανάπτυξη των Διαδικτυακών εφαρμογών. Στις μέρες μας, εξακολουθούμε α χρησιμοποιούμε το σχεσιακό μοντέλο Βάσεων Δεδομένων (SQL), ωστόσο έχουν αναπτυχθεί και μη σχεσιακές τεχνολογίες (Not only SQL, NoSQL) οι οποίες χρησιμοποιούνται κυρίως σε πολύ μεγάλο όγκο δεδομένων. Παραδείγματα σχεσιακών Βάσεων Δεδομένων, είναι η Oracle, η DB2, η SQL Server, η MySQL και άλλες, ενώ μη σχεσιακών είναι η MongoDB, η Apache Cassandra και η Hadoop. Έτσι, τα ΣΔΒΔ σχεσιακά και μη έχουν την δυνατότητα να δημιουργηθούν, να αναπτυχθούν και να διαχειριστούν μέσω των νέων εξελιγμένων τεχνολογιών [17].

## 2.7 Σάρωση Ιστοσελίδων (Web Scraping)

Το Web Scraping, είναι μια μέθοδος συλλογής δεδομένων από ιστοσελίδες, μέσω προγραμμάτων ή αυτοματοποιημένων εργαλείων. Μετά την εμφάνιση του παγκόσμιου ιστού (World Wide Web) το 1989, το πρώτο Web Robot, δημιουργήθηκε το 1993 και ονομάστηκε World Wide Web Wanderer, με στόχο την καταμέτρηση του μεγέθους των ιστών. Στα τέλη του ίδιου έτους, δημιουργήθηκε η πρώτη μηχανή αναζήτησης που βασιζόταν σε crawler για να συλλέγει δεδομένα αυτόματα, διαφοροποιούμενη από άλλες μηχανές αναζήτησης που βασίζονταν σε ανθρώπους για τη διαχείριση των συνδέσμων. Σταδιακά, με τη δημιουργία του πρώτου Web API το 2000, η διαδικασία συλλογής δεδομένων έγινε πιο συστηματική και ευκολότερα προσβάσιμη. Έτσι, μεγάλες εταιρείες όπως το eBay και το Salesforce ξεκίνησαν να δίνουν πρόσβαση σε προγραμματιστές στα δεδομένα τους μέσω αυτών των API. Σήμερα, το Web Scraping αποτελεί βασικό εργαλείο για τη συλλογή μεγάλων όγκων δεδομένων από το διαδίκτυο και είναι ευρέως διαδεδομένο σε επιχειρήσεις και μεμονωμένους χρήστες.

Το web scraping, βασίζεται σε αυτόματες διαδικασίες για τη συλλογή δεδομένων από ιστοσελίδες. Η διαδικασία περιλαμβάνει τα παρακάτω βασικά βήματα:

1. Αρχικά, ο Web Scraper στέλνει ένα αίτημα (request) σε μια ιστοσελίδα χρησιμοποιώντας το πρωτόκολλο HTTP ή HTTPS, όπως όταν ένας χρήστης επισκέπτεται κανονικά την ιστοσελίδα από τον περιηγητή του.
2. Στη συνέχεια, ιστοσελίδα απαντά με τον κώδικα HTML της, ο οποίος περιέχει τα δεδομένα που εμφανίζονται στον χρήστη.
3. Έπειτα, το πρόγραμμα "αναλύει" τον HTML κώδικα για να βρει συγκεκριμένες πληροφορίες, όπως κείμενο, εικόνες ή άλλα δεδομένα.
4. Και τέλος, οι εξαγόμενες πληροφορίες αποθηκεύονται σε μια βάση δεδομένων ή σε κάποιο αρχείο (π.χ., CSV, JSON) για περαιτέρω επεξεργασία ή ανάλυση, καθώς πολλές φορές εμφανίζονται χωρίς σωστή δομή.

Το Web Scraping, αν και πλέον είναι ευρέως διαδεδομένο και χρήσιμο εργαλείο, εγείρει σημαντικά νομικά ζητήματα, καθώς η αυτοματοποιημένη εξαγωγή δεδομένων από ιστοσελίδες μπορεί να παραβιάζει τους όρους χρήσης ή τη νομοθεσία προστασίας δεδομένων. Ωστόσο, οι νομικές πτυχές του, διαφέρουν ανάλογα με τη δικαιοδοσία και το νομικό πλαίσιο κάθε χώρας, ενώ σημαντικό ρόλο παίζουν και οι πολιτικές των ίδιων των ιστοσελίδων [18], [19].

## 2.8 Chatbot

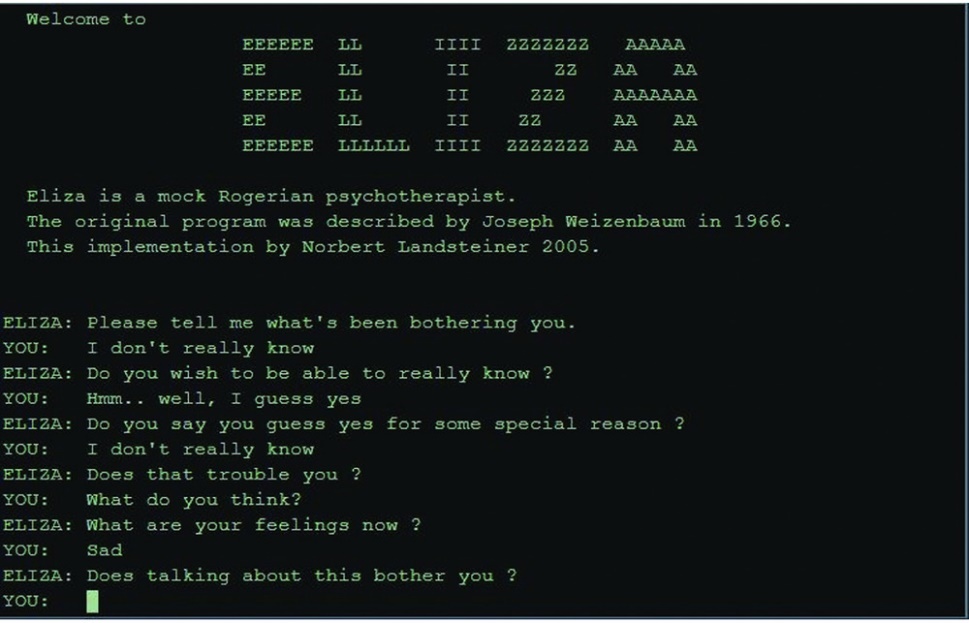
### 2.8.1 Ορισμός

Chatbot, ονομάζουμε ένα λογισμικό ή μια εφαρμογή που χρησιμοποιεί Τεχνητή Νοημοσύνη για να αλληλοεπιδρά με χρήστες μέσω γραπτού ή προφορικού λόγου. Ο βασικός σκοπός ενός Chatbot, είναι να πραγματοποιεί συνομιλίες με τους χρήστες του, παρέχοντάς τους πληροφορίες, απαντώντας στις ερωτήσεις τους και εκτελώντας ενέργειες βασισμένες σε ερωτήματα ή εντολές που του δίνονται. Για να καταφέρει να ανταπεξέλθει στις απαιτήσεις των χρηστών, χρησιμοποιεί την Επεξεργασία Φυσικής Γλώσσας ώστε να κατανοεί τα γλωσσικά δεδομένα που του εισάγουν και να απαντάει κατάλληλα.

### 2.8.2 Ιστορική αναδρομή στα Chatbot

Οι διαλογικοί πράκτορες, γνωστοί και ως Chatbots, έχουν μια μακρά και ενδιαφέρουσα ιστορία που αντανακλά την εξέλιξη της Τεχνητής Νοημοσύνης και της Επεξεργασίας Φυσικής Γλώσσας με την πάροδο των χρόνων. Η ιδέα των διαλογικών πρακτόρων ξεκίνησε από τα πρώτα χρόνια ανάπτυξης της Τεχνητής Νοημοσύνης, με τις πρώτες προσπάθειες να εστιάζουν στη δημιουργία προγραμμάτων που θα μπορούσαν να "κατανοήσουν" και να "απαντήσουν" σε ανθρώπινες ερωτήσεις με φυσική γλώσσα.

Το 1950, ο Alan Turing στο διάσημο άρθρο του "Computing Machinery and Intelligence" [8], πρότεινε το Turing Test ως έναν τρόπο για να κρίνεται αν μια μηχανή μπορεί να θεωρηθεί "έξυπνη" ή όχι. Η ιδέα ήταν ότι αν ένας άνθρωπος δεν μπορούσε να ξεχωρίσει αν συνομιλεί με έναν άνθρωπο ή μια μηχανή, τότε η μηχανή θα μπορούσε να θεωρηθεί νοήμων. Έτσι αυτό έθεσε τις βάσεις για την ανάπτυξη διαλογικών συστημάτων και έπειτα από λίγα χρόνια, το 1966 αναπτύχθηκε από τον Joseph Weizenbaum στο MIT, το πρώτο προγράμματα που μπορούσε να προσομοιώσει μια συνομιλία με άνθρωπο, γνωστό ως ELIZA. Το ELIZA, μπορούσε να προσποιηθεί διάφορους χαρακτήρες, ωστόσο, ο πιο διάσημος "χαρακτήρας" του ήταν ο "ψυχοθεραπευτής". Το πρόγραμμα βασιζόταν σε ένα σύνολο κανόνων που αντιστοιχούσαν συγκεκριμένα μοτίβα (pattern matching) στην είσοδο του χρήστη με προκαθορισμένες απαντήσεις και δεν απαιτούσαν κατανόηση της σημασίας των λέξεων ή των φράσεων από το πρόγραμμα. Για παράδειγμα, αν ο χρήστης έγραφε "Νιώθω άσχημα σήμερα", το ELIZA μπορούσε να απαντήσει με "Γιατί νιώθεις άσχημα σήμερα;", βασιζόμενο στον κανόνα που ανιχνεύει τη λέξη "νιώθω" και προσθέτοντας μια ερώτηση γύρω από αυτή [20].

  
**Εικόνα 8: Συνομιλία με το ELIZA**

Πηγή: <https://community.neontools.io/chatbots/>

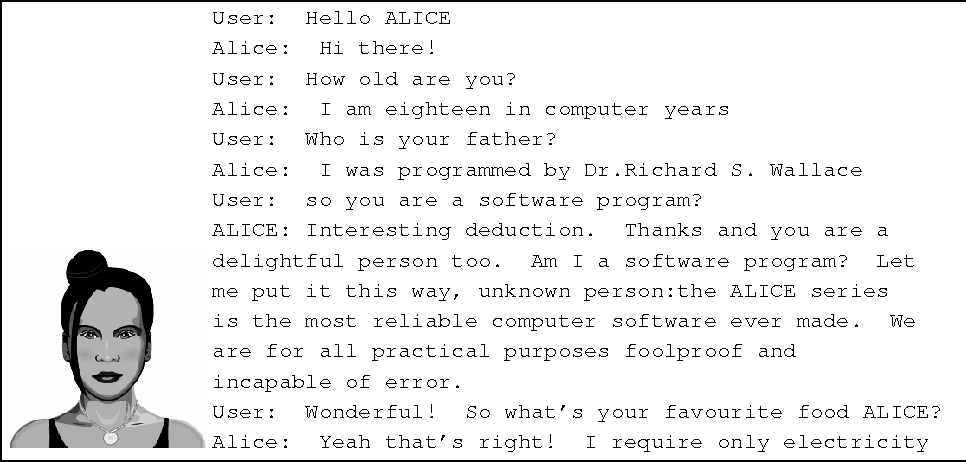
Οι επόμενες δεκαετίες μέχρι το 1990, ήταν σημαντικές για την εξέλιξη των διαλογικών πρακτόρων, καθώς σηματοδότησαν τη μετάβαση από τις αρχικές ακαδημαϊκές έρευνες σε πιο προχωρημένα συστήματα που άρχισαν να βρίσκουν εφαρμογές σε πραγματικά επιχειρηματικά μοντέλα. Ένα από τα πρώτα παραδείγματα ενός διαλογικού πράκτορα που προσπαθούσε να προσομοιώσει την ανθρώπινη σκέψη και συγκεκριμένα τη σκέψη ενός ατόμου με παρανοϊκή σχιζοφρένεια ήταν το PARRY. Αναπτύχθηκε το 1972 από τον Kenneth Colby, ψυχίατρο και επιστήμονα πληροφορικής στο Πανεπιστήμιο του Στάνφορντ. Το PARRY χρησιμοποιούσε κανόνες και προγράμματα λογισμικού για να αναπαράγει τα συμπτώματα της παρανοϊκής σχιζοφρένειας. Σε αντίθεση με το ELIZA, το οποίο ήταν απλό και βασιζόταν σε αναγνώριση λέξεων, το PARRY ήταν πιο περίπλοκο και ενσωμάτωνε ένα μοντέλο που προσπαθούσε να "σκεφτεί" όπως ένας παρανοϊκός ασθενής [21]

Το 1983, δημιουργήθηκε ένα πρόγραμμα από τον William Chamberlain και τον Thomas Etter, το οποίο ήταν προσανατολισμένο στη δημιουργία αφηγηματικών κειμένων και φράσεων που προσομοίωναν τη δημιουργικότητα στη γλώσσα και ονομαζόταν Racter. Σε αντίθεση με τα προηγούμενα συστήματα, το Racter δεν προσπαθούσε απλώς να ανταποκριθεί στις εισαγωγές του χρήστη, αλλά δημιουργούσε αυτόνομες αφηγήσεις και διαλόγους. Χρησιμοποιούσε ένα σύνολο προκαθορισμένων κανόνων για τη σύνθεση γλώσσας, κάτι που του επέτρεπε να παράγει "κείμενα" που σε μερικές περιπτώσεις ήταν λογικά συνεπή και αστεία, αλλά σε άλλες περιπτώσεις ασύνδετα ή παραληρηματικά. Ήταν ένα από τα πρώτα προγράμματα που δοκίμασαν την αυτόνομη δημιουργία περιεχομένου, και παρόλο που δεν είχε καμία πραγματική κατανόηση της γλώσσας ή του περιεχομένου, έδειξε τις δυνατότητες που θα μπορούσαν να έχουν τα συστήματα διαλόγου στην παραγωγή δημιουργικού λόγου [20]. Κατά τη διάρκεια αυτής της περιόδου, οι επιχειρήσεις άρχισαν να ενδιαφέρονται για τα διαλογικά συστήματα, αναγνωρίζοντας τις δυνατότητες τους σε τομείς όπως η εξυπηρέτηση πελατών και η αυτοματοποίηση εργασιών. Έτσι στα τέλη της δεκαετίας του 1980 και στις αρχές της δεκαετίας του 1990, άρχισαν να αναπτύσσονται τα πρώτα εμπορικά chatbots για τη διαχείριση τηλεφωνικών κέντρων και την παροχή υποστήριξης πελατών

Τα χρόνια που ακολούθησαν, από τη δεκαετία του 1990 έως τη δεκαετία του 2010, σημειώθηκε μια σημαντική πρόοδος στην ανάπτυξη διαλογικών πρακτόρων με αυξανόμενη ικανότητα κατανόησης φυσικής γλώσσας. Αυτή η εξέλιξη, οδήγησε στην άνοδο πιο εξελιγμένων συστημάτων που μπόρεσαν να κατανοήσουν καλύτερα τα συμφραζόμενα, να επεξεργάζονται σύνθετες ερωτήσεις και να απαντούν με πιο σχετικές και «ανθρώπινες» απαντήσεις. Τη δεκαετία του 1990 και τις αρχές της δεκαετίας του 2000, τα διαλογικά συστήματα είδαν επίσης σημαντική πρόοδο στην Επεξεργασία Φυσικής Γλώσσας, ένα πεδίο που ασχολείται με την κατανόηση και τη διαχείριση της φυσικής γλώσσας από υπολογιστές. Κατά τη διάρκεια αυτής της περιόδου, διάφορα συστήματα και πλατφόρμες που βασίζονταν σε τεχνολογίες NLP άρχισαν να κάνουν την εμφάνισή τους και να υιοθετούνται από επιχειρήσεις και καταναλωτές.

Το Jabberwacky, ήταν ένα διαλογικό σύστημα που δημιουργήθηκε από τον Rollo Carpenterτο 1988 αλλά δημοσιεύτηκε το 1997 [20]. Σχεδιάστηκε για να μιμείται ανθρώπινη συζήτηση μέσω συνεχούς αλληλεπίδρασης με τους χρήστες και μαθαίνοντας από τις απαντήσεις τους. Επιπλέον, βασιζόταν σε στατιστικές μεθόδους και χρησιμοποιούσε τεχνικές αντιστοίχισης προτύπων για να παράγει απαντήσεις που συχνά ήταν απροσδόκητες ή χιουμοριστικές, προσπαθώντας να συμπεριφέρεται όσο το δυνατόν πιο ανθρώπινα και να προσφέρει ψυχαγωγία [21], [22].

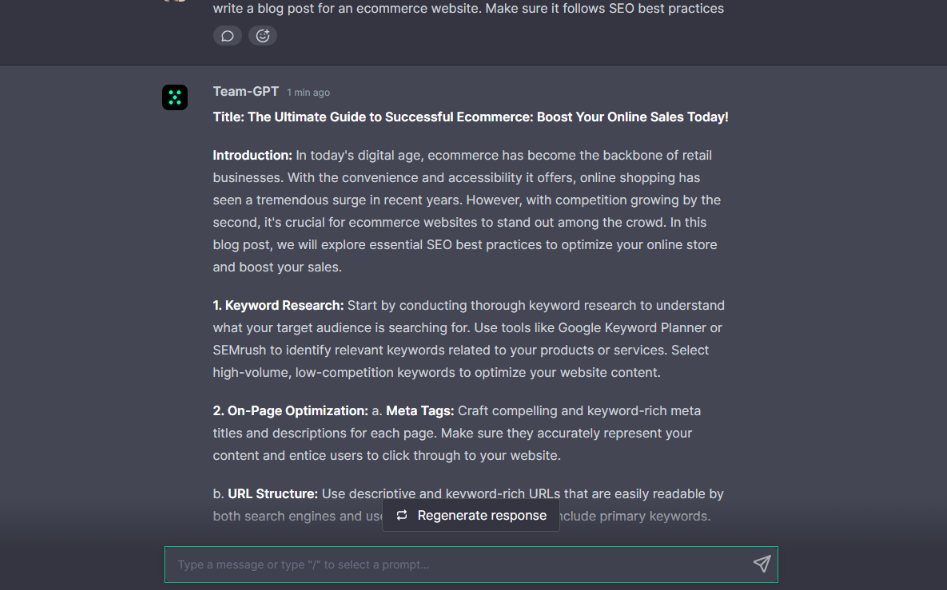
Το 1995, αναπτύχθηκε από τον Richard Wallace το ALICE (Artificial Linguistic Internet Computer Entity) που ήταν ένα από τα πιο προηγμένα συστήματα διαλόγου εκείνης της εποχής. Χρησιμοποιούσε μια γλώσσα σήμανσης που ονομάζεται AIML (Artificial Intelligence Markup Language) για να καθορίσει τους κανόνες διαλόγου. Το ALICE αποτέλεσε ένα σημείο καμπής στην ιστορία των διαλογικών πρακτόρων, καθώς έδειξε ότι τα συστήματα αυτά μπορούσαν να γίνουν πιο εξελιγμένα και πιο φυσικά στις απαντήσεις τους [23].

**Εικόνα 9: Συνομιλία με το ALICE**

Πηγή: <https://www.semanticscholar.org/paper/Chatbots%3A-Are-they-Really-Useful-Shawar-Atwell/8d8284bfba7ebcb4e2575d864ec7c16ea6a168f0/figure/2>

Το 2011 κυκλοφόρησε η Siri, ο ψηφιακός βοηθός της Apple και αποτέλεσε ένα από τα πρώτα συστήματα που ενσωμάτωσαν την επεξεργασία φυσικής γλώσσας με αναγνώριση φωνής για την παροχή χρήσιμων πληροφοριών και την εκτέλεση καθημερινών εργασιών. Η Siri ανέβασε τον πήχη για το τι μπορούσαν να πετύχουν οι διαλογικοί πράκτορες, ενσωματώνοντας τεχνολογίες όπως η αναγνώριση ομιλίας και η πρόσβαση σε διάφορες πηγές δεδομένων στο διαδίκτυο. Παρόμοιοι ψηφιακοί βοηθοί που αναπτύχθηκαν μετέπειτα, αποτελούν το Google Assistant, το Microsoft Cortana και η Alexa της Amazon. Οι βοηθοί αυτοί δεν περιορίστηκαν μόνο σε φωνητικές εντολές αλλά μπορούσαν επίσης να ενσωματωθούν σε άλλες εφαρμογές και υπηρεσίες, δημιουργώντας ένα ευρύ οικοσύστημα προϊόντων και υπηρεσιών [20], [21].

Τα τελευταία χρόνια, οι διαλογικοί πράκτορες εξελίχθηκαν ραγδαία χάρη στην πρόοδο της Μηχανικής Μάθησης, της Τεχνητής Νοημοσύνης και των Νευρωνικών Δικτύων. Τα μοντέλα που βασίζονται σε βαθιά Νευρωνικά Δίκτυα, όπως το GPT, έχουν τη δυνατότητα να κατανοούν το πλαίσιο των συνομιλιών και να δημιουργούν συνεχείς και σχετικούς διαλόγους. Αυτό επιτυγχάνεται μέσω της εκπαίδευσης αυτών των μοντέλων σε τεράστια σύνολα δεδομένων που περιλαμβάνουν κείμενα από βιβλία, άρθρα, και το διαδίκτυο. Το GPT βασίζεται στην αρχιτεκτονική Transformer, η οποία επιτρέπει στο μοντέλο να κατανοεί και να επεξεργάζεται κείμενα με έναν εξαιρετικά αποτελεσματικό τρόπο. Το GPT-2, το οποίο παρουσιάστηκε το 2019, ήταν η πρώτη σημαντική έκδοση αυτής της τεχνολογίας που έδειξε πόσο καλά μπορεί ένα μοντέλο να δημιουργεί συνεκτικά και σχετικά κείμενα. Ακολούθησε το GPT-3 το 2020, το οποίο ενσωμάτωσε 175 δισεκατομμύρια παραμέτρους, καθιστώντας το ένα από τα πιο ισχυρά γλωσσικά μοντέλα της εποχής του. Το GPT-3 μπορούσε να παράγει κείμενα που συχνά ήταν δύσκολο να ξεχωρίσει κανείς αν είχαν γραφτεί από άνθρωπο ή από μηχανή. Με την κυκλοφορία του GPT-3 και, αργότερα, του GPT-4, η OpenAI εισήγαγε το ChatGPT, ένα προηγμένο chatbot που εκμεταλλεύεται τις δυνατότητες αυτών των μοντέλων για να προσφέρει συνομιλίες υψηλού επιπέδου. Το ChatGPT είναι η εφαρμογή του GPT-3 και GPT-4 που έχει σχεδιαστεί για να διεκπεραιώνει συνομιλίες με τους χρήστες. Έχει εκπαιδευτεί σε ένα ευρύ φάσμα θεμάτων, καθιστώντας το ικανό να απαντά σε ερωτήσεις, να παρέχει πληροφορίες, να βοηθά στη λήψη αποφάσεων και ακόμη και να δημιουργεί δημιουργικά κείμενα. Το ChatGPT, είναι σε θέση να κατανοεί το πλαίσιο της συνομιλίας, να διατηρεί τη συνοχή της και να ανταποκρίνεται με φυσικότητα και πολυπλοκότητα. Επιπλέον, βρίσκει εφαρμογή σε διάφορους τομείς της καθημερινότητας όπως είναι οι επιχειρήσεις, για την αυτοματοποίηση της εξυπηρέτησης πελατών, τη δημιουργία περιεχομένου και την υποστήριξη εσωτερικών λειτουργιών, όπως η ανάλυση δεδομένων και η παροχή συστάσεων [20].

**Εικόνα 10: Συνομιλία με το ChatGPT**

Πηγή: <https://team-gpt.com/blog/chatgpt-examples/>

Η ανάπτυξη των διαλογικών πρακτόρων αναμένεται να συνεχιστεί με έμφαση στη βελτίωση της κατανόησης της φυσικής γλώσσας, της συναισθηματικής νοημοσύνης και της ικανότητας προσαρμογής σε διαφορετικά πλαίσια και πολιτισμικές αναφορές. Η ενσωμάτωση με άλλες τεχνολογίες, όπως η εικονική πραγματικότητα και το Διαδίκτυο των πράγματων (Internet of Things, IoT), μπορεί να οδηγήσει σε ακόμα πιο ενδιαφέρουσες εφαρμογές οι οποίες θα διευρυνθούν σε διάφορους τομείς, όπως η εκπαίδευση, η υγεία, και οι επιχειρήσεις, με στόχο την υποστήριξη και βελτίωση της ανθρώπινης ζωής και εργασίας.

### 2.8.3 Τύποι των Chatbot

Ανάλογα με τις λειτουργίες και τεχνολογίες που χρησιμοποιούν για την αποκρυπτογράφηση των ερωτημάτων των χρηστών, τα Chatbots διαχωρίζονται σε 6 μεγάλες κατηγορίες.

Ένας από τους τύπους, είναι τα Menu or button-based Chatbots, τα οποία είναι από τα πιο χρησιμοποιούμενα καθώς οι χρήστες μπορούν να αλληλοεπιδρούν με αυτά απλά επιλέγοντας το κατάλληλο κουμπί από το προκαθορισμένο μενού το οποίο αντιπροσωπεύει καλύτερα τις ανάγκες τους. Τα Chatbots αυτά, λειτουργούν σαν ένα δέντρο αποφάσεων δηλαδή, ανάλογα με την εκάστοτε επιλογή του χρήστη αυτό τον προτρέπει σε ένα άλλο σύνολο επιλογών μέχρι να φτάσει στην επιθυμητή επιλογή.

Μία ακόμη κατηγορία, αποτελούν τα Rules-based Chatbots, τα οποία βασίζονται στην λογική if/then για την ανάπτυξη αυτοματοποιημένων ροών συνομιλίας. Λειτουργούν ορίζοντας προκαθορισμένους συνδυασμούς επιλογών ερωτήσεων και απαντήσεων, ώστε το Chatbot να μπορεί να κατανοήσει τα δεδομένα του χρήστη και να ανταποκριθεί με ακρίβεια. Αυτή η κατηγορία ταιριάζει ιδανικά σε επιχειρήσεις οι οποίες γνωρίζουν ποιες ερωτήσεις θα λάβουν από τους χρήστες και ποιες είναι οι πιθανές απαντήσεις για αυτές.

Ένας επιπλέον τύπος, είναι τα AI-Powered Chatbots, κατά τα οποία γίνεται χρήση της Μηχανικής Μάθησης και της Επεξεργασία Φυσικής Γλώσσας για να κατανοούν και να μαθαίνουν από τις συνομιλίες με τους χρήστες και επομένως, να βελτιώνουν τις επιδόσεις τους. Με αυτόν το τρόπο καταφέρνουν να δίνουν πιο σύνθετες και εξατομικευμένες απαντήσεις, όπως χρειάζεται στην υποστήριξη πελατών ή στις πωλήσεις.

Επίσης, ένα εργαλείο συνομιλίας που έχει αναπτυχθεί είναι τα Voice chatbots, τα οποία επιτρέπουν στους χρήστες να επικοινωνούν μαζί τους με προφορικό λόγο. Τα Voice Chatbots, αναγνωρίζουν και κατανοούν ομιλία, χρησιμοποιώντας τεχνολογίες Φωνητικής Αναγνώρισης και Επεξεργασίας Φυσικής Γλώσσας. Στη συνέχεια, απαντούν φωνητικά μέσω σύνθεσης ομιλίας (Text-To-Speech). Αυτό που τα χαρακτηρίζει είναι η ικανότητά τους να προσφέρουν πιο φυσική και άμεση επικοινωνία που είναι εξαιρετικά χρήσιμα σε ψηφιακούς βοηθούς, υποστήριξη πελατών, έξυπνες συσκευές και άλλες εφαρμογές [24].

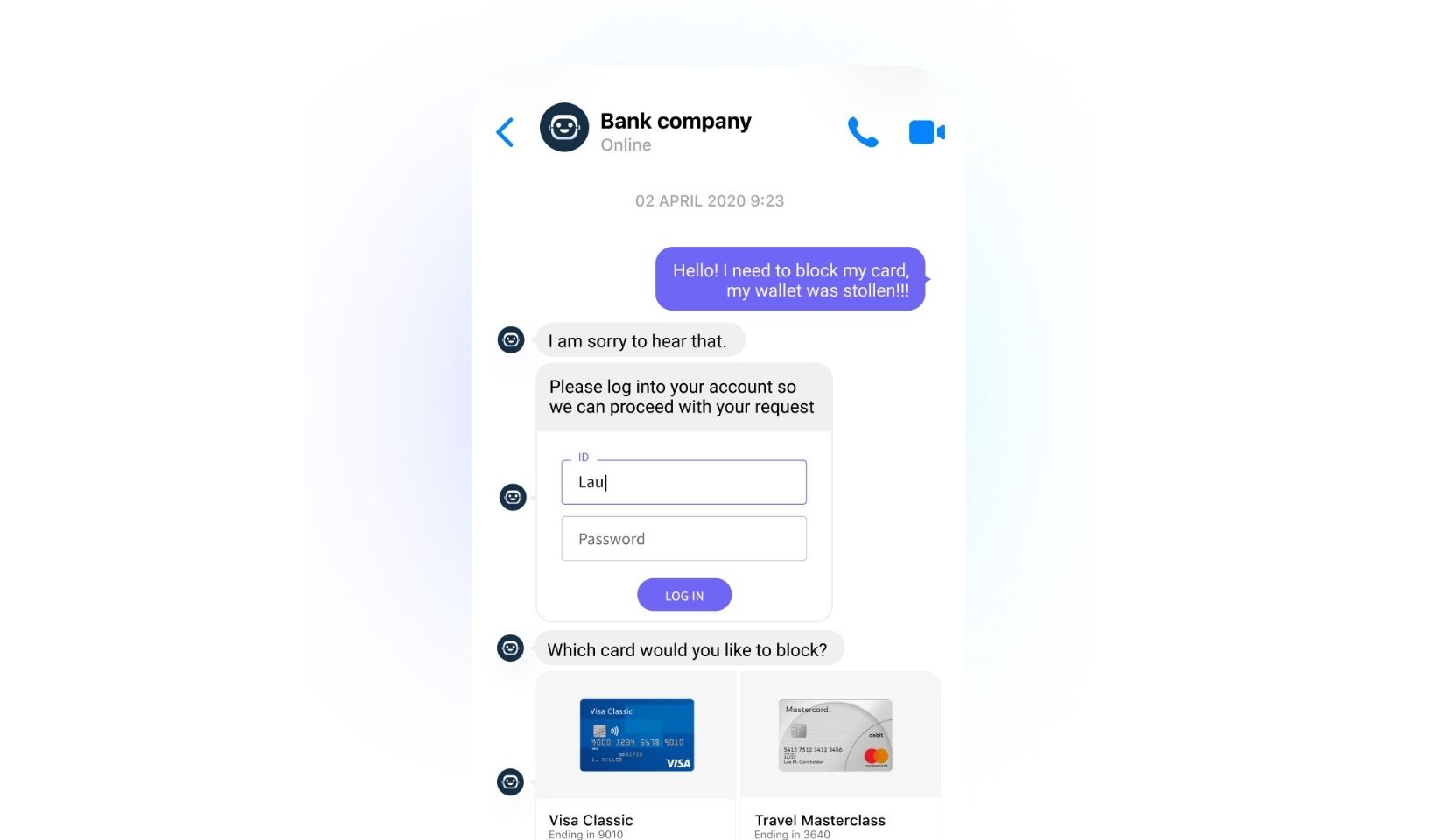
Προ τελευταία κατηγορία, αποτελούν τα Hybrid Chatbots, τα οποία κάνουν χρήση ενός συνδυασμού τεχνολογιών των AI και Rule-based Chatbots, με σκοπό να παρέχουν την καλύτερη δυνατή εξυπηρέτηση των χρηστών. Αυτό που καθιστά σπουδαία τα Hybrid Chatbots είναι ότι επιτρέπουν την ταυτόχρονη χρήση των Chatbots και του Live-Chat. Αυτό σημαίνει ότι μπορούν να προσομοιώνουν συνομιλίες όσο τα ερωτήματα είναι απλά και όταν γίνουν πιο σύνθετα τα αιτήματα, μπορεί το bot να αντικατασταθεί από την ομάδα εξυπηρέτησης και να βοηθήσει γρήγορα έναν πελάτη χωρίς να διακόψει τη ροή της συνομιλίας. Επομένως, πρόκειται για τα πιο πρακτικά Chatbots από αυτά που αναφέρονται στην παρούσα εργασία [25].

Τέλος, αξίζει να αναφέρουμε τον τύπο Generative AI Chatbots, ο οποίος ανήκει στην επόμενη γενιά των Chatbots και μπορεί να προσφέρει πιο βελτιωμένες λειτουργίες αφού διαθέτει ευχέρεια στην κατανόηση της φυσικής γλώσσας και έχει πρόσβαση στα δεδομένα σε πραγματικό χρόνο. Χαρακτηριστικό παράδειγμα αποτελούν τα μοντέλα GPT-3 και GPT-4 τα οποία παρέχουν υψηλή ακρίβεια και φυσικότητα στις απαντήσεις τους [24].

### 2.8.4 Χρησιμότητα των Chatbots

Την εποχή που διανύουμε, τα Chatbots έχουν εισέλθει σε κάθε τομέα της καθημερινότητάς μας, σε κάποιους περισσότερο και σε κάποιους άλλους λιγότερο μέχρι στιγμής. Αυτό έχει ως συνέπεια, να επηρεαστεί σημαντικά ο τρόπος με τον οποίο οι άνθρωποι εξυπηρετούνται και πραγματοποιούν σήμερα τις αγορές τους, τις τραπεζικές τους συναλλαγές, την διαδικασία εκπαίδευσής τους, την ιατρική τους περίθαλψη, την ψυχαγωγία τους καθώς και πολλές άλλες διαδικασίες.

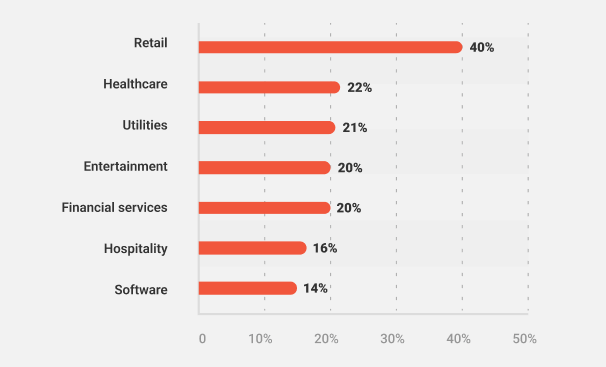
Στον τομέα της υποστήριξης πελατών και του ηλεκτρονικού εμπορίου, τα Chatbots έχουν βελτιώσει σημαντικά την απόδοσή τους, ικανοποιώντας τις προσδοκίες των πελατών και έχοντας κυριολεκτικά πυροδοτήσει την οικονομία του ηλεκτρονικού εμπορίου. Αυτό συμβαίνει επειδή οι έννοιες Chatbot και υποστήριξη πελατών έχουν συνυφαστεί με τη ταχύτητα, την αποτελεσματικότητα και την εύκολη προσβασιμότητα, γεγονός που αναπόφευκτα μεταφράζεται σε διευκόλυνση της καθημερινής ζωής των ανθρώπων [26]. Παράδειγμα χρήσης των Chatbots στην εξυπηρέτηση πελατών, αποτελούν οι τράπεζες. Τα Chatbots που εφαρμόζονται στον χρηματοπιστωτικό κλάδο, βοηθούν τους πελάτες στη διαχείριση οικονομικών συναλλαγών, όπως είναι η διαχείριση ενός λογαριασμού (έλεγχο εσόδων-εξόδων), η αναφορά απώλειας ή κλοπής καρτών, η πραγματοποίηση πληρωμών και μεταφορών, η επιστροφή χρημάτων και πολλές ακόμη υπηρεσίες, που απαλλάσσουν τους πελάτες από τις τεράστιες ουρές αναμονής στα τραπεζικά καταστήματα και τα τηλεφωνικά κέντρα [27].

**Εικόνα 11: Παράδειγμα χρήσης Chatbot από τράπεζα**

Πηγή: <https://www.hubtype.com/blog/banking-chatbots-101>

Επίσης, η πρόοδος των τεχνολογιών της επεξεργασίας και ανάλυσης δεδομένων έχει αλλάξει ριζικά την ιατρική περίθαλψη, προσφέροντας ψηφιακές λύσεις οι οποίες πρόκειται να μεταμορφώσουν ολόκληρη τη διαδικασία αυτής, ώστε να γίνει πιο αποτελεσματική, λιγότερο δαπανηρή και πιο ποιοτική. Τα chatbots υγείας μπορούν να παρέχουν πολλές διαφορετικές υπηρεσίες. Μπορούν για παράδειγμα να παρέχουν, στον χρήστη πληροφορίες σχετικές με την υγεία, να βοηθήσουν στον καθορισμό ραντεβού και να στείλουν υπενθυμίσεις γι' αυτά. Και παρόλο που δεν μπορούν να κάνουν, ακόμα, επίσημες διαγνώσεις, αν πληροφορηθούν τα συμπτώματα, μπορούν να δώσουν μια πιθανή διάγνωση και συμβουλές αντιμετώπισης. Σε ορισμένες περιπτώσεις, τα Chatbots υγείας είναι επίσης σε θέση να συνδέουν τους ασθενείς με κλινικούς γιατρούς για διάγνωση ή θεραπεία [28]. Έτσι, ένα από τα Chatbots υγείας που έχουν σχεδιαστεί είναι ο Endurance, ένα Chatbot που συνομιλεί με χρήστες που πάσχουν από άνοια. Επίσης, το Chatbot Casper έχει υλοποιηθεί για να βοηθά τους ανθρώπους που πάσχουν από αϋπνία λόγω μοναξιάς, να περάσουν τις νύχτες τους, προσφέροντάς τους συντροφιά. Επιπλέον, ο MedWhat είναι ένα Chatbot που απαντά σε συχνές ερωτήσεις υγείας και παρέχει πληροφορίες για διάφορες ασθένειες και τα συμπτώματά τους [29]. Τέλος, ένα ακόμη Chatbot με χρήση στον ιατρικό τομέα είναι το HOLMeS (Health On-Line Medical Suggestions), που έχει σχεδιαστεί για να επικοινωνεί με τον χρήστη, κατανοώντας τη φυσική γλώσσα και ενεργώντας σαν κανονικός γιατρός, κάνοντάς τον να αισθάνεται άνετα και να ζητώντας ευγενικά από τον χρήστη τις απαιτούμενες πληροφορίες (όπως ηλικία, ύψος, βάρος κλπ.), όπως ακριβώς θα γινόταν στην πραγματική ζωή [30].

Ένας ακόμη κλάδος, στον οποίο βρίσκουν εφαρμογή οι διαλογικοί πράκτορες είναι αυτός της εκπαίδευσης, στον οποίο η χρήση τους είναι σε πρωταρχικό στάδιο, ωστόσο υπάρχουν προοπτικές για μελλοντική εξέλιξη, αφού μπορούν να προσφέρουν πολλά οφέλη τόσο στους εκπαιδευόμενους όσο και στους διδάσκοντες. Μερικά από αυτά, είναι η υποστήριξη της συνεχής μάθησης, δηλαδή η δυνατότητα των εκπαιδευόμενων να έχουν πρόσβαση στο εκπαιδευτικό υλικό και να επαναλαμβάνουν την ανάγνωσή του όσες φορές χρειάζεται προκειμένου να γίνει κατανοητό. Επίσης, καθιστά την επίλυση αποριών πιο ευχάριστη αφού συνήθως είναι προτιμότερη η συζήτηση με ένα Chatbot παρά με τους συμμαθητές ή τους εκπαιδευτικούς. Επιπλέον, ενισχύονται τα κίνητρα μάθησης καθώς είναι πιο ελκυστικό και καινοτόμο εργαλείο που προσελκύει την προσοχή εκπαιδευόμενων και εκπαιδευτών. Ακόμη, βελτιώνουν την ικανότητα ανάγνωσης και ακρόασης μέσω διαδραστικών και εξατομικευμένων μεθόδων, καθώς η ικανότητά τους να παρέχουν άμεση ανατροφοδότηση, να προσαρμόζονται στα ατομικά μαθησιακά επίπεδα και να προσφέρουν ευκαιρίες εξάσκησης ανά πάσα στιγμή, τα καθιστά πολύτιμη προσθήκη στις εκπαιδευτικές στρατηγικές [1]. Ήδη, έχει διαπιστωθεί ότι αρχίζουν να ενσωματώνονται σε διοικητικές εργασίες εκπαιδευτικών ιδρυμάτων, ενώ αρχίζει όλο και περισσότερο να χρησιμοποιείται και στην εκπαιδευτική διαδικασία τόσο σαν ένα εργαλείο αξιολόγησης, όσο και για ενίσχυσης της γνώσης. Φυσικά, η δια ζώσης διδασκαλία είναι αναντικατάστατη, παρόλα αυτά, είναι σημαντικό να ενισχύεται με νέα καινοτόμα τεχνολογικά εφόδια, καθώς η τεχνολογία μπορεί να προσφέρει μεγάλη υποστήριξη στην εκπαιδευτική διαδικασία, όπως φάνηκε και στην πρόσφατη εφαρμογή της τηλεκπαίδευσης την περίοδο της πανδημίας Covid-19 [31].

**Εικόνα 12: Αξιοποίηση των Chatbots σε διάφορους κλάδους**

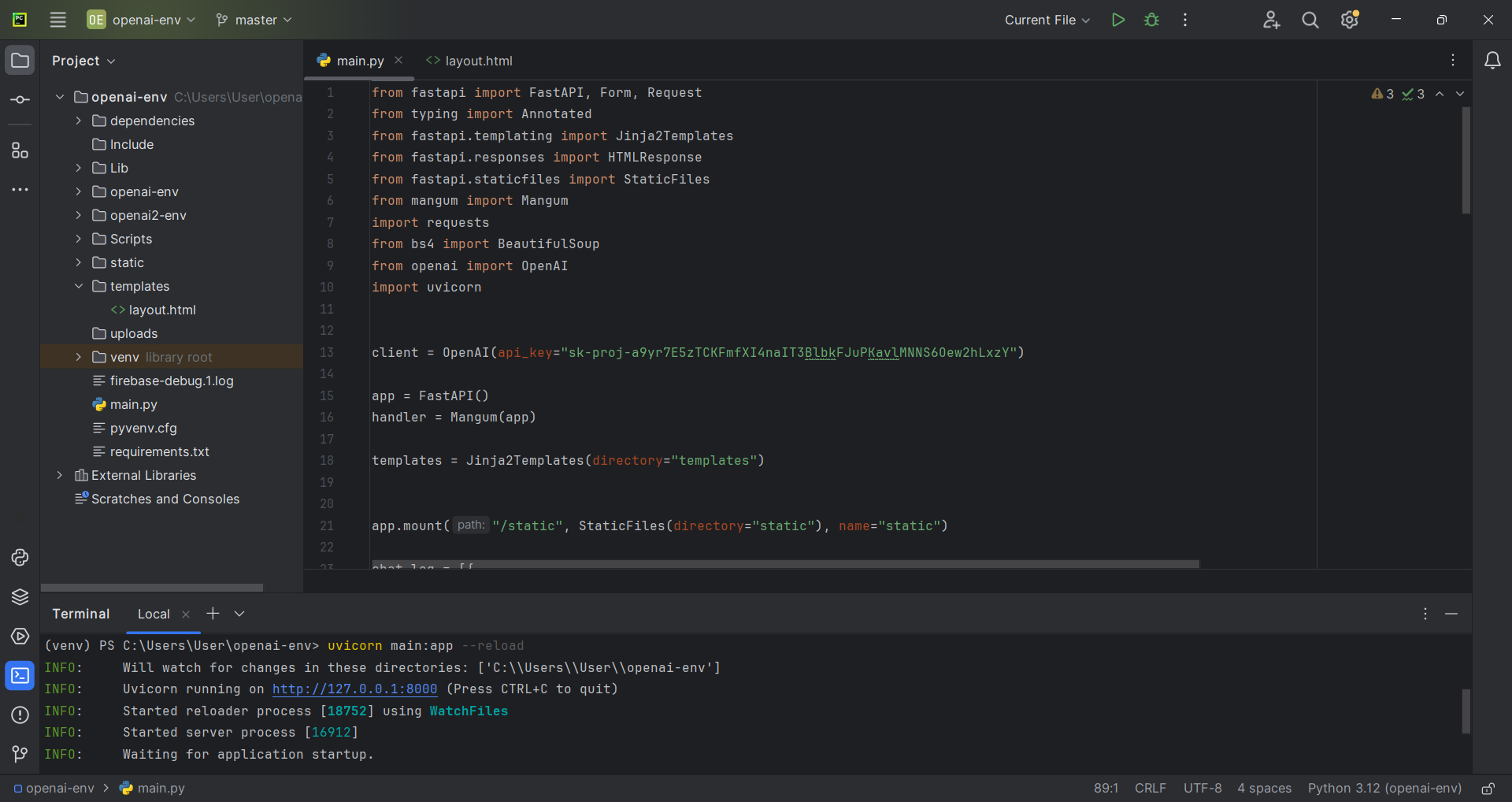
Πηγή: <https://masterofcode.com/blog/chatbot-statistics>

# 3ο Κεφάλαιο: Εργαλεία και Βιβλιοθήκες

## 3.1 PyCharm

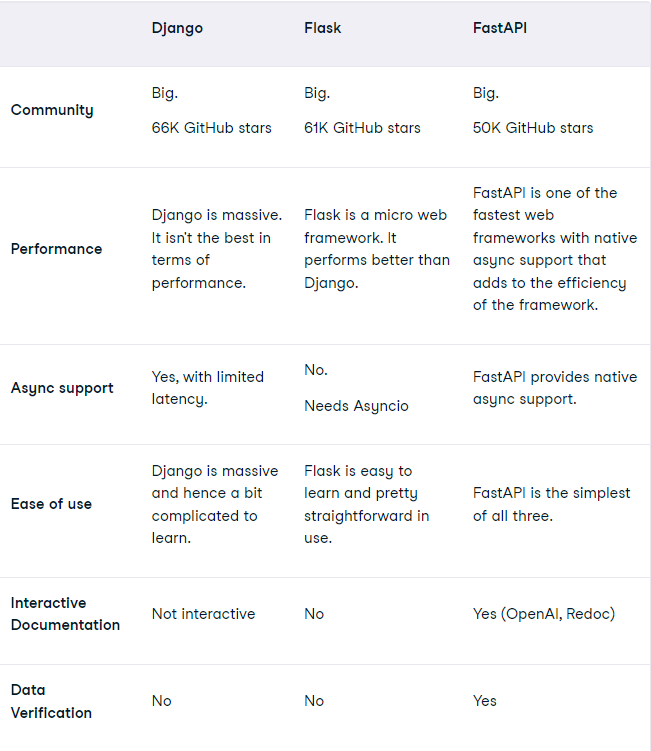
Το PyCharm, είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment, IDE) για την γλώσσα προγραμματισμού Python το οποίο αναπτύχθηκε από την εταιρία JetBrains. Η ανάπτυξη του λογισμικού ξεκίνησε το 2008 και είχε ως στόχο την δημιουργία ενός ισχυρού IDE που να υποστηρίζει την Python, η οποία αποκτούσε όλο και περισσότερη απήχηση. Η πρώτη έκδοση που κυκλοφόρησε το 2010 ήταν η Professional Edition και επρόκειτο για μια επί πληρωμή έκδοση μη οποία διέθετε χαρακτηριστικά που γρήγορα τράβηξαν την προσοχή της προγραμματιστικής κοινότητας. Έπειτα από μερικά χρόνια, το 2013, κυκλοφόρησε η Community Edition, η δωρεάν έκδοση, που κατέστησε το PyCharm προσιτό σε ένα ευρύτερο κοινό καθώς περιλάμβανε βασικά εργαλεία που χρειάζεται ένας προγραμματιστής Python αλλά χωρίς τις προχωρημένες δυνατότητες της Professional Edition. Σήμερα, το PyCharm έχει εξελιχθεί σημαντικά, ενσωματώνοντας υποστήριξη για νέες τεχνολογίες, και είναι ένα από τα πιο δημοφιλή IDE, με εκατομμύρια χρήστες σε όλο τον κόσμο. Συνοψίζοντας, το PyCharm αποτελεί μια άριστη επιλογή για την ανάπτυξη των προγραμμάτων Python, προσφέροντας πολλά πλεονεκτήματα έναντι των υπολοίπων IDEs. Καταρχήν, έχει εξειδικευτεί στη γλώσσα αυτή και διαθέτει ακριβή αυτόματη συμπλήρωση και ανάλυση συντακτικών λαθών, που είναι προσαρμοσμένη στις ιδιαιτερότητες της. Παράλληλα υποστηρίζει επίσης, άλλες γλώσσες προγραμματισμού, π.χ. JavaScript και SQL, παρέχοντας μεγαλύτερη ευελιξία στην ανάπτυξη εφαρμογών που συνδυάζουν ποικιλία τεχνολογιών. Τέλος, το PyCharm υποστηρίζει μεγάλη γκάμα από επεκτάσεις που επεκτείνουν τη λειτουργικότητά του και το καθιστούν προσαρμόσιμο στις συγκεκριμένες ανάγκες του κάθε έργου. Αυτές οι επιμέρους ιδιότητές του, μαζί με την υψηλή απόδοση και σταθερότητα ακόμη και σε μεγάλες εφαρμογές, κάνουν το PyCharm πλήρες, ασφαλές και προσιτό όργανο για κάθε προγραμματιστή.

Για τους παραπάνω λόγους, και αφού είχε προεπιλεχθεί η γλώσσα προγραμματισμού Python για την ανάπτυξη του Chatbot, προτιμήθηκε το PyCharm IDE. Το πρόγραμμα χρησιμοποιεί τεχνολογίες όπως το FastAPI που είναι υπεύθυνο για τη δημιουργία του διαδικτυακού Framework, και κώδικα στις γλώσσες HTML και JavaScript για την διεπαφή χρήστη.

  
**Εικόνα 13: Το project σε περιβάλλον PyCharm**

## 3.2 FastAPI

Το FastAPI είναι ένα σύγχρονο, υψηλής απόδοσης, διαδικτυακό πλαίσιο (Web Framework) για τη δημιουργία APIs με Python, το οποίο κυκλοφόρησε πρώτη φορά το 2018 και δημιουργήθηκε από τον Sebastián Ramírez. Σχεδιάστηκε για να συνδυάζει την υψηλή απόδοση με την ευχρηστία, για να αξιοποιεί πλήρως τις δυνατότητες της Python, όπως είναι οι υποδείξεις τύπων (type hints) και η ασύγχρονη εκτέλεση (async/await). Επίσης, είναι “χτισμένο” πάνω στο Starlette το οποίο λειτουργεί ως πυρήνας παρέχοντας τις βασικές λειτουργίες ενός Web Framework, και το Pydantic το οποίο είναι ένα εργαλείο για την επικύρωση και τη διαχείριση δεδομένων. Το FastAPI είναι επίσης, ένα ισχυρό εργαλείο για την ανάπτυξη εφαρμογών που χρησιμοποιούν μοντέλα τεχνητής νοημοσύνης, συμπεριλαμβανομένων αυτών που παρέχονται από την OpenAI. Για την εκτέλεση αυτών των εφαρμογών, το FastAPI, συνεργάζεται άψογα με τον Uvicorn, έναν ταχύτατο, ασύγχρονο Web Server που υποστηρίζει το πρωτόκολλο ASGI, εξασφαλίζοντας την αποδοτική διαχείριση πολλαπλών αιτημάτων ταυτόχρονα.

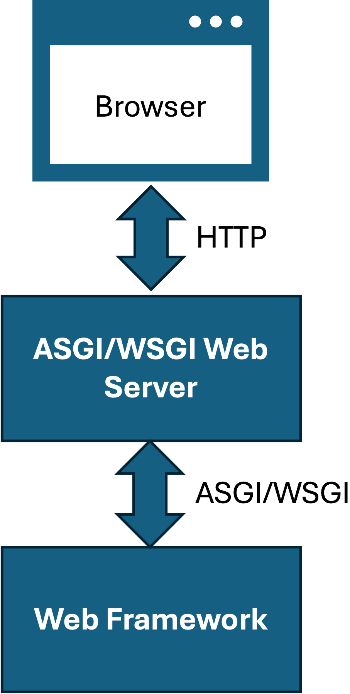
  
**Εικόνα 14: Σύγκριση του FastAPI με άλλα Web Frameworks**

Πηγή: <https://www.datacamp.com/tutorial/introduction-fastapi-tutorial>

Η επιλογή αυτού του Web Framework για την εργασία έγινε λόγω των πλεονεκτημάτων που προσφέρει το FastAPI στις εφαρμογές, όπως περιεγράφηκε παραπάνω. Συγκεκριμένα, για την υλοποίηση του προγράμματος χρησιμοποιήθηκαν οι εξής βιβλιοθήκες: η ‘FastAPI’ για τη δημιουργία της εφαρμογής και τη διαχείριση των HTTP αιτημάτων και των αποκρίσεών της, η 'Form’ για την επεξεργασία των δεδομένων φόρμας που στέλνονται μέσω της μεθόδου POST, η ‘Request’ για την πρόσβαση στις πληροφορίες των αιτημάτων HTTP, επιτρέποντας την ανάγνωση των δεδομένων που στέλνονται από τον χρήστη, η ‘Jinja2Templates’, η οποία προέρχεται από τη βιβλιοθήκη ‘fastapi.templating’, για την απόδοση της HTML σελίδας χρησιμοποιώντας τα δεδομένα της εφαρμογής, η ‘HTMLResponse’ για την επιστροφή του HTML περιεχομένου ως απόκριση σε ένα HTTP αίτημα, και η βιβλιοθήκη ‘StaticFiles’ για την εξυπηρέτηση στατικών αρχείων που εμφανίζονται στην εφαρμογή, όπως CSS, JavaScript και εικόνες. Συγκεκριμένα, από τον καθορισμένο φάκελο ‘static’, ο οποίος περιέχει τον φάκελο ‘images’, εμφανίζεται η εικόνα ‘logo.png’ [32].

## 3.3 Uvicorn

Το Uvicorn, είναι ένας διακομιστής ιστού (web server) που διαχειρίζεται την επικοινωνία μέσω δικτύου, λαμβάνει αιτήματα από τους χρήστες εφαρμογών και στέλνει απαντήσεις σε αυτούς. Το FastAPI δεν περιέχει ενσωματωμένο διακομιστή ανάπτυξης, γι' αυτό χρησιμοποιεί το Uvicorn. Το Uvicorn επικοινωνεί με το FastAPI μέσω της ασύγχρονης διεπαφής πύλης διακομιστή (Asynchronous Server Gateway Interface, ASGI), που είναι ένα πρότυπο API για διακομιστές ιστού Python που εκτελούν ασύγχρονο κώδικα. Αυτό, επιτρέπει στο FastAPI και στον κώδικα της εφαρμογής να επικεντρώνονται στη διαχείριση των HTTP αιτημάτων και στη λογική της εφαρμογής, χωρίς να ασχολούνται με τις λεπτομέρειες χαμηλού επιπέδου και των συνδέσεων δικτύου, καθιστώντας την εκτέλεση της εφαρμογής ταχύτατη [33].

  
**Εικόνα 15: Αρχιτεκτονική Web Εφαρμογών**

Πηγή: <https://tonybaloney.github.io/posts/fine-tuning-wsgi-and-asgi-applications.html>

## 3.4 OpenAI

Η OpenAI, είναι ένα αμερικάνικο ερευνητικό εργαστήριο τεχνητής νοημοσύνης (AI) που ιδρύθηκε το 2015 στο Σαν Φρανσίσκο της Καλιφόρνιας από τους Σαμ Άλτμαν, Ριντ Χόφμαν, Τζέσικα Λίβινγκστον, Ίλον Μασκ, Ιλία Σουτσκέβερ, Πίτερ Τιλ και άλλους. Η αποστολή της είναι η ανάπτυξη "ασφαλούς και ωφέλιμης" Γενικής Τεχνητής Νοημοσύνης ( [Artificial General Intelligence](https://en.wikipedia.org/wiki/Artificial_general_intelligence), AGI). Η OpenAI έχει δημιουργήσει πολλά από τα πιο προηγμένα μοντέλα AI, όπως το GPT. Παράδειγμα τέτοιου μοντέλου αποτελεί το GPT-4 που χρησιμοποιείται για τη δημιουργία κειμένων, απαντήσεων, μεταφράσεων και σε άλλες εργασίες που σχετίζονται με τη γλώσσα. Χρησιμοποιείται επίσης, για την υλοποίηση του παρόντος Chatbot. Το ChatGPT, είναι ένα εργαλείο που παρέχει η OpenAI και επιτρέπει στους χρήστες να αλληλοεπιδρούν με τα GPT μοντέλα μέσω συνομιλιών, παρέχοντας πληροφορίες, καθοδήγηση και βοήθεια στη σύνταξη κειμένων. Ένα ακόμη μοντέλο της OpenAI είναι το Codex, το οποίο είναι εκπαιδευμένο στην κατανόηση και συγγραφή κώδικα και χρησιμοποιείται στο GitHub Copilot για αυτόματη δημιουργία κώδικα και υποστηρίζει πληθώρα γλωσσών προγραμματισμού. Επιπλέον, το μοντέλο DALL·E δημιουργεί εικόνες από περιγραφές σε φυσική γλώσσα, ενώ το Whisper, ένα προηγμένο σύστημα αυτόματης μετατροπής προφορικού λόγου σε κείμενο, υποστηρίζει πολλές γλώσσες και χρησιμοποιείται για μεταγραφές, υποτίτλους και άλλες φωνητικές εφαρμογές. Τέλος, η OpenAI προσφέρει APIs που επιτρέπουν στους προγραμματιστές να ενσωματώνουν τις τεχνολογίες της, παρέχοντας πρόσβαση στα μοντέλα GPT, Codex και άλλα εργαλεία για την ανάπτυξη εξατομικευμένων AI προγραμμάτων [34].

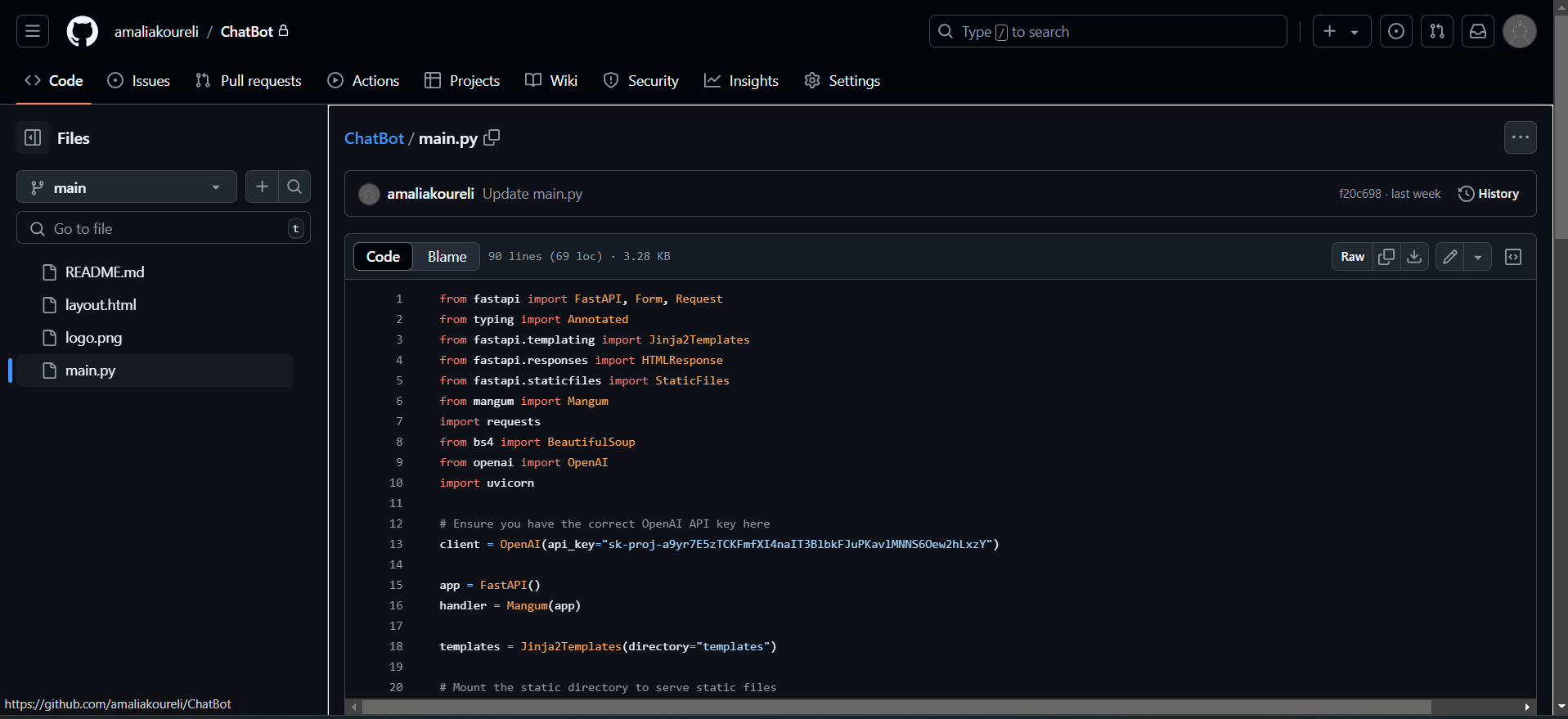
Στο συγκεκριμένο πρόγραμμα, χρησιμοποιείται η βιβλιοθήκη ‘OpenAI’ για την αλληλεπίδραση με το μοντέλο GPT-4o μέσω του OpenAI API. Ο πελάτης (client) αρχικοποιείται με τη μέθοδο ‘OpenAI(api\_key="...")’, η οποία επιτρέπει την επικοινωνία με την πλατφόρμα της OpenAI μέσω του API κλειδιού που παρέχεται από αυτή. Η κύρια μέθοδος που χρησιμοποιείται για την επεξεργασία της συνομιλίας είναι η ‘client.chat.completions.create’, η οποία καλεί το μοντέλο GPT για να δημιουργήσει απαντήσεις στα αιτήματα των πελατών. Στην κλήση αυτή, το πρόγραμμα καθορίζει ποιο μοντέλο θα χρησιμοποιηθεί (στην προκειμένη περίπτωση το "gpt-4o") και στέλνει ως είσοδο τόσο τα μηνύματα του χρήστη όσο και τις πληροφορίες που συλλέχθηκαν μέσω του Web scraping. Η απάντηση αυτή επιστρέφεται από την OpenAI στο πρόγραμμα, το οποίο την προσθέτει στη συνομιλία για να εμφανιστεί στον χρήστη.

## 3.5 GitHub

Το GitHub, είναι μια πλατφόρμα φιλοξενίας και διαχείρισης κώδικα που δημιουργήθηκε το 2008 από τους Tom Preston-Werner, Chris Wanstrath, P.J. Hyett και Scott Chacon. Το GitHub, είναι βασισμένο στο σύστημα ελέγχου εκδόσεων (Version Control) Git και επιτρέπει στους προγραμματιστές να παρακολουθούν τις αλλαγές στον κώδικα και να συνεργάζονται αποτελεσματικά, είτε εργάζονται ατομικά είτε σε ομάδες. Πρόκειται για μια cloud-based υπηρεσία, η οποία προσφέρει πληθώρα εργαλείων για τη διαχείριση του κώδικα, την αποθήκευση και την κοινοποίηση project, καθιστώντας το, ιδανικό για τη φιλοξενία τόσο ανοιχτού όσο και ιδιωτικού κώδικα.

Στο GitHub, οι κώδικες οργανώνονται σε repositories (αποθετήρια), τα οποία μπορούν να είναι δημόσια ή ιδιωτικά, ανάλογα με το αν οι χρήστες επιθυμούν να μοιραστούν το έργο τους με άλλους. Η πλατφόρμα, παρέχει το εργαλείο pull requests, όπου οι προγραμματιστές μπορούν να προτείνουν αλλαγές, επιτρέποντας σε άλλους να αναθεωρήσουν τον κώδικα, να σχολιάσουν, καθώς και να εγκρίνουν τις αλλαγές πριν αυτές ενσωματωθούν στο κυρίως έργο. Ένα ακόμη εργαλείο, είναι το fork, που δίνει τη δυνατότητα στους χρήστες να δημιουργούν αντίγραφα ενός project στο δικό τους αποθετήριο για πειραματισμό ή βελτίωση χωρίς να επηρεάζουν το πρωτότυπο. Ένα σημαντικό χαρακτηριστικό, αποτελεί το issues σύστημα, όπου οι χρήστες μπορούν να αναφέρουν σφάλματα, να προτείνουν βελτιώσεις και να συζητούν τις ιδέες τους. Το GitHub, δεν είναι μόνο μια τεχνολογική πλατφόρμα, αλλά και ένα κοινωνικό δίκτυο για προγραμματιστές, αφού δίνει στους χρήστες τη δυνατότητα να προωθούν τη δουλειά τους, δημιουργώντας portfolios με τα έργα τους, τα οποία είναι ορατά σε παγκόσμια κλίμακα. Έτσι, διευκολύνεται η διαδικασία προβολής των δεξιοτήτων τους και η επικοινωνία τους με άλλους επαγγελματίες, ανοίγοντας τον δρόμο για συνεργασίες και νέες εργασιακές ευκαιρίες. Ως εκ τούτου, το GitHub αποτελεί όχι μόνο ένα ισχυρό εργαλείο ανάπτυξης λογισμικού, αλλά και έναν κόμβο για τη κοινωνική διασύνδεση και την επαγγελματική εξέλιξη των προγραμματιστών. Τον Ιανουάριο του 2023, το GitHub ξεπέρασε τους 100 εκατομμύρια χρήστες ενώ τα repositories έφτασαν τα 370 εκατομμύρια, καθιστώντας το, αν όχι την μεγαλύτερη, μια από τις μεγαλύτερες πλατφόρμες για την ανάπτυξη λογισμικού και την ανοιχτή συνεργασία.

Για την παρούσα εργασία, η πλατφόρμα του GitHub χρησιμοποιήθηκε για την εναπόθεση όλων των αρχείων της εφαρμογής προκειμένου να είναι ευκολότερη η διαχείριση τους από οποιοδήποτε απομακρυσμένο περιβάλλον και για την διαχείριση των αλλαγών που υπήρξαν στον κώδικα με την πάροδο του χρόνου [35].

**** **Εικόνα 16: Το Project σε περιβάλλον GitHub**

# 4ο Κεφάλαιο: Περιγραφή Λογισμικού

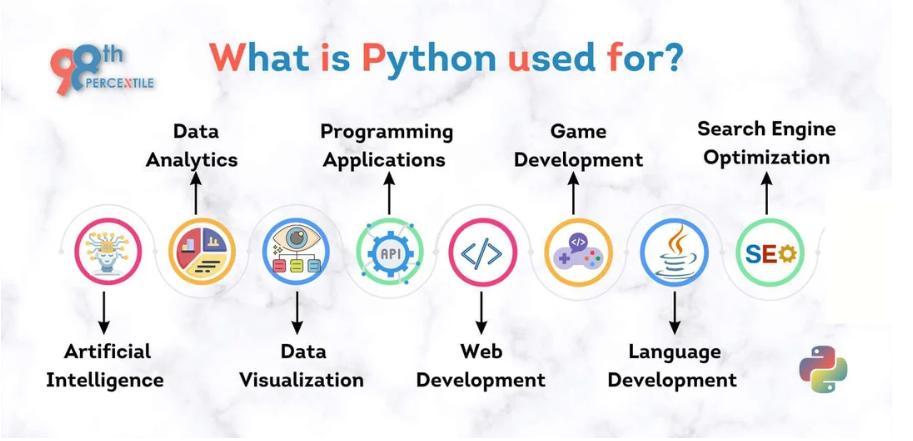
## 4.1 Γλώσσες προγραμματισμού

Για την υλοποίηση της παρούσας εργασίας, χρησιμοποιήθηκαν τέσσερις διαφορετικές γλώσσες προγραμματισμού όπου η καθεμιά εκτελεί ένα συγκεκριμένο σκοπό. Ο συνδυασμός διαφορετικών γλωσσών προγραμματισμού σε πολυσύνθετα προγράμματα, όπως αυτό, είναι σημαντικός, καθώς επιτρέπει την εξειδίκευση κάθε γλώσσας στο ρόλο της, προσφέροντας βελτιστοποίηση στην αποδοτικότητα. Επίσης, ο διαχωρισμός του Front-end από το Back-end διευκολύνει τη συντήρηση και την επέκταση της εφαρμογής, ενώ η χρήση εξειδικευμένων βιβλιοθηκών επιτρέπει τη γρήγορη ανάπτυξη με ασφάλεια και ευελιξία. Επιπλέον, ο συνδυασμός γλωσσών προσφέρει καλύτερη κλιμάκωση, καθώς κάθε τμήμα της εφαρμογής μπορεί να βελτιστοποιηθεί ανεξάρτητα, διασφαλίζοντας καλύτερη εμπειρία χρήστη και ασφάλεια στο σύστημα.

### 4.1.1 Python

Η Python είναι μια διερμηνευμένη, αντικειμενοστραφής γλώσσα προγραμματισμού υψηλού επιπέδου, με δυναμική σημασιολογία. Δημιουργήθηκε τη δεκαετία του 1980 από τον Guido van Rossum στην Ολλανδία. Η πρώτη έκδοση που κυκλοφόρησε το 1991, είχε ελάχιστους ενσωματωμένους τύπους δεδομένων και παρείχε βασική λειτουργικότητα. Ενώ, η πιο πρόσφατη έκδοση της Python είναι η 3.13, η οποία κυκλοφόρησε τον Οκτώβριο του 2024 και περιλαμβάνει βελτιώσεις απόδοσης, διορθώσεις σφαλμάτων, νέα χαρακτηριστικά και βελτιώσεις στην εξσφαλμάτωση.

Πρόκειται για μια από τις πιο εύκολες αλλά και πιο χρήσιμες γλώσσες προγραμματισμού, καθώς έχει απλή και καθαρή σύνταξη, η οποία μοιάζει αρκετά με την ανθρώπινη γλώσσα και αυτό καθιστά την εκμάθησή της πιο προσιτή και ευχάριστη στους αρχάριους προγραμματιστές. Επιπλέον, διαθέτει ευρεία γκάμα από βιβλιοθήκες και frameworks για διάφορες εφαρμογές. Παραδείγματα framework, αποτελούν το FastAPI και το Flask για την ανάπτυξη ιστοσελίδων, το NumPy, το Pandas και το TensorFlow για την ανάλυση δεδομένων και τη Μηχανική Μάθηση, το Keras και το PyTorch για εφαρμογές της Τεχνητής Νοημοσύνης κ.α. Μερικές από τις βιβλιοθήκες της χρησιμοποιούνται και στο συγκεκριμένο project, όπως η FastAPI χρησιμοποιείται για τη δημιουργία του web server και τη διαχείριση των HTTP αιτημάτων και απαντήσεων, επιτρέποντας τη γρήγορη και αποτελεσματική δημιουργία APIs. Για την κατασκευή δυναμικών HTML σελίδων μέσω του template rendering, που είναι ενσωματωμένο με το FastAPI, χρησιμοποιείται η Jinja2. Η βιβλιοθήκη Requests χρησιμοποιείται για να εξυπηρετεί την αποστολή HTTP αιτημάτων και την ανάκτηση δεδομένων από εξωτερικές ιστοσελίδες. Επιπροσθέτως, η βιβλιοθήκη BeautifulSoup χρησιμοποιείται για την ανάλυση και εξαγωγή δεδομένων από HTML. Ακόμη, για την αλληλεπίδραση με το API του OpenAI και τη δημιουργία των απαντήσεων του Chatbot, χρησιμοποιείται η βιβλιοθήκη openai. Τέλος, ο Uvicorn, ως web server, τρέχει το FastAPI app. Η Python, είναι διαλειτουργική με άλλες γλώσσες προγραμματισμού όπως η C/C++ και η Java, καθώς και με πλατφόρμες, καθιστώντας τη κατάλληλη για πολυδιάστατα έργα. Έχει ακόμη, την δυνατότητα να λειτουργεί σε διάφορα λειτουργικά συστήματα (Windows, macOS και Linux), επιτρέποντας να αναπτύσσονται εφαρμογές που λειτουργούν σε οποιοδήποτε λογισμικό ανεμπόδιστα. Τέλος, διαθέτει μια τεράστια και ενεργή κοινότητα προγραμματιστών που προσφέρει υποστήριξη, με tutorials και λύσεις για σχεδόν κάθε πρόβλημα που μπορεί να προκύψει, ενισχύοντας περαιτέρω τη χρηστική αξία της [36], [37].

  
**Εικόνα 17: Παραδείγματα χρήσης της Python**

Πηγή: <https://www.98thpercentile.com/blog/what-is-python-used-for/>

### 4.1.2 HTML

Η HTML (HyperText Markup Language), είναι η θεμελιώδης γλώσσα για τη δημιουργία και τον σχεδιασμό των ιστοσελίδων και διαδικτυακών εφαρμογών, η οποία χρησιμοποιείται από δισεκατομμύρια προγραμματιστές καθημερινά. Παρουσιάστηκε το 1991, στο CERN, ως μια απλή γλώσσα σήμανσης και έκτοτε, έχει εξελιχθεί, μέσω εκδόσεων, μέχρι την HTML5 που κυκλοφόρησε το 2014, η οποία από τότε ανανεώνεται με βελτιώσεις και νέες δυνατότητες.

Για να πετύχει τον σκοπό της, χρησιμοποιεί ένα σύνολο ετικετών (tags), που αποτελούν τα δομικά στοιχεία των ιστοσελίδων. Έχουν δημιουργηθεί κατάλληλες ετικέτες για οποιαδήποτε επιθυμητή λειτουργεία των ιστοσελίδων. Για παράδειγμα, μέσω αυτών, η HTML μπορεί να οργανώσει το περιεχόμενο σε δομικά τμήματα, επιτρέποντας τη δημιουργία επικεφαλίδων ( <h1>, <h2>...), παραγράφων (<p>) και λιστών (<ul>, <οl>). Επίσης, επιτρέπει την ενσωμάτωση πολυμέσων με ετικέτες όπως <img> για εικόνες, <video> για βίντεο, και <audio> για αρχεία ήχου, κάνοντας πιο διαδραστική την εκάστοτε ιστοσελίδα και συνεπώς πιο φιλική προς τον χρήστη. Και για τη δημιουργία υπερσυνδέσμων, χρησιμοποιείται η ετικέτα <a>, επιτρέποντας την πλοήγηση μεταξύ σελίδων και εξωτερικών πόρων.

Η HTML, παράλληλα, συνεργάζεται με βιβλιοθήκες template rendering, όπως η Jinja2, για τη δημιουργία δυναμικών HTML σελίδων, βελτιώνοντας τη διαχείριση περιεχομένου. Πιο συγκεκριμένα, στην παρούσα εργασία, το FastAPI χρησιμοποιεί το Jinja2 για να προωθήσει HTML templates στους χρήστες και να εμφανίσει το δυναμικό περιεχόμενο. Τέλος, έχει την δυνατότητα να συνεργάζεται με άλλες γλώσσες όπως η CSS και η JavaScript που χρησιμοποιούνται αντίστοιχα για την πλήρη μορφοποίηση και διαδραστικότητα των ιστοσελίδων, αποτελώντας τη βάση για τη σύγχρονη ανάπτυξη ιστότοπων.

### 4.1.3 CSS

Η CSS (*Cascading Style Sheets* – διαδοχικά φύλλα στυλ), είναι μια γλώσσα που αναπτύχθηκε από τους Håkon Wium Lie και Bert Bos, κυκλοφόρησε το 1996 μέσω του W3C (World Wide Web Consortium) και χρησιμοποιείται για την περιγραφή της εμφάνισης και του σχεδιασμού των ιστοσελίδων. Χρησιμοποιείται σε συνδυασμό με την HTML για να καθορίσει πώς πρέπει να εμφανίζονται τα στοιχεία στον ιστότοπου. Γενικά, η CSS είναι μια γλώσσα υπολογιστή, απαραίτητη για την ανάπτυξη ιστοσελίδων, η οποία προσφέρει τεράστιες δυνατότητες για την προσαρμογή της εμφάνισης και της συμπεριφοράς των στοιχείων σε μια ιστοσελίδα. Έχει τη δυνατότητα να διαμορφώνει χαρακτηριστικά, όπως τα χρώματα, η στοίχιση, οι γραμματοσειρές και πολλά άλλα, βελτιώνοντας την εμφάνιση και την αίσθηση ενός ιστότοπου εφαρμόζοντας στυλ σε στοιχεία HTML.

Χρησιμοποιώντας CSS μειώνεται η ποσότητα του κώδικα στα αρχεία HTML, απαλλάσσοντας από τις περιττές επαναλήψεις εντολών, διευκολύνοντας τη συντήρηση και την ενημέρωσή του. Επιπλέον, βοηθά στο σχεδιασμό ιστότοπων που λειτουργούν καλά σε διάφορες συσκευές και μεγέθη οθόνης, παρέχοντας μεγαλύτερη ευελιξία. Επιτρέπει στους προγραμματιστές να διαχωρίζουν το στυλ από το περιεχόμενο και τη δομή της εφαρμογής ιστού, γεγονός που διευκολύνει την ανάγνωση και την συντήρηση του κώδικα.

Η CSS, μπορεί να εφαρμοστεί στα HTML αρχεία με τρεις διαφορετικούς τρόπους. Αρχικά υπάρχει η επιλογή inline, το οποίο εφαρμόζεται απευθείας σε ένα στοιχείο μέσα στην HTML. Αυτή η μέθοδος, είναι ιδανική για μικρές αλλαγές σε ένα συγκεκριμένο στοιχείο και για μοναδικές περιπτώσεις αφού μπορεί εύκολα και γρήγορα να συντηρηθεί. Ωστόσο, δεν προτιμάται σε περιπτώσεις που μπορεί να χρειαστούν αλλαγές σε πολλά στοιχεία γιατί είναι χρονοβόρο. Επίσης, η CSS μπορεί να χρησιμοποιηθεί μέσα στο HTML αρχείο με την μέθοδο internal (εσωτερικά), κατά την οποία τοποθετείται μέσα στο <style> tag στην κεφαλίδα (head) του αρχείου. Όλα τα στυλ που καθορίζονται σε αυτό το τμήμα ισχύουν για ολόκληρο το συγκεκριμένο αρχείο. Με αυτόν το τρόπο, παρέχεται καλύτερη οργάνωση από το inline CSS, αφού όλα τα στυλ για το συγκεκριμένο HTML αρχείο είναι συγκεντρωμένα σε ένα μέρος. Επίσης, είναι χρήσιμο αν πρόκειται για την διαχείριση μόνο μίας σελίδας, διαφορετικά, αν πρέπει να διαχειριστούν πολλές σελίδες στον ιστότοπο, θα πρέπει να επαναληφθεί το ίδιο στυλ σε κάθε σελίδα, και έτσι θα αυξηθεί ο πλεονασμός και η δυσκολία συντήρησης της ιστοσελίδας. Τελευταία μέθοδος, με την οποία μπορεί να εφαρμοστεί η CSS στον HTML κώδικα είναι χρησιμοποιώντας ένα εξωτερικό (external) αρχείο.css αποκλειστικά για τον CSS κώδικα, το οποίο συνδέεται με το HTML αρχείο, μέσω του <link> tag στην κεφαλίδα. Με αυτή την μέθοδο, χωρίζεται πλήρως το περιεχόμενο από το στυλ, κάνοντας τον κώδικα πιο εύκολο στη συντήρηση και επιτρέπει την επαναχρησιμοποίησή του σε πολλές σελίδες, μειώνοντας τις επαναλήψεις [38].

  
**Εικόνα 18: Παράδειγμα χρήσης CSS κώδικα**

Πηγή: <https://www.bitdegree.org/learn/inline-css>

### 4.1.4 JavaScript

Η JavaScript, είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου που δημιουργήθηκε το 1995 από τον Brendan Eich. Χρησιμοποιείται για την προσθήκη διαδραστικότητας και λειτουργικότητας στις ιστοσελίδες, ενισχύοντας τις δυνατότητες που προσφέρει η HTML. Είναι μια διερμηνευόμενη γλώσσα που εκτελείται στην πλευρά του πελάτη (client-side) από το πρόγραμμα περιήγησης και επιτρέπει τη δυναμική αλλαγή του περιεχομένου της ιστοσελίδας χωρίς την ανάγκη ανανέωσης της σελίδας.

Επιπλέον, η JavaScript διευκολύνει την επικοινωνία με διακομιστές (servers) μέσω ασύγχρονων αιτήσεων. Έχουν, ακόμη, δημιουργηθεί πολλά δημοφιλή frameworks και βιβλιοθήκες που βασίζονται στην JavaScript. Το React, το οποίο αναπτύχθηκε από το Facebook, χρησιμοποιείται ευρέως για την κατασκευή διεπαφών χρήστη και single-page εφαρμογών. Το Angular, που δημιουργήθηκε από την Google, είναι ένα ισχυρό framework για τη δημιουργία σύνθετων εφαρμογών ιστού. Στην πλευρά του διακομιστή, το Node.js επιτρέπει την ανάπτυξη εφαρμογών διακομιστή (server-side) με JavaScript, κάνοντάς την κατάλληλη για την ανάπτυξη ολοκληρωμένων εφαρμογών ιστού[39].

Ο κώδικας JavaScript μπορεί να ενσωματωθεί σε έγγραφα HTML χρησιμοποιώντας την ετικέτα <script>, διαφορετικά μπορεί να γραφτεί σε ένα ξεχωριστό αρχείο και να συνδεθεί με αυτό. Η άμεση ενσωμάτωση της JavaScript στο HTML είναι χρήσιμη για μικρό κώδικα που σχετίζεται άμεσα με το περιεχόμενο της συγκεκριμένης σελίδας. Ενώ, η σύνδεση με ένα εξωτερικό αρχείο JavaScript, το οποίο έχει κατάληξη .js, γίνεται μέσω της ετικέτας <script src="path/to/file.js"></script>. Αυτή η μέθοδος επιτρέπει τη διαχείριση μεγαλύτερων ή επαναχρησιμοποιούμενων τμημάτων κώδικα για καλύτερη οργάνωση, διευκόλυνση της ανάπτυξης και συντήρηση μεγάλων εφαρμογών ιστού.

Η JavaScript, παραμένει στην αιχμή των τεχνολογικών εξελίξεων, ως βασικό εργαλείο για τη δημιουργία δυναμικών και διαδραστικών διαδικτυακών εμπειριών. Στη συγκεκριμένη εργασία, η JavaScript έχει σημαντικό ρόλο στη διαχείριση της διεπαφής χρήστη και της δυναμικής αλληλεπίδρασης με το chatbot. Παρόλο που το πρόγραμμα βασίζεται κυρίως στην Python για τη λογική του server-side και την αλληλεπίδραση με APIs (όπως το OpenAI και το Web Scraping), η JavaScript είναι απαραίτητη για την υποβολή των φορμών και την άμεση απόκριση της εφαρμογής χωρίς να απαιτείται ανανέωση της σελίδας. Επομένως, χρησιμοποιώντας JavaScript, οι εισαγωγές των δεδομένων του χρήστη αποστέλλονται ασύγχρονα στο backend, βελτιώνοντας την εμπειρία του χρήστη με την ταχύτητα και την απόδοση της εφαρμογής.

## 4.2 Αρχιτεκτονική συστήματος

Σε αυτή την ενότητα, περιγράφεται αναλυτικά ο τρόπος με τον οποίο σχεδιάστηκε η υλοποίηση του chatbot στα πλαίσια της παρούσας εργασίας, δηλαδή η αρχιτεκτονική του. Συγκεκριμένα, θα αναλύσουμε τις προδιαγραφές του συστήματος όπως είναι οι χρήστες, οι λειτουργικές και οι μη απαιτήσεις του. Επίσης, θα παρουσιαστούν διαγράμματα τα οποία θα βοηθήσουν στην πλήρη κατανοητή της λειτουργίας και της συνδεσιμότητας του Chatbot. Για την δημιουργία των διαγραμμάτων χρησιμοποιήθηκε η εφαρμογή yEdGraph Editor, η οποία δίνει την δυνατότητα στους χρήστες να σχεδιάζουν διάφορων ειδών διαγράμματα, παρέχοντας βασικά εργαλεία όπως σχήματα, συνδέσεις, συσχετίσεις κ.α.

### 4.2.1 Χρήστες

Ένα Chatbot, για μπορέσει να λειτουργήσει χρειάζεται απαραιτήτως χρήστες με τους οποίους θα μπορεί να αλληλοεπιδρά. Για να γίνει σωστός σχεδιασμός του συστήματος που θέλουμε να δημιουργήσουμε, είναι σημαντικό να έχουμε μια ξεκάθαρη εικόνα των χρηστών που θα το χρησιμοποιούν. Ποιες είναι οι ανάγκες που έχουν, ποιες είναι οι απαιτήσεις τους από το σύστημα αυτό καθώς και ποιες είναι οι δράσεις που θα μπορούν οι ίδιοι να πραγματοποιήσουν κατά τη χρήση του.

Το Chatbot, απευθύνεται σε χρήστες της ιστοσελίδας του τμήματος Πληροφορικής του Δημοκρίτειου Πανεπιστήμιου Θράκης στους οποίους μπορεί να χρειαστεί κάποια διευκρίνηση, οδηγία ή πληροφορία σχετικά με το τμήμα και την φοίτηση σε αυτό. Ειδικότερα, οι χρήστες μπορεί να είναι φοιτητές, καθηγητές ή ακόμα και «περαστικοί» που χρησιμοποιώντας την ιστοσελίδα τους δημιουργήθηκε κάποιο ερώτημα. Οι χρήστες τότε, θα έχουν την δυνατότητα να απευθυνθούν στον ακαδημαϊκό βοηθό, του οποίου η ιστοσελίδα είναι συνδεδεμένη με τον πανεπιστημιακό ιστότοπο, για να λύσουν τις απορίες τους γρήγορα, εύκολα και αποτελεσματικά.

### 4.2.2 Λειτουργικές Απαιτήσεις

Για τον σκοπός υλοποίησης του συγκεκριμένου λογισμικού, δημιουργήθηκαν μερικές λειτουργικές απαιτήσεις οι οποίες καθορίζουν τις υπηρεσίες που πρέπει να προσφέρει το σύστημα, καθώς και τον τρόπο με τον οποίο θα αντιδρά αυτό στις ενέργειες των χρηστών. Έτσι, ορίζονται οι παρακάτω λειτουργικές απαιτήσεις:

* Λ.Α.1: Ο χρήστης-φοιτητής πρέπει να έχει πρόσβαση στο Chatbot ανά πάσα ώρα και στιγμή, μέσω των συνομιλιών στην ιστοσελίδα του.
* Λ.Α.2: Το chatbot πρέπει να μπορεί να κατανοεί και να επεξεργάζεται την ανθρώπινη γλώσσα, τόσο σε γραπτή όσο και σε προφορική μορφή.
* Λ.Α.3: Το σύστημα πρέπει να σαρώνει την ιστοσελίδα της σχολής σε κάθε αίτημα των χρηστών-φοιτητών με τη χρήση API κλειδιού για την ανάκτηση πληροφοριών.
* Λ.Α.4: Το chatbot πρέπει να παρέχει πρόσβαση σε δεδομένα πραγματικού χρόνου.
* Λ.Α.5: Το chatbot πρέπει να απαντάει στις ερωτήσεις των χρηστών-φοιτητών γρήγορα και στοχευμένα.
* Λ.Α.6: Το chatbot πρέπει να εμφανίζει το ιστορικό της κάθε συζήτησης.
* Λ.Α.7: Το σύστημα πρέπει να μπορεί να υποστηρίξει μεγάλο όγκο χρηστών και πληροφοριών, χωρίς να δημιουργείται κάποιο πρόβλημα στη λειτουργία του.
* Λ.Α.8: Το σύστημα πρέπει να μπορεί να διαχειριστεί πολλές ταυτόχρονες συνομιλίες.

### 4.2.3 Μη Λειτουργικές Απαιτήσεις

Οι μη λειτουργικές απαιτήσεις ενός chatbot, είναι αυτές που ορίζουν όλες τις ιδιότητες και τους περιορισμούς του συστήματος, που σχετίζονται με την ποιότητα, την απόδοση και την εμπειρία που παρέχει. Παρακάτω αναφέρονται οι κύριες μη λειτουργικές απαιτήσεις του Chatbot:

* + Μ.Λ.Α.1:Η διεπαφή του Chatbot πρέπει να είναι φιλική προς τον χρήστη και να παρέχει ευχάριστη εμπειρία αλληλεπίδρασης.
  + Μ.Λ.Α.2:Το Chatbot πρέπει να λειτουργεί σωστά σε διαφορετικές πλατφόρμες και να είναι συμβατό με διάφορα λειτουργικά συστήματα και περιηγητές.
  + Μ.Λ.Α.3:Η συντήρηση και η αναβάθμιση του συστήματος πρέπει να είναι εύκολες και χωρίς μεγάλες διακοπές.
  + Μ.Λ.Α.4:Το Chatbot πρέπει να παραμένει σε λειτουργία, ακόμη και σε περιόδους υψηλής ζήτησης ή σε περιπτώσεις σφαλμάτων.
  + Μ.Λ.Α.5:Το σύστημα πρέπει να μπορεί να επικοινωνεί με εξωτερικά APIs.
  + Μ.Λ.Α.6:Το Chatbot πρέπει να είναι διαθέσιμο online όλη την ημέρα.
  + Μ.Λ.Α.7:Το σύστημα θα πρέπει να παρέχει προστασία των προσωπικών δεδομένων του χρήστη τηρώντας τον Γενικό Κανονισμό Προστασίας Προσωπικών Δεδομένων (GDPR).
  + Μ.Λ.Α.8:Το chatbot πρέπει να παρέχει όλες τις απαραίτητες και ζητούμενες πληροφορίες για το συγκεκριμένο Πανεπιστημιακό τμήμα στους χρήστες-φοιτητές.
  + Μ.Λ.Α.9:Το σύστημα θα πρέπει να αποφορτίσει τον όγκο εργασίας που έχει η γραμματεία και το υποστηρικτικό προσωπικό της σχολής για την ενημέρωση των φοιτητών.
  + Μ.Λ.Α.10:Το σύστημα θα πρέπει να βελτιώσει την ποιότητα παροχής των υπηρεσιών που λαμβάνουν οι χρήστες-φοιτητές από το Πανεπιστήμια στα θέματα ενημέρωσης.
  + Μ.Λ.Α.11:Το chatbot πρέπει να παρέχει κατάλληλες πληροφορίες στους ενδιαφερόμενους υποψήφιους χρήστες-φοιτητές σχετικά με το τμήμα που θέλουν να εισαχθούν.

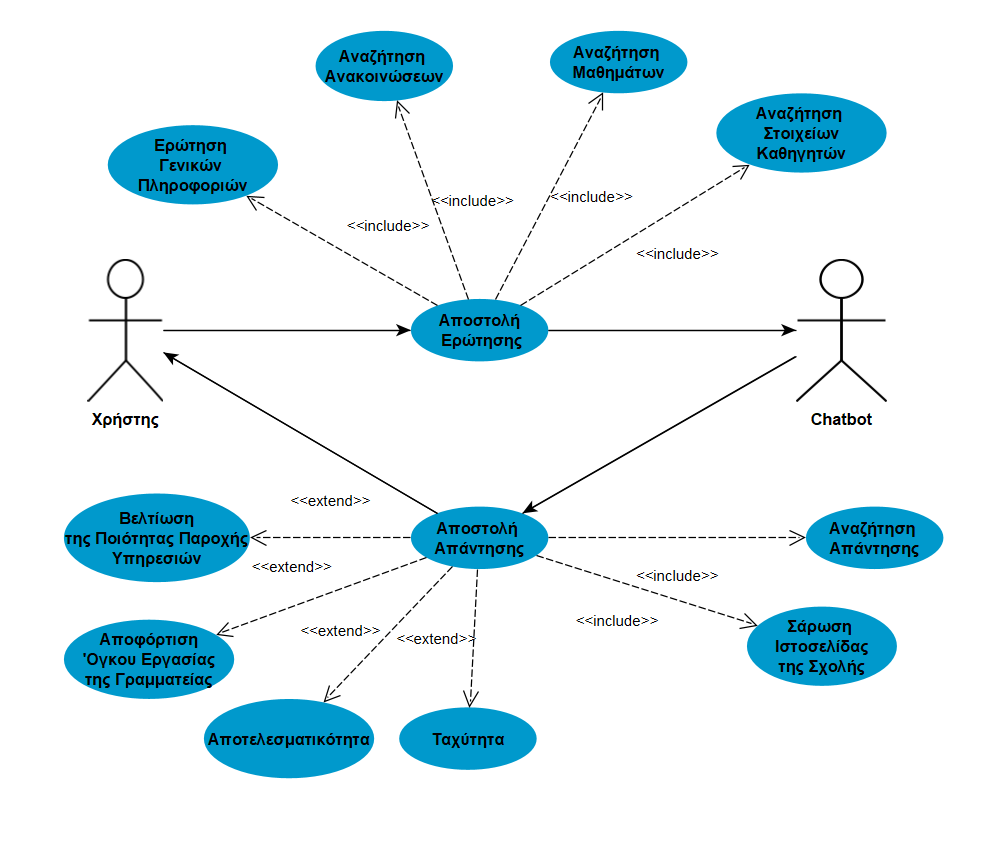
## 4.3 Διαγράμματα Συστήματος

Σε αυτή την ενότητα, θα παρουσιαστούν και θα αναλυθούν όλα τα απαραίτητα διαγράμματα για το σύστημα, προκειμένου να γίνει πιο εύκολα κατανοητή η υλοποίηση και οι μέθοδοι που χρησιμοποιήθηκε. Για την δημιουργία των διαγραμμάτων αξιοποιήθηκε η εφαρμογή yEd Graph Editor που μπορεί να χρησιμοποιηθεί για τη γρήγορη και αποτελεσματική δημιουργία διαγραμμάτων υψηλής ποιότητας. Το yEd διατίθεται για όλα τα λογισμικά συστήματα: Windows, Unix/Linux και macOS.

### 4.3.1 UML Διάγραμμα Περιπτώσεων Χρήσης (Use Case Diagram)

Τα διαγράμματα περιπτώσεων χρήσης, ανήκουν στα διαγράμματα συμπεριφοράς της UML και είναι ιδιαίτερα χρήσιμα για την κατανόηση των απαιτήσεων του συστήματος. Δείχνουν πώς αλληλεπιδρούν οι χρήστες με το σύστημα, παρουσιάζοντας τη λειτουργικότητά του. Αυτός ο τύπος διαγράμματος, είναι θεμελιώδης στη γλώσσα UML και μας δείχνει ποιες ενέργειες μπορούν να εκτελέσουν οι χρήστες, χωρίς να μπαίνει σε λεπτομέρειες για το πώς υλοποιούνται αυτές οι λειτουργίες στο εσωτερικό του συστήματος.

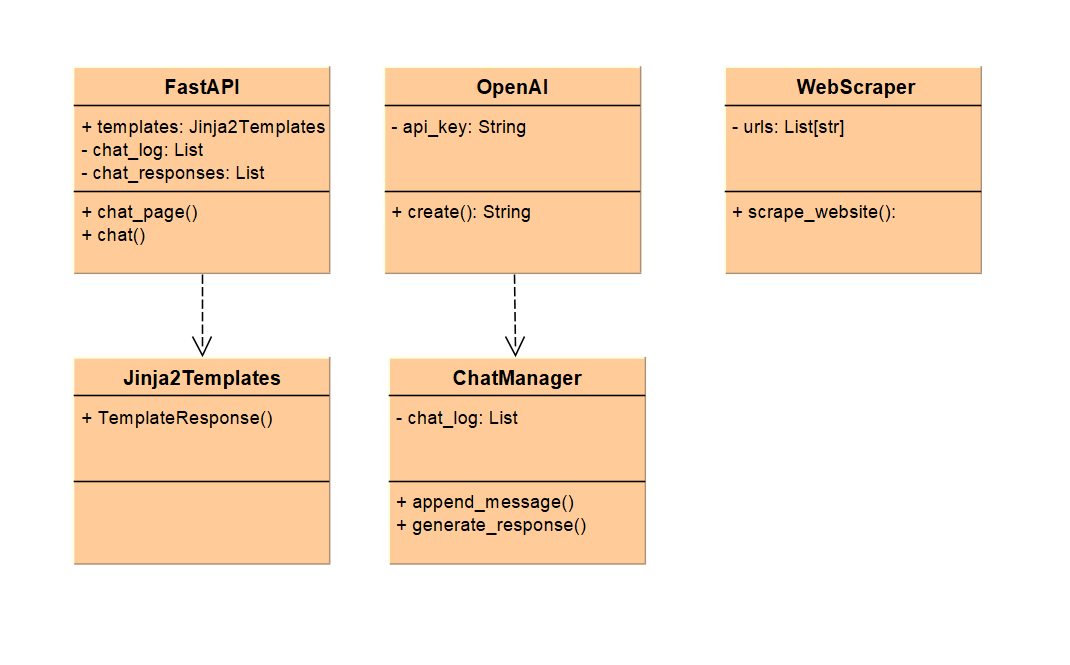
Στο συγκεκριμένο διάγραμμα περιπτώσεων χρήσης, το σύστημα αλληλοεπιδρά με δύο βασικούς χρήστες: τον Χρήστη και το Chatbot. Αρχικά ο χρήστης υποβάλλει κάποιο ερώτημα μέσω της περίπτωσης χρήσης «Αποστολή Ερώτησης», το οποίο μπορεί να αφορά την αναζήτηση ανακοινώσεων, μαθημάτων ή στοιχείων καθηγητών. Το Chatbot, στη συνέχεια, λαμβάνει την ερώτηση και επιστρέφει μια απάντηση μέσω της περίπτωσης «Αποστολή Απάντησης», η οποία περιλαμβάνει επιμέρους λειτουργίες, όπως η «Σάρωση της Ιστοσελίδας της Σχολής», η «Αναζήτηση Απάντησης» κ.α. Η λειτουργικότητα του συστήματος επεκτείνεται μέσω της δυνατότητας να βελτιώνει την ποιότητα των παρεχόμενων υπηρεσιών, καθώς και να μειώνει τον φόρτο εργασίας της γραμματείας, με γρήγορο και αποτελεσματικό τρόπο.

  
**Εικόνα 19: UML Διάγραμμα Περιπτώσεων Χρήσης**

### 4.3.2 UML Διάγραμμα Κλάσεων (Class Diagrams)

Τα διαγράμματα κλάσεων, ανήκουν στα διαγράμματα δομής της UML και βοηθούν να γίνει κατανοητή η δομή ενός συστήματος, δείχνοντας τις κλάσεις του, τα χαρακτηριστικά τους και τις σχέσεις που έχουν μεταξύ τους. Κάθε κλάση παρουσιάζεται με τα χαρακτηριστικά και τις μεθόδους της, και οι σχέσεις μεταξύ των κλάσεων, όπως η κληρονομικότητα, η σύνθεση και η συσχέτιση, απεικονίζονται με γραμμές και σύμβολα. Αυτά τα διαγράμματα, προσφέρουν μια απλοποιημένη εικόνα της αρχιτεκτονικής του συστήματος, διευκολύνοντας την κατανόηση των δομικών στοιχείων και των σχέσεών τους, χωρίς να αναφέρονται σε λεπτομέρειες για το πώς υλοποιούνται οι μέθοδοι ή οι εσωτερικές λειτουργίες.

Το UML διάγραμμα κλάσεων που υλοποιήθηκε για το συγκεκριμένο πρόγραμμα, παρουσιάζει τη στατική δομή και τη λογική του συστήματος. Η κλάση FastAPI, διαχειρίζεται τις αιτήσεις του χρήστη. Μέσω αυτής, γίνεται η απόδοση της σελίδας συνομιλίας και η επεξεργασία των εισόδων του χρήστη. Επίσης, αποθηκεύει το ιστορικό και τις απαντήσεις στις λίστες chat\_log και chat\_responses. Η κλάση OpenAI, είναι υπεύθυνη για την αλληλεπίδραση με το API της OpenAI. Μέσω του api\_key, αποκτά πρόσβαση στο GPT μοντέλο και δημιουργήσει απαντήσεις σε πραγματικό χρόνο μέσω της μεθόδου create(). Η κλάση WebScraper, συλλέγει δεδομένα από ιστότοπους. Η μέθοδος scrape\_website(), χρησιμοποιεί την βιβλιοθήκη BeautifulSoup για να εξάγει το κείμενο από τις ιστοσελίδες και να το επιστρέψει, για να παρέχει την απάντηση στο ερώτημα του χρήστη. Ο ChatManager, λειτουργεί ως διαχειριστής συνομιλιών. Η κλάση αυτή αποθηκεύει το ιστορικό των συνομιλιών και προσθέτει νέα μηνύματα μέσω της μεθόδου append\_message(). Παράλληλα, με τη μέθοδο generate\_response(), δημιουργούνται οι απαντήσεις με βάση το ερώτημα του χρήστη. Τέλος, η κλάση Jinja2Templates, είναι υπεύθυνη για την απόδοση των HTML σελίδων, με τη χρήση της μεθόδου TemplateResponse(), αφού διαχειρίζεται την εμφάνιση και αλληλεπίδραση του chatbot. Το διάγραμμα κλάσεων, καθιστά σαφές πώς οι κλάσεις που αναφέρθηκαν προηγουμένως, συνεργάζονται για να πετύχουν την ομαλή λειτουργεία του συστήματος.

  
**Εικόνα 20: UML Διάγραμμα Κλάσεων**

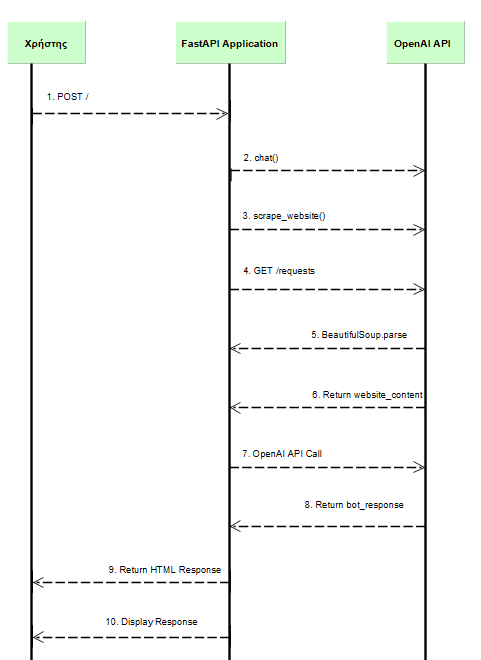
### 4.3.3 UML Διάγραμμα Ακολουθίας (Sequence Diagram)

Τα διαγράμματα ακολουθίας, δείχνουν τις αλληλεπιδράσεις που συμβαίνουν όταν εκτελείται μια συγκεκριμένη περίπτωση χρήσης του συστήματος και τη σειρά με την οποία αλληλοεπιδρούν τα διάφορα στοιχεία που το απαρτίζουν για να εκτελέσουν μια λειτουργία.

Για το συγκεκριμένο πρόγραμμα που χρησιμοποιεί FastAPI και OpenAI API, το διάγραμμα ακολουθίας συμβάλλει στη ανάλυση της ροής δεδομένων, δηλαδή δείχνει πώς τα δεδομένα, περνούν από τον χρήστη στην εφαρμογή FastAPI, στη συνέχεια, πώς η εφαρμογή αλληλοεπιδρά με άλλες βιβλιοθήκες και υπηρεσίες (όπως το BeautifulSoup και το OpenAI API) και τέλος, πώς τα αποτελέσματα επιστρέφονται στον χρήστη. Επίσης, βοηθάει στην κατανόηση της συνολικής διαδικασίας περιγράφοντάς τη, με τη χρονική σειρά των αλληλεπιδράσεων. Επιπλέον, διασφαλίζει ότι η εφαρμογή λειτουργεί σωστά και ότι όλα τα απαραίτητα βήματα και οι αλληλεπιδράσεις είναι σωστά ενσωματωμένα.

Η διαδικασία που ακολουθείται από το σύστημα για μια επιτυχή αλληλεπίδραση του Chatbot με τον χρήστη είναι η εξής:

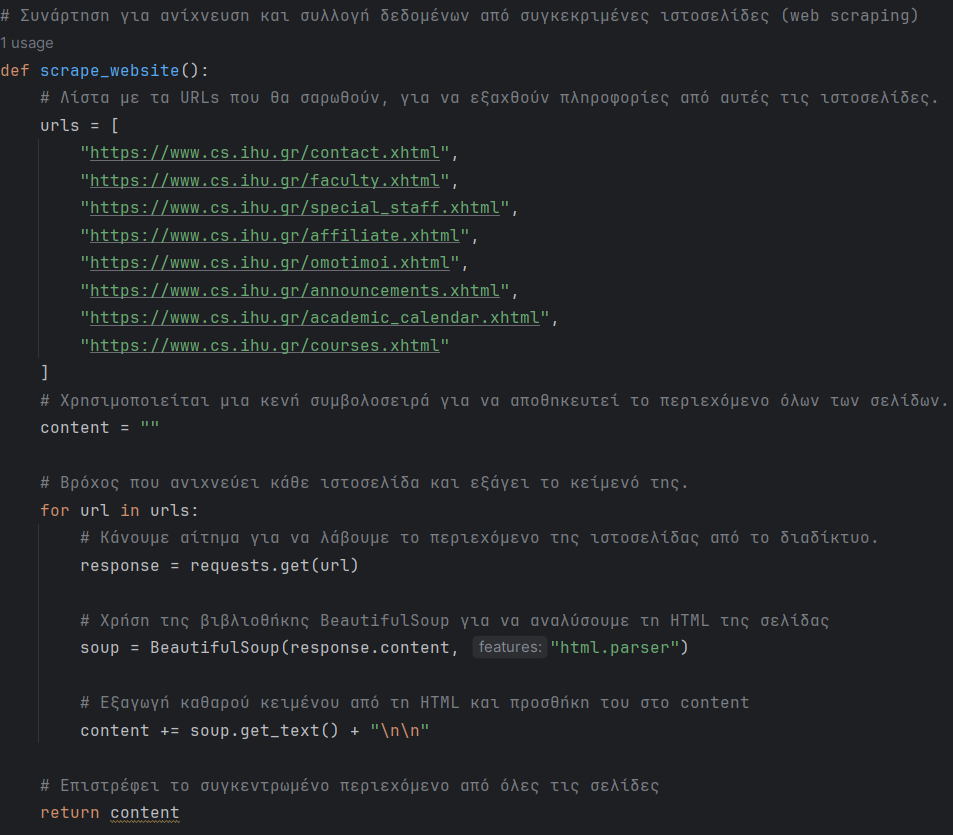
1. Ο χρήστης στέλνει ένα POST request στην εφαρμογή FastAPI. Το POST request περιέχει την είσοδο του χρήστη (π.χ. μια ερώτηση) μέσω μιας φόρμας.
2. Η εφαρμογή FastAPI λαμβάνει το POST request από τον χρήστη. Όταν το request φτάσει, η εφαρμογή καλεί τη συνάρτηση chat() για να επεξεργαστεί το αίτημα του χρήστη.
3. Η συνάρτηση chat() αρχικά ενημερώνει το chat\_log, το οποίο κρατάει ιστορικό των αλληλεπιδράσεων μεταξύ του χρήστη και του ακαδημαϊκού βοηθού. Στη συνέχεια, καλεί τη συνάρτηση scrape\_website() για να συλλέξει περιεχόμενο από τις ιστοσελίδες που έχουν καθοριστεί.
4. Η συνάρτηση scrape\_website() εκτελεί ένα αιτήματα GET για να αντλήσει περιεχόμενο από τις καθορισμένες ιστοσελίδες. Χρησιμοποιεί το BeautifulSoup για να αναλύσει το περιεχόμενο των σελίδων και να εξάγει το κείμενο από αυτές.
5. Μετά από τη συλλογή του περιεχομένου των ιστοσελίδων, η συνάρτηση chat() δημιουργεί το prompt για το OpenAI API. Το prompt περιλαμβάνει το περιεχόμενο των ιστοσελίδων και την είσοδο του χρήστη. Στη συνέχεια, η συνάρτηση καλεί το OpenAI API για να λάβει την απάντηση του βοηθού.
6. Το OpenAI API επεξεργάζεται το prompt και επιστρέφει την απάντηση του βοηθού στο FastAPI.
7. Το FastAPI ενημερώνει το chat\_log και το chat\_responses με την απάντηση του Chatbot. Στη συνέχεια, επιστρέφει την HTML σελίδα με την απάντηση στον χρήστη.
8. Και τέλος, ο χρήστης βλέπει την απάντηση του βοηθού στη σελίδα του browser του.

  
**Εικόνα 21: UML Διάγραμμα Ακολουθίας**

## 4.4 Χαρακτηριστικά τμήματα κώδικα

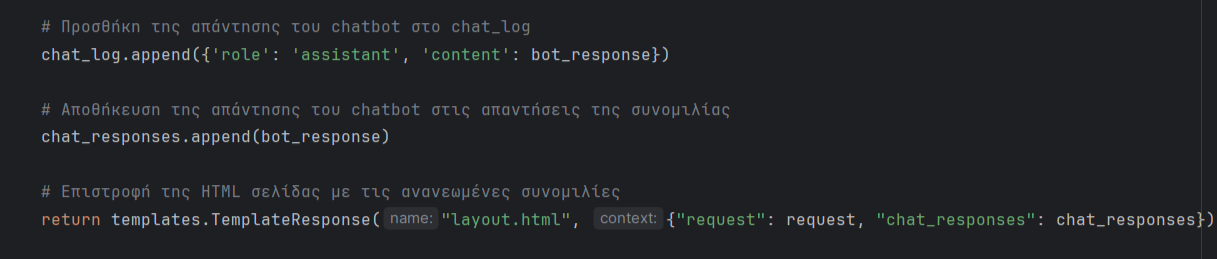
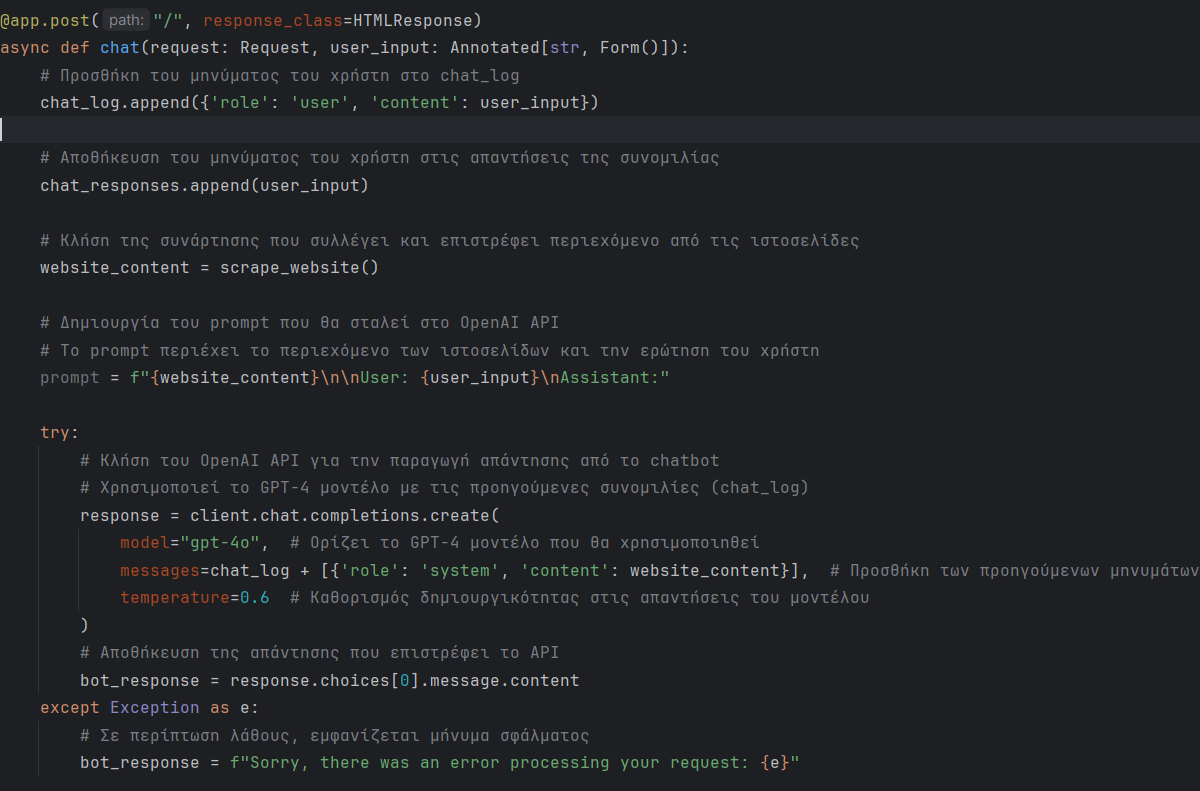
### Web Scraper

Η συνάρτηση scrape\_website, καλείται στο σημείο που δημιουργείται το prompt για το chatbot, και το περιεχόμενο των σελίδων παρέχεται ως επιπλέον πληροφορία στο μοντέλο, για να μπορεί να απαντήσει με βάση τις πληροφορίες του ιστότοπου. Συγκεκριμένα, ανιχνεύει τα URLs που του ζητούνται και περιέχουν πληροφορίες σχετικές με το τμήμα πληροφορικής του Δημοκρίτειου Πανεπιστήμιου Θράκης. Στη συνέχεια, συλλέγει το κείμενο από τις ιστοσελίδες και το επιστρέφει ως μία συμβολοσειρά που μπορεί να χρησιμοποιηθεί για άλλες διεργασίες, όπως η δημιουργία του prompt στο chatbot. Παρακάτω, βλέπουμε αναλυτικά τον κώδικα που χρησιμοποιήθηκε με σχόλια για την καλύτερη κατανόησή του.

 **Εικόνα 22: Η συνάρτηση scrape\_website με εσωτερική τεκμηρίωση**

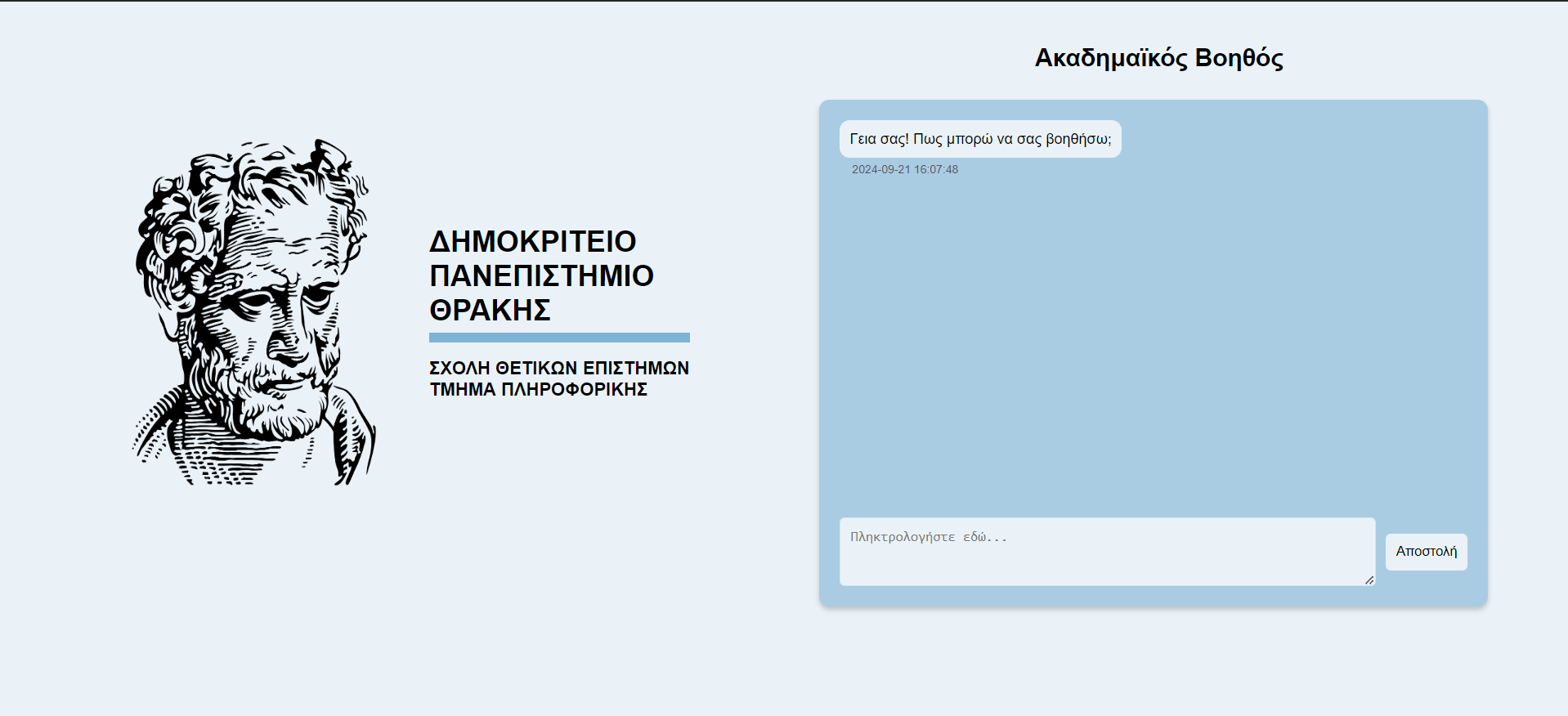
### Διαχείριση Επικοινωνίας Χρήστη-Chatbot.

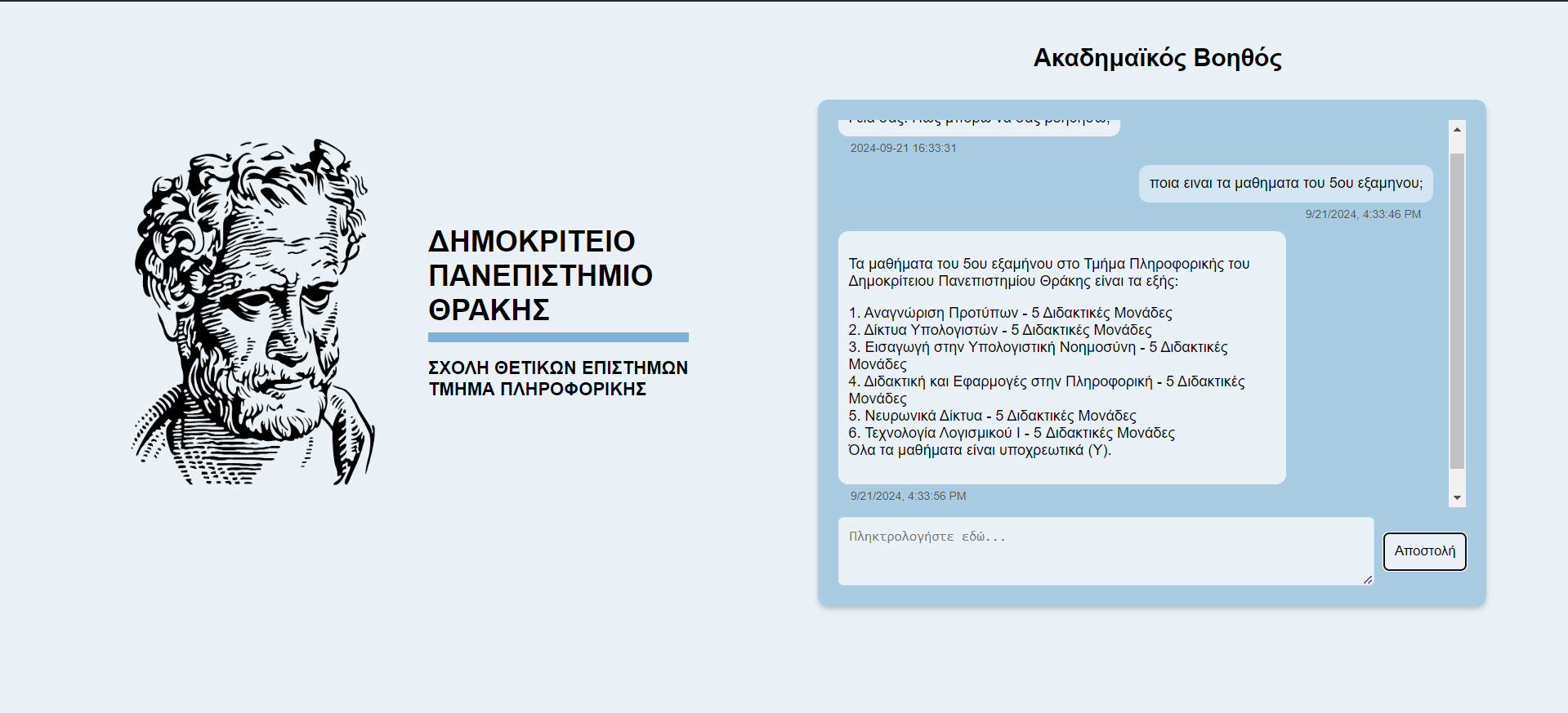
Η συνάρτηση chat(), αποτελεί το πιο σημαντικό τμήμα του προγράμματος που συνδέει όλες τις λειτουργίες μεταξύ τους. Εδώ ο χρήστης εισάγει δεδομένα και το chatbot επεξεργάζεται και επιστρέφει απάντηση.

  
**Εικόνα 23: Η συνάρτηση chat() με εσωτερική τεκμηρίωση**

## Περιγραφή Παραθύρου Εφαρμογής

Η εφαρμογή "Ακαδημαϊκός Βοηθός", έχει σχεδιαστεί για να προσφέρει φιλική διεπαφή προς τον χρήστη, επιτρέποντας την αλληλεπίδραση του το chatbot για την παροχή χρήσιμων πληροφοριών. Στο αριστερό τμήμα της οθόνης, υπάρχει το λογότυπο του Δημοκρίτειου Πανεπιστημίου Θράκης με τον τίτλο της σχολής. Δεξιά, βρίσκεται το κύριο μέρος της εφαρμογής που αποτελείται από το παράθυρο συνομιλίας. Εκεί, ο χρήστης μπορεί να ξεκινήσει μια συζήτηση με το σύστημα σε μορφή διαλόγου, όπου τα μηνύματα παρουσιάζονται με χρονοσήμανση για κάθε ανταλλαγή, ώστε να παρακολουθείται η ροή της επικοινωνίας σε πραγματικό χρόνο. Ο χρήστης βλέπει την ερώτησή του να εμφανίζεται, αφού έχει πατήσει το κουμπί send, με ανοιχτό γαλάζιο πλαίσιο στα δεξιά του πλαισίου, ενώ οι απαντήσεις του chatbot εμφανίζονται σε άσπρο χρώμα στα αριστερά, δημιουργώντας οπτική διαφοροποίηση για την καλύτερη κατανόηση του διαλόγου. Εάν η συνομιλία είναι μεγάλη, ο χρήστης έχει την δυνατότητα να μετακινηθεί πάνω- κάτω από την μπάρα δεξιά της συνομιλίας για να δει προηγουμενες απαντήσεις. Το περιβάλλον είναι απλό, εστιάζοντας στη λειτουργικότητα και την άμεση επικοινωνία.

 **Εικόνα 24: Αρχική Σελίδα της Εφαρμογής**

 **Εικόνα 25: Παράδειγμα συνομιλίας με τον Ακαδημαϊκό Βοηθό**

**Πίνακας 1: Συγκεντρωτικός Πίνακας ερωτήσεων που απαντάει το Chatbot**

|  |  |
| --- | --- |
| **Σχετικές Ερωτήσεις** | **Απαντήσεις** |
| Στοιχεία Επικοινωνίας με την Σχολή | Ναι |
| Πληροφορίες για Καθηγητές | Ναι |
| Ανακοινώσεις Τμήματος | Ναι |
| Πρόγραμμα Σπουδών | Ναι |
| Πτυχιακή Εργασία | Ναι |

**Πίνακας 2: Καταστάσεις Λειτουργίας**

|  |  |  |  |
| --- | --- | --- | --- |
| **Κατάσταση** | **Περιγραφή** | **Ενέργεια Χρήστη** | **Επόμενη Κατάσταση** |
| Αρχική Κατάσταση | Η εφαρμογή εμφανίζει το παράθυρο συνομιλίας με το καλωσόρισμα από τον βοηθό. | Ο χρήστης πληκτρολογεί μια ερώτηση. | Αναμονή απάντησης |
| Αναμονή απάντησης | Η εφαρμογή περιμένει την απάντηση από το chatbot αφού ο χρήστης έχει υποβάλει ερώτηση. | Το chatbot επεξεργάζεται την ερώτηση. | Παρουσίαση απάντησης |
| Παρουσίαση απάντησης | Η εφαρμογή εμφανίζει την απάντηση του chatbot στη συνομιλία. | Ο χρήστης βλέπει την απάντηση. | Αρχική Κατάσταση (για νέα ερώτηση) |
| Σφάλμα συστήματος (προαιρετική) | Σε περίπτωση αποτυχίας της λειτουργίας (π.χ. πρόβλημα σύνδεσης με το AI API). | Το σύστημα εμφανίζει μήνυμα σφάλματος. | Αρχική Κατάσταση (για νέα ερώτηση) |

# 5ο Κεφάλαιο: Αποτελέσματα Χρήσης

Στο κεφάλαιο αυτό, θα παρουσιαστούν τα αποτελέσματα χρήσης του Chatbot που υλοποιήθηκε σε ρεαλιστικά σενάρια, με ελέγχους καλής λειτουργίας, μετρήσεις χρόνου απόκρισης και πειραματικά αποτελέσματα. Για την καλύτερη ανάλυση των αποτελεσμάτων χρησιμοποιήθηκαν πίνακες, ένα ραβδόγραμμα και μία πίτα (pie chart), τα οποία σχεδιάστηκαν στη διαδικτυακή πλατφόρμα Canva, που επιτρέπει τη εύκολη δημιουργία γραφικών για κάθε είδους ανάγκη, όπως λογότυπα, παρουσιάσεις, διαγράμματα, αφίσες και πολλά άλλα. Επίσης, διαθέτει μια πλούσια συλλογή από έτοιμα πρότυπα, εικόνες, γραμματοσειρές, φίλτρα και άλλα εργαλεία, που με την κατάλληλη προσαρμογή τους στην εκάστοτε περίπτωση μπορούν να δώσουν εξαιρετικά αποτελέσματα.

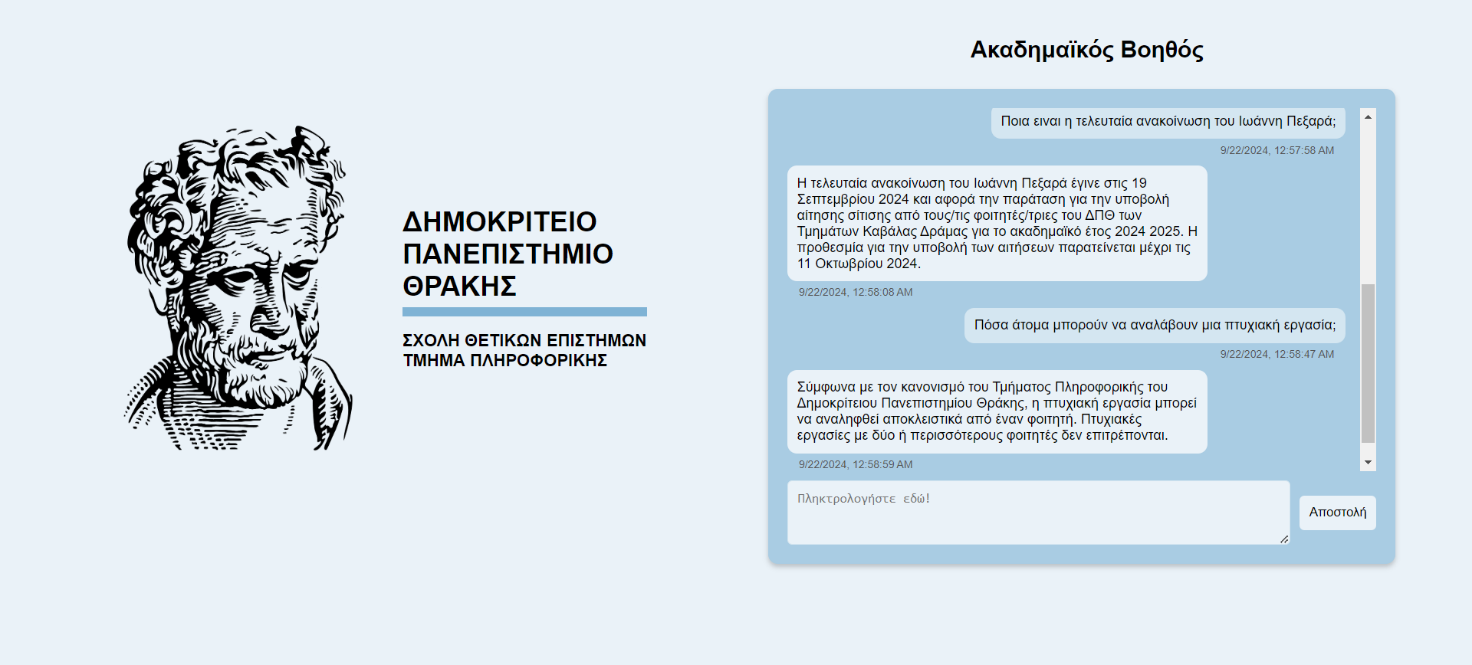
## 5.1 Έλεγχος Ποιότητας Απαντήσεων του Chatbot

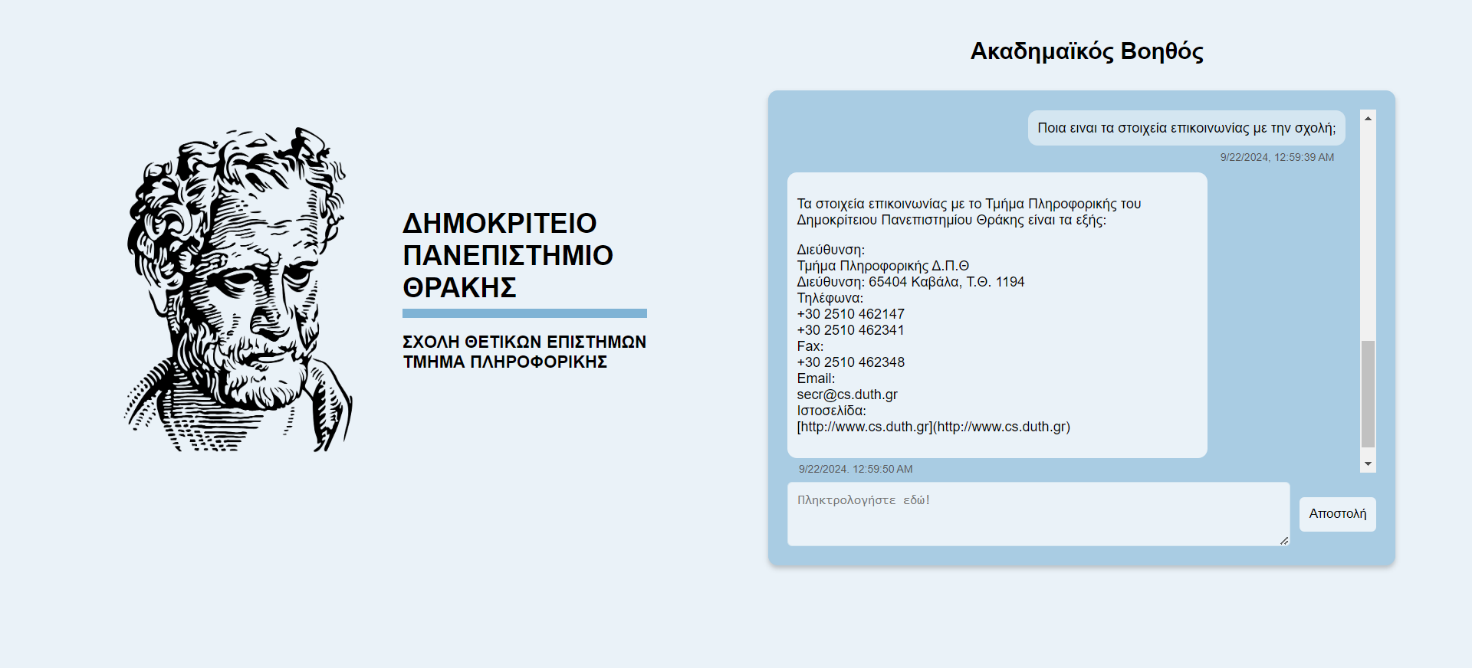
Για την αξιολόγηση της απόδοσης του chatbot, πραγματοποιήθηκαν δοκιμές με βάση ένα σύνολο διαφορετικών ερωτήσεων, οι οποίες κάλυπταν το φάσμα των θεματικών ενοτήτων και τους βαθμού δυσκολίας που υποστηρίζει το Chatbot, και στη συνέχεια αναλύθηκαν. Οι ερωτήσεις επιλέχθηκαν προσεκτικά, ώστε να αντικατοπτρίζουν ρεαλιστικές ερωτήσεις των χρηστών, διασφαλίζοντας ότι το σύστημα αξιολογείται σε πραγματικές συνθήκες.

Από τις ερωτήσεις αυτές, το σύστημα παρείχε σωστή και πλήρη απάντηση σε ένα ποσοστό 90% των περιπτώσεων. Οι απαντήσεις αυτές ήταν ακριβείς, σαφείς και κάλυπταν πλήρως τις ερωτήσεις. Ωστόσο, το 10% των περιπτώσεων ήταν αποτυχημένο, καθώς το Chatbot είτε δεν μπόρεσε να επεξεργαστεί σωστά την ερώτηση είτε η παρεχόμενη απάντηση ήταν ελλιπής ή μη σχετική, γεγονός που υποδεικνύει ότι το σύστημα μπορεί να βελτιστοποιηθεί περαιτέρω, .

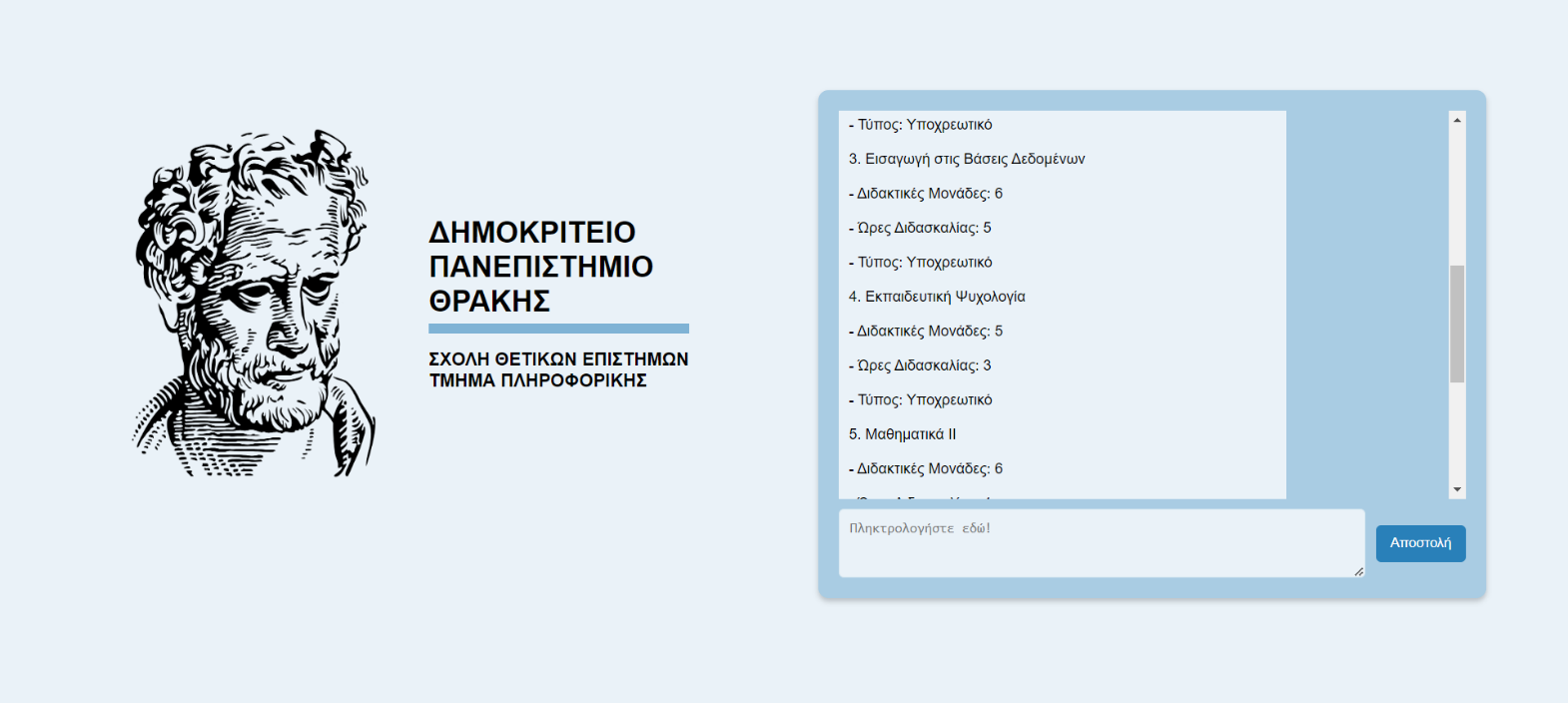
Τα αποτελέσματα αυτά είναι ενθαρρυντικά, καθώς το υψηλό ποσοστό επιτυχίας δείχνει ότι το Chatbot είναι σε θέση να ανταποκριθεί ικανοποιητικά στις περισσότερες ερωτήσεις που αφορούν τις ιστοσελίδες που έχει προγραμματιστεί να σαρώνει για να εξάγει απαντήσεις. Παράλληλα, οι περιπτώσεις αποτυχίας παρέχουν πολύτιμη ανατροφοδότηση για τη βελτίωση του αλγορίθμου του συστήματος, με στόχο την επίτευξη ακόμη υψηλότερης ακρίβειας στο μέλλον. Παρακάτω, βλέπουμε μερικά από τα παραδείγματα συνομιλιών καθώς και μια γραφική παράσταση σε μορφή πίτας (pie chart) για την καλύτερη κατανόηση των δυνατοτήτων της εφαρμογής:

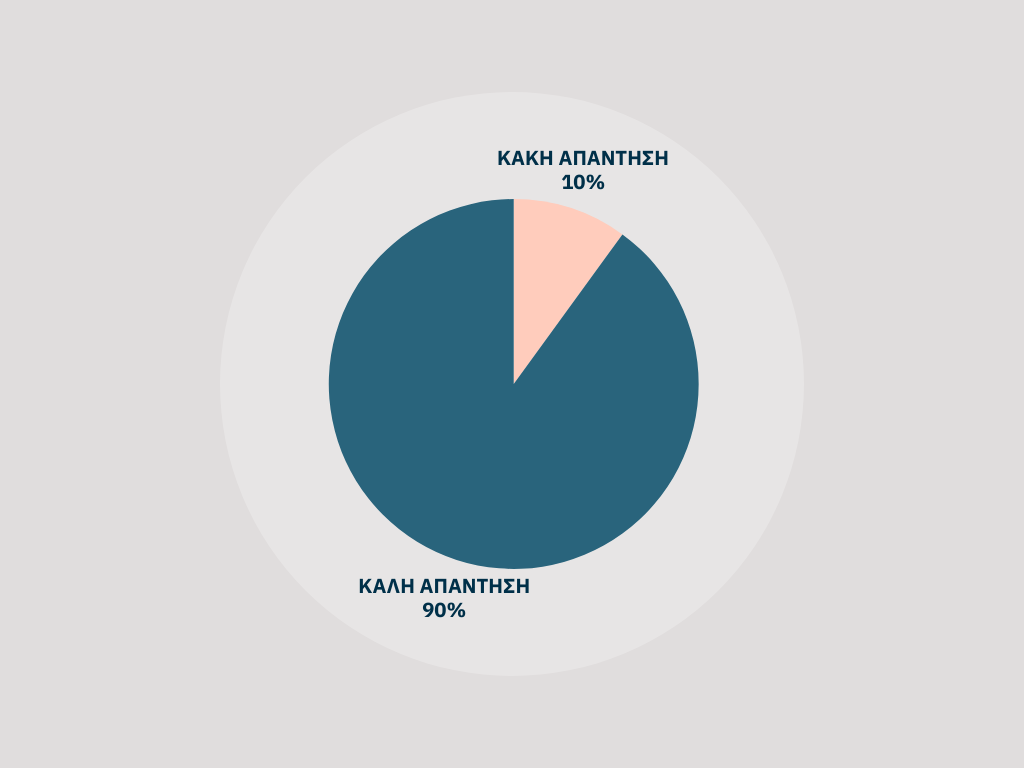
**Εικόνα 26: Παράδειγμα συνομιλίας (1)**

**Εικόνα 27: Παράδειγμα συνομιλίας (2)**

****

**Εικόνα 28: Παράδειγμα συνομιλίας (3)**

******Εικόνα 29: Παράδειγμα συνομιλίας (4)**

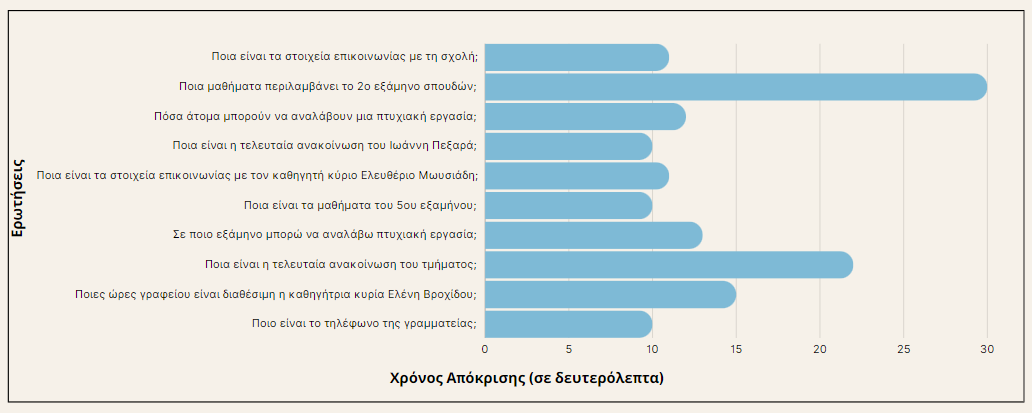
  
**Εικόνα 30: Διάγραμμα Ποσοστών Επιτυχίας-Αποτυχίας**

## 5.2 Έλεγχος Χρόνων Απόκρισης του Chatbot

Αφού πραγματοποιήθηκαν έλεγχοι με διαφορετικές ερωτήσεις και παρατηρήθηκε η ορθή λειτουργία του συστήματος, καταγράφηκαν οι χρόνοι απόκρισης για κάθε ερώτηση. Τα αποτελέσματα αυτά αποτυπώθηκαν στον Πίνακα 3 και στη γραφική παράσταση της εικόνας 28 για καλύτερη ανάλυση.

Οι χρόνοι απόκρισης του συστήματος σε μια ερώτηση εξαρτώνται από διάφορους παράγοντες. Αρχικά, ο χρόνος που απαιτείται για την αποστολή του αιτήματος στο OpenAI και η λήψη της απάντησης εξαρτάται από τη δικτυακή σύνδεση, τον φόρτο στους OpenAI servers και την πολυπλοκότητα του αιτήματος. Επίσης, η συνάρτηση scrape\_website() που συλλέγει δεδομένα από τις ιστοσελίδες, απαιτεί επιπλέον χρόνο για κάθε αίτηση HTTP και για την επεξεργασία του HTML περιεχομένου μέσω του BeautifulSoup. Οι χρόνοι απόκρισης μπορούν να επηρεαστούν ακόμη, από τον αριθμό των ιστοσελίδων που πρέπει να "σαρωθούν", καθώς και την ταχύτητα φόρτωσης των σελίδων. Επιπλέον, η μορφοποίηση της απάντησης (π.χ., ο καθαρισμός δεδομένων, ο διαχωρισμός επικοινωνιακών πληροφοριών και η δημιουργία λιστών μαθημάτων για την ευαναγνωσία των απαντήσεων) απαιτεί επεξεργασία, η οποία μπορεί να επιβραδύνει το σύστημα ανάλογα με την πολυπλοκότητα της εισόδου και των κανόνων μορφοποίησης. Επιπροσθέτως, ο FastAPI server που διαχειρίζεται τα αιτήματα και τις απαντήσεις μπορεί να επιβραδυνθεί από την ταυτόχρονη εξυπηρέτηση πολλών αιτήσεων, την κατάσταση των πόρων του διακομιστή (π.χ. CPU, μνήμη) και την υποδομή του Uvicorn. Τέλος, το σύστημα τρέχει σε AWS Lambda μέσω του Mangum, επομένως, ο χρόνος μπορεί να επηρεαστεί από την κατάσταση της πλατφόρμας Cloud και τον χρόνο εκκίνησης (cold start) του serverless περιβάλλοντος. Συνολικά, οι χρόνοι απόκρισης εξαρτώνται από τη δικτυακή απόδοση, τη φόρτωση εξωτερικών πόρων και την επεξεργασία δεδομένων στο backend.

Έχοντας λοιπόν υπόψιν όλα τα παραπάνω, και παρατηρώντας τα αποτελέσματα του Πίνακα 3 και του διαγράμματος, αντιλαμβανόμαστε ότι είναι πολλοί οι παράγοντες που επηρεάζουν τους χρόνους απόκρισης του Chatbot και καταλήγουμε στο συμπέρασμα ότι οι απαντήσεις δίνονται σε ικανοποιητικό χρόνο για τα δεδομένα της εργασίας, με μέσο όρο απάντησης (για τις 10 ερωτήσεις) τα 14.4 δευτερόλεπτα. Παρακάτω, παρουσιάζονται μερικές ερωτήσεις με τους χρόνους απόκρισής τους μέσω ενός διαγράμματος και ενός πίνακα:

  
**Εικόνα 31: Διάγραμμα Χρόνου Απόκρισης ανά Ερώτηση**

**Πίνακας 3: Χρόνοι απόκρισης του συστήματος**

|  |  |  |
| --- | --- | --- |
| **Ερώτηση** | **Χρόνος Απόκρισης Συστήματος** | **Ποιότητα Απάντησης** |
| Ποια είναι τα μαθήματα του 5ου εξαμήνου; (Εικόνα 23) | 10s | ΚΑΛΗ |
| Ποια είναι τα στοιχεία επικοινωνίας με τον καθηγητή κύριο Ελευθέριο Μωυσιάδη; (Εικόνα 24) | 11s | ΚΑΛΗ |
| Ποια είναι η τελευταία ανακοίνωση του Ιωάννη Πεξαρά;  (Εικόνα 25 ερώτηση 1) | 10s | ΚΑΛΗ |
| Πόσα άτομα μπορούν να αναλάβουν μια πτυχιακή εργασία;  (Εικόνα 25 ερώτηση 2) | 12s | ΚΑΛΗ |
| Ποια είναι τα στοιχεία επικοινωνίας με τη σχολή; (Εικόνα 26) | 11s | ΚΑΛΗ |
| Ποια μαθήματα περιλαμβάνει το 2ο εξάμηνο σπουδών;  (Εικόνα 27) | 30s | ΚΑΛΗ |
| Σε ποιο εξάμηνο μπορώ να αναλάβω πτυχιακή εργασία; | 13s | ΚΑΛΗ |
| Ποια είναι η τελευταία ανακοίνωση του τμήματος; | 22s | ΚΑΚΗ  (σε περίπτωση καρφιτσωμένης ανακοίνωσης) |
| Ποιες ώρες γραφείου είναι διαθέσιμη η καθηγήτρια κυρία Ελένη Βροχίδου; | 15s | ΚΑΛΗ |
| Ποιο είναι το τηλέφωνο της γραμματείας; | 10s | ΚΑΛΗ |

## 5.3 Ανάλυση Πειραματικών Αποτελεσμάτων

Η ανάλυση των πειραματικών αποτελεσμάτων καταδεικνύει ότι το Chatbot λειτουργεί αξιόπιστα, παρέχοντας σχετικά γρήγορες και ακριβείς απαντήσεις στην πλειονότητα των ερωτήσεων. Η υψηλή ακρίβεια που παρατηρήθηκε, σε συνδυασμό με τους χαμηλούς χρόνους απόκρισης, επιβεβαιώνει ότι το σύστημα είναι βελτιστοποιημένο και ανταποκρίνεται επαρκώς στα περισσότερα σενάρια χρήσης. Έτσι, το ποσοστό επιτυχίας, το οποίο φτάνει το 90%, ενισχύει την εμπιστοσύνη στις δυνατότητες του Chatbot να παρέχει χρήσιμες και ακριβείς πληροφορίες.

Οι χρόνοι απόκρισης που αξιολογήθηκαν με βάση τα δεδομένα και τις γραφικές αναπαραστάσεις (πίνακας και διάγραμμα), αποδείχθηκε ότι βρίσκονται σε ικανοποιητικά επίπεδα για εφαρμογές αυτού του τύπου. Επομένως, το Chatbot καταφέρνει να απαντά γρήγορα, γεγονός που βελτιώνει την εμπειρία των χρηστών, επιτρέποντας την άμεση κάλυψη των πληροφοριακών τους αναγκών.

Ωστόσο, η ανάλυση των περιπτώσεων αποτυχίας υποδεικνύει την ανάγκη για περαιτέρω βελτιώσεις στο σύστημα. Οι αποτυχίες εντοπίστηκαν κυρίως σε πιο σύνθετες ή ασαφείς ερωτήσεις, όπου το Chatbot δεν μπόρεσε να δώσει ακριβείς ή κατάλληλες απαντήσεις. Αυτά τα ζητήματα οφείλονται, πιθανώς, είτε σε περιορισμούς της κατανόησης φυσικής γλώσσας είτε σε ελλείψεις στη βάση δεδομένων του συστήματος. Παραδείγματος χάριν, στην περίπτωση ερώτησης του χρήστη για την τελευταία ανακοίνωση του τμήματος, εάν δεν υπάρχει κάποια καρφιτσωμένη ανακοίνωση, η απάντηση θα δοθεί ολόσωστα, αντιθέτως αν υπάρχει καρφιτσωμένη ανακοίνωση, η απάντηση θα περιέχει αυτή την ανακοίνωση και όχι την χρονολογικά τελευταία. Για την επίλυση αυτών των προβλημάτων, θα ήταν καλό να προστεθούν αλγόριθμοι για την βελτίωση της κατανόησης και της προσαρμοστικότητας του συστήματος.

Συνοψίζοντας, τα αποτελέσματα των δοκιμών επιβεβαιώνουν τη σταθερή και επιτυχημένη λειτουργία του Chatbot, δίνοντας περιθώριο για περαιτέρω βελτιστοποίηση, κυρίως σε πιο εξειδικευμένα και απαιτητικά σενάρια, ενισχύοντας τη συνολική αποτελεσματικότητά του.

# 6ο Κεφάλαιο: Συμπεράσματα, Προβλήματα και Μελλοντικές προτάσεις

Στο τελευταίο κεφάλαιο αυτής της εργασίας, θα αναφερθούν τα προβλήματα που παρουσιάστηκαν και ο τρόπος με τον οποίο επιλύθηκαν. Επίσης, θα περιγραφούν τα συμπεράσματα που προέκυψαν από την υλοποίηση και την χρήσης του Project, καθώς και οι προοπτικές βελτίωσης και περαιτέρω ανάπτυξης που θα εμπλούτιζαν τις δυνατότητές του Chatbot.

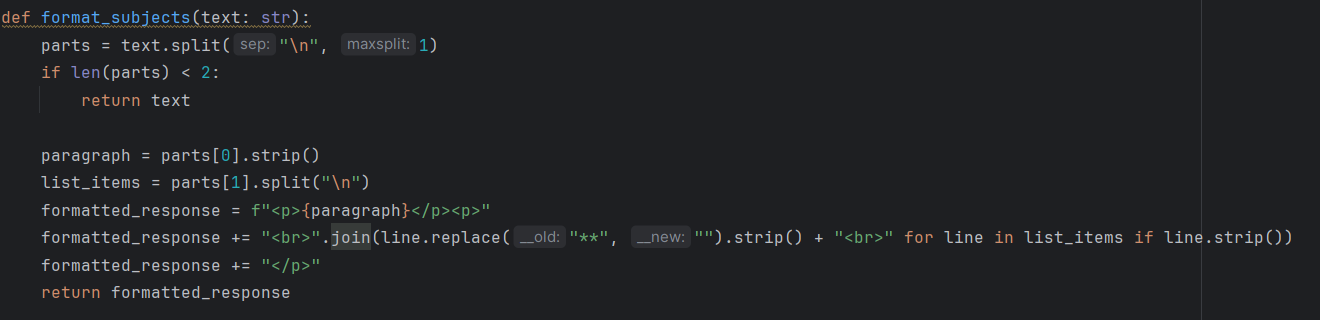
## 6.1 Προβλήματα που παρουσιάστηκαν

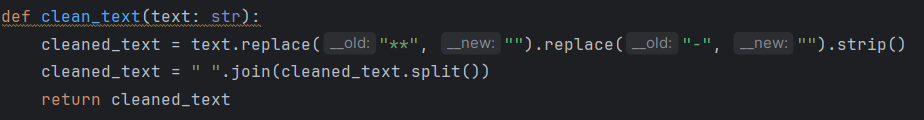
Το πρώτο πρόβλημα που παρουσιάστηκε κατά την εκπόνηση της εργασίας ήταν η υπέρβαση του ορίου των input tokens στο GPT-4ο, το οποίο και περιόρισε σημαντικά τις δυνατότητες του συστήματος. Παρόλα αυτά, η κατάσταση διαχειρίστηκε επιλέγοντας όσο το δυνατό ιστοσελίδες με μικρότερο αριθμό εισαγόμενων tokens ώστε να καλύπτει εύρος πληροφοριών από διάφορα θέματα. Αυτό γενικά μπορεί να προκληθεί από τους εξής λόγους:

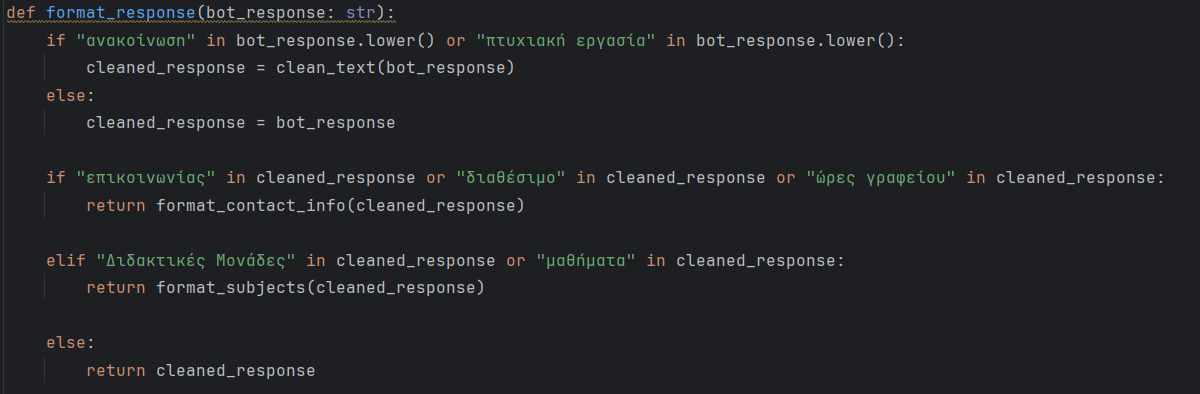
1. Όγκος του περιεχομένου που θα σαρωθεί από τις ιστοσελίδες: Κάθε φορά που προστίθεται ένα νέο URL στη συνάρτηση scrape\_website και εξάγει περιεχόμενο από την ιστοσελίδα, αυξάνεται ο όγκος του κειμένου που πρέπει να επεξεργαστεί το μοντέλο. Ολόκληρο το περιεχόμενο των σελίδων που γίνεται scraping προστίθεται στο prompt που στέλνεται στο GPT, αυξάνοντας έτσι τα input tokens που χρειάζεται να διαχειριστεί.
2. Tokens των συνομιλιών: Εκτός από το περιεχόμενο των ιστοσελίδων, κάθε φορά που γίνεται μια νέα ερώτηση, το chat\_log προσθέτει και τα προηγούμενα μηνύματα στη λίστα των tokens. Εάν υπάρχει συνομιλία με πολλές ερωτήσεις, αυτό μπορεί να αυξήσει τον αριθμό των tokens που αποστέλλονται στο μοντέλο.
3. Όριο του GPT-4o: Το μοντέλο GPT-4o έχει συγκεκριμένο όριο tokens (8.000 για τη δωρεάν έκδοση). Εάν το σύνολο του prompt μαζί με τις απαντήσεις, τα περιεχόμενα από τις σελίδες, και το ιστορικό συνομιλιών υπερβαίνουν αυτό το όριο, τότε το μοντέλο θα αποτύχει να επεξεργαστεί σωστά το αίτημα. Πέρα από τη δωρεάν έκδοση που χρησιμοποιήθηκε στο πρόγραμμα, υπάρχει και έκδοση με όριο τα 32000 input tokens, ωστόσο είναι επί πληρωμή.

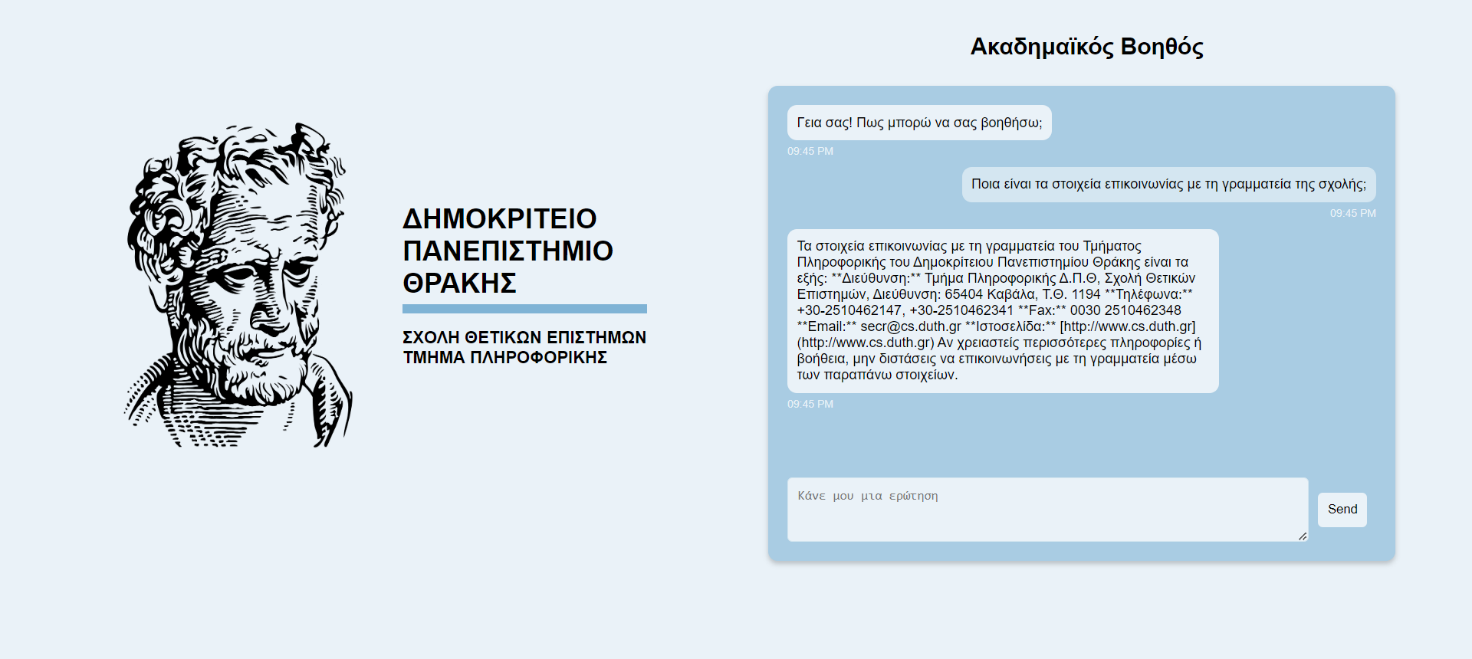
Στη συνέχεια, προέκυψε ένα πρόβλημα με την δομή των απαντήσεων που έβλεπε ο χρήστης στην οθόνη. Συγκεκριμένα, το περιεχόμενο που εξαγόταν από τις ιστοσελίδες μέσω του web scraping ήταν άμορφο και ακατέργαστο. Τα δεδομένα πολλές φορές περιείχαν HTML ετικέτες, ανεπιθύμητα κενά διαστήματα, ή αχρείαστα σημεία στίξης όπως "\*\*" και "-". Όλα αυτά, έκαναν τις απαντήσεις δυσανάγνωστες και υποβίβαζαν την εμπειρία διεπαφής χρήστη. Για την αντιμετώπιση αυτού του προβλήματος, χρησιμοποιήθηκαν συναρτήσεις που δομούν τις απαντήσεις και εξασφαλίζουν ότι το περιεχόμενο παρουσιάζεται με τρόπο κατανοητό και ευανάγνωστο. Παρακάτω, βλέπουμε τον κώδικα των συναρτήσεων που χρησιμοποιήθηκαν για αυτόν τον σκοπό καθώς και σύγκριση των συνομιλιών πριν και μετά την εφαρμογή τους.

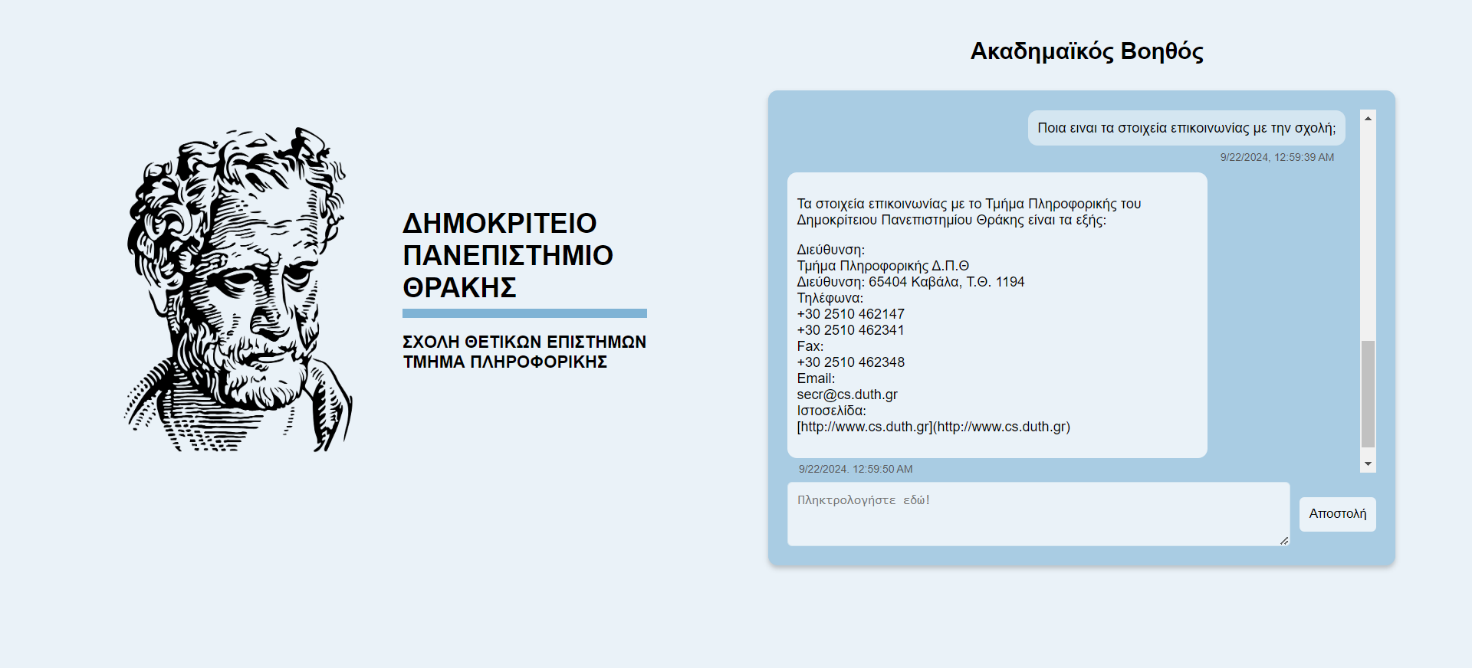
  
**Εικόνα 32: Συνάρτηση μορφοποίησης στοιχείων επικοινωνίας**

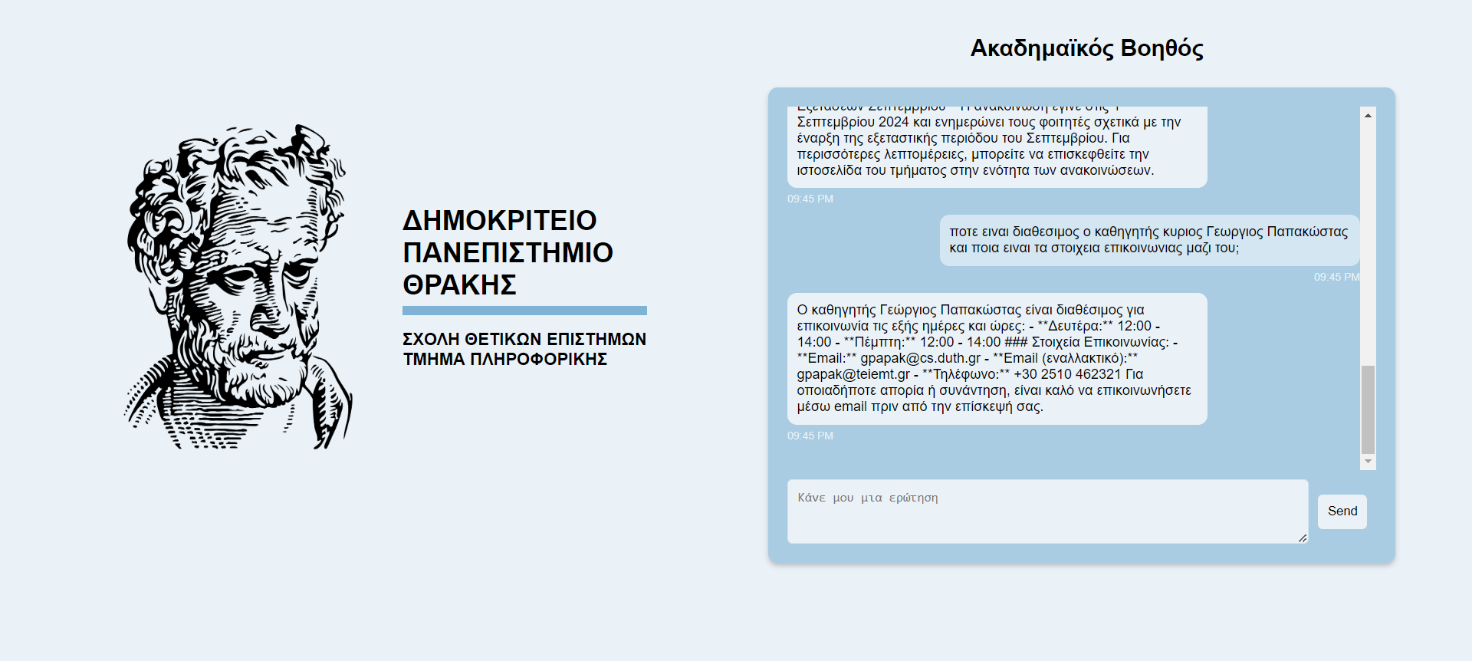
  
**Εικόνα 33: Συνάρτηση μορφοποίησης μαθημάτων**

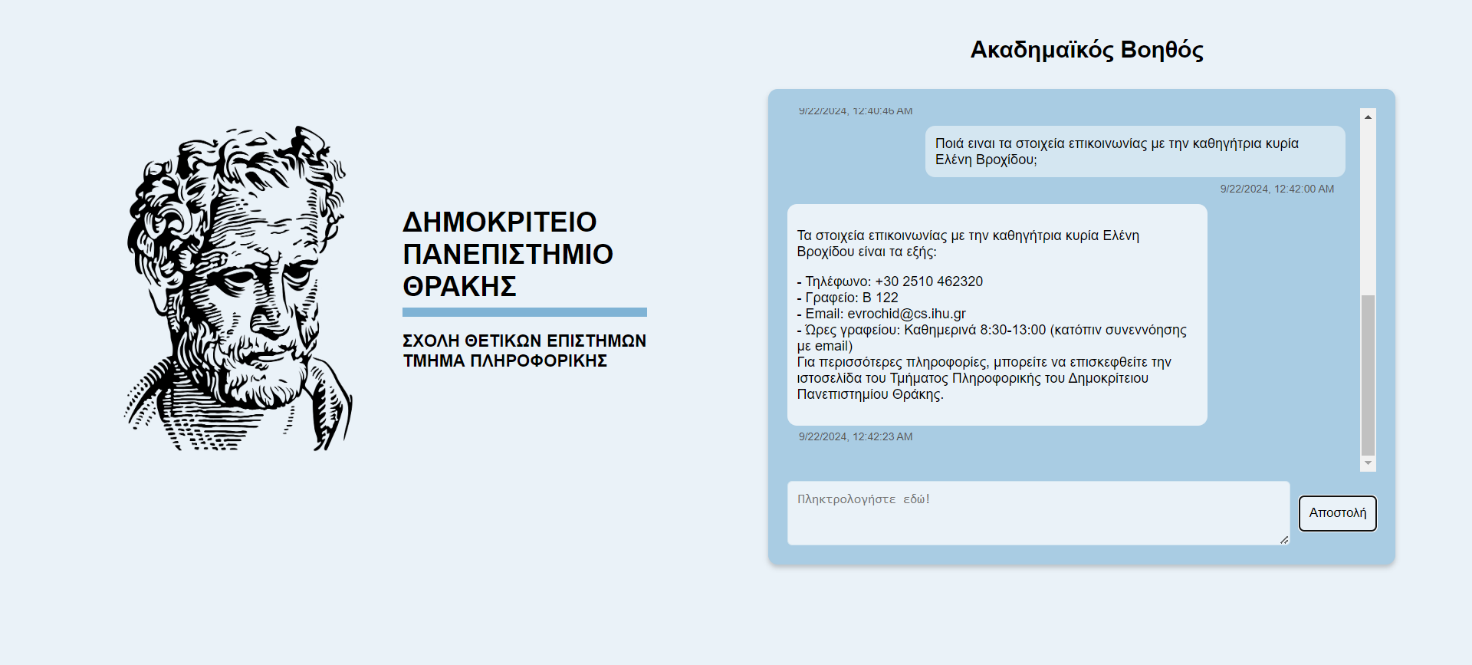
  
**Εικόνα 34: Συνάρτηση αφαίρεσης περιττών σημείων στίξης**

  
**Εικόνα 35: Συνάρτηση κλήσης των συναρτήσεων μορφοποίησης**

  
**Εικόνα 36: Δομή απάντησης πριν την μορφοποίηση (1)**

**Εικόνα 37: Δομή απάντησης μετά την μορφοποίηση (1)**

**Εικόνα 38: Δομή απάντησης πριν την μορφοποίηση (2)**

  
**Εικόνα 39: Δομή απάντησης μετά την μορφοποίηση (2)**

## 6.2 Συμπεράσματα

Στόχος της παρούσας εργασίας, όπως αναφέρθηκε και στην αρχή, είναι η ανάπτυξη ενός συστήματος ικανού να απαντά με φυσικό λόγο (Chatbot) σε απορίες φοιτητών, καθηγητών και γενικότερα «επισκεπτών» του τμήματος Πληροφορικής του Δημοκρίτειου Πανεπιστήμιου Θράκης και να τους παρέχει χρήσιμες πληροφορίες σχετικά με το τμήμα και τις σπουδές τους σε αυτό. Δημιουργήθηκε έτσι, ο «Ακαδημαϊκός Βοηθός», παρέχοντας μια φιλική διεπαφή στον χρήστη, με τη συνεργασία πολλών καινοτόμων εργαλείων και τεχνολογιών ώστε να επιτευχθεί το επιθυμητό αποτέλεσμα. Μετά την ολοκλήρωση του προγράμματος ακολούθησε η αξιολόγηση των επιδόσεών του με δοκιμές που βασίζονται σε σενάρια που θα μπορούσαν να αποτελούν ερωτήσεις των πραγματικών χρηστών στο Chatbot. Συμπεραίνουμε, λοιπόν, ότι το πανεπιστήμιο θα μπορούσε να αξιοποιήσει μια τέτοιου είδους τεχνολογία για να παρέχει πιο εύκολα, γρήγορα και ευχάριστα τις επιθυμητές πληροφορίες στους φοιτητές και σε όλους τους συσχετιζόμενους με αυτό.

Κατά την εκπόνηση της εργασίας, δημιουργήθηκαν και απαντήθηκαν τα παρακάτω βασικά ερωτήματα:

* Ποια είναι τα στάδια ανάπτυξης ενός Chatbot;

Για την ανάπτυξη του Ακαδημαϊκού Βοηθού, χρειάστηκε να δημιουργηθεί ένα OpenAI API κλειδί το οποίο επιτρέπει τη επικοινωνία με το μοντέλο GPT-4o. Επίσης, χρησιμοποιήθηκε το FastAPI framework για την διαχείριση των HTTP αιτημάτων και μια συνάρτηση για την σάρωση των επιλεγμένων ιστοσελίδων του τμήματος και την εξαγωγή περιεχομένου από αυτές. Όλα αυτά, σε συνδυασμό με την φιλική προς τον χρήστη διεπαφή, προσφέρουν μια ολοκληρωμένη εμπειρία χρήσης ενός αξιοπρεπούς Chatbot.

* Ποιες ερωτήσεις μπορεί να απαντήσει ο Ακαδημαϊκός Βοηθός;

Το Chatbot έχει προγραμματιστεί να απαντάει σε ερωτήσεις που σχετίζονται με τις ιστοσελίδες που σαρώνει. Συγκεκριμένα, μπορεί να σαρώσει τις παρακάτω ιστοσελίδες οι οποίες σχετίζονται με το πρόγραμμα σπουδών, τους καθηγητές, τις ανακοινώσεις, τις πτυχιακές εργασίες και την επικοινωνία με τη γραμματεία του τμήματος παρέχοντας αξιόπιστες πληροφορίες που ανταποκρίνονται στην πραγματικότητα :

<https://www.cs.ihu.gr/faculty.xhtml>

<https://www.cs.ihu.gr/affiliate.xhtml>

<https://www.cs.ihu.gr/contact.xhtml>

<https://www.cs.ihu.gr/courses.xhtml>

<https://www.cs.ihu.gr/omotimoi.xhtml>

<https://www.cs.ihu.gr/dissertation.xhtml>

<https://www.cs.ihu.gr/special_staff.xhtml>

<https://www.cs.ihu.gr/announcements.xhtml>

* Θα βοηθούσε στην ομαλή λειτουργία του Πανεπιστημίου ένα Chatbot;

Ο κόσμος πλέον έχει αρχίσει να συνηθίζει και να εκμεταλλεύεται τα Chatbot προς όφελός του σε προσωπικό επίπεδο και σε επαγγελματικό. Στην προκειμένη περίπτωση, η χρήση ενός Chatbot θα μείωνε τον φόρτο εργασίας των εργαζομένων στη γραμματεία του τμήματος επιτρέποντάς τους να ασχοληθούν με πιο μείζον θέματα και όχι να παρέχουν απλές πληροφορίες.

Από την άλλη, οι ενδιαφερόμενοι θα λαμβάνουν γρήγορες και εύστοχες απαντήσεις και θα γλυτώνουν χρόνο από τις ουρές αναμονής στη γραμματεία, εφόσον με το Chatbot μπορούν να εξυπηρετηθούν πολλοί χρήστες ταυτόχρονα. Επομένως, δεν υπάρχουν μελανά σημεία στην αξιοποίηση των Chatbot από τα Πανεπιστήμια. Για να επιτευχθεί όμως αυτό, θα βοηθούσε ο Ακαδημαϊκός Βοηθός να ενσωματωθεί στην ιστοσελίδα του τμήματος ώστε να μπορούν οι χρήστες να έρθουν εύκολα σε επαφή μαζί του.

## 6.3 Προοπτικές βελτίωσης

Κατά την υλοποίησης της εργασίας, γινόταν διαρκής προσπάθεια για εύρεση τρόπων με τους οποίους θα μπορούσε να βελτιωθεί η απόδοση του ήδη υπάρχοντος συστήματος ή ακόμη και πως θα μπορούσαν να εμπλουτιστούν οι λειτουργίες του μελλοντικά. Δυστυχώς όμως, κάποια στιγμή υπήρξε περιορισμός στους πόρους που χρησιμοποιήθηκαν και τότε έγινε κατανοητό ότι προκειμένου να μπορέσει να επεκταθεί το σύστημα και να μπορεί να υποστηρίξει περισσότερες λειτουργίες χρειάζονται κάποιες επιπλέον ενέργειες που για τη συγκεκριμένη χρονική στιγμή δεν θα μπορούσαν να πραγματοποιηθούν. Παρακάτω λοιπόν περιγράφονται αυτές αλλά και τι περιθώρια και προοπτικές βελτίωσης υπάρχουν γενικότερα για τον Ακαδημαϊκό Βοηθό.

Όπως παρατηρήσαμε στο προηγούμενο κεφάλαιο, οι απαντήσεις που λαμβάνουμε από το Chatbot είναι σε μεγάλο ποσοστό ικανοποιητικές, ωστόσο, υπάρχει ένα μικρό ποσοστό ανεπιτυχών απαντήσεων. Αυτή η παρατήρηση υποδεικνύει ότι το σύστημα μπορεί να βελτιωθεί, προκειμένου να καλύπτει αποτελεσματικά όλο το φάσμα των ερωτήσεων. Για να επιτευχθεί αυτό, θα ήταν σκόπιμο να εξεταστεί η χρήση ενός άλλου μοντέλου ΑΙ που να μπορεί να διαχειριστεί περισσότερα tokens εισαγωγής. Αυτό θα επέτρεπε την εκτενή σάρωση περισσότερων ιστοσελίδων και την εξαγωγή περισσότερων και πιο εμπεριστατωμένων πληροφοριών. Ωστόσο, τα μοντέλα της OpenAI δεν διαφέρουν σημαντικά μεταξύ τους όσον αφορά αυτή τη δυνατότητα, γεγονός που καθιστά αναγκαία την αναμονή για κάποια ενημέρωση ή για την ανάπτυξη νέων μοντέλων που θα μπορούσαν να προσφέρουν βελτιωμένα χαρακτηριστικά.

Επιπλέον, για να διευρυνθούν οι δυνατότητες του Chatbot και να τελειοποιηθεί ως εργαλείο, ιδανικά θα έπρεπε να ενσωματωθεί στην ιστοσελίδα της σχολής. Αυτό θα διευκόλυνε την άμεση επαφή με πραγματικούς χρήστες, επιτρέποντας στο Chatbot να παρέχει τις υπηρεσίες του σε πραγματικό χρόνο. Αυτή η ενσωμάτωση θα μπορούσε να περιλαμβάνει πέρα από τις ήδη υπάρχουσες λειτουργίες, και άλλες όπως είναι η δυνατότητα προγραμματισμού ραντεβού με καθηγητές κ.α.

Τέλος, θα ήταν πολύ χρήσιμο να προστεθεί ένας ακόμη τρόπος αλληλεπίδρασης με τους χρήστες αφού προς το παρόν επικοινωνεί μόνο με μηνύματα, πράγμα που καθιστά αδύνατη τη χρήση του από χρήστες με προβλήματα όρασης. Η προσθήκη φωνητικών εντολών ή η δυνατότητα αλληλεπίδρασης μέσω φωνής θα μπορούσε να διευρύνει την προσβασιμότητα και να βελτιώσει την εμπειρία των χρηστών. Επίσης, η υιοθέτηση λειτουργιών όπως η αναγνώριση φυσικής γλώσσας (NLP) θα μπορούσε να ενισχύσει την κατανόηση και την αλληλεπίδραση με τους χρήστες, επιτρέποντας πιο φυσικές και ρεαλιστικές συνομιλίες.

# Αναφορές

1. Almahri, F. A., Merhi , M., & Bell , D. (2020). *Understanding Student Acceptance and Use of Chatbots in the United Kingdom Universities: A Structural Equation Modelling Approach.*
2. De Santo, M., Lombardi, M., Pascale, F., Pietrosanto, A., & Colace, F. (2018). *Chatbot for E-Learning: A Case of Study.* Italy: University of Salerno.
3. Huang , X. (2021). *CHATBOT: DESIGN, ARCHITECUTRE, AND APPLICATIONS.* University of Pennsylvania.
4. ΑΝΑΣΤΑΣΙΟΥ, Μ. (2021). *ΚΑΤΑΣΚΕΥΗ CHATBOT ΓΙΑ ΕΝΗΜΕΡΩΣΗ ΦΟΙΤΗΤΩΝ.* Πάτρα: ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ.
5. Μπασιά, Α. (2021). *Δημιουργία Διαλογικού Πράκτορα (Chat Bot) σε περιβάλλον Python.* Θεσσαλονίκη: ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ.
6. ΑΚΡΙΤΙΔΟΥ, Ε. (2023). *Δημιουργία Εικονικού έξυπνου Βοηθού Chatbot για υποβοήθηση της Ακαδημαϊκής Κοινότητας.* Κοζάνη: Πανεπιστήμιο Δυτικής Μακεδονίας.
7. Τσιρίδης, Γ., & Σταύρου, Ι. (2021). *Πρωτοετάκης: Ανάπτυξη Έξυπνου Βοηθού (Chatbot) για Φοιτητές ΑΕΙ.* Πανεπιστήμιο Αιγαίου.
8. Turing, A. (1950). *Computing Machinery and Intelligence.*
9. Βλαχάβας, Κεφαλάς, Βασιλειάδης, Κόκκορας, & Σακελλαρίου. (2020). *Τεχνητή Νοημοσύνη.* Θεσσαλονίκη: ΕΚΔΟΣΕΙΣ ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΜΑΚΕΔΟΝΙΑΣ.
10. ΝΑΪΝΤΕΝΟΒΑ, Β. (2024). *Ανάπτυξη Συστήματος Ανάλυσης Συναισθήματος με Χρήση Επεξεργασίας.* ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ.
11. Georgouli, A. (2015). *Τεχνητή νοημοσύνη.*
12. *Supervised and Unsupervised learning*. (2024, Μάρτιος 13). Ανάκτηση από https://www.geeksforgeeks.org/supervised-unsupervised-learning/
13. Αβούρης, Ν., Κατσάνος, Χ., Τσέλιος, Ν., & Μουστάκας, Κ. (2015). *Εισαγωγή στην Αλληλεπίδραση Ανθρώπου-Υπολογιστή.* Εθνικό Μετσόβιο Πολυτεχνείο.
14. Aggarwal, K., & Singh, Y. (2005). *Software Engineering Programs, Documentation, Operating Procedures.*
15. Γιακουμάκης, Ε. Α., & Διαμαντίδης, Ν. Α. (2021). *Τεχνολογία Λογισμικού.* Αθήνα: UNIBOOKS.
16. *Software Development Life Cycle (SDLC)*. (2024, Ιουλίου 18). Ανάκτηση από https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/
17. Hoffer, J. A., Ramesh, V., & Topi, H. (2024). *Βάσεις Δεδομενων Σύγχρονη Διαχείριση .* ΕΚΔΟΣΕΙς ΤΖΙΟΛΑ.
18. *What is Web Scraping and How to Use It?* (2024, July 16). Ανάκτηση από Geeks for Geeks: https://www.geeksforgeeks.org/what-is-web-scraping-and-how-to-use-it/
19. *Web Scraping*. (2023, April). Ανάκτηση από WIKIPEDIA: https://en.wikipedia.org/wiki/Web\_scraping
20. Al-Amin, M., Shazed Ali, M., Salam, A., Khan, A., Ali, A., Ullah, A., . . . Chowdhury, S. (2024). *History of generative Artificial Intelligence (AI) chatbots: past, present, and future development.*
21. Adamopoulou, E., & Moussiades, L. (2020). *Chatbots: History, technology, and applications.* Καβάλα: Department of Computer Science, International Hellenic University.
22. Verma, S., Sahni, L., & Sharma, M. (2020). *COMPARATIVE ANALYSIS OF CHATBOTS.*
23. Jay, B. (2024). *Chatbots: history, technology, and a case analysis.*
24. Church, B. (2023, September 5). *5 types of chatbot and how to choose the right one for your business*. Ανάκτηση από https://www.ibm.com/think/topics/chatbot-types
25. Sapardic, J. (2024, Ιούνιος 11). *6 Different Types of Chatbots*. Ανάκτηση από https://www.tidio.com/blog/chatbot-types/#NLP-hatbots
26. Misischia, C., Poecze, F., & Strauss, C. (2022). *Chatbots in customer service: Their relevance and impact on service quality.* Porto, Portugal.
27. ALT, M.-A., VIZELI, I., & SĂPLĂCAN , Z. (2021). *BANKING WITH A CHATBOT – A STUDY ON TECHNOLOGY ACCEPTANCE.* STUDIA UNIVERSITATIS BABEȘ-BOLYAI OECONOMICA.
28. Vryoni, V. (2021). *Chatbots in Healthcare: Towards AI-enabled general diagnosis and medical support.* Athens: UNIVERSITY OF PIRAEUS.
29. Bhirud, N., Tataale, S., Randive, S., & Nahar, S. (2019). *A Literature Review On Chatbots In Healthcare Domain.*
30. Amato, F., Marrone, S., Moscato, V., Piantadosi, G., Picariello, A., & Sansone, C. (2017). *Chatbots meet eHealth: automatizing healthcare.* Napoli.
31. Θεοχαρόπουλος , Φ. (2020). *Η ΧΡΗΣΗ ΤΩΝ CHATBOTS ΣΤΗΝ ΕΚΠΑΙΔΕΥΣΗ.* Θεσσαλονίκη: ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ.
32. Moez, A. (2022, September 13). *FastAPI Tutorial: An Introduction to Using FastAPI*. Ανάκτηση από https://www.datacamp.com/tutorial/introduction-fastapi-tutorial?dc\_referrer=https%3A%2F%2Fwww.google.com%2F
33. Y, D. (2024, April 15). *Understand the purpose of Uvicorn in FastAPI applications*. Ανάκτηση από https://sentry.io/answers/understand-the-purpose-of-uvicorn-in-fastapi-applications/
34. *OpenAI*. (χ.χ.). Ανάκτηση από WIKIPEDIA : https://en.wikipedia.org/wiki/OpenAI
35. *About GitHub and Git*. (2024). Ανάκτηση από GitHub: https://docs.github.com/en/get-started/start-your-journey/about-github-and-git#about-github
36. Φαλλιέρος, Δ., Δεδούση, Μ., & Κωσταπαππάς, Λ. (2021, April 21). *Να γιατί η Python είναι τόσο δημοφιλής*. Ανάκτηση από Kariera.gr: https://resources.kariera.gr/el/blog/why-python-is-popular/
37. *What is Python? Its Uses and Applications*. (2024, September 12). Ανάκτηση από Geeks for Geeks: https://www.geeksforgeeks.org/what-is-python/
38. *CSS*. (χ.χ.). Ανάκτηση από WIKIPEDIA: https://en.wikipedia.org/wiki/CSS
39. *JavaScript*. (χ.χ.). Ανάκτηση από WIKIPEDIA: https://simple.wikipedia.org/wiki/JavaScript

# **Παράρτημα Ι**

**Οδηγίες εγκατάστασης των εργαλείων και της εφαρμογής**

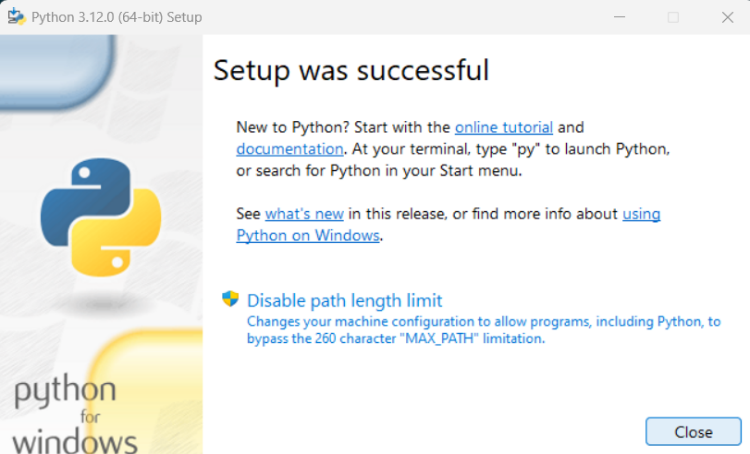
1. Εγκατάσταση Python 3.12.0.

Για να ξεκινήσει την ανάπτυξη του προγράμματος, θα πρέπει να έχει εγκατασταθεί η γλώσσα προγραμματισμού Python 3.12.0 στον υπολογιστή.

Βήματα εγκατάστασης Python 3.12.0:

1. Εγκατάσταση της Python 3.12.0 για το λειτουργικό σύστημα που χρησιμοποιείται (Windows, macOS, ή Linux) και τις προδιαγραφές του (32-bit, 64-bit κλπ.) από την ιστοσελίδα (<https://www.python.org/downloads/release/python-3120/>).
2. Εκτέλεση του αρχείου εγκατάστασης που εγκαταστάθηκε (.exe).   
   Έπειτα εκτελούνται τις παρακάτω ενέργειες:

Στο πρώτο παράθυρο, επιλέγεται το κουτάκι "Add Python to PATH" για να είναι διαθέσιμη η Python από τον τερματικό. Και στη συνέχεια επιλέγεται το Install Now για να ξεκινήσει η εγκατάσταση. Μόλις αυτή ολοκληρωθεί, επιλέγεται το close για να κλείσει η καρτέλα.

**Εικόνα 40: Εγκατάσταση Python σε Windows**

2. Εγκατάσταση PyCharm 2024.2.2 (IDE για Python).

Μετά την εγκατάσταση της Python, ακολουθεί η εγκατάσταση του PyCharm, που χρησιμοποιείται για την ανάπτυξη κώδικα σε Python.

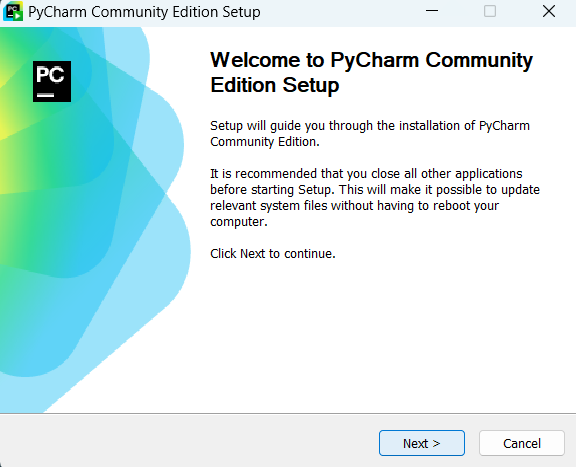
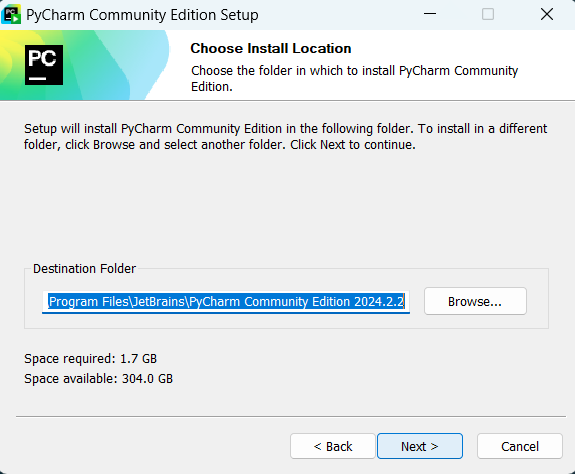
Βήματα εγκατάστασης PyCharm:

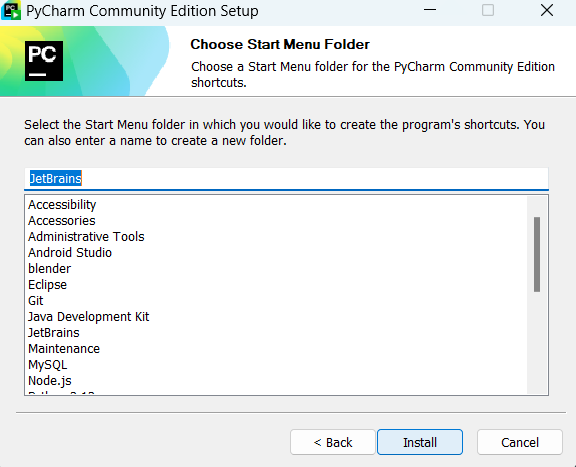
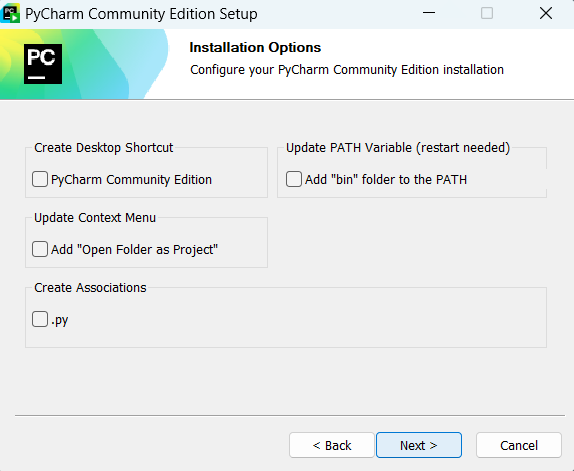
1. Από την [επίσημη σελίδα του PyCharm](https://www.jetbrains.com/pycharm/download/) γίνεται λήψη της Community Edition για το εκάστοτε λογισμικό

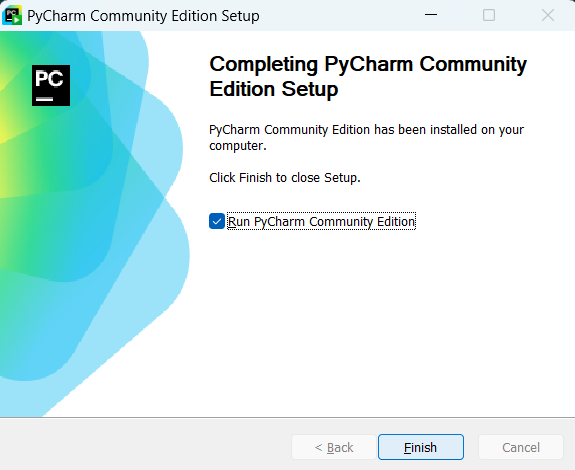
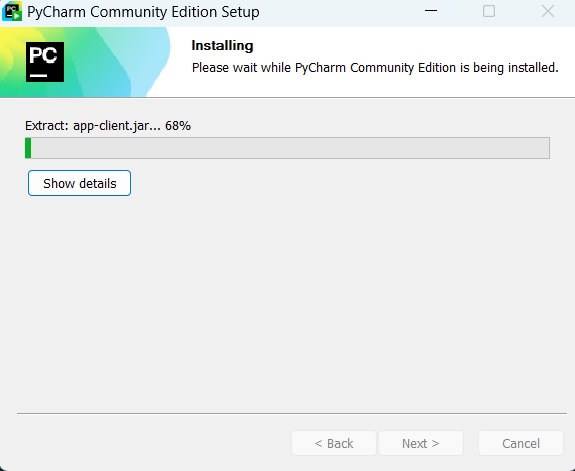
( <https://www.jetbrains.com/pycharm/download/?section=windows> ).

1. Εκτέλεση του αρχείου εγκατάστασης που λήφθηκε (.exe).

Έπειτα εκτελούνται οι σκιασμένες ενέργειες:

 **Εικόνα 41: Εγκατάσταση Pycharm σε Windows (βήμα 1 και 2)**

**Εικόνα 42:Εγκατάσταση Pycharm σε Windows (βήμα 3 και 4)**

  
**Εικόνα 43:Εγκατάσταση Pycharm σε Windows (βήμα 4 και 5)**

3.Σύνδεση/Δημιουργία Λογαριασμού στην OpenAI.

Στη συνέχεια γίνεται σύνδεση ή δημιουργείται ένας λογαριασμός στην OpenAI για την απόκτηση του API Key που θα χρησιμοποιείται στην εφαρμογή.

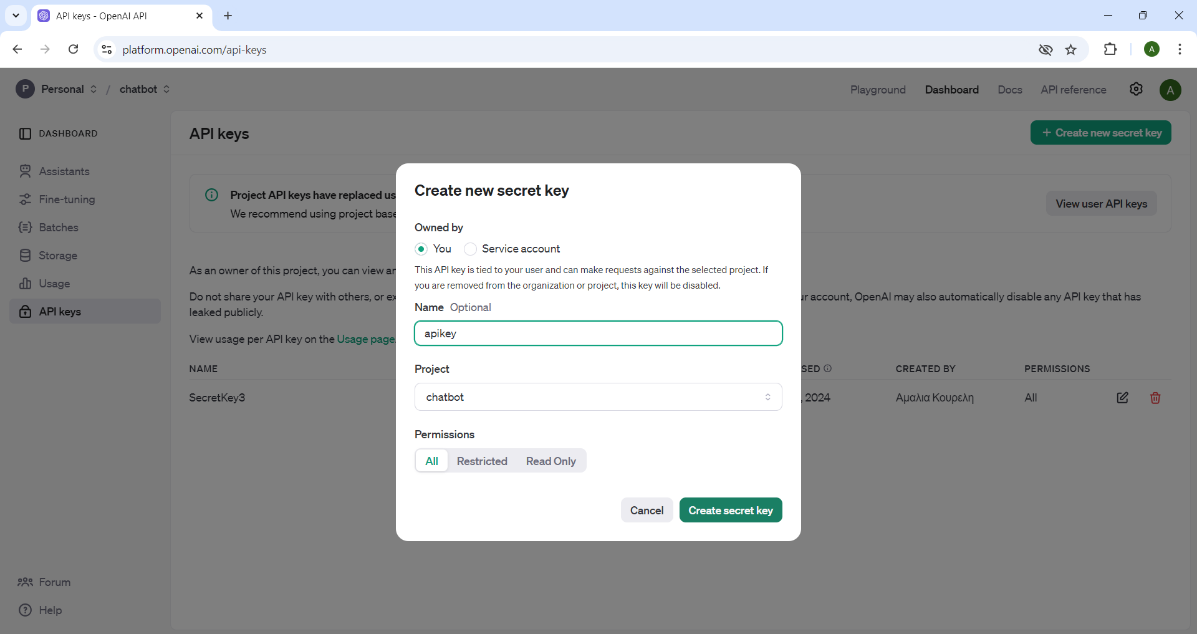
Βήματα δημιουργίας λογαριασμού στην OpenAI:

1. Επίσκεψη της σελίδας ([platform.openai.com/signup](https://platform.openai.com/signup)) και δημιουργία λογαριασμού ή σύνδεση σε υπάρχον.
2. Επαλήθευση στοιχείων (email και το τηλέφωνο).
3. Για την απόκτηση του API Key, ακολουθούν οι παρακάτω ενέργειες:

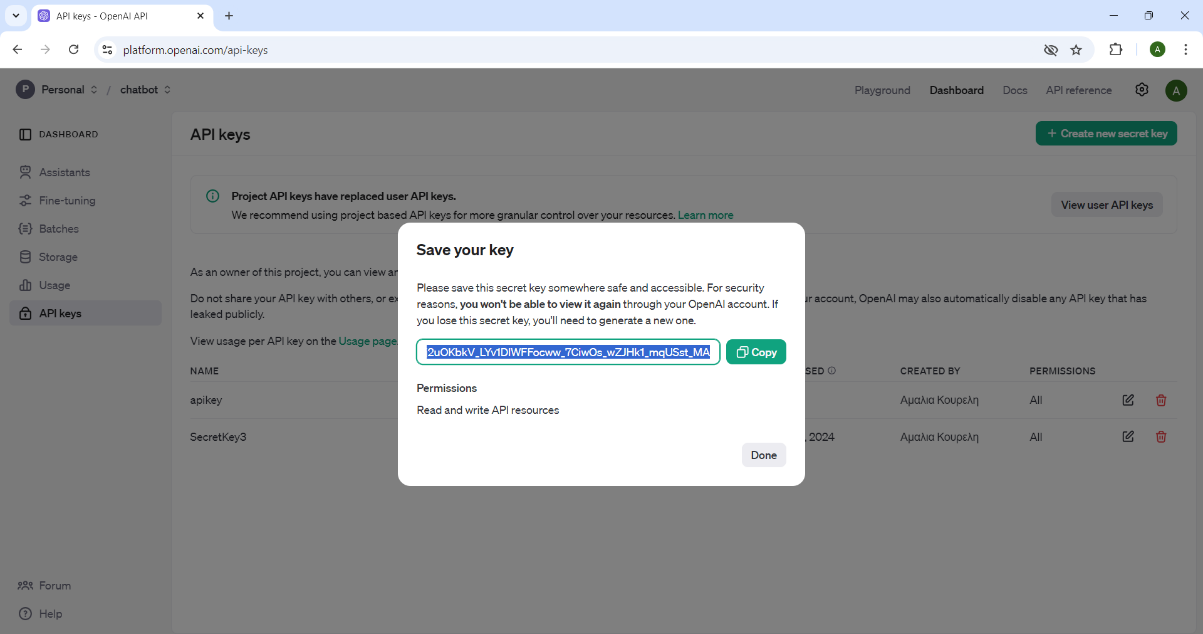
Τσεκάρεται το Dashboard επάνω δεξιά. Και από το αριστερό μενού, επιλέγεται το API Keys και έπειτα το Create New Secret Key.

  
**Εικόνα 44: Απόκτηση API Key (βήμα 1)**

Δίνεται όνομα στο κλειδί και τσεκάρεται το Create secret key.

  
**Εικόνα 45 :Απόκτηση API Key (βήμα 2)**

Αντιγράφεται το API Key και φυλάσσεται σε ασφαλές μέρος.

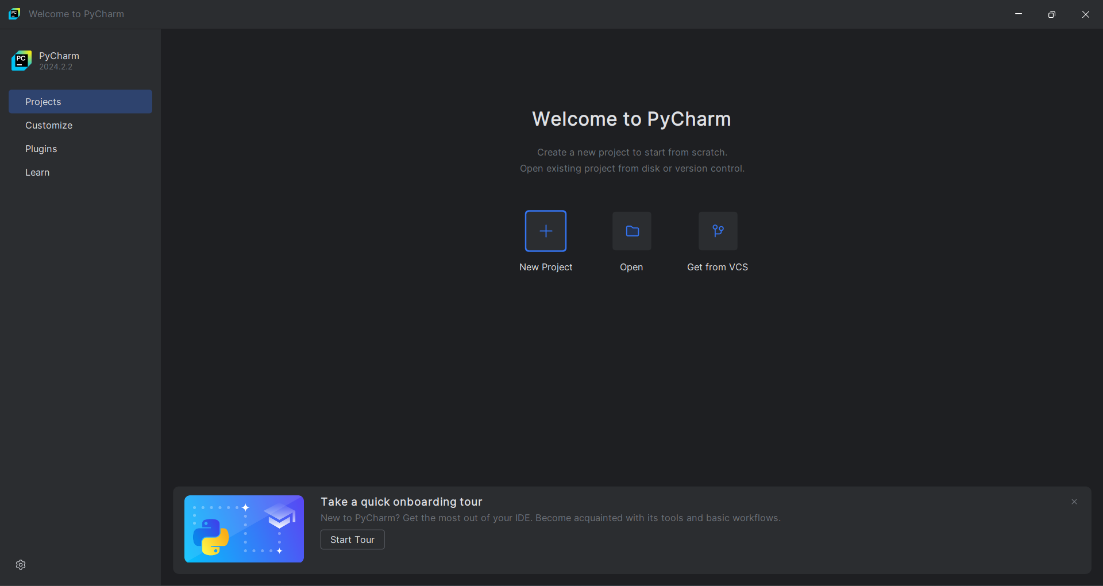
  
**Εικόνα 46: Απόκτηση API Key (Βήμα 3)**

4. Δημιουργία Νέου Project στο PyCharm.

Αφού έχουν εγκατασταθεί όλα τα απαραίτητα εργαλεία μπορεί να ξεκινήσει η ανάπτυξη του προγράμματος όπως φαίνεται και στις εικόνες.

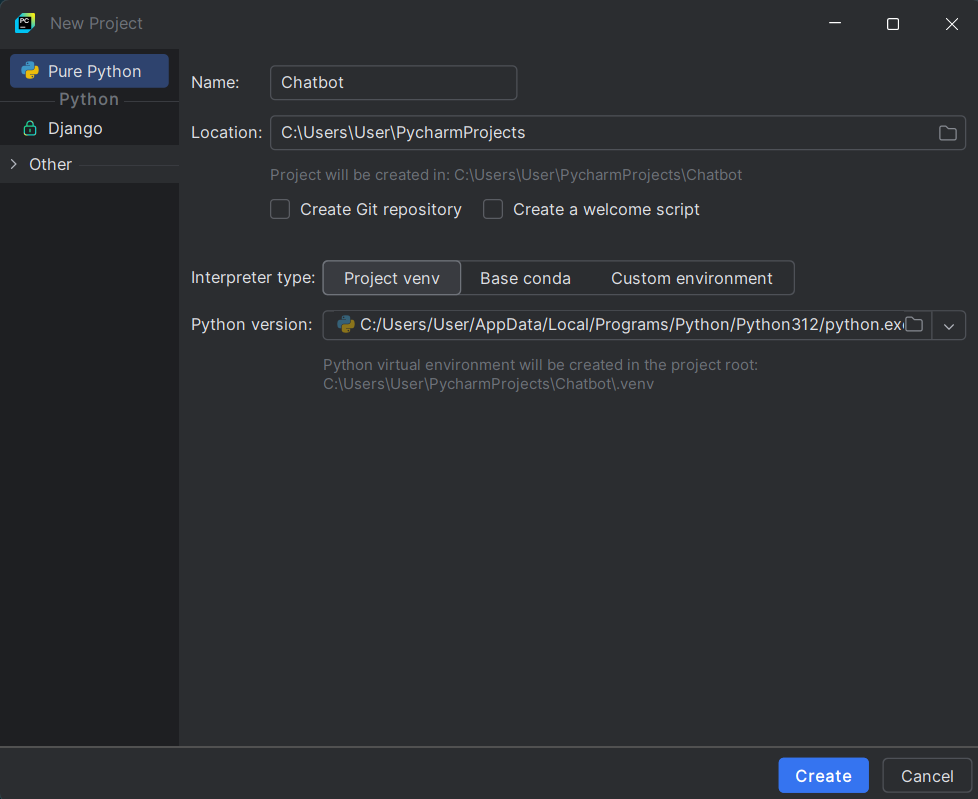
Βήματα δημιουργίας νέου Project στο PyCharm:

1. Άνοιγμα του PyCharm και δημιουργία New Project.



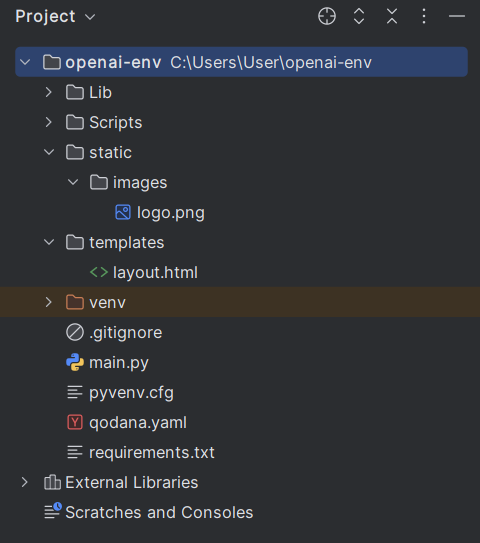
**Εικόνα 47:Δημιουργία Νέου Project στο PyCharm (βήμα 1).**

1. Επιλογή ονόματος, τοποθεσίας, interpreter, Python version και Create project.

  
**Εικόνα 48:Δημιουργία Νέου Project στο PyCharm (βήμα 2)**

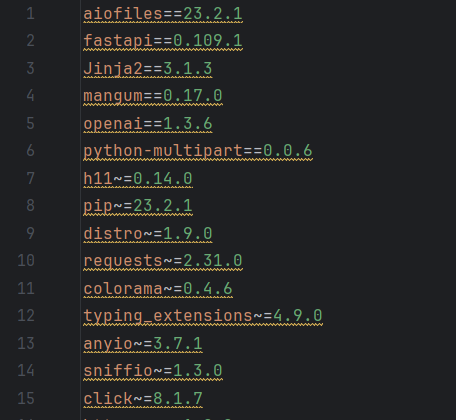
5. Δημιουργία Αρχείων.

Δημιουργία των κατάλληλων directories και αρχείων ώστε η δομή του project να είναι η ακόλουθη:

  
**Εικόνα 49: Δομή των αρχείων Project**

6.Εγκατάσταση Βιβλιοθηκών

Όπως φαίνεται στην παραπάνω εικόνα, υπάρχει ένα αρχείο με το όνομα requirements.txt, σε αυτό αναγράφονται όλες οι απαραίτητες βιβλιοθήκες που θα χρειαστούν για την ανάπτυξη της εφαρμογής οι οποίες φαίνονται στην παρακάτω εικόνα και για την εγκατάστασή τους εκτελείται στο terminal η εντολή: pip install requirements.txt

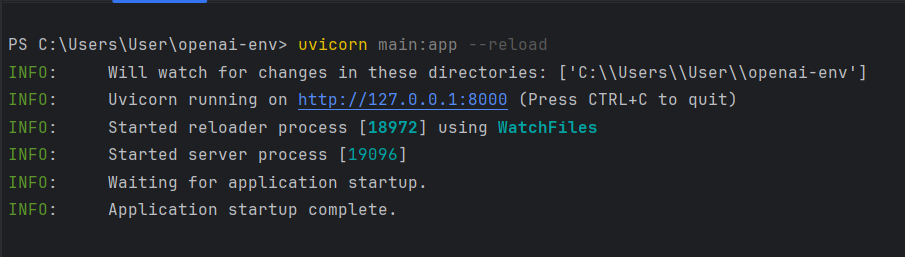
  
**Εικόνα 50: Απαραίτητες Βιβλιοθήκες για το Project**

7.Σύνταξη του Κώδικα

Στα αρχεία main.py και layout.html που δημιουργήθηκαν συντάσσεται ο κώδικας του project. Επίσης, στο directory template εντάσσεται το logo της εφαρμογής.

8.Εκτέλεση της Εφαρμογής

1. Για την εκτέλεση της εφαρμογής πληκτρολογείται στο terminal η εντολή (uvicorn main:app –reload).
2. Μόλις εμφανιστεί το μήνυμα (Application startup complete.) μπορεί να ανοίξει η εφαρμογή στην ηλεκτρονική διεύθυνση (<http://127.0.0.1:8000>)

  
**Εικόνα 51: Εκτέλεση της εφαρμογής**

# **Παράρτημα ΙΙ**

**Ο κώδικας που αναπτύχθηκε με σχόλια**

**main.py**

**# Εισαγωγές απαραίτητων βιβλιοθηκών.**

**# FastAPI για το API, Form και Request για φόρμες και αιτήματα.**

from fastapi import FastAPI, Form, Request

from typing import Annotated **# Annotated για τύπους δεδομένων (π.χ., στη φόρμα).**

from fastapi.templating import Jinja2Templates **# Jinja2Templates για τη διαχείριση templates.**

**# HTMLResponse για να επιστρέφουμε HTML απαντήσεις.**

from fastapi.responses import HTMLResponse

from fastapi.staticfiles import StaticFiles **# StaticFiles για τη διαχείριση στατικών αρχείων.**

from mangum import Mangum **# Mangum για υποστήριξη AWS Lambda εκτέλεσης.**

import requests **# requests για αποστολή HTTP αιτημάτων.**

from bs4 import BeautifulSoup **# BeautifulSoup για την εξαγωγή δεδομένων από HTML.**

from openai import OpenAI **# Βιβλιοθήκη OpenAI για πρόσβαση στο GPT μοντέλο.**

import uvicorn **# uvicorn για την εκκίνηση του FastAPI server.**

from datetime import datetime **# datetime για τη διαχείριση και μορφοποίηση ημερομηνιών.**

**# Ρυθμίσεις OpenAI κλειδιού API.**

client = OpenAI(api\_key="sk-BIONkgx4BAQF2yEMGevgRYzHA3Njw0dfCvo91Xfw1oT3BlbkFJOhQYmtxI9kydqBn56A1njrz0nywo1EbGs6vB3MywEA")

app = FastAPI() **# Δημιουργία FastAPI εφαρμογής.**

handler = Mangum(app) **# Ορίζουμε handler για AWS Lambda υποστήριξη με το Mangum.**

**# Ορισμός του φακέλου templates για HTML rendering με Jinja2.**

templates = Jinja2Templates(directory="templates")

**# Ορισμός του φακέλου στατικών αρχείων (π.χ., CSS, εικόνες).**

app.mount("/static", StaticFiles(directory="static"), name="static")

**# Αρχικό σύστημα, που περιέχει τις βασικές πληροφορίες και οδηγίες για το chatbot.**

chat\_log = [{

'role': 'system',

'content': 'Είσαι ένας ακαδημαϊκός βοηθός που παρέχει πληροφορίες '

'για μαθήματα, καθηγητές και διοικητικές υποθέσεις σχετικά με τις σπουδές '

'σε φοιτητές και μη, του Τμήματος Πληροφορικής του Δημοκρίτειου Πανεπιστημίου Θράκης.'

'Δώσε συγκεκριμένες και ακριβείς απαντήσεις, με επίκεντρο την ερώτηση.'

}]

**# Αρχικό μήνυμα του chatbot για το UI.**

chat\_responses = [

{

'message': "Γεια σας! Πως μπορώ να σας βοηθήσω;",

**# Φόρμα ημερομηνίας και ώρας.**

'timestamp': datetime.now().strftime("%Y-%m-%d %H:%M:%S")

}

]

**# Συνάρτηση για τη μορφοποίηση κειμένου που σχετίζεται με πληροφορίες επικοινωνίας #(σχολής/καθηγητών).**

def format\_as\_contact\_info(text: str):

**# Χωρίζει το κείμενο σε δύο μέρη με βάση την πρώτη αλλαγή γραμμής.**

parts = text.split("\n", 1)

if len(parts) < 2:

return text **# Επιστρέφει το αρχικό κείμενο αν δεν υπάρχει δεύτερο μέρος.**

introduction = parts[0].strip() **# Παίρνει την εισαγωγή του χρήστη.**

contact\_items = parts[1].split("\n") **# Διαχωρίζει τα στοιχεία επικοινωνίας.**

**# Καθαρισμός γραμμών.**

contact\_items = [line.replace("\*\*", "").strip() for line in contact\_items if line.strip()]

formatted\_intro = f"<p>{introduction}</p>" **# Μορφοποιεί την εισαγωγή ως παράγραφο.**

**# Μορφοποιεί τα στοιχεία επικοινωνίας.**

formatted\_contact\_info = "<p>" + "<br>".join(contact\_items) + "</p>"

return formatted\_intro + formatted\_contact\_info

**# Συνάρτηση μορφοποίησης κειμένου που σχετίζεται με τα μαθήματα του προγράμματος σπουδών # σε παράγραφο και λίστα.**

def format\_as\_paragraph\_and\_list(text: str):

parts = text.split("\n", 1)

if len(parts) < 2:

return text **# Επιστρέφει το αρχικό κείμενο αν δεν υπάρχει δεύτερο μέρος.**

paragraph = parts[0].strip() **# Παίρνει την εισαγωγή (παράγραφο).**

list\_items = parts[1].split("\n") **# Παίρνει τα στοιχεία της λίστας.**

**# Μορφοποιεί την απάντηση.**

formatted\_response = f"<p>{paragraph}</p><p>" + "<br>".join(line.replace("\*\*", "").strip() for line in list\_items if line.strip()) + "</p>"

return formatted\_response

**# Συνάρτηση καθαρισμού κειμένου ανακοινώσεων.**

def clean\_announcement\_text(text: str):

**# Αφαιρεί ειδικούς χαρακτήρες και κενά.**

cleaned\_text = text.replace("\*\*", "").replace("-", "").strip()

cleaned\_text = " ".join(cleaned\_text.split()) **# Συνενώνει πολλαπλά κενά.**

return cleaned\_text

**# Συνάρτηση επιλογής της κατάλληλης απάντησης του chatbot για την εκάστοτε ερώτηση.**

def format\_response(bot\_response: str):

**#Αν η ερώτηση περιέχει τις λέξεις "ανακοίνωση" ή "πτυχιακή", επιστρέφεται η απάντηση με την # κατάλληλη μορφοποίηση.**

if "ανακοίνωση" in bot\_response.lower() or "πτυχιακή" in bot\_response.lower():

cleaned\_response = clean\_announcement\_text(bot\_response)

else:

cleaned\_response = bot\_response **# Αλλιώς αφήνει το κείμενο ως έχει.**

**#Αν η ερώτηση περιέχει τη λέξη "επικοινωνίας", επιστρέφεται η απάντηση με την κατάλληλη # μορφοποίηση.**

if "επικοινωνίας" in cleaned\_response:

return format\_as\_contact\_info(cleaned\_response)

**#Αλλιώς αν η ερώτηση περιέχει τις λέξεις "Διδακτικές Μονάδες" ή "μαθήματα", επιστρέφεται η # απάντηση με την κατάλληλη #μορφοποίηση.**

elif "Διδακτικές Μονάδες" in cleaned\_response or "μαθήματα" in cleaned\_response:

return format\_as\_paragraph\_and\_list(cleaned\_response)

else:

return cleaned\_response **# Αλλιώς επιστρέφει το κείμενο χωρίς επιπλέον μορφοποίηση.**

**# Συνάρτηση που σαρώνει περιεχόμενο από τις ιστοσελίδες.**

def scrape\_website():

urls = [

"https://www.cs.ihu.gr/contact.xhtml",

"https://www.cs.ihu.gr/faculty.xhtml",

"https://www.cs.ihu.gr/special\_staff.xhtml",

"https://www.cs.ihu.gr/affiliate.xhtml",

"https://www.cs.ihu.gr/omotimoi.xhtml",

"https://www.cs.ihu.gr/announcements.xhtml",

"https://www.cs.ihu.gr/dissertation.xhtml",

"https://www.cs.ihu.gr/courses.xhtml"

]

content = "" **# Αρχικό κενό περιεχόμενο.**

for url in urls: **# Για κάθε URL στη λίστα.**

response = requests.get(url) **# Στέλνουμε HTTP αίτημα στη σελίδα.**

soup = BeautifulSoup(response.content, "html.parser") **# Ανάλυση της HTML σελίδας.**

content += soup.get\_text() + "\n\n" **# Προσθήκη του καθαρού κειμένου.**

return content

**# Route για την αρχική σελίδα του chat.**

@app.get("/", response\_class=HTMLResponse)

async def chat\_page(request: Request):

if len(chat\_responses) == 0: **# Αν δεν υπάρχουν απαντήσεις, προσθέτει μία.**

chat\_responses.append({

'message': "Γεια σας! Πως μπορώ να σας βοηθήσω;",

'timestamp': datetime.now().strftime("%Y-%m-%d %H:%M:%S")

})

return templates.TemplateResponse("layout.html", {"request": request, "chat\_responses": chat\_responses})

**# Route για την αποστολή και επεξεργασία απάντησης από τον chatbot.**

@app.post("/get-response", response\_class=HTMLResponse)

async def chat(request: Request, user\_input: Annotated[str, Form()]):

current\_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")  **# Τρέχουσα ώρα.**

**# Προσθήκη εισόδου χρήστη στο chat log.**

chat\_log.append({'role': 'user', 'content': user\_input})

website\_content = scrape\_website() **# Scraping περιεχομένου από τις σελίδες.**

**# Δημιουργία prompt με βάση το περιεχόμενο της εισόδου του χρήστη.**

prompt = f"{website\_content}\n\nUser: {user\_input}\nAssistant:"

try:

**# Κλήση στο GPT μοντέλο για να δημιουργήσει την απάντηση.**

response = client.chat.completions.create(

model="gpt-4o", **# Χρήση του GPT-4 μοντέλου.**

**# Προσθήκη του chat log.**

messages=chat\_log + [{'role': 'system', 'content': website\_content}],

temperature=0.5 **# Ελεγχόμενη δημιουργικότητα.**

)

bot\_response = response.choices[0].message.content **# Παίρνει την απάντηση του bot.**

formatted\_response = format\_response(bot\_response) **# Μορφοποίηση της απάντησης.**

except Exception as e:

**# Σε περίπτωση λάθους, επιστρέφει μήνυμα λάθους.**

formatted\_response = f"Sorry, there was an error processing your request: {e}"

**# Προσθήκη της απάντησης του chatbot στο chat log και στο UI.**

chat\_log.append({'role': 'assistant', 'content': formatted\_response})

chat\_responses.append({

'message': formatted\_response,

'timestamp': current\_time

})

return formatted\_response **# Επιστροφή της απάντησης.**

**# Εκκίνηση του FastAPI server.**

if \_\_name\_\_ == "\_\_main\_\_":

**# Εκκίνηση του server στη διεύθυνση 0.0.0.0 και θύρα 8000.**

uvicorn.run(app, host="0.0.0.0", port=8000)

**layout.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8"> **<!-- Ορίζει την κωδικοποίηση χαρακτήρων σε UTF-8 -->**

**<!-- Ρύθμιση του viewport για ορθή εμφάνιση σε κινητές συσκευές -->**

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Chatbot</title> **<!-- Τίτλος της σελίδας -->**

**<!—Χρήση CSS για την προσαρμογή του στυλ της ιστοσελίδας-->**

<style>

**/\* Γενικό στυλ για το σώμα της σελίδας \*/**

body {

font-family: Arial, sans-serif; **/\* Ορίζει τη γραμματοσειρά \*/**

margin: 0; **/\* Μηδενίζει τα περιθώρια \*/**

padding: 0; **/\* Μηδενίζει την απόσταση \*/**

background-color: #EAF2F8; **/\* Χρώμα φόντου \*/**

display: flex; **/\* Χρήση flexbox \*/**

flex-direction: column; **/\* Στοίχιση σε στήλες \*/**

align-items: center; **/\* Κεντρική στοίχιση περιεχομένου \*/**

**/\* Στυλ για το πάνω δεξιά μέρος της σελίδας \*/**

.top-right {

position: absolute; **/\* Απόλυτη τοποθέτηση \*/**

top: 10px; **/\* Απόσταση από την κορυφή \*/**

right: 280px; **/\* Απόσταση από τη δεξιά πλευρά \*/**

font-size: 1.5rem; **/\* Μέγεθος γραμματοσειράς \*/**

margin: 0; **/\* Καταργεί το προεπιλεγμένο περιθώριο \*/**

font-weight: bold; **/\* Έντονη γραφή \*/**

}

**/\* Στυλ για το header που περιέχει το λογότυπο και το όνομα του πανεπιστημίου \*/**

.header-container {

display: flex; **/\* Χρήση flexbox για διάταξη \*/**

align-items: center; **/\* Κεντράρισμα περιεχομένου \*/**

position: absolute; **/\* Απόλυτη τοποθέτηση \*/**

top: 130px; **/\* Απόσταση από την κορυφή \*/**

left: 100px; **/\* Απόσταση από την αριστερή πλευρά \*/**

}

**/\* Στυλ για το λογότυπο του πανεπιστημίου \*/**

.image-demo img {

margin-right: 20px;  **/\* Δεξί περιθώριο \*/**

height: 350px; **/\* Ύψος εικόνας \*/**

width: 300px; **/\* Πλάτος εικόνας \*/**

}

**/\* Στυλ για το όνομα του πανεπιστημίου \*/**

.university-name {

font-size: 1.8rem; **/\* Μέγεθος γραμματοσειράς \*/**

font-weight: bold; **/\* Έντονη γραφή \*/**

color: #000; **/\* Χρώμα κειμένου \*/**

display: flex; /\* **Χρήση flexbox για διάταξη \*/**

flex-direction: column; **/\* Καθορισμός διάταξης στηλών \*/**

align-items: left; **/\* Ευθυγράμμιση αριστερά \*/**

padding-bottom: 5px; **/\* Εσωτερικό περιθώριο κάτω \*/**

}

**/\* Στυλ για την μπλε γραμμή κάτω από το πανεπιστήμιο \*/**

.blue-line {

border-bottom: 10px solid #7FB3D5; **/\* Κάτω περίγραμμα \*/**

width: 100%; **/\* Πλάτος γραμμής \*/**

margin: 5px 0; **/\* Κάτω και πάνω περιθώριο \*/**

}

**/\* Στυλ για το όνομα του τμήματος \*/**

.department-name {

font-size: 1.1rem; **/\* Μέγεθος γραμματοσειράς \*/**

font-weight: bold; **/\* Έντονη γραφή \*/**

margin-top: 10px; **/\* Άνω περιθώριο \*/**

text-align: left; **/\* Ευθυγράμμιση αριστερά \*/**

}

**/\* Στυλ για το container του chat \*/**

.chat-container {

display: flex; **/\* Χρήση flexbox για διάταξη \*/**

flex-direction: column; **/\* Καθορισμός διάταξης στηλών \*/**

width: 40%; **/\* Πλάτος του chat \*/**

height: 65vh; **/\* Ύψος του chat \*/**

margin-top: 1rem; **/\* Άνω περιθώριο \*/**

background-color: #A9CCE3; **/\* Χρώμα φόντου \*/**

border-radius: 10px; **/\* Στρογγυλεμένες γωνίες \*/**

box-shadow: 0 3px 5px rgba(0, 0, 0, 0.2); **/\* Σκιά \*/**

padding: 20px; **/\* Εσωτερικό περιθώριο \*/**

position: absolute; **/\* Απόλυτη τοποθέτηση \*/**

top: 80px; **/\* Απόσταση από την κορυφή \*/**

right: 80px; **/\* Απόσταση από τη δεξιά πλευρά \*/**

font-size: 0.9rem; **/\* Μέγεθος γραμματοσειράς \*/**

}

**/\* Στυλ για το chat box \*/**

.chat-box {

flex-grow: 1; **/\* Επιτρέπει στο chat box να μεγαλώνει \*/**

overflow-y: auto; **/\* Κύλιση αν το περιεχόμενο ξεπερνά το ύψος \*/**

padding-bottom: 20px; **/\* Εσωτερικό περιθώριο κάτω \*/**

}

**/\* Στυλ για τα μηνύματα του chat \*/**

.chat-message {

display: flex; **/\* Χρήση flexbox για διάταξη \*/**

flex-direction: column; **/\* Καθορισμός διάταξης στηλών \*/**

align-items: flex-start; **/\* Ευθυγράμμιση αριστερά \*/**

margin-bottom: 10px; **/\* Κάτω περιθώριο \*/**

}

**/\* Στυλ για τα μηνύματα του χρήστη \*/**

.chat-message.user {

align-items: flex-end; **/\* Ευθυγράμμιση δεξιά \*/**

}

**/\* Στυλ για το περιεχόμενο των μηνυμάτων \*/**

.message-content {

background-color: #EAF2F8; **/\* Χρώμα φόντου του μηνύματος \*/**

color: black; **/\* Χρώμα κειμένου \*/**

padding: 10px; **/\* Εσωτερικό περιθώριο \*/**

border-radius: 10px; **/\* Στρογγυλεμένες γωνίες \*/**

max-width: 70%; **/\* Μέγιστο πλάτος \*/**

}

**/\* Στυλ για το περιεχόμενο των μηνυμάτων του χρήστη \*/**

.message-content.user {

background-color: #D4E6F1; **/\* Χρώμα φόντου για μηνύματα χρήστη \*/**

margin-right: 15px; **/\* Δεξί περιθώριο \*/**

}

**/\* Στυλ για τα timestamps των μηνυμάτων \*/**

.message-timestamp {

font-size: 0.7rem; **/\* Μέγεθος γραμματοσειράς \*/**

color: #555; **/\* Χρώμα κειμένου \*/**

margin-top: 5px; **/\* Άνω περιθώριο \*/**

text-align: left; **/\* Ευθυγράμμιση αριστερά \*/**

padding-left: 2%; **/\* Αριστερό εσωτερικό περιθώριο \*/**

}

**/\* Στυλ για το timestamp των μηνυμάτων του χρήστη \*/**

.chat-message.user .message-timestamp {

text-align: right; **/\* Ευθυγράμμιση δεξιά \*/**

padding-left: 0; **/\* Καταργεί το αριστερό εσωτερικό περιθώριο \*/**

padding-right: 4.5%; **/\* Δεξί εσωτερικό περιθώριο \*/**

}

**/\* Στυλ για τη φόρμα εισόδου \*/**

.input-form {

display: flex; **/\* Χρήση flexbox για διάταξη \*/**

width: 100%; **/\* Πλάτος φόρμας \*/**

align-items: center; **/\* Κεντράρισμα περιεχομένου \*/**

margin-top: 10px;  **/\* Άνω περιθώριο \*/**

}

**/\* Στυλ για το πεδίο κειμένου εισόδου \*/**

textarea {

width: 85%; **/\* Πλάτος textarea \*/**

padding: 10px; **/\* Εσωτερικό περιθώριο \*/**

border: 1px solid #D4E6F1; **/\* Όριο \*/**

border-radius: 5px; **/\* Στρογγυλεμένες γωνίες \*/**

resize: vertical; **/\* Επιτρέπει την κατακόρυφη αλλαγή μεγέθους \*/**

background-color: #EAF2F8; **/\* Χρώμα φόντου \*/**

color: black; **/\* Χρώμα κειμένου \*/**

}

**/\* Στυλ για το κουμπί \*/**

.btn {

background-color: #EAF2F8; **/\* Χρώμα φόντου κουμπιού \*/**

color: black; **/\* Χρώμα κειμένου \*/**

padding: 10px 10px; **/\* Εσωτερικό περιθώριο \*/**

border: none; **/\* Χωρίς όριο \*/**

border-radius: 5px; **/\* Στρογγυλεμένες γωνίες \*/**

cursor: pointer; **/\* Εμφάνιση χεριού κατά το hover \*/**

margin-left: 10px; **/\* Αριστερό περιθώριο \*/**

}

**/\* Στυλ για το κουμπί αποστολής \*/**

.btn-primary {

background-color: #EAF2F8; **/\* Χρώμα φόντου για το κουμπί αποστολής \*/**

}

**/\* Στυλ για τις αριθμημένες λίστες \*/**

ol {

margin-left: 20px; **/\* Αριστερό περιθώριο \*/**

padding-left: 20px; **/\* Αριστερό εσωτερικό περιθώριο \*/**

}

**/\* Στυλ για τα στοιχεία λίστας \*/**

li {

margin-bottom: 10px; **/\* Κάτω περιθώριο \*/**

}

</style>

</head>

<body>

<div class="top-right">

<h4>Ακαδημαϊκός Βοηθός</h4> **<!-- Τίτλος του chatbot -->**

</div>

<div class="header-container">

<div class="image-demo">

**<!-- Εικόνα λογοτύπου πανεπιστημίου -->**

<img src="{{ url\_for('static', path='images/logo.png') }}" alt="Logo" height="100">

</div>

<div class="university-name">

<div>ΔΗΜΟΚΡΙΤΕΙΟ</div>

<div>ΠΑΝΕΠΙΣΤΗΜΙΟ</div>

<div>ΘΡΑΚΗΣ</div>

<div class="blue-line"></div>

<div class="department-name">

<div>ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ</div>

<div>ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ</div>

</div>

</div>

</div>

<div class="chat-container"> **<!-- Container για το chat -->**

<div class="chat-box"> **<!-- Πλαίσιο εμφάνισης μηνυμάτων -->**

**<!-- Επανάληψη για κάθε απάντηση του chatbot -->**

{% for response in chat\_responses %}

**<!-- Ορισμός της κλάσης για το μήνυμα -->**

<div class="chat-message {% if loop.index % 2 == 0 %}user{% endif %}">

**<!-- Ορισμός της κλάσης για το περιεχόμενο του μηνύματος -->**

<div class="message-content {% if loop.index % 2 == 0 %}user{% endif %}">

{{ response.message | safe }} **<!-- Εμφάνιση του μηνύματος -->**

</div>

<div class="message-timestamp"> **<!-- Περιοχή εμφάνισης timestamp -->**

{{ response.timestamp }} **<!-- Εμφάνιση της ημερομηνίας και ώρας -->**

</div>

</div>

{% endfor %} **<!-- Τέλος επανάληψης -->**

</div>

<div id="loading-message" style="display: none; color: #555; margin-top: 10px;">

Γίνεται αναζήτηση της απάντησης... **<!-- Μήνυμα φόρτωσης της απάντησης -->**

</div>

<form class="input-form" method="post"> **<!-- Φόρμα για την είσοδο του χρήστη -->**

<textarea name="user\_input" placeholder="Πληκτρολογήστε εδώ!" rows="3"></textarea> **<!-- Πεδίο κειμένου για εισαγωγή μηνύματος -->**

**<!-- Κουμπί αποστολής -->**

<button class="btn btn-primary" type="submit">Αποστολή</button>

</form>

</div>

<script>

**// Χρήση JavaScript για ρυθμίσεις υποβολής της φόρμας**

document.querySelector('form').addEventListener('submit', async function(event) {

event.preventDefault(); **// Αποτροπή της προεπιλεγμένης υποβολής φόρμας**

**// Αποθήκευση της εισόδου του χρήστη**

let userInput = document.querySelector('textarea').value;

if (!userInput.trim()) return; **// Έλεγχος αν το πεδίο είναι κενό**

const chatBox = document.querySelector('.chat-box'); **// Αναφορά στο chat box**

**// Λήψη τρέχουσας ημερομηνίας και ώρας**

const currentTime = new Date().toLocaleString();

**// Δημιουργία του μηνύματος του χρήστη**

const userMessage = `

<div class="chat-message user">

<div class="message-content user">${userInput}</div>

<div class="message-timestamp">${currentTime}</div>

</div>

`;

**// Προσθήκη του μηνύματος του χρήστη στο chat box**

chatBox.innerHTML += userMessage;

**// Εκκαθάριση του πεδίου κειμένου εισόδου**

document.querySelector('textarea').value = '';

**// Αναφορά στο μήνυμα φόρτωσης**

const loadingMessage = document.getElementById('loading-message');

loadingMessage.style.display = 'block'; **// Εμφάνιση του μηνύματος φόρτωσης**

const formData = new FormData(); **// Δημιουργία νέας φόρμας δεδομένων**

formData.append('user\_input', userInput); **// Προσθήκη της εισόδου του χρήστη**

try {

**// Αποστολή αιτήματος για λήψη απάντησης από το server**

const response = await fetch('/get-response', {

method: 'POST',

body: formData **// Σώμα αιτήματος με δεδομένα χρήστη**

});

if (!response.ok) {

throw new Error('Network response was not ok'); **// Έλεγχος για σφάλμα δικτύου**

}

const botResponse = await response.text(); **// Λήψη της απάντησης του chatbot**

loadingMessage.style.display = 'none'; **// Απόκρυψη του μηνύματος φόρτωσης**

**// Δημιουργία του μηνύματος του chatbot**

const botMessage = `

<div class="chat-message">

<div class="message-content">${botResponse}</div>

<div class="message-timestamp">${new Date().toLocaleString()}</div>

</div>

`;

**// Προσθήκη του μηνύματος του chatbot στο chat box**

chatBox.innerHTML += botMessage;

chatBox.scrollTop = chatBox.scrollHeight; **// Κύλιση στο κάτω μέρος του chat box**

} catch (error) {

console.error('Error:', error); **// Καταγραφή σφάλματος**

loadingMessage.style.display = 'none'; **// Απόκρυψη του μηνύματος φόρτωσης**

}

});

</script>

</body>

</html>