ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Ανάπτυξη Εφαρμογής Ευφυούς Διαλογικού Πράκτορα»

Του φοιτητή
Βραχά Χαρίλαου
Αρ. Μητρώου: 185157

Επιβλέπων
Ονοματεπώνυμο Δημοσθένης Σταμάτης
Βαθμίδα ……

Ημερομηνία ……

Τίτλος Δ.Ε. ….
Κωδικός Δ.Ε. …
Ονοματεπώνυμο φοιτητή/τών ….
Ονοματεπώνυμο εισηγητή …
Ημερομηνία ανάληψης Δ.Ε. …
Ημερομηνία περάτωσης Δ.Ε. …

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

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία τ__ φοιτητ__ _____ που την εκπόνησε/αν. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης, ο συγγραφέας/δημιουργός εκχωρεί στο Διεθνές Πανεπιστήμιο της Ελλάδος άδεια χρήσης του δικαιώματος αναπαραγωγής, δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσης της εργασίας διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος. Η ανοικτή πρόσβαση στο πλήρες κείμενο της εργασίας, δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού, ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, πώληση, εμπορική χρήση, διανομή, έκδοση, μεταφόρτωση (downloading), ανάρτηση (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

«Στην Αγαπημένη μου

Και στην Οικογένεια μου»

Πρόλογος

Η παρούσα πτυχιακή εργασία σηματοδοτεί μια σημαντική προσπάθεια να αναδείξει τη σημασία της τεχνητής νοημοσύνης και των τεχνητών νευρωνικών δικτύων στον σύγχρονο κόσμο. Σε μια εποχή όπου η τεχνολογία εξελίσσεται με αστρονομική ταχύτητα και τα δεδομένα διογκώνονται αδιάκοπα, αυτή η εργασία αναλύει πώς οι τεχνητές νευρωνικών δικτύων μπορούν να λειτουργήσουν ως ισχυρό εργαλείο για την εξαγωγή σημαντικών πληροφοριών από τα δεδομένα αυτά.

Αυτή η εργασία δεν είναι μόνο μια τεχνική ανάλυση. Είναι μια αναζήτηση για την κατανόηση της δύναμης της ανθρώπινης δημιουργίας, μια προσπάθεια να κατανοήσουμε πώς μπορούμε να διαμορφώσουμε το μέλλον. Αναδεικνύει πώς οι ανθρώπινες δυνατότητες συνδυάζονται με την τεχνολογία για να δημιουργήσουν καινοτόμες λύσεις σε προβλήματα που καλούνται να αντιμετωπίσουν η κοινωνία και η βιομηχανία.

Στην πορεία της εργασίας, θα ανακαλύψετε πώς οι τεχνητές νευρωνικές δικτυακές δομές μπορούν να αποτελέσουν ένα κλειδί για την ανάπτυξη νέων προϊόντων και υπηρεσιών. Ελπίζουμε ότι αυτή η εργασία θα εμπνεύσει και θα κινητοποιήσει τους αναγνώστες να εξερευνήσουν περισσότερο αυτόν το συναρπαστικό και δυναμικό κόσμο.

Ευχόμαστε καλή ανάγνωση και ελπίζουμε ότι αυτή η εργασία θα συμβάλει στην εμπλούτιση της γνώσης και της σκέψης σας σε αυτό τον συναρπαστικό τομέα.

Με εκτίμηση,

Χαρίλαος Βραχάς

Περίληψη

Η παρούσα πτυχιακή εργασία αναλύει την εφαρμογή των τεχνητών νευρωνικών δικτύων και της τεχνητής νοημοσύνης στη δημιουργία ενός διαλογικού πράκτορα *Chatbot*. Επίσης συγκρίνει τις τρέχουσες καταστάσεις *(state of the art)* των εργαλείων ανάπτυξης ευφυών διαλογικών πρακτόρων.

Η εργασία αναλύει τις βασικές αρχές και τεχνικές που απαιτούνται για την ανάπτυξη ενός αποτελεσματικού *Chatbot*, από την αναγνώριση του λόγου και την επεξεργασία του φυσικού γλωσσικού, μέχρι την αυτόματη απόκτηση γνώσης και την απόκριση σε ερωτήσεις.

Η εργασία παρουσιάζει επίσης πρακτικά παραδείγματα εφαρμογής, καθώς και μια διεξοδική ανάλυση των προκλήσεων και των ευκαιριών στον τομέα της τεχνητής νοημοσύνης στον κόσμο των *Chatbots*. Επιπλέον, προτείνονται μελλοντικές επεκτάσεις και βελτιώσεις στην εφαρμογή, όπως η πολυγλωσσική υποστήριξη, η ενσωμάτωση προηγμένης τεχνητής νοημοσύνης και μέτρα ασφαλείας και απορρήτου.

Αυτή η πτυχιακή εργασία προσφέρει μια εις βάθος κατανόηση των τεχνολογιών που βρίσκονται πίσω από τους διαλογικούς πράκτορες και τους *Chatbots* και παρέχει μια βάση για την περαιτέρω έρευνα και ανάπτυξη σε αυτόν τον αναπτυσσόμενο τομέα.

Ελπίζουμε ότι αυτή η εργασία θα προσφέρει χρήσιμες εισηγήσεις και ενδιαφέρουσες προοπτικές για όσους ενδιαφέρονται να εξερευνήσουν τη σχέση μεταξύ τεχνητής νοημοσύνης και διαλογικών πρακτόρων στον κόσμο της τεχνολογίας.

Εύχομαστε καλή ανάγνωση!

Development of an Intelligent Conversational Agent application

Charilaos Vrachas

Abstract

The present thesis analyzes the application of artificial neural networks and artificial intelligence in creating a conversational agent, Chatbot. It also compares the current state of development tools for intelligent conversational agents.

The work explores the fundamental principles and techniques required for developing an effective Chatbot, from speech recognition and natural language processing to automatic knowledge acquisition and responding to questions.

Additionally, practical application examples are presented, along with a detailed analysis of challenges and opportunities in the field of artificial intelligence in the realm of Chatbots. Future expansions and improvements in the application are proposed, such as multilingual support, integration of advanced artificial intelligence, and security and privacy measures.

This thesis provides an in-depth understanding of the technologies behind conversational agents and Chatbots, laying the groundwork for further research and development in this evolving field.

We hope that this work will offer useful insights and interesting perspectives for those interested in exploring the relationship between artificial intelligence and conversational agents in the world of technology.

We wish you an enjoyable reading experience!

Ευχαριστίες

Η παρούσα πτυχιακή εργασία πραγματοποιήθηκε στα πλαίσια των σπουδών μου στο Διεθνές Πανεπιστήμιο της Ελλάδος και συγκεκριμένα στο τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων κατά το έτος 2023.

Η ολοκλήρωση της πτυχιακής αυτής εργασίας δεν θα μπορούσε να είχε ολοκληρωθεί χωρίς της πολύτιμη στήριξη, καθώς και συνεχή επικοινωνία, του επιβλέποντα καθηγητή μου, Καθηγητής ΔΙΠΑΕ, Κο Δημοσθένη Σταμάτη. Εκφράσω ένα τεράστιο ευχαριστώ για όλη την καθοδήγηση που μου προσέφερε, καθώς και την εμπιστοσύνη που μου έδειξε για να αναλάβω το θέμα.

Ευχαριστώ πολύ από καρδίας την κοπέλα μου, η οποία δεν έπαψε ποτέ να με στηρίζει σε αυτή την δύσκολη προσπάθεια της ολοκλήρωσης των σπουδών μου, και της ολοκλήρωσης της πτυχιακής εργασίας.

Τέλος, θέλω να ευχαριστήσω την οικογένειά μου η οποία με στήριξε και ψυχολογικά και οικονομικά και τους οποίους τους ευχαριστώ πολύ.

Περιεχόμενα

Πρόλογος v
Περίληψη vi
Abstract vii
Ευχαριστίες viii
Περιεχόμενα ix
Κατάλογος Σχημάτων xii
Κατάλογος Πινάκων xii
Συντομογραφίες xiii
Κεφάλαιο 1ο: Εισαγωγή 1
1.1 Εισαγωγή στην Ανάπτυξη Ευφυών Διαλογικών Πρακτόρων 1
1.2 Σκοπός και Στόχοι της Πτυχιακής Εργασίας 2
1.3 Μελέτη Περίπτωσης ενός Διαλογικού Πράκτορα - Βοηθό Μαθήματος 2
1.3.1 Σχεδιασμός του Διαλογικού Πράκτορα 3
1.3.2 Λειτουργίες του Διαλογικού Πράκτορα 3
1.3.3 Αλληλεπίδραση με τον Χρήστη 3
1.3.4 Αξιολόγηση και Βελτίωση 4
1.4 Μέθοδος και Προσέγγιση 4
1.5 Επίλογος 5
Κεφάλαιο 2ο: Θεωρητικό Υπόβαθρο 7
2.1 Εισαγωγή στην Τεχνητή Νοημοσύνη 7
2.2 Μηχανική Μάθηση 8
2.2.1 Επιβλεπόμενη (Supervised Learning) 8
2.2.2 Μη Επιβλεπόμενη (Unsupervised Learning) 9
2.2.3 Ημι-Επιβλεπόμενη (Semi-Supervised Learning) 9
2.2.4 Ενισχυμένη (Reinforcement Learning) 10
2.3 Βαθιά Μηχανική Μάθηση 10
2.3.1 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks CNNs) 11
2.3.2 Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Recurrent Neural Networks RNNs) 11
2.3.3 Μετασχηματιστής (Transformer) 12
2.4 Νευρωνικά Δίκτυα 12
2.5 Επίλογος 13

Κεφάλαιο 3ο: Συστηματική Συγκριτική Μελέτη των Εργαλείων Ανάπτυξης Ευφυών Διαλογικών Πρακτόρων 14
3.1 Εργαλεία Ανάπτυξης Ευφυών Διαλογικών Πρακτόρων 14
3.1.1 DialogFlow 14
3.1.2 Microsoft Bot Framework 15
3.1.3 Rasa 15
3.2 Συγκριτική Ανάλυση 16
3.2.1 Εισαγωγή Χρήστη 18
3.2.2 Διάλογος 18
3.2.3 Ανάπτυξη 19
3.2.4 Διεπαφή 19
3.2.5 Ανάπτυξη και Δοκιμές 19
3.2.6 Εκτέλεση 20
3.2.7 Ασφάλεια 20
3.3 Επίλογος 21
Κεφάλαιο 4ο: Επιλογή Πλατφόρμας Ανάπτυξης 21
4.1 Εξέταση της Πλατφόρμας DialogFlow της Google 22
4.1.1 Εισαγωγή στο DialogFlow 22
4.1.2 Δυνατότητες του DialogFlow 23
4.2 Ασφάλεια και Προστασία Δεδομένων στο DialogFlow 28
4.2.1 Διαχείριση Δεδομένων Χρηστών 28
4.2.2 Προστασία Προσωπικών Δεδομένων 29
4.2.3 Κρυπτογράφηση Δεδομένων 29
4.3 Επίλογος 30
Κεφάλαιο 5ο: Σχεδιασμός της Δομής Διαλόγου 30
5.1 Ορισμός των Λειτουργικών Απαιτήσεων για τον Διαλογικό Πράκτορα-Βοηθό Μαθήματος 31
5.2 Σχεδιασμός του Διαλόγου και των Πιθανών Περιπτώσεων Χρήσης 31
5.3 Επίλογος 32
Κεφάλαιο 6ο: Υλοποίηση της Εφαρμογής 34
6.1 Δημιουργία Βάσης Δεδομένων 34
6.1.1 Αυτοματοποιημένη Εισαγωγή Δεδομένων 35
6.2 Ανάπτυξη του Server με FastAPI και Python 39
6.2.1 FastAPI 39
6.2.2 Εξωτερική Πρόσβαση με το ngrok 40

6.2.3 Python 41
6.3 Υλοποίηση των Λειτουργιών του Πράκτορα-Βοηθού Μαθήματος 44
6.4 Κριτήρια Αξιολόγησης της Απόδοσης του Πράκτορα 46
6.5 Επίλογος 48
Κεφάλαιο 7ο: Συμπεράσματα και Μελλοντική Επέκταση 48
7.1 Συνοψίζοντας τα Κύρια Αποτελέσματα και τα Συμπεράσματα 48
7.2 Προτάσεις για Μελλοντική Επέκταση και Βελτίωση της Εφαρμογής 49
7.3 Επίλογος 51
ΒΙΒΛΙΟΓΡΑΦΙΑ 52
ΠΑΡΑΡΤΗΜΑ Α : ΚΩΔΙΚΑΣ SCRAPPER 53
ΠΑΡΑΡΤΗΜΑ Β : ΚΩΔΙΚΑΣ PYTHON 55
ΠΑΡΑΡΤΗΜΑ C : ΚΩΔΙΚΑΣ ENTITY - SUBJECT 56
ΠΑΡΑΡΤΗΜΑ D : ΚΩΔΙΚΑΣ ENTITY - INFORMATION 57

Κατάλογος Σχημάτων

Εικόνα 2.1: Τεχνολογίες Τεχνιτής Νοημοσύνης.....**Error! Bookmark not defined.**
Εικόνα 4.1.2.1: DialogFlow 23
Εικόνα 6.1.1.1: Python Scrapping (1) 36
Εικόνα 6.1.1.2: Python Scrapping (2) 37
Εικόνα 6.1.1.3: Python Scrapping (3) 38
Εικόνα 6.2.3.1: Python Main (1) 41
Εικόνα 6.2.3.2: Python Main (2) 43

Κατάλογος Πινάκων

Πίνακας 3.3.1: Σύγκριση Εργαλείων Ανάπτυξης Διαλογικών Πρακτόρων..... 17

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
ΤΝ	Τεχνητή Νοημοσύνη
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
NN	Neural Networks
CNN	Convolutional Neural Network
RNN	Recurrent Neural Networks
NLP	Natural Language Processing

Κεφάλαιο 1ο: Εισαγωγή

Στο πρώτο κεφάλαιο της πτυχιακής εργασίας παρουσιάζεται η εισαγωγή, με στόχο να προσφέρει μια γενική εικόνα του θέματος που θα διερευνηθεί στην εργασία, καθώς και του σκοπού και των στόχων που έχουν τεθεί.

1.1 Εισαγωγή στην Ανάπτυξη Ευφυών Διαλογικών Πρακτόρων

Οι τελευταίες δεκαετίες έχουν μαρτυρήσει μια επανάσταση στον τομέα της τεχνολογίας, με ιδιαίτερη έμφαση στην ανάπτυξη των ευφυών διαλογικών πρακτόρων. Οι διαλογικοί πράκτορες, επίσης γνωστοί ως *Chatbots* ή *Conversational Agents*, αποτελούν σημαντικό μέρος της τεχνητής νοημοσύνης και έχουν καταφέρει να αλλάξουν τον τρόπο με τον οποίο αλληλεπιδρούμε με την τεχνολογία και τα συστήματα πληροφορικής.

Οι διαλογικοί πράκτορες είναι λογισμικά που έχουν τη δυνατότητα να αλληλεπιδρούν με ανθρώπους μέσω φυσικής γλώσσας. Αυτό σημαίνει ότι οι χρήστες μπορούν να επικοινωνούν με αυτούς όπως θα κάναν με έναν ανθρώπινο συνομιλητή, χρησιμοποιώντας φυσική γλώσσα και λεκτικές εκφράσεις. Αυτό έχει επιφέρει τον επαναπροσδιορισμό της επικοινωνίας μεταξύ του ανθρώπου και της τεχνολογίας, ανοίγοντας νέες προοπτικές και ευκαιρίες σε διάφορους τομείς.

Ο κύριος λόγος που η ανάπτυξη ευφυών διαλογικών πρακτόρων έχει καταφέρει να κεντρίσει το ενδιαφέρον τόσο της ακαδημαϊκής όσο και της βιομηχανικής κοινότητας είναι η δυνατότητα αυτών των πρακτόρων να παρέχουν αποτελεσματικές λύσεις σε ποικίλα προβλήματα και εργασίες. Αναφορικά με την εκπαίδευση, οι διαλογικοί πράκτορες μπορούν να λειτουργήσουν ως βοηθοί, παρέχοντας πληροφορίες, διαλέξεις και εξηγήσεις για διάφορα μαθήματα και εκπαιδευτικά θέματα.

Για να επιτευχθεί αυτό, η παρούσα πτυχιακή εργασία προσεγγίζει με συστηματικό τρόπο την ανάπτυξη ευφυών διαλογικών πρακτόρων. Θα πραγματοποιηθεί εξονυχιστική έρευνα και ανάλυση των τρεχουσών τάσεων, τεχνολογιών και εργαλείων για την ανάπτυξή τους. Ταυτόχρονα, θα παρουσιαστεί μια μελέτη περίπτωσης, επικεντρώνοντας στην ανάπτυξη ενός διαλογικού πράκτορα που θα λειτουργεί ως βοηθός μαθήματος. Για την υλοποίηση αυτής της μελέτης περίπτωσης, θα χρησιμοποιηθεί η πλατφόρμα ανάπτυξης διαλογικών πρακτόρων *Dialogflow* της *Google*, η οποία παρέχει εκτεταμένες δυνατότητες στον τομέα της κατανόησης φυσικής γλώσσας και της διαχείρισης διαλόγων.

Με την παρούσα εργασία, αναζητούμε να κατανοήσουμε τη σημασία και τον ρόλο των ευφυών διαλογικών πρακτόρων στην σύγχρονη τεχνολογική εξέλιξη, εξετάζοντας τις δυνατότητες τους να βοηθούν σε διάφορους τομείς, με έμφαση στην εκπαίδευση και την αλληλεπίδραση με τους χρήστες.

1.2 Σκοπός και Στόχοι της Πτυχιακής Εργασίας

Ένας από τους στόχους της πτυχιακής εργασίας είναι να εξεταστεί η ανάπτυξη ευφυών διαλογικών πρακτόρων, με έμφαση στη χρήση τεχνικών μηχανικής μάθησης. Η εργασία περιλαμβάνει τη μελέτη περίπτωσης ενός διαλογικού πράκτορα-βοηθού μαθήματος.

Ένας ακόμη στόχος αυτής της εργασίας είναι η συστηματική σύγκριση εργαλείων ανάπτυξης, και ο σχεδιασμός και η υλοποίηση ενός τέτοιου πράκτορα. Τα συμπεράσματα και οι προοπτικές για μελλοντική επέκταση αποτελούν επίσης σημαντικό στόχος αυτής της εργασίας.

Για την επίτευξη των παραπάνω στόχων, η πτυχιακή εργασία έχει καθορίσει τις ακόλουθες συγκεκριμένες στρατηγικές στόχους:

- **Ανάλυση των Τρεχουσών Τάσεων:** Επιδιώκουμε να αναλύσουμε εκτενώς την εξέλιξη των διαλογικών πρακτόρων τα τελευταία χρόνια. Θα εξετάσουμε τις πρόσφατες τάσεις και τις τεχνολογικές προόδους που έχουν διαμορφώσει τη λειτουργία και τις δυνατότητες αυτών των πρακτόρων. Θα αναλύσουμε τα διάφορα μοντέλα μηχανικής μάθησης και επεξεργασίας φυσικής γλώσσας που χρησιμοποιούνται, παρέχοντας μια εικόνα της τρέχουσας κατάστασης στον τομέα.

- **Ανάπτυξη Εφαρμογής Μελέτης Περίπτωσης:** Σε αυτή τη φάση, θα αναλάβουμε τον σχεδιασμό και την υλοποίηση ενός διαλογικού πράκτορα που θα λειτουργεί ως βοηθός μαθήματος. Θα αναλύσουμε λεπτομερώς τη δομή και τις λειτουργίες της εφαρμογής. Επικεντρωνόμαστε στην αναγνώριση προθέσεων των χρηστών και την αποτελεσματική ανταπόκριση σε διαφορετικά σενάρια αλληλεπίδρασης.

- **Αξιολόγηση Απόδοσης:** Σε αυτό το στάδιο, προτιθέμεθα να προτείνουμε πιθανούς τρόπους αξιολόγησης της απόδοσης του διαλογικού πράκτορα. Αυτό περιλαμβάνει κάποιες προτάσεις για μετρικές όπως η ακρίβεια και η κατανόηση των προθέσεων των χρηστών, οι οποίες θα ήταν χρήσιμες για την εγκυρότητα του διαλογικού πράκτορα.

Οι στόχοι αυτοί θα υλοποιηθούν με τη χρήση μιας εκτενούς μεθοδολογίας που περιλαμβάνει την συλλογή και ανάλυση βιβλιογραφίας, τον σχεδιασμό και την υλοποίηση της εφαρμογής μελέτης περίπτωσης και την παράθεση τρόπων για την αξιολόγηση της απόδοσης του διαλογικού πράκτορα.

1.3 Μελέτη Περίπτωσης ενός Διαλογικού Πράκτορα - Βοηθό Μαθήματος

Η παρούσα υποενότητα αποτελεί τον πυρήνα της πτυχιακής εργασίας, αναλύοντας λεπτομερώς τη μελέτη περίπτωσης ενός διαλογικού πράκτορα που εκτελεί τον ρόλο του βοηθού μαθήματος. Στο επίκεντρο της προσομοίωσης βρίσκεται η ενσωμάτωση προηγμένων τεχνικών διαλόγου και τεχνητής νοημοσύνης σε ένα εκπαιδευτικό πλαίσιο, επιτρέποντας την αποτελεσματική εκμάθηση και αλληλεπίδραση των χρηστών.

1.3.1 Σχεδιασμός του Διαλογικού Πράκτορα

Ο σχεδιασμός ενός διαλογικού πράκτορα είναι ουσιώδης για την επιτυχία της εφαρμογής. Στο πλαίσιο της μελέτης περίπτωσης, αναλύουμε τον σχεδιασμό του πράκτορα που βασίζεται στην πλατφόρμα *Dialogflow* της *Google*. Εκμεταλλευόμενοι τις προηγμένες δυνατότητες της πλατφόρμας όπως θα αναλύσουμε σε επόμενη ενότητα, ο πράκτορας θα αναγνωρίζει και θα απαντά σε προθέσεις, ερωτήματα και αιτήματα των χρηστών.

1.3.2 Λειτουργίες του Διαλογικού Πράκτορα

Ο διαλογικός πράκτορας θα προσφέρει ένα φάσμα λειτουργιών που θα εξυπηρετούν τους χρήστες της πλατφόρμας:

- **Γενική Ενημέρωση:** Μέσω του διαλογικού πράκτορα, οι χρήστες μπορούν να ενημερωθούν για το υπάρχον πρόγραμμα μαθημάτων του εκάστοτε εξαμήνου που τους ενδιαφέρει.

- **Παροχή Πληροφοριών:** Ο πράκτορας θα παρέχει στους χρήστες βασικές πληροφορίες σχετικά με το μάθημα, τα θέματα που καλύπτονται και τους στόχους του, συμβάλλοντας στην κατανόηση της δομής του μαθήματος.

- **Απαντήσεις σε Ερωτήσεις:** Ο πράκτορας θα μπορεί να απαντά σε ερωτήσεις που αφορούν το μάθημα, παρέχοντας λεπτομερείς και περιεκτικές απαντήσεις που βοηθούν τους χρήστες να κατανοήσουν καλύτερα το περιεχόμενο.

1.3.3 Αλληλεπίδραση με τον Χρήστη

Στο πλαίσιο αυτής της υποενότητας, εξετάζουμε την αλληλεπίδραση μεταξύ των χρηστών και του διαλογικού πράκτορα, υπογραμμίζοντας τη φυσικότητα και την ομαλότητα της διαδικασίας. Η διασφάλιση μιας φυσικής και ευχάριστης εμπειρίας αλληλεπίδρασης αποτελεί έναν σημαντικό παράγοντα για την επιτυχία και την αποδοτικότητα του διαλογικού πράκτορα.

Όταν οι χρήστες αντιλαμβάνονται την αλληλεπίδραση με τον πράκτορα ως φυσική, αυτό διευκολύνει την επικοινωνία και αυξάνει την προσήλωσή τους στην εκπαιδευτική διαδικασία. Τα διαλείμματα και οι καθυστερήσεις που μπορεί να προκύψουν από ατέλειωτες τεχνικές παρεμβάσεις μεταξύ χρήστη και πράκτορα αποφεύγονται, δίνοντας έμφαση στην ομαλή ροή επικοινωνίας.

Ο διαλογικός πράκτορας πρέπει να είναι ικανός να ανταποκρίνεται με ανθρώπινο τρόπο, επιδεικνύοντας κατανόηση των ερωτημάτων και των αιτημάτων των χρηστών. Η χρήση προηγμένων τεχνικών φυσικής γλώσσας, όπως η αναγνώριση προθέσεων και η αντίστοιχη παραγωγή απαντήσεων, επιτρέπει στον πράκτορα να επικοινωνεί με τον χρήστη με τρόπο που αποτελείται από φυσικές και ευνόητες συνομιλίες.

Η επιτυχημένη αλληλεπίδραση δεν περιορίζεται μόνο στην αναγνώριση των ερωτημάτων. Περιλαμβάνει επίσης τη δυνατότητα του πράκτορα να αντιλαμβάνεται τον συναισθηματικό τόνο των ερωτημάτων και τις απαιτήσεις των χρηστών. Αυτό δημιουργεί μια προσωπική σύνδεση μεταξύ του χρήστη και του πράκτορα, προσφέροντας μια εμπειρία που προσομοιάζει σε ανθρώπινη συνομιλία.

Συνολικά, η αλληλεπίδραση με τον διαλογικό πράκτορα αποτελεί ένα κρίσιμο στοιχείο για την επιτυχία του εκπαιδευτικού του ρόλου. Η φυσική, ευχάριστη και προσαρμοσμένη ανταπόκριση του πράκτορα στις ανάγκες των χρηστών συμβάλλει στην αποτελεσματικότητα της μάθησης και στην επίτευξη των εκπαιδευτικών στόχων.

1.3.4 Αξιολόγηση και Βελτίωση

Η διαδικασία αξιολόγησης και βελτίωσης αποτελεί κρίσιμο βήμα στον κύκλο ζωής του διαλογικού πράκτορα, καθώς διασφαλίζει την εξέλιξη και την προσαρμογή του πράκτορα πάνω στις ανάγκες των χρηστών και της εκπαιδευτικής διαδικασίας.

Η βασική πηγή αξιολόγησης είναι οι ανατροφοδοτήσεις από τους χρήστες του πράκτορα. Οι απόψεις, οι παρατηρήσεις και οι προτάσεις που προκύπτουν από την αλληλεπίδρασή τους με τον πράκτορα παρέχουν πολύτιμη πληροφορία για την απόδοση του και την ανταπόκρισή του στις ανάγκες τους. Αυτές οι ανατροφοδοτήσεις μπορούν να αφορούν την κατανόηση των ερωτημάτων, την ποιότητα των απαντήσεων, αλλά και τον γενικό χαρακτήρα της αλληλεπίδρασης.

Η συλλογή και η ανάλυση αυτών των ανατροφοδοτήσεων βοηθά την εφαρμογή, με στόχο την εξαγωγή πρότυπων και την αναγνώριση πιθανών προκλήσεων και βελτιστοποιήσεων. Η σύγκριση των ανατροφοδοτήσεων με τους προκαθορισμένους στόχους της εφαρμογής βοηθά στον προσδιορισμό των αδυναμιών και των προτεραιοτήτων για βελτίωση.

Με βάση τις ανατροφοδοτήσεις, θα εφαρμόζονται συστηματικές αλλαγές στον πράκτορα με στόχο τη βελτίωση του. Αυτές οι βελτιώσεις μπορεί να αφορούν την αύξηση της ακρίβειας και της ποιότητας των απαντήσεων, την αποτελεσματικότητα στην αναγνώριση των προθέσεων των χρηστών, αλλά και τη βελτίωση της ροής της συνομιλίας.

Το αποτέλεσμα αυτής της διαδικασίας είναι ένας διαλογικός πράκτορας που συνεχώς προσαρμόζεται και βελτιώνεται, προσφέροντας μια υψηλής ποιότητας εκπαιδευτική εμπειρία για τους χρήστες. Με αυτόν τον τρόπο, η μελέτη περίπτωσης προσφέρει ένα ζωντανό παράδειγμα του πώς οι αξιολογήσεις και οι βελτιώσεις συμβάλλουν στη συνεχή εξέλιξη και αναβάθμιση της τεχνολογίας των διαλογικών πρακτόρων, ενισχύοντας την αποτελεσματικότητα και την αποδοτικότητά τους σε εκπαιδευτικά περιβάλλοντα.

1.4 Μέθοδος και Προσέγγιση

Ενόψει της ανάπτυξης και υλοποίησης ενός διαλογικού πράκτορα-βοηθού μαθήματος, η ενότητα αυτή εξηγεί αναλυτικά τη συστηματική προσέγγιση που θα ακολουθηθεί για την επίτευξη των προκαθορισμένων στόχων. Αυτή η ενότητα αποτελεί τον οδηγό για την πορεία που θα ακολουθηθεί κατά τη διάρκεια της έρευνας, του σχεδιασμού, της υλοποίησης, και της αξιολόγησης του πράκτορα.

Αρχικά, η μέθοδος περιλαμβάνει μια εκτενή βιβλιογραφική έρευνα για την ανασκόπηση των πρόσφατων εξελίξεων στον τομέα των διαλογικών πρακτόρων, της τεχνητής νοημοσύνης, και της μηχανικής μάθησης. Αυτή η ανάλυση θα εξετάσει τις πιο πρόσφατες τάσεις, τις τεχνολογικές προόδους και τις καινοτομίες που διαμορφώνουν το πεδίο. Πηγές όπως επιστημονικά άρθρα, συνέδρια, και βιβλία θα αξιοποιηθούν για την απόκτηση συνολικής κατανόησης του τομέα και των προκλήσεων που αντιμετωπίζει.

Στη συνέχεια, θα προχωρήσουμε στον σχεδιασμό του διαλογικού πράκτορα. Αφού αναλύσουμε και συγκρίνουμε τα σημερινά εργαλεία ανάπτυξης διαλογικών πρακτόρων, θα επιλέξουμε την πλατφόρμα *Dialogflow* της *Google*, λόγω των προηγμένων δυνατοτήτων της στην αναγνώριση φυσικής γλώσσας. Θα προγραμματίσουμε τις λειτουργίες του πράκτορα, συμπεριλαμβανομένης της αναγνώρισης προθέσεων χρηστών, της απόκρισης σε ερωτήματα, και της παροχής ενεργειακών ενεργειών.

Η φάση της υλοποίησης θα επικεντρωθεί στον αναλυτικό σχεδιασμό και την υλοποίηση της εφαρμογής. Θα αξιοποιήσουμε τις δυνατότητες της πλατφόρμας για τη δημιουργία των διαφόρων λειτουργιών του πράκτορα.

Στην τελική φάση, θα προταθούν τρόποι για την γόνιμη αξιολόγηση του πράκτορα. Η μελλοντική υλοποίηση της αξιολόγησης αυτής θα είναι κρίσιμη για τη βελτίωση του πράκτορα και την προσαρμογή του στις ανάγκες των χρηστών.

1.5 Επίλογος

Στον επίλογο αυτού του κεφαλαίου, αναδεικνύεται η σημασία και η πολυπλοκότητα της ανάπτυξης ενός διαλογικού πράκτορα-βοηθού μαθήματος. Ο διαλογικός αυτός πράκτορας επιδιώκει να ανταποκριθεί στις ανάγκες της σύγχρονης εκπαιδευτικής διαδικασίας, βελτιώνοντας την εμπειρία μάθησης και την αλληλεπίδραση με τον χρήστη. Η ανάπτυξη του πράκτορα αυτού απαιτεί σύγχρονες τεχνολογίες, προηγμένες μεθόδους, και συνεχή προσαρμογή στις ανάγκες των χρηστών.

Κατά τη διάρκεια αυτού του κεφαλαίου, διερευνήσαμε τις βασικές αρχές και στόχους που καθοδηγούν την ανάπτυξη του διαλογικού πράκτορα, με εστίαση στον ρόλο του ως βοηθού μαθήματος. Αναλύσαμε τη σημασία της τεχνητής νοημοσύνης και των ευφυών διαλογικών πρακτόρων στον τομέα της εκπαίδευσης και πώς μπορούν να ενσωματωθούν στη διαδικασία μάθησης.

Στη συνέχεια, εξετάσαμε τη μεθοδολογία που θα ακολουθηθεί για την ανάπτυξη του πράκτορα. Αναφέραμε τη συστηματική προσέγγιση που περιλαμβάνει τη βιβλιογραφική έρευνα, τον σχεδιασμό και την υλοποίηση των λειτουργιών του πράκτορα, καθώς και την μελλοντική αξιολόγηση και βελτίωσή του βάσει των ανατροφοδοτήσεων των χρηστών.

Τέλος, αναδείξαμε ότι η ανάπτυξη αυτού του διαλογικού πράκτορα είναι ένα πολυδιάστατο εγχείρημα που απαιτεί συνεργασία μεταξύ διαφόρων τομέων όπως η τεχνητή νοημοσύνη, η εκπαίδευση και η ανθρώπινη-υπολογιστική αλληλεπίδραση. Με τον προσανατολισμό αυτό, ανοίγουμε τον δρόμο για μια πιο αποτελεσματική, προσαρμοστική και ανθρωποκεντρική διαδικασία μάθησης.

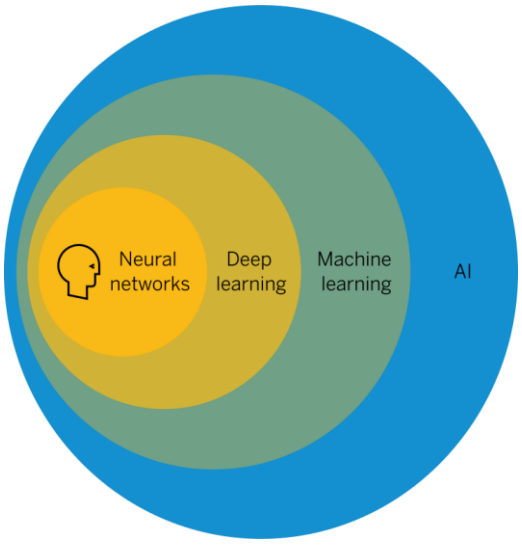
Κεφάλαιο 2ο: Θεωρητικό Υπόβαθρο

Οι διαλογικοί πράκτορες είναι προγράμματα υπολογιστή που σχεδιάστηκαν για να αλληλεπιδρούν με τους ανθρώπους με τρόπο παρόμοιο με την ανθρώπινη γλώσσα. Χρησιμοποιούνται σε πολλές εφαρμογές, από εικονικούς βοηθούς σε έξυπνα συστήματα συνομιλίας σε ιστοσελίδες και εφαρμογές κινητών. Για να λειτουργήσουν αποτελεσματικά, πρέπει να κατανοούν τις ανθρώπινες ανάγκες και να παράγουν προσαρμοσμένες απαντήσεις.

Σε αυτό το κεφάλαιο, θα εξετάσουμε τις τεχνικές μηχανικής μάθησης που χρησιμοποιούνται για την εκπαίδευση των διαλογικών πρακτόρων, καθώς και τη σημασία της ανάλυσης της φυσικής γλώσσας για την κατανόηση και τη δημιουργία ανθρώπινου διαλόγου. Ακόμη, θα εξετάσουμε παραδείγματα πρόσφατων εφαρμογών διαλογικών πρακτόρων για να κατανοήσουμε τον τρόπο με τον οποίο αυτοί οι πράκτορες εφαρμόζονται στην πράξη.

2.1 Εισαγωγή στην Τεχνητή Νοημοσύνη

Ο όρος της Τεχνητής Νοημοσύνης *(Artificial Intelligence)* αναφέρεται στην προσομοίωση ανθρωπίνων διαδικασιών από ειδικά υπολογιστικά συστήματα. Οι σημαντικές τεχνολογίες τεχνητής νοημοσύνης περιλαμβάνουν διάφορες υποκατηγορίες. Όπως φαίνεται και στην Εικόνα 2.1, η Μηχανική Μάθηση *(Machine Learning)*, το Βαθύ Μάθημα *(Deep Learning)*, τα Νευρωνικά Δίκτυα *(Neural Networks)*, αποτελούν κάποιες από αυτές αλλά όχι όλες. Αυτά τα συστήματα μπορούν να εκπαιδευτούν για να βελτιώσουν την επίδοσή τους σε συγκεκριμένες εργασίες, χρησιμοποιώντας δεδομένα και αλγόριθμους.



Εικόνα 2.1 Τεχνολογίες ΤΝ.

(Πηγή: https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence)

Η τεχνητή νοημοσύνη εφαρμόζεται σε πολλούς τομείς, όπως η ρομποτική, η υγειονομική περίθαλψη, η αυτόνομη οδήγηση, η αναγνώριση φωνής και εικόνας, η πρόβλεψη δεδομένων, και πολλοί άλλοι. Η τεχνητή νοημοσύνη αντιπροσωπεύει έναν σημαντικό τομέα έρευνας και ανάπτυξης με δυνατότητες επιρροής σε πολλές πτυχές της καθημερινής ζωής και της βιομηχανίας.

Παρακάτω θα αναλύσουμε τις σημαντικές τεχνολογίες της ΤΝ οι οποίες είναι η Μηχανική Μάθηση *(Machine Learning)*, η Βαθιά Μηχανική Μάθηση *(Deep Learning)*, τα Νευρωνικά Δίκτυα *(Neural Networks)*.

2.2 Μηχανική Μάθηση

Η μηχανική μάθηση είναι η κατηγορία τεχνητής νοημοσύνης που επιτρέπει στους υπολογιστές να μαθαίνουν από δεδομένα. Οι αλγόριθμοι μηχανικής μάθησης χρησιμοποιούνται για να αναγνωρίσουν πρότυπα, να προβλέπουν τάσεις και να προσαρμόζονται σε νέα δεδομένα.

Παρακάτω θα εξερευνήσουμε τις βασικές τεχνικές μηχανικής μάθησης που αποτελούν την κινητήρια δύναμη για την ανάπτυξη και λειτουργία των διαλογικών πρακτόρων. Αυτές οι τεχνικές δημιουργούν το θεμέλιο της εξελισσόμενης τεχνολογίας που δίνει τη δυνατότητα για την αναγνώριση, την κατανόηση και την απόκριση σε ανθρώπινη γλώσσα, αποτελώντας τη βάση για μια φυσική και αποτελεσματική επικοινωνία μεταξύ ανθρώπου και μηχανής.

Το κλειδί για την επιτυχία των διαλογικών πρακτόρων βρίσκεται στη δυνατότητα τους να αναπτύσσουν ικανότητες μέσα από την εμπειρία. Στο πλαίσιο αυτό, η μηχανική μάθηση παίζει καθοριστικό ρόλο διότι επιτρέπει τη δημιουργία πρακτόρων που μπορούν να μάθουν από τα δεδομένα και να προσαρμόζονται αυτόματα σε νέες πληροφορίες και περιβάλλοντα.

Συνοψίζοντας, η μηχανική μάθηση ανοίγει νέους ορίζοντες στη δυνατότητα δημιουργίας διαλογικών πρακτόρων με εξελιγμένες ικανότητες αντίληψης, κατανόησης και προσαρμογής σε διάφορες συνθήκες και απαιτήσεις. Μέσα από την αξιοποίηση της ανάλυσης δεδομένων και την εξαγωγή γνώσης, η μηχανική μάθηση ανοίγει νέες προοπτικές για την εξέλιξη της τεχνολογίας των διαλογικών πρακτόρων.

Παρακάτω θα δούμε τις βασικές τεχνικές μηχανικής μάθησης, οι οποίες είναι: Επιβλεπόμενη *(Supervised Learning)*, Μη Επιβλεπόμενη *(Unsupervised Learning)*, Ημι-Επιβλεπόμενη *(Semi-Supervised Learning)*, Ενισχυμένη *(Reinforcement Learning)*.

2.2.1 Επιβλεπόμενη (Supervised Learning)

Η επιβλεπόμενη μάθηση αντιπροσωπεύει το πρώτο από τα τέσσερα μοντέλα μηχανικής μάθησης. Στο πλαίσιο των επιβλεπόμενων αλγορίθμων μάθησης, η μηχανή εκπαιδεύεται μέσω της ανάληψης παραδειγμάτων. Ο στόχος είναι να μάθει το μοντέλο να παράγει σωστές προβλέψεις για νέες, προηγουμένως άγνωστες εισόδους. Ένα κλασσικό παράδειγμα είναι η κατηγοριοποίηση εικόνων.

Έχουμε ένα σύνολο εικόνων, και για κάθε εικόνα, υπάρχει μια ετικέτα που λέει το αντικείμενο στην εικόνα (π.χ., γάτα ή σκύλος). Το μοντέλο εκπαιδεύεται να μάθει να συσχετίζει τα χαρακτηριστικά των εικόνων με τις σωστές κατηγορίες.

Μέσω του αλγορίθμου, το σύστημα επεξεργάζεται τα δεδομένα εκπαίδευσης στη διάρκεια του χρόνου και αναπτύσσει συσχετίσεις, ανιχνεύει διαφορές και αναγνωρίζει τα στοιχεία της λογικής, καταλήγοντας στη δυνατότητα πρόβλεψης απαντήσεων χωρίς εξωτερική καθοδήγηση.

2.2.2 Μη Επιβλεπόμενη (Unsupervised Learning)

Στη μη επιβλεπόμενη μάθηση, το μοντέλο εκπαιδεύεται σε ένα σύνολο δεδομένων χωρίς ετικέτες. Σκοπός είναι να ανακαλύψει μοτίβα ή δομές στα δεδομένα χωρίς προηγούμενη γνώση των αποτελεσμάτων.

Ένα παράδειγμα είναι όταν δίνονται σε έναν υπολογιστή φωτογραφίες ανθρώπων χωρίς να γνωρίζει ποιος είναι ποιος. Ο υπολογιστής θα προσπαθήσει να ανακαλύψει μόνος του τα χαρακτηριστικά που κάνουν τους ανθρώπους διαφορετικούς, όπως τα χρώματα του δέρματος, τα μάτια, κλπ.

Σε απλά λόγια, είναι σαν να προσπαθεί ο υπολογιστής να βρει πρότυπα ή να ομαδοποιήσει αυτά τα δεδομένα χωρίς να ξέρει τι να αναζητήσει. Συνηθισμένα παραδείγματα εφαρμογών μη επιβλεπόμενης μάθησης περιλαμβάνουν την αναγνώριση προσώπου (*Face ID*), την ανάλυση γονιδιακών ακολουθιών και την κυβερνοασφάλεια.

2.2.3 Ημι-Επιβλεπόμενη (Semi-Supervised Learning)

Η ημι-εποπτευόμενη μάθηση αποτελεί το τρίτο μοντέλο μηχανικής μάθησης από τα τέσσερα. Σε έναν ιδανικό κόσμο, όλα τα δεδομένα θα ήταν δομημένα και επισημασμένα πριν εισαχθούν σε ένα σύστημα. Εντούτοις, καθώς αυτό δεν είναι εφικτό, η ημι-εποπτευόμενη μάθηση αποτελεί μια πρακτική λύση όταν υπάρχουν μεγάλες ποσότητες μη δομημένων δεδομένων.

Αυτό το μοντέλο περιλαμβάνει την εισαγωγή μικρών ποσοτήτων δεδομένων με ετικέτα για να επεκτείνει τα σύνολα δεδομένων χωρίς ετικέτα. Τα επισημασμένα δεδομένα λειτουργούν ως αρχικό σημείο εκκίνησης για το σύστημα και βελτιστοποιούν σημαντικά την ταχύτητα και την ακρίβεια της μάθησης. Ένας ημι-εποπτευόμενος αλγόριθμος μαθαίνει τη μηχανή να αναλύσει τα επισημασμένα δεδομένα για συσχετίσεις που μπορούν να εφαρμοστούν στα δεδομένα χωρίς ετικέτα.

Παρά τα πλεονεκτήματα, υπάρχουν και κίνδυνοι σχετιζόμενοι με την ημι-εποπτευόμενη μάθηση, όπου ελαττώματα στα επισημασμένα δεδομένα μπορούν να μαθαίνουν και να αναπαράγονται από το σύστημα. Εταιρείες που χρησιμοποιούν επιτυχώς αυτή τη μέθοδο εξασφαλίζουν την ύπαρξη πρωτοκόλλων βέλτιστης πρακτικής. Η ημι-εποπτευόμενη μάθηση εφαρμόζεται στην ομιλία και τη γλωσσική ανάλυση, στην πολύπλοκη ιατρική έρευνα, όπως στην κατηγοριοποίηση πρωτεϊνών, και στον υψηλού επιπέδου εντοπισμό απάτης.

2.2.4 Ενισχυμένη (Reinforcement Learning)

Αυτή η προσέγγιση της μηχανικής μάθησης χρησιμοποιείται συχνά στα παιχνίδια, στην ανάπτυξη αυτόνομων συστημάτων και στη ρομποτική. Η ενισχυτική μάθηση διαφέρει από την εποπτευόμενη μάθηση διότι δεν απαιτεί παρουσίαση συγκεκριμένων ζευγών εισόδου/εξόδου και δεν επιβάλλει συγκεκριμένες διορθώσεις σε μη βέλτιστες ενέργειες. Αντιθέτως, η κύρια έμφαση έγκειται στον εντοπισμό μιας ισορροπίας ανάμεσα στην εξερεύνηση (ανακάλυψη μη χαρτογραφημένων περιοχών) και την εκμετάλλευση (χρήση της υπάρχουσας γνώσης).

Η βασική διαφορά μεταξύ των κλασικών μεθόδων δυναμικού προγραμματισμού και των αλγορίθμων ενισχυτικής μάθησης είναι ότι οι τελευταίοι λειτουργούν χωρίς την ανάγκη για γνώση ενός ακριβούς μαθηματικού μοντέλου του Μοντέλου Δυναμικού Συστήματος *(MDP)*. Επιπλέον, οι αλγόριθμοι ενισχυτικής μάθησης αντιμετωπίζουν μεγάλα *MDP*, όπου οι ακριβείς μέθοδοι δυναμικού προγραμματισμού γίνονται ανέφικτες λόγω του υψηλού υπολογιστικού κόστους. Και αυτό είναι το κύριο πλεονέκτημα των αλγορίθμων ενισχυτικής μάθησης, καθώς επιτρέπουν την αντιμετώπιση προβλημάτων σε περιβάλλοντα όπου η δομή του *MDP* είναι άγνωστη ή πολύπλοκη.

2.3 Βαθιά Μηχανική Μάθηση

Η βαθιά μάθηση ονομάζεται λόγω της ύπαρξης ενός νευρωνικού δικτύου,το οποίο θα αναλυθεί αργότερα, με τρία ή περισσότερα επίπεδα, καθώς και λόγω της χρήσης μεγάλου όγκου σύνθετων και διαφορετικών δεδομένων. Είναι επίσης υποσύνολο της μηχανικής μάθησης καθώς και της τεχνητής νοημοσύνης. Για να επιτύχει τη βαθιά μάθηση, το σύστημα συνδέεται με πολλαπλά επίπεδα στο δίκτυο, εξάγοντας σταδιακά πιο περίπλοκες εξόδους σε υψηλότερα επίπεδα.

Για παράδειγμα, ένα σύστημα βαθιάς μάθησης που αναγνωρίζει εικόνες φύσης και αναζητά μαργαρίτες μπορεί αρχικά να αναγνωρίσει ένα φυτό στο πρώτο επίπεδο. Καθώς προχωρά στα νευρωνικά στρώματα, θα αναγνωρίσει στη συνέχεια ένα λουλούδι, μια μαργαρίτα, και τελικά μια συγκεκριμένη μαργαρίτα. Παραδείγματα εφαρμογών βαθιάς μάθησης περιλαμβάνουν τον αναγνώστη ομιλίας, την ταξινόμηση εικόνας και τη φαρμακευτική ανάλυση.

Στην κλασική μηχανική μάθηση, οι αλγόριθμοι εργάζονται με δομημένα δεδομένα, τα οποία έχουν συγκεκριμένα χαρακτηριστικά που ορίζονται από τα δεδομένα εισόδου και οργανώνονται σε πίνακες. Αν και μπορεί να χρησιμοποιούν και μη δομημένα δεδομένα, συνήθως απαιτείται προεπεξεργασία για την οργάνωση τους σε δομημένη μορφή.

Αντίθετα, η βαθιά εκμάθηση απορροφά και επεξεργάζεται μη δομημένα δεδομένα χωρίς την ίδια έκταση προεπεξεργασίας. Αυτοί οι αλγόριθμοι μπορούν να αυτοματοποιήσουν την εξαγωγή χαρακτηριστικών, ελαχιστοποιώντας την ανάγκη για χειροκίνητη προεπεξεργασία από ειδικούς. Για παράδειγμα, εάν εφαρμόζαμε βαθιά εκμάθηση σε ένα σύνολο φωτογραφιών λουλουδιών για κατηγοριοποίηση σε "παπαρούνα", "τριαντάφυλλο", "τουλίπα" κ.λπ., ο αλγόριθμος θα μπορούσε αυτόνομα να αναγνωρίσει τα σημαντικά χαρακτηριστικά (π.χ., χρώμα) για να διακρίνει τα διάφορα λουλούδια.

Κάποιες βασικές τεχνικές **Βαθιάς Μηχανικής Μάθησης** είναι οι παρακάτω:

2.3.1 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks CNNs)

Τα **Convolutional Neural Networks *(CNNs)*** ξεχωρίζονται από άλλα νευρωνικά δίκτυα λόγω της υψηλής τους απόδοσης στην επεξεργασία εικόνων, ήχου ή σημάτων. Έχουν τρία βασικά είδη στρωμάτων:

- **Συνελικτικό Επίπεδο *(Convolutional layer)*:** Είναι το πρώτο επίπεδο ενός *CNN* και χρησιμοποιείται για τον εντοπισμό χαρακτηριστικών στην εικόνα, όπως χρώματα και ακμές.

- **Επίπεδο Συγκέντρωσης *(Pooling layer)*:** Το συγκεκριμένο επίπεδο χρησιμοποιείται για τη μείωση της διαστατικότητας των χαρακτηριστικών, διατηρώντας τα σημαντικότερα.

- **Πλήρως Συνδεδεμένο Επίπεδο *(Fully-connected layer)*:** Είναι το τελευταίο επίπεδο ενός *CNN* και συμβάλλει στον τελικό προσδιορισμό του αντικειμένου. Καθώς η εικόνα προχωρά μέσα από τα επίπεδα, το *CNN* αναγνωρίζει πιο πολύπλοκα χαρακτηριστικά.

Με αυτό τον τρόπο, τα πρώτα επίπεδα επικεντρώνονται σε απλά χαρακτηριστικά, όπως χρώματα και ακμές, ενώ τα τελευταία επίπεδα αναγνωρίζουν μεγαλύτερα στοιχεία ή σχήματα, καταλήγοντας τελικά στον αναγνωρισμό του επιθυμητού αντικειμένου.

2.3.2 Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Recurrent Neural Networks RNNs)

Τα Αναδρομικά Νευρωνικά Δίκτυα *(Recurrent Neural Networks - RNNs)* είναι ένα είδος νευρωνικών δικτύων που έχουν σχεδιαστεί για να διαχειρίζονται ακολουθίες δεδομένων ή χρονοσειρές. Σε αντίθεση με τα συμβατικά νευρωνικά δίκτυα, τα οποία θεωρούν κάθε είσοδο ως ανεξάρτητη από τις προηγούμενες, τα *RNNs* διατηρούν μια κατάσταση *(state)* που ενημερώνεται με κάθε νέα είσοδο.

Η ιδέα πίσω από τα *RNNs* είναι να εισάγουν την έννοια της μνήμης στα νευρωνικά δίκτυα, καθιστώντας τα ικανά να διατηρούν πληροφορίες για προηγούμενες καταστάσεις. Αυτό είναι χρήσιμο όταν ασχολούμαστε με ακολουθίες δεδομένων, όπως κείμενο, χρονοσειρές, ή οποιοδήποτε άλλο είδος δεδομένων που έχει χρονική ή σειριακή διάσταση.

Το βασικό μειονέκτημα των απλών *RNNs* είναι ότι δυσκολεύονται να "θυμηθούν" πληροφορίες από μακρινές παρελθοντικές καταστάσεις, λόγω του προβλήματος της εξαφάνισης *(Vanishing)* ή Εκρηκτικής *(exploding)* κλίσης. Για να αντιμετωπίσουν αυτό το πρόβλημα, έχουν αναπτυχθεί πιο προηγμένες μορφές των *RNNs*, όπως τα Long Short-Term Memory Networks *(LSTMs)* και τα Gated Recurrent Units *(GRUs)*, τα οποία έχουν σχεδιαστεί για να αντιμετωπίζουν αποτελεσματικότερα τα προβλήματα μνήμης σε μακροπρόθεσμες ακολουθίες.

Τα *RNNs* χρησιμοποιούνται σε πολλές εφαρμογές, όπως αναγνώριση φωνής, μετάφραση μηχανής, αναγνώριση προτύπων σε χρονοσειρές, και γενικότερα σε καθετοποιημένα προβλήματα με χρονική διάσταση.

2.3.3 Μετασχηματιστής (Transformer)

Ο *Transformer* είναι ένα είδος αρχιτεκτονικής νευρωνικού δικτύου που πρωτοπαρουσιάστηκε από τους Vaswani και συνεργάτες σε μια έρευνα που δημοσιεύτηκε το 2017 με τον τίτλο "Attention is All You Need". Οι αρχιτέκτονες *Transformer* αναδείχθηκαν αρχικά για την επίλυση προβλημάτων μηχανικής μετάφρασης, αλλά από τότε έχουν καταστεί ευρέως χρησιμοποιούμενες σε πολλές εφαρμογές τεχνητής νοημοσύνης.

Ο κύριος καινοτομία του *Transformer* είναι ο μηχανισμός αναστροφής προσοχής *(Self-Attention Mechanism)*, ο οποίος επιτρέπει στο δίκτυο να επικεντρωθεί δυναμικά σε διάφορα μέρη της εισόδου κατά την επεξεργασία. Αυτό επιτρέπει στο δίκτυο να αντιμετωπίσει ακολουθίες δεδομένων με διαφορετικές σημαντικότητες.

Οι *Transformers* έχουν εφαρμοστεί με επιτυχία σε πολλούς τομείς, συμπεριλαμβανομένης της επεξεργασίας φυσικής γλώσσας *(Natural Language Processing - NLP)*, αναγνώριση εικόνων, μετάφραση μηχανής, και γενικά σε καθετοποιημένα προβλήματα που απαιτούν την αντιμετώπιση συνεκτικών σχέσεων σε δεδομένα με δομή.

Η επιτυχία των Transformers οδήγησε επίσης στην ανάπτυξη πολλών παραλλαγών τους, όπως τα *BERT, GPT (Generative Pre-trained Transformer)*, και άλλα, που έχουν κατακτήσει πολύ υψηλές επιδόσεις σε διάφορα καθήκοντα μηχανικής μάθησης και νοημοσύνης.

Παρακάτω θα δούμε λίγα λόγια για τα ***BERT*** και ***GPT*** για τυχόν σύγχυση του αναγνώστη:

Το BERT *(Bidirectional Encoder Representations from Transformers)* και το GPT *(Generative Pre-trained Transformer)* είναι δύο διακεκριμένα μοντέλα βαθιάς μάθησης που βασίζονται στην αρχιτεκτονική των *Transformers*. Το *BERT* είναι σε θέση να κατανοήσει το κείμενο και συχνά χρησιμοποιείται για κατακερματισμό λέξεων, εντοπισμό οντοτήτων, κλπ. Από την άλλη πλευρά το *GPT* χρησιμοποιεί και το *BERT* απλή η διαφορά τους είναι ότι το *GPT* μπορεί να παράγει και κείμενο. Για παράδειγμα αν δώσουμε το κομμάτι κειμένου "Σε μια μακρινή γαλαξία, ένας ήρωας μετράει τα αστέρια", το *GPT* μπορεί να συνεχίσει την ιστορία και να παράγει κείμενο που συνδέεται με το προηγούμενο.

2.4 Νευρωνικά Δίκτυα

Μια ακόμη κατηγορία της τεχνητής νοημοσύνης και υποκατηγορία της βαθιάς μηχανικής μάθησης και της μηχανικής μάθησης είναι τα νρυωνικά δίκτυα. Τα νευρωνικά δίκτυα είναι υπολογιστικά μοντέλα που εμπνέονται από τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου. Κάθε

"νευρώνας" σε ένα νευρωνικό δίκτυο εκτελεί απλές λειτουργίες. Αυτοί οι νευρώνες συνδέονται μεταξύ τους, και το δίκτυο μαθαίνει από τα δεδομένα που του παρέχονται.

Για παράδειγμα, σε ένα νευρωνικό δίκτυο που μαθαίνει να αναγνωρίζει λουλούδια, κάθε νευρώνας μπορεί να ανταποκριθεί σε συγκεκριμένα χαρακτηριστικά, όπως το σχήμα του λουλουδιού ή το χρώμα. Καθώς το δίκτυο εκπαιδεύεται με παραδείγματα Λουλουδιών, προσαρμόζει τις συνδέσεις των νευρώνων για να αναγνωρίζει αυτά τα χαρακτηριστικά. Συνολικά, τα νευρωνικά δίκτυα είναι σαν εκπαιδευόμενοι "εγκεφάλοι" που μπορούν να μάθουν να αντιμετωπίζουν διάφορα είδη εργασιών, όπως η αναγνώριση εικόνων, η φωνητική αναγνώριση, και άλλα.

Οι τεχνικές **Νευρωνικών Δικτύων** που χρησιμοποιούνται είναι σχεδόν ίδιες με της τεχνικές στην Βαθιά Μηχανική Μάθηση οι οποίες συνεργάζονται. Ονομαστικά κάποιες από αυτές είναι:

- **Συνελικτικό Νευρωνικό Δίκτυο *(Convolutional Neural Network - CNN)***: Είναι σχεδιασμένο για την αναγνώριση προτύπων σε εικόνες. Χρησιμοποιεί φίλτρα (καταστρώματα) για την εντοπισμό χαρακτηριστικών, όπως ακμές και γωνίες, όπως αναλύσαμε και προηγούμενός.

- **Αναδρομικό Νευρωνικό Δίκτυο *(Recurrent Neural Network - RNN)***: Όπως είδαμε προηγουμένως, αυτό είναι σχεδιασμένο για την επεξεργασία ακολουθιών δεδομένων, όπως κείμενο και χρονοσειρές. Οι *RNNs* διατηρούν κατάσταση (μνήμη) που επιτρέπει την εκμάθηση μακροπρόθεσμων εξαρτήσεων.

- **Το Πλήρες Διασυνδεδεμένο Δίκτυο *(Fully Connected Neural Network)***: Είναι το πιο απλό είδος νευρωνικού δικτύου, όπου κάθε νευρώνας σε ένα επίπεδο είναι συνδεδεμένος με κάθε νευρώνα στο επόμενο επίπεδο. Είναι κοινό σε προβλήματα κατηγοριοποίησης και πρόβλεψης.

2.5 Επίλογος

Στο πλαίσιο αυτού του κεφαλαίου, αναλύσαμε τη σημασία της μηχανικής μάθησης ως βασικού εργαλείου για την ανάπτυξη των διαλογικών πρακτόρων, και πώς η ικανότητα της μηχανής να εξάγει πρότυπα από δεδομένα επιτρέπει τη δημιουργία πρακτόρων που μπορούν να προσαρμόζονται στις ανάγκες και τις προτιμήσεις των χρηστών.

Είδαμε επίσης ότι οι διάφορες τεχνικές μηχανικής μάθησης, όπως η επιβλεπόμενη, η μη επιβλεπόμενη και η ενισχυτική μάθηση, καθώς και τα νευρωνικά δίκτυα, έχουν ανοίξει νέους ορίζοντες στην αντιμετώπιση προβλημάτων που παλιότερα θεωρούνταν δύσκολα.

Συνοψίζοντας, η βαθιά μηχανική μάθηση και τα νευρωνικά δίκτυα συνδέονται στενά και χρησιμοποιούν πολλές κοινές τεχνικές και αλγόριθμους. Οι αλγόριθμοι εκπαίδευσης, βελτιστοποίησης, αξιολόγησης, και επεξεργασίας δεδομένων παίζουν καίριο ρόλο στην επίτευξη υψηλής απόδοσης των μοντέλων.

Το κεφάλαιο αυτό παρέθεσε μια περιεκτική επισκόπηση του πεδίου των διαλογικών πρακτόρων και των τεχνολογιών που τους υποστηρίζουν. Καταδείξαμε τον συναρπαστικό ρόλο που διαδραματίζουν στη βελτίωση της αλληλεπίδρασης ανθρώπου-μηχανής και μέσα από την ανάλυση της Τεχνικής Νοημοσύνης διαπιστώνουμε πώς οι προοπτικές για το μέλλον είναι λαμπρές. Με την ταχεία

εξέλιξη της τεχνολογίας, αναμένουμε ότι οι διαλογικοί πράκτορες θα συνεχίσουν να διαμορφώνουν τον τρόπο με τον οποίο αλληλεπιδρούμε και αξιοποιούμε τις ψηφιακές υπηρεσίες στο μέλλον.

Κεφάλαιο 3ο: Συστηματική Συγκριτική Μελέτη των Εργαλείων Ανάπτυξης Ευφυών Διαλογικών Πρακτόρων

Το τρίτο κεφάλαιο της πτυχιακής εργασίας αφορά την "Συγκριτική μελέτη των εργαλείων ανάπτυξης ευφυών διαλογικών πρακτόρων" και μαζί με τα επόμενα κεφάλαιο αποτελεί τον πυλώνα της πτυχιακής εργασίας. Σε αυτό το κεφάλαιο, θα εξεταστούν τα διάφορα εργαλεία διαλογικών πρακτόρων που υπάρχουν, θα αναλυθούν τα χαρακτηριστικά τους, έπειτα θα αξιολογηθούν και θα τα συγκριθούν και τέλος θα γίνει η επιλογή κάποιου εργαλείου για την ανάπτυξη ενός διαλογικού πράκτορα.

3.1 Εργαλεία Ανάπτυξης Ευφυών Διαλογικών Πρακτόρων

Ο τομέας της ανάπτυξης ευφυών διαλογικών πρακτόρων εμπεριέχει ποικίλα εργαλεία, βιβλιοθήκες και πλατφόρμες που χρησιμοποιούνται για τη δημιουργία αυτών των προηγμένων συστημάτων. Κάποια από αυτά τα εργαλεία είναι:

3.1.1 DialogFlow

Το *DialogFlow* είναι μια υπηρεσία που παρέχεται από τη *Google* και χρησιμοποιείται για τη δημιουργία chatbots και εφαρμογών που ανταποκρίνονται σε φωνητικές εντολές. Αρχικά δημιουργήθηκε με το όνομα "*API.AI*" και αργότερα ανακοινώθηκε ως *DialogFlow* από τη *Google*. Ο στόχος του *DialogFlow* είναι να παρέχει ένα εργαλείο που επιτρέπει στους προγραμματιστές να δημιουργούν εύκολα συστήματα συνομιλίας. Κάποια από τα πλεονεκτήματα και τα μειονεκτήματά του είναι τα εξής:

Πλεονεκτήματα:

- **Εύκολη Χρήση**: Έχει εύχρηστο γραφικό περιβάλλον για την ανάπτυξη *Chatbots*, καθιστώντας το κατάλληλο για αρχάριους προγραμματιστές.

- **Ενσωμάτωση με Άλλες Υπηρεσίες Google**: Είναι ευέλικτο και μπορεί να ενσωματωθεί εύκολα με άλλες υπηρεσίες της *Google*.

- **Δωρεάν Προτεινόμενη Χρήση**: Προσφέρει ένα δωρεάν επίπεδο χρήσης, κατάλληλο για περιορισμένες εφαρμογές.

Μειονεκτήματα:

Περιορισμένος Έλεγχος Συνομιλίας: Ορισμένες φορές οι προγραμματιστές ενδέχεται να αντιμετωπίσουν περιορισμούς στον έλεγχο της συνομιλίας.

Εξαρτηση από την Google: Είναι στενά συνδεδεμένο με τις υπηρεσίες της *Google*, κάτι που μπορεί να περιορίζει την ευελιξία.

3.1.2 Microsoft Bot Framework

Το *Microsoft Bot Framework* είναι ένα πλαίσιο που αναπτύχθηκε από τη *Microsoft* για τη δημιουργία, τη διαχείριση και την εκτέλεση *Chatbots*. Είναι ένα εργαλείο που επιτρέπει στους προγραμματιστές να δημιουργήσουν εφαρμογές που επικοινωνούν με τους χρήστες μέσω φυσικής γλώσσας, εκμεταλλευόμενοι τεχνολογίες όπως η αναγνώριση φωνής και η κατανόηση φυσικής γλώσσας. Όπως και το *DialogFlow* έτσι και εδώ θα δούμε κάποια θετικά και αρνητικά, όπως τα εξής:

Πλεονεκτήματα:

- **Ευέλικτο**: Υποστηρίζει πολλές γλώσσες προγραμματισμού και είναι ευέλικτο σε ό,τι αφορά την ανάπτυξη.

- **Ενσωμάτωση με Άλλα Προϊόντα Microsoft**: Μπορεί εύκολα να συνδεθεί και να ενσωματωθεί με άλλα προϊόντα και υπηρεσίες της *Microsoft*, όπως το *Azure* και το *Microsoft 365*.

- **Καλή Τεκμηρίωση**: Διαθέτει καλή τεκμηρίωση και ενεργή κοινότητα υποστήριξης.

Μειονεκτήματα:

- **Περίπλοκη Εγκατάσταση**: Η εγκατάσταση και η ρύθμιση μπορεί να είναι πιο πολύπλοκη σε σύγκριση με άλλα πλαίσια.

- **Περιορισμένη Δωρεάν Χρήση**: Το κόστος μπορεί να αυξηθεί για μεγαλύτερες και πιο προηγμένες εφαρμογές.

3.1.3 Rasa

Το *Rasa* είναι ένα λογισμικό ανάπτυξης διαλογικών πρακτόρων ανοιχτού κώδικα, που σημαίνει ότι ο κώδικας του είναι προσβάσιμος και μπορεί να τροποποιηθεί από το κοινό. Αυτό παρέχει μεγάλη ευελιξία και δυνατότητα προσαρμογής. Κάποια από τα πλεονεκτήματα και τα μειονεκτήματά του είναι τα εξής:

Πλεονεκτήματα:

Ανοιχτού Κώδικα: Είναι ελεύθερο και ανοιχτού κώδικα, παρέχοντας πλήρη έλεγχο και προσαρμογή.

Ευελιξία στον Σχεδιασμό: Οι προγραμματιστές μπορούν να καθορίσουν την αρχιτεκτονική του *Chatbot* σύμφωνα με τις ανάγκες τους. Για παράδειγμα, μπορούν να προσθέσουν προσαρμοσμένους κανόνες συνομιλίας που ανταποκρίνονται σε συγκεκριμένα σενάρια.

Στοχαστική Συνομιλία: Με τη στοχαστική συνομιλία, το *Rasa* μπορεί να παράγει ποικίλες απαντήσεις για τις ίδιες ερωτήσεις. Αυτό δημιουργεί μια πιο φυσική αίσθηση στην αλληλεπίδραση, καθώς οι άνθρωποι δεν παράγουν πάντα τις ίδιες ακριβείς απαντήσεις.

Μειονεκτήματα:

Απαιτεί Εμπειρογνώμονα: Η ανάπτυξη με *Rasa* μπορεί να απαιτεί περισσότερη εμπειρία σε σχέση με άλλες πλατφόρμες.

Εξαρτηση από τον Προγραμματιστή: Ενδέχεται να χρειάζεται περισσότερη προσοχή και συντήρηση από τους προγραμματιστές.

3.2 Συγκριτική Ανάλυση

Η αυξανόμενη δημοτικότητα των chatbots οδήγησε στην εμφάνιση πολλών εργαλείων για τη δημιουργία τους, όπου κάποια από αυτά αναλύθηκαν σε προηγούμενη ενότητα του κεφαλαίου.

Ο Πίνακας 3.3.1 συγκρίνει τις κύριες διαθέσιμες επιλογές λογισμικού για τη δημιουργία chatbot. Περιλαμβάνει προτάσεις τόσο από μεγάλες εταιρείες *(Dialogflow by Google, Watson by IBM, Lex by Amazon, and Bot Framework and LUIS by Microsoft)* όσο και από νεότερες εταιρείες ειδικευόμενες σε *Chatbot [FlowXO, Landbot.io, Chatfuel, Rasa, SmartLoop, Xenioo, Botkit* (το οποίο πρόσφατα εξαγοράστηκε από τη Microsoft), *ChatterBot*, και *Pandorabots]*. Όλα αυτά είναι ανεξάρτητα από τον τομέα, εκτός από το *Chatfuel*, το οποίο επικεντρώνεται σε εφαρμογές μάρκετινγκ.

			Dialogflow (Google) [v2]	Watson (IBM) [v2]	Lex (Amazon) [07/06/2020]	Bot Framework + LUIS (Microsoft) [v4]	FlowXO [07/06/2020]	Landbot.io [07/06/2020]	Chatfuel [07/06/2020]	Rasa [1.0.1.2]	SmartLoop [07/06/2020]	Xenioo [07/06/2020]	Botkit (part of Bot Framework) [4.9.0]	LUIS [05/19/2020]	ChatterBot [1.0.5]	Pandorabots [07/06/2020]
Technical Factors		1. Kind (Library, Framework, Platform, Service)	P	P	P	F	P	P	P	F	P	P	F	S	L	P
	Input processing	2. Regular expressions/patterns	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓
		3. NLP for phrase match	✓	✓	✓	✓			✓	✓	✓	✓		✓	✓	
		4. Text processing to obtain phrase parameters	✓	✓	✓	✓				✓	✓	✓		✓		✓
		5. Number of languages: very high (≥50), high (≥10), some (<10), 1 (represented with flag)	h	h	🇺🇸	h	h		h	v		s		h	v	
		6. Sentiment analysis	✓	✓	✓	✓								✓		
		7. Speech recognition	✓	✓	✓	✓						✓		✓		
	Dialogue	8. Storage of phrase parameters: volatile, persistent, both	b	b	b	b	b	v	v	b	v	v		v		v
		9. Support for intents	✓	✓	✓	✓	✓			✓	✓			✓		✓
		10. Support for entities: predefined, user-def, both	b	b	b	b	p	p		b	b	b		b		
		11. Dialogue structure: tree, followup intents, dsl	f	f	f	f	t	t	t	t	f	t			f	d
		12. Utterances to reengage users					✓		✓		✓	✓				
		13. Specification of chatbot answers	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓
	Deployment	14. Integration with social networks/websites: high (≥10), some (<10), 1 (represented with logo)	h	s	s	h	s	🟢	🔵	h	s	s	s			s
		15. Interaction support for specific social networks	✓			✓						✓				✓
	Sys. integ.	16. Call to services from chatbot	✓	✓	✓	✓	✓	✓	✓	✓					✓	
		17. Chatbot usage via API	✓	✓	✓						✓	✓		✓		✓
	Development and testing	18. Pre-built components: chatbot templates, intents, small talks, services	cts	c	i	cs	c	c							c	t
		19. Version control: native, code-based	n	n	n	c				c			c		c	
		20. Chat console for testing	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				✓
		21. Debug mechanisms	✓			✓				✓			✓		✓	
		22. Validation support	✓													
	Execution	23. Hosted deployment	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓		✓
		24. Support for analytics	✓	✓		✓	✓	✓			✓	✓				
		25. User message persistence	✓		✓	✓	✓	✓	✓			✓	✓			
	Sec	26. Cloud security	✓	✓	✓	✓								✓		
Managerial Factors	Organization	27. Pricing model: free, pay-as-you-go, quota, advanced feats	fp	fp	fp	fpa	fq	fa	fa	fa	fa	fq	f	fq	f	fqa
		28. Developer expertise: low, high	l	l	l	h	l	l	l	h	l	l	h	h	h	l
	Development	29. Code hosting: external, on-premises	e	e	e	o	e	e	e	o	e	e	o	o	o	e
		30. Group work				✓				✓			✓	✓	✓	
		31. i8n	✓													
		32. Open source								✓			✓		✓	
	Operational	33. New channels								✓			✓	✓	✓	
		34. No vendor lock-in								✓			✓	✓	✓	

Πίνακας 3.3.1 Σύγκριση Εργαλείων Ανάπτυξης Διαλογικών Πρακτόρων

(https://ieeexplore.ieee.org/abstract/document/9364349)

Στον Πίνακα 3.3.1 κάνουμε διάκριση ανάμεσα σε τεχνικά χαρακτηριστικά (παράδειγμα, επεξεργασία εισόδου), τα οποία συζητώνται σε αυτήν την ενότητα, και διαχειριστικά χαρακτηριστικά (όπως το μοντέλο τιμολόγησης), με τα οποία δεν θα ασχοληθούμε στα πλαίσια αυτής της πτυχιακής εργασίας.

Η πρώτη σειρά στον Πίνακα 3.3.1 υποδεικνύει εάν το λογισμικό είναι βιβλιοθήκη, πλαίσιο, πλατφόρμα ή υπηρεσία. Οι σειρές 2–26 στον Πίνακα 3.3.1 αναλύουν καθοριστικές τεχνικές διαστάσεις κατά την επιλογή ενός εργαλείου ανάπτυξης chatbot. Αυτά περιλαμβάνουν πτυχές που σχετίζονται με την επεξεργασία του κειμένου εισαγωγής χρήστη (σειρές 2-7), την υποστήριξη διαλόγου (σειρές 8-13), την ανάπτυξη του *Chatbot* (σειρές 14-15), την ενοποίηση με άλλα συστήματα (σειρές 16-17), υποστήριξη δοκιμών και ανάπτυξης (σειρές 18–22), υποστήριξη εκτέλεσης (σειρές 23–25) και πτυχές ασφαλείας (σειρά 26).

Παρακάτω θα αναλύσουμε τις κατηγορίες που αφορούν τον πυλώνα για την σύγκριση των εργαλείων ανάπτυξης διαλογικών πρακτόρων.

3.2.1 Εισαγωγή Χρήστη

Κάποιες προσεγγίσεις επιτρέπουν τον καθορισμό των αναμενόμενων φράσεων εισόδου χρησιμοποιώντας κανονικές εκφράσεις ή πρότυπα (σειρά 2), ενώ άλλες επιτρέπουν τον καθορισμό των προθέσεων μέσω φράσεων εκπαίδευσης και στη συνέχεια εφαρμόζουν την *NLP* (σειρά 3).

Επιπλέον, πλατφόρμες όπως το *Landbot.io* υποστηρίζουν επίσης τις εισόδους των χρηστών μέσω κουμπιών και *widgets*. Οι περισσότερες προσεγγίσεις που βασίζονται στην *NLP* μπορούν να αναγνωρίσουν παραμέτρους στις φράσεις εισόδου, με εξαίρεση το *Chatfuel* και το *ChatterBot* (σειρά 4). Ένα άλλο σημαντικό στοιχείο της *NLP* είναι η υποστήριξη της γλώσσας (σειρά 5). Όλες οι προσεγγίσεις λαμβάνουν υπόψη τους κάποιες από τις πιο διαδεδομένες γλώσσες (όπως η αγγλική και η ισπανική), και ορισμένες πλατφόρμες ξεχωρίζουν για την ευρεία υποστήριξή τους στις γλώσσες (για παράδειγμα, το *DialogFlow* περιλαμβάνει 22 γλώσσες).

Ενδιαφέρον παρουσιάζει το Rasa, το οποίο μπορεί να χρησιμοποιήσει προεκπαιδευμένα μοντέλα γλώσσας (π.χ., διανυσματικά λέξεων *fastText* είναι διαθέσιμα για εκατοντάδες γλώσσες)9, αλλά οι προγραμματιστές μπορούν να εκπαιδεύσουν τα δικά τους. Λίγες προσεγγίσεις μόνο - οι υπηρεσίες *NLP LUIS, Watson, Lex, Bot Framework* και η μη δωρεάν έκδοση Enterprise του *DialogFlow* - παρέχουν ανάλυση συναισθήματος πρότασης, που μπορεί να είναι χρήσιμη σε συγκεκριμένους τομείς όπως ο μάρκετινγκ.

Τέλος, εκτός από το κείμενο, πολλές προσεγγίσεις υποστηρίζουν φυσικά την αλληλεπίδραση βασισμένη στη φωνή (σειρά 7). Αυτή η μορφή αλληλεπίδρασης θα μπορούσε να προστεθεί χειροκίνητα σε προσεγγίσεις που βασίζονται σε γλώσσες προγραμματισμού (π.χ., *Botkit*) ή που είναι ανοικτού κώδικα.

3.2.2 Διάλογος

Αυτή η διάσταση εξετάζει τις δυνατότητες οργάνωσης της ροής της συνομιλίας. Όλες οι πλατφόρμες και η πλειονότητα των πλαισίων αποθηκεύουν αυτόματα τις τιμές παραμέτρων που εξάγονται από τις φράσεις των χρηστών για να επιτρέψουν την επαναχρησιμοποίησή τους στο μέλλον, ενώ οι βιβλιοθήκες απαιτούν προγραμματισμό για αυτήν τη δυνατότητα (σειρά 8). Αυτή η αποθήκευση μπορεί να είναι προσωρινή (δηλαδή, ενεργή μόνο κατά τη διάρκεια της τρέχουσας αλληλεπίδρασης με τον χρήστη) ή μόνιμη.

Οι προθέσεις και οντότητες (σειρές 9 και 10) είναι κοινά βασικά στοιχεία πλατφορμών όπως το *DialogFlow*, το *Watson* και το *Lex*. Προσεγγίσεις που υποστηρίζουν την *NLP* καθορίζουν προθέσεις με σύνολα φράσεων εκπαίδευσης. Αυτές οι φράσεις μπορεί να είναι παραδείγματα αναμενόμενων δηλώσεων χρηστών ή φράσεις για τη βελτίωση της εμπειρίας του χρήστη και μπορεί να προέλθουν από υπάρχοντα αρχεία καταγραφής συνομιλιών (για παράδειγμα, κατά τη μετάβαση ενός παραδοσιακού συστήματος υποστήριξης πελατών σε ένα *Chatbot*).

Όσον αφορά τη δομή της συνομιλίας (σειρά 11), βρίσκουμε δύο κύρια στυλ ορισμού: **ρητά μέσω ενός δέντρου συνομιλίας** ή **έμμεσα μέσω εξαρτημένων πλαισίων και προθέσεων** που ενεργοποιούνται ανάλογα με το αν ταίριαξε με τη γονική πρόθεση (για παράδειγμα, μια πρόθεση για το κλείσιμο ραντεβού). Διαφορετικότερα, το *Pandorabots* χρησιμοποιεί τη γλώσσα σήμανσης τεχνητής

νοημοσύνης *(AI Markup Language - AIML)*. Είναι βασισμένο σε πρότυπα, αντίθετα προς τις σύγχρονες προσεγγίσεις που βασίζονται στην *NLP*. Ορισμένες πλατφόρμες επιτρέπουν επίσης τον καθορισμό φράσεων που το *Chatbot* μπορεί να χρησιμοποιήσει για την επανέναρξη της επικοινωνίας με ανενεργούς χρήστες (σειρά 12). Τέλος, όλες οι προσεγγίσεις εκτός από το *LUIS* και το *Botkit* επιτρέπουν τον καθορισμό των απαντήσεων του *Chatbot* (σειρά 13).

3.2.3 Ανάπτυξη

Ενώ κάποιες προσεγγίσεις επιτρέπουν την ανάπτυξη chatbots σε πολλά κοινωνικά δίκτυα, άλλες επικεντρώνονται σε συγκεκριμένα (σειρά 14). Για παράδειγμα, τα chatbots του *Chatfuel* είναι ειδικά σχεδιασμένα για το *Facebook Messenger*, ενώ τα *chatbot.io* μπορούν να αναπτυχθούν μόνο σε *WhatsApp Business* και ιστότοπους.

Αντίθετα, το *DialogFlow* προσφέρει ενσωματώσεις για 15 διάφορα κανάλια, συμπεριλαμβανομένων ιστοτόπων, υπηρεσιών όπως το *Skype*, έξυπνα ηχεία, και κοινωνικά δίκτυα όπως το *Slack*, το *Viber*, το *Twitter* και το *Telegram*. Βιβλιοθήκες και υπηρεσίες δεν παρέχουν επιλογές ανάπτυξης, καθώς δεν είναι στο πλαίσιο τους.

Τέλος, το *DialogFlow*, το *Bot Framework*, το *Xenioo* και το *Pandorabots* επιτρέπουν την ενσωμάτωση προσαρμοσμένων μηχανισμών αλληλεπίδρασης για το επιλεγμένο κανάλι, όπως κουμπιά στο Telegram (σειρά 15).

3.2.4 Διεπαφή

Σε πολλά εργαλεία δίνεται η δυνατότητα κλήσης υπηρεσιών για τον επιπλέον προγραμματισμό του διαλογικού πράκτορα (σειρά 16). Σε ορισμένες περιπτώσεις, όπως στο *DialogFlow*, αυτό επιτυγχάνεται συσχετίζοντας τη διεύθυνση *URL* της υπηρεσίας με μια πρόθεση, έτσι ώστε η αντιστοίχιση της πρόθεσης να ενεργοποιεί ένα μήνυμα *POST* προς την υπηρεσία.

Σε άλλες περιπτώσεις, είναι δυνατή η κατασκευή προγραμμάτων με προσαρμοσμένο κώδικα. Για τον σκοπό αυτό, το *DialogFlow* υποστηρίζει το *Cloud Functions* για το *Firebase*, ενώ το *Lex* υποστηρίζει το *Amazon* Web Services (*AWS*) *Lambda*. Αντίθετα, ορισμένες προσεγγίσεις προσφέρουν μια διεπαφή προγραμματισμού εφαρμογών (*API*) που επιτρέπει την ολοκλήρωση τμημάτων των *Chatbots* με υπάρχουσες εφαρμογές (σειρά 17). Για παράδειγμα, τα *Chatbots* του *DialogFlow* μπορούν να χρησιμοποιηθούν μέσω προγραμματισμού για να ελέγξουν την πιο πιθανή πρόθεση αντιστοίχισης δεδομένης μιας φράσης χρήστη.

3.2.5 Ανάπτυξη και Δοκιμές

Κάποια εργαλεία προσφέρουν προκατασκευασμένα στοιχεία που μπορούν να ενσωματωθούν σε νέα *Chatbots* (σειρά 18). Αυτά περιλαμβάνουν γενικά πρότυπα *Chatbot* (π.χ. για καφενεία ή συστήματα

κρατήσεων ξενοδοχείων), προκαθορισμένες προθέσεις, προκαθορισμένες μικρές συνομιλίες (π.χ. απαντήσεις σε απλές φράσεις και ερωτήσεις) ή υπηρεσίες (π.χ. δημιουργία ερωτήσεων και απαντήσεων *Chatbot* από βάση γνώσεων).

Όσον αφορά τον έλεγχο έκδοσης (σειρά 19), όλα τα πλαίσια και οι βιβλιοθήκες βασίζονται σε κώδικα και μπορούν να χρησιμοποιηθούν με οποιοδήποτε σύστημα ελέγχου έκδοσης, ενώ μόνο ορισμένες πλατφόρμες (π.χ. *DialogFlow, Watson* και *Lex*) παρέχουν εγγενή υποστήριξη για την έκδοση εκδόσεων, αν και αυτό είναι απλούστερο από τα σύγχρονα συστήματα έκδοσης όπως το *GitHub*. Όσον αφορά τη δοκιμή, οι περισσότερες προσεγγίσεις παρέχουν μια διαδικτυακή κονσόλα συνομιλίας για τη μη αυτόματη δοκιμή των *Chatbots* (σειρά 20).

Για τον εντοπισμό σφαλμάτων (σειρά 21), τα πλαίσια και οι βιβλιοθήκες μπορούν να βασίζονται στην υποστήριξη της γλώσσας προγραμματισμού, ενώ μόνο μία πλατφόρμα (*DialogFlow*) προσφέρει εγκαταστάσεις εντοπισμού σφαλμάτων για την επιθεώρηση της αντιστοιχισμένης πρόθεσης και των σχετικών πληροφοριών. Επιπλέον, το *DialogFlow* ενσωματώνει ελέγχους ποιότητας του *Chatbot*, όπως τον εντοπισμό προθέσεων με παρόμοιες φράσεις εκπαίδευσης (σειρά 22).

3.2.6 Εκτέλεση

Μόλις οριστεί ένα *Chatbot*, όλες οι πλατφόρμες και τα περισσότερα πλαίσια υποστηρίζουν την εκτέλεσή του στο cloud (σειρά 23). Αυτή η λύση μπορεί να είναι η βέλτιστη για πολλές εταιρείες, ειδικά εάν χρησιμοποιούν ήδη τις υπηρεσίες *cloud* του προμηθευτή της πλατφόρμας (για παράδειγμα, *Google, Azure* - σύννεφο της Microsoft για το *Bot Framework* και *LUIS* - ή *AWS*). Ωστόσο, αυτό μπορεί να μην είναι πάντα κατάλληλο. Σε ορισμένες περιπτώσεις, όπως η *Watson*, υπάρχει ένα ειδικό σχέδιο τιμολόγησης για την ανάπτυξη του *Chatbot* σε *cloud* τρίτων.

3.2.7 Ασφάλεια

Τα *Chatbots*, ιδίως όταν χειρίζονται προσωπικά δεδομένα χρήστη, πρέπει να ενσωματώνουν πτυχές ασφάλειας. Ενώ η υλοποίηση των δυνατοτήτων ασφάλειας είναι ευθύνη των προγραμματιστών, ορισμένα εργαλεία παρέχουν επιπλέον επίπεδα ασφαλείας πάνω από το cloud όπου αναπτύσσεται το *Chatbot* (σειρά 26).

Συγκεκριμένα, τα *DialogFlow, Watson, Lex* και *Azure* προσφέρουν δυνατότητες όπως τείχη προστασίας, έλεγχο ταυτότητας και εξουσιοδότηση μέσω *API*, και ασφαλείς συνδέσεις μέσω πρωτοκόλλων όπως το Secure Sockets Layer *(SSL)* ή το HTTPS/Transport Layer Security *(TLS)*. Σε αντίθεση, προσεγγίσεις χωρίς υπηρεσίες ανάπτυξης δεν προσφέρουν αυτή την ενσωματωμένη δυνατότητα. Επιπλέον, κοινωνικά δίκτυα όπως το *WhatsApp* ή το *Telegram* υποστηρίζουν κρυπτογράφηση μηνυμάτων και έλεγχο ταυτότητας χρήστη.

3.3 Επίλογος

Συνοπτικά, το κεφάλαιο αυτό προσφέρει μια εμβαθυσμένη ανάλυση των χαρακτηριστικών και των τεχνικών πτυχών που σχετίζονται με την ανάπτυξη *Chatbots*. Καταδεικνύεται η διάκριση μεταξύ των τεχνικών και διαχειριστικών χαρακτηριστικών, ενώ παρουσιάζονται σε λεπτομέρεια τα στοιχεία που επηρεάζουν την επιλογή ενός εργαλείου ανάπτυξης chatbot.

Αναλύονται επιπλέον οι διάφορες διαστάσεις, όπως η επεξεργασία της εισόδου χρήστη, η υποστήριξη διαλόγου, η ανάπτυξη, η ολοκλήρωση με άλλα συστήματα, η δοκιμή και η ασφάλεια. Κάθε πτυχή εξετάζεται σε βάθος, αναδεικνύοντας τα πλεονεκτήματα και τις προκλήσεις που σχετίζονται με την επιλογή κατάλληλου εργαλείου ανάπτυξης.

Κλείνοντας, με αυτό τον τρόπο, το κεφάλαιο παρέχει μια πλήρη εικόνα του τοπίου ανάπτυξης chatbot, επιτρέποντας στους αναγνώστες να κατανοήσουν τις τεχνικές διαφορές και τα ιδιαίτερα χαρακτηριστικά κάθε εργαλείου.

Κεφάλαιο 4ο: Επιλογή Πλατφόρμας Ανάπτυξης

Το τέταρτο κεφάλαιο της πτυχιακής εργασίας αφορά την "Επιλογή πλατφόρμας ανάπτυξης" και αποτελεί ένα από τα βασικά μαθησιακά σημεία της έρευνάς μας. Σε αυτό το κεφάλαιο, θα εξετάσουμε διεξοδικά τη δημοφιλή πλατφόρμα ανάπτυξης *DialogFlow* της *Google*, αναδεικνύοντας τις δυνατότητες και λειτουργίες της που την καθιστούν ένα ισχυρό εργαλείο για τη δημιουργία διαλογικών πρακτόρων.

Η επιλογή της κατάλληλης πλατφόρμας ανάπτυξης αποτελεί κρίσιμη απόφαση σε κάθε έργο που σχετίζεται με την δημιουργία διαλογικών πρακτόρων. Σε αυτή την πορεία, η πλατφόρμα *DialogFlow* ξεχωρίζει ως μια πρωτοποριακή λύση, όπως με βάση τα κριτήρια αξιολόγησης που είδαμε πριν, που επιτρέπει σε προγραμματιστές την ανάπτυξη διαλογικών πρακτόρων και τη δημιουργία εφαρμογών που επικοινωνούν με τους χρήστες μέσω φυσικής γλώσσας.

Αυτό το κεφάλαιο θα εξετάσει λεπτομερώς την πλατφόρμα *DialogFlow*, ξεκινώντας με την παρουσίασή της και την εξήγηση του γιατί είναι αξιόλογη για την ανάπτυξη διαλογικών πρακτόρων. Στη συνέχεια, θα εξετάσουμε αναλυτικά τις βασικές λειτουργίες και δυνατότητες που προσφέρει, αναδεικνύοντας παραδείγματα πρακτικής εφαρμογής και προσφέροντας οδηγίες βήμα-προς-βήμα για τη χρήση της πλατφόρμας.

Με την επιλογή της συγκεκριμένης πλατφόρμας, ο αναγνώστης θα έχει τη δυνατότητα να αναπτύξει, να προσαρμόσει και να διαμορφώσει διαλογικούς πράκτορες σύμφωνα με τις ανάγκες του. Αυτό το κεφάλαιο αποσκοπεί στην κατανόηση και αξιοποίηση του *DialogFlow* ως εργαλείου για την ανάπτυξη προηγμένων και αποτελεσματικών διαλογικών πρακτόρων.

4.1 Εξέταση της Πλατφόρμας DialogFlow της Google

Η αποτελεσματική ανάπτυξη διαλογικών πρακτόρων αποτελεί καίριο στάδιο στην επιτυχημένη υλοποίηση πολλών εφαρμογών και υπηρεσιών στον τομέα της τεχνητής νοημοσύνης και της αλληλεπίδρασης ανθρώπου-μηχανής. Σε αυτό το πλαίσιο, αυτή η ενότητα εστιάζει σε μια από τις κορυφαίες πλατφόρμες διαλόγου, την *DialogFlow* της *Google*.

Η υιοθέτηση ενός διαλογικού πράκτορα στο πλαίσιο μιας εφαρμογής απαιτεί μια πλατφόρμα που να ενσωματώνει προηγμένη επεξεργασία φυσικής γλώσσας και δυνατότητες αυτόματης ανταπόκρισης σε ερωτήσεις και εντολές των χρηστών. Αυτή ακριβώς είναι η πρόκληση που αντιμετωπίζει η πλατφόρμα *DialogFlow*, παρέχοντας ένα ολοκληρωμένο σύστημα για την κατανόηση και απόκριση σε ανθρώπινη γλώσσα.

Στο πλαίσιο αυτού του κεφαλαίου, θα εξετάσουμε βαθύτερα την πλατφόρμα *DialogFlow* της *Google*. Αρχικά, θα εξηγήσουμε τι είναι το *DialogFlow* και ποιος είναι ο σκοπός της χρήσης της. Θα δούμε πώς αυτή η πλατφόρμα διευκολύνει τη δημιουργία διαλογικών πρακτόρων και ποια είναι τα βασικά στοιχεία της. Στη συνέχεια, θα αναλύσουμε αναλυτικά τις δυνατότητες και τις λειτουργίες της πλατφόρμας, παρέχοντας παραδείγματα και βήματα για την αξιοποίησή τους. Στην πορεία του κεφαλαίου, θα εξετάσουμε πώς μπορείτε να δημιουργήσετε τον δικό σας διαλογικό πράκτορα με τη χρήση του *DialogFlow*. Θα παρέχουμε σαφείς οδηγίες για την εγκατάσταση και τη ρύθμιση του περιβάλλοντος ανάπτυξης, καθώς και τη δημιουργία των πρώτων διαλόγων με τους χρήστες.

Η ενότητα αυτή αποτελεί έναν πολύτιμο οδηγό για όσους επιθυμούν να αξιοποιήσουν το *DialogFlow* της *Google* για τη δημιουργία διαλογικών πρακτόρων με εξαιρετική ακρίβεια και αποτελεσματικότητα. Από την αρχική επεξήγηση μέχρι την πρακτική εφαρμογή, αυτό το κεφάλαιο καλύπτει κάθε βήμα για την κατανόηση και την αξιοποίηση του *DialogFlow* και των δυνατοτήτων του.

4.1.1 Εισαγωγή στο DialogFlow

Ο διαλογικός πράκτορας αποτελεί τον πυρήνα του *DialogFlow* και είναι το ενεργό στοιχείο που αλληλεπιδρά με τους χρήστες μέσω φυσικής γλώσσας. Είναι ένα προηγμένο σύστημα που αντλεί την ιδέα του από τον τρόπο που οι άνθρωποι επικοινωνούν μεταξύ τους, με σκοπό να επιτρέψει την φυσική και εύκολη αλληλεπίδραση ανθρώπου-υπολογιστή. Ο διαλογικός πράκτορας λειτουργεί με δύο βασικούς τρόπους: την **αναγνώριση και την παραγωγή**. Κατά την αναγνώριση, ερμηνεύει τη φυσική γλώσσα που χρησιμοποιεί ο χρήστης και τη μετατρέπει σε δομημένες προθέσεις και οντότητες, κατανοώντας έτσι την πρόθεση και τα αιτήματα του. Κατά την παραγωγή, δημιουργεί αποτελεσματικές απαντήσεις στους χρήστες, οι οποίες αποτελούν την απόκριση του συστήματος.

Ο διαλογικός πράκτορας αλληλεπιδρά με τους χρήστες με τον τρόπο που θα αναμέναμε να επικοινωνούμε με έναν άνθρωπο, αξιοποιώντας τη φυσική γλώσσα ως μέσο επικοινωνίας. Αυτό δίνει την αίσθηση ότι ο χρήστης διαδραματίζει ένα πραγματικό διάλογο, κάνοντας την αλληλεπίδραση με τον υπολογιστή πιο φυσική και άνετη. Όταν ο χρήστης ξεκινά την αλληλεπίδραση, εκφράζει το αίτημά του χρησιμοποιώντας φυσική γλώσσα. Ο διαλογικός πράκτορας, μέσω της δυνατότητας της αναγνώρισης προθέσεων και οντοτήτων, αντιλαμβάνεται τον σκοπό της αναζήτησης του χρήστη. Στη

συνέχεια, απαντά με μια συνεκτική και ενημερωτική απάντηση, προσαρμοσμένη στο περιεχόμενο της ερώτησης.

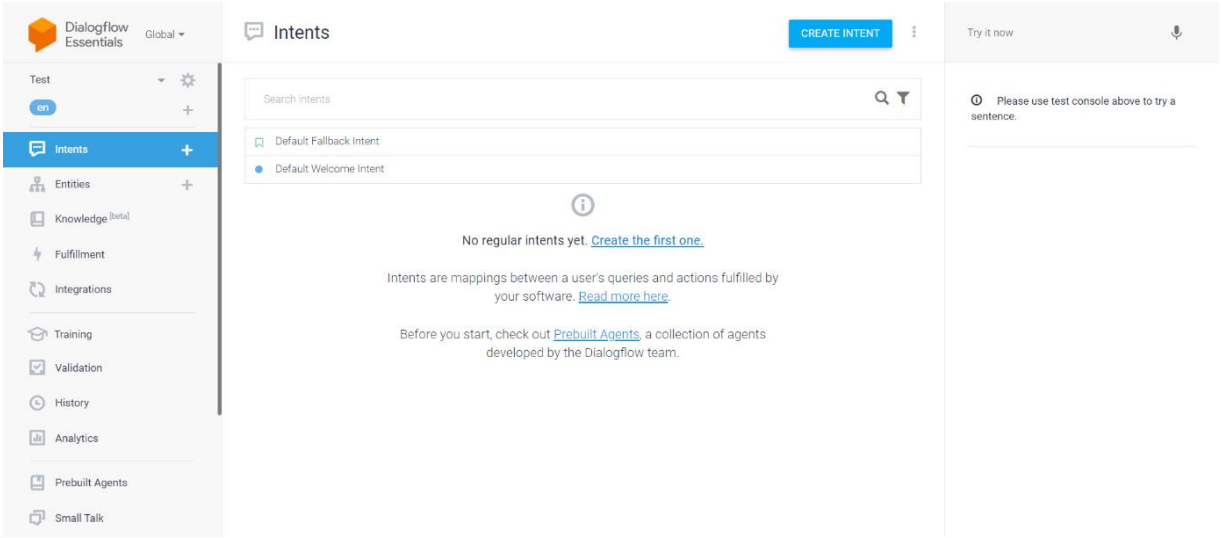
Είναι σημαντικό να τονίσουμε ότι ο διαλογικός πράκτορας δεν παρέχει απλά στατικές απαντήσεις, αλλά μπορεί να προσαρμόζεται στον συγκεκριμένο χρήστη και το περιβάλλον του. Με την χρήση τεχνικών μηχανικής μάθησης, ο διαλογικός πράκτορας μπορεί να εξατομικεύει τις απαντήσεις του βάσει των προηγούμενων αλληλεπιδράσεων και των προτιμήσεων του χρήστη. Αυτό δημιουργεί μια αίσθηση προσωπικής και εξατομικευμένης εμπειρίας για τον χρήστη. Επομένως, η αλληλεπίδραση με τους χρήστες γίνεται φυσική, εύκολη και αποτελεσματική μέσω της φυσικής γλώσσας, επιτρέποντας την άνετη επικοινωνία μεταξύ ανθρώπου και υπολογιστή.

Ο ρόλος της φυσικής γλώσσας σε αυτήν τη διαδικασία είναι ζωτικής σημασίας. Ο διαλογικός πράκτορας είναι σχεδιασμένος να κατανοεί και να χειρίζεται την ανθρώπινη γλώσσα με τρόπο φυσικό και ευέλικτο. Αντιλαμβάνεται τα διάφορα είδη γλωσσικής παραλλαγής, συναισθηματικής φόρτισης και ακόμα και τον λογοτεχνικό ύφος. Αυτό επιτρέπει την επικοινωνία με το σύστημα με τρόπο που να αισθάνεται φυσικός και άνετος για τον χρήστη.

Ο διαλογικός πράκτορας, λοιπόν, αποτελεί τον κυρίαρχο σύνδεσμο ανάμεσα στον ανθρώπινο χρήστη και τον κόσμο της τεχνολογίας. Με την αντιστοίχιση προθέσεων, την επεξεργασία οντοτήτων και την ευφυή απόκριση, διαμορφώνει μια δυναμική διαδραστική εμπειρία που ενισχύει την αποτελεσματικότητα και την ικανοποίηση των χρηστών.

4.1.2 Δυνατότητες του DialogFlow

Ο *DialogFlow* παρέχει πληθώρα λειτουργιών για τη δημιουργία προηγμένων διαλογικών πρακτόρων. Παρακάτω θα δείτε μια εικόνα του περιβάλλοντος εργασίας του *DialogFlow*, μαζί με τις αντίστοιχες εξηγήσεις για κάθε σημαντικό στοιχείο, έτσι ώστε στην επόμενη ενότητα να προχωρήσουμε στην δημιουργία του διαλογικού πράκτορα που έχουμε επιλέξει:



Εικόνα 4.1.2.1 DialogFlow

Intents (Σκοποί): Οι *Intents* είναι ο βασικός μηχανισμός μέσω του οποίου ο πράκτορας του Dialogflow αναγνωρίζει και κατανοεί τις προθέσεις των χρηστών. Ουσιαστικά, το *Intent* καθορίζει τον τρόπο με τον οποίο ο πράκτορας θα ανταποκρίνεται στις διάφορες ερωτήσεις και αιτήσεις που λαμβάνει.

Κάθε *Intent* αντιπροσωπεύει μια συγκεκριμένη πρόθεση ή επιθυμία του χρήστη. Για παράδειγμα, αν ένας χρήστης λέει "Πείτε μου τον καιρό σήμερα," υπάρχει ένας *Intent* που είναι σχεδιασμένος να αναγνωρίσει αυτήν την επιθυμία για πληροφορίες για τον καιρό. Ο σκοπός *(Intent)* είναι προκαθορισμένος και δεν αναγνωρίζει απλώς τις λέξεις, αλλά κατανοεί την σκοπιμότητα πίσω από την ερώτηση.

Για να διδαχτεί ο πράκτορας πώς να αναγνωρίζει συγκεκριμένες προθέσεις, χρησιμοποιείται εκπαίδευση με βάση παραδείγματα. Αναλύοντας εκατοντάδες ή και χιλιάδες παραδείγματα επικοινωνίας με τους χρήστες, ο πράκτορας μαθαίνει τις πιθανές εκφράσεις που σχετίζονται με κάθε Intent. Ακόμα, μπορούν να οριστούν παράμετροι για κάθε *Intent*, που αναγνωρίζουν στοιχεία όπως την ημερομηνία, τον τόπο, τα ονόματα κ.λπ.

Αυτό επιτρέπει στον πράκτορα να προσδιορίσει πληροφορίες από τον διάλογο με τον χρήστη και να τις χρησιμοποιήσει για πιο εξατομικευμένες απαντήσεις. Συνολικά, οι *Intents* είναι το κύριο μέσο για την αναγνώριση των προθέσεων των χρηστών και την κατανόηση του τι θέλουν να επιτύχουν μέσω της αλληλεπίδρασης.

Entities (Οντότητες): Οι οντότητες *(Entities)* αποτελούν ένα ακόμα σημαντικό στοιχείο του *DialogFlow* που συμβάλλει στην καλύτερη κατανόηση της φυσικής γλώσσας και τη διευκόλυνση της διαδικασίας επικοινωνίας με τον διαλογικό πράκτορα. Ουσιαστικά, οι οντότητες είναι **κλειδιά έννοιες** ή στοιχεία τα οποία μπορεί να εμφανίζονται στις φράσεις του χρήστη και που πρέπει να αναγνωρίζονται για την καλύτερη επεξεργασία του διαλόγου.

Για να το κατανοήσουμε καλύτερα, ας υποθέσουμε ότι δημιουργούμε έναν διαλογικό πράκτορα για παραγγελίες φαγητού. Ο χρήστης μπορεί να πει "Θέλω να παραγγείλω μια πίτσα με παρμεζάνα και φέτα". Σε αυτήν τη φράση, οι οντότητες είναι οι διάφορες κατηγορίες τροφίμων όπως "πίτσα", "παρμεζάνα" και "φέτα". Κάθε οντότητα αντιστοιχεί σε μια συγκεκριμένη κατηγορία που είναι σημαντική για τον σκοπό της επεξεργασίας της παραγγελίας. Η δημιουργία και ορισμός των οντοτήτων επιτρέπει στον διαλογικό πράκτορα να αναγνωρίζει τις κατηγορίες αυτές στις φράσεις των χρηστών και να τις εξάγει για περαιτέρω επεξεργασία.

Για το παράδειγμα μας, ο διαλογικός πράκτορας θα κατανοήσει ότι ο χρήστης θέλει μια πίτσα με τα συγκεκριμένα υλικά που αναφέρθηκαν. Οι οντότητες δημιουργούνται και προσαρμόζονται ανάλογα με την εφαρμογή και τον σκοπό του διαλογικού πράκτορα. Η προσθήκη τους βελτιώνει την ακρίβεια της αναγνώρισης και την απόκριση στις εντολές των χρηστών, καθιστώντας την διαδικασία πιο φυσική και ευέλικτη.

Fulfillment: Το *Fulfillment* στο *DialogFlow* αναφέρεται στη δυνατότητα παροχής απαντήσεων και εκτέλεσης ενεργειών πέρα από τις απλές στατικές απαντήσεις που μπορεί να προσφέρει ο διαλογικός πράκτορας. Με άλλα λόγια, το *Fulfillment* επιτρέπει τη διασύνδεση του διαλογικού πράκτορα με εξωτερικές υπηρεσίες, βάσεις δεδομένων, *API* και λοιπές πηγές δεδομένων, προκειμένου να παράγει δυναμικές και εξατομικευμένες απαντήσεις. Αναλυτικότερα, το *Fulfillment* επιτρέπει την εξής λειτουργία:

- **Εκτέλεση Ενεργειών:** Μέσω του *Fulfillment*, μπορείτε να δημιουργήσετε προσαρμοσμένες ενέργειες που ο διαλογικός πράκτορας μπορεί να εκτελέσει κατόπιν απόφασης του χρήστη. Για παράδειγμα, αν κάποιος χρήστης επιλέξει να παραγγείλει ένα προϊόν, το *Fulfillment* μπορεί να αναλάβει τη διαδικασία της παραγγελίας μέσω ενσωματωμένων λειτουργιών ή με την επικοινωνία με εξωτερικά συστήματα.

- **Δυναμικές Απαντήσεις:** Το *Fulfillment* επιτρέπει τη δημιουργία δυναμικών απαντήσεων που βασίζονται σε δεδομένα από εξωτερικές πηγές. Μπορείτε να επιστρέψετε πληροφορίες από βάσεις δεδομένων, να υπολογίσετε τιμές, να εμφανίσετε δεδομένα από *APIs* και άλλα.

- **Προσαρμοσμένες Ενέργειες:** Μπορείτε να δημιουργήσετε προσαρμοσμένες ενέργειες που ανταποκρίνονται σε συγκεκριμένες εντολές των χρηστών. Για παράδειγμα, αν κάποιος χρήστης ζητήσει πληροφορίες για τον καιρό, το *Fulfillment* μπορεί να ενεργοποιήσει τη σχετική υπηρεσία και να επιστρέψει τα αποτελέσματα.

Συνολικά, το *Fulfillment* προσθέτει βάθος και ευελιξία στις δυνατότητες του διαλογικού πράκτορα, επιτρέποντας την επικοινωνία με εξωτερικά συστήματα και τη δημιουργία πλούσιων και εξατομικευμένων απαντήσεων.

Integrations: Η λειτουργία *Integrations* στο *DialogFlow* αναφέρεται στη δυνατότητα του προγράμματος να συνδεθεί και να ενσωματωθεί με διάφορες εξωτερικές πλατφόρμες και υπηρεσίες. Βασικά, οι *Integrations* επιτρέπουν στον διαλογικό πράκτορα που δημιουργήθηκε στο *DialogFlow* να επικοινωνεί με άλλα συστήματα, παρέχοντας έναν τρόπο για τη ροή πληροφοριών μεταξύ αυτών των πλατφορμών και του πράκτορα.

Οι **Integrations** παρέχουν το εξής:

- **Διασύνδεση με Πλατφόρμες Επικοινωνίας:** Μπορείτε να συνδέσετε τον διαλογικό σας πράκτορα με διάφορες πλατφόρμες επικοινωνίας, όπως το *Facebook Messenger*, το *Slack*, το *WhatsApp* και άλλα. Αυτό επιτρέπει στον πράκτορα να αλληλεπιδρά με τους χρήστες μέσα από τις αγαπημένες τους εφαρμογές επικοινωνίας.

- **Ενσωμάτωση σε Ιστοσελίδες:** Μπορείτε να ενσωματώσετε τον διαλογικό πράκτορα σε ιστοσελίδες, παρέχοντας ένα *Chatbot* ή ένα *widget* επικοινωνίας που επιτρέπει στους επισκέπτες της ιστοσελίδας να αλληλεπιδρούν με τον πράκτορα.

- **Ενσωμάτωση με Εφαρμογές:** Μπορείτε να ενσωματώσετε τον διαλογικό πράκτορα σε εφαρμογές, επιτρέποντας την αλληλεπίδραση με τον πράκτορα μέσω εφαρμογών.

- **Δυναμικές Ενέργειες:** Οι *Integrations* επιτρέπουν την εκτέλεση δυναμικών ενεργειών με βάση τις ανάγκες του χρήστη. Μπορείτε να επιτρέψετε στον πράκτορα να προσφέρει λειτουργίες όπως αναζήτηση πληροφοριών, κρατήσεις, παραγγελίες και άλλα μέσω των συνδεδεμένων πλατφορμών.

Οι *Integrations* επεκτείνουν τη λειτουργικότητα του διαλογικού πράκτορα σε πολλαπλές πλατφόρμες, ενσωματώνοντας τον σε διαφορετικά περιβάλλοντα επικοινωνίας και προσφέροντας ευκολία και επιλογές στους χρήστες για αλληλεπίδραση με τον πράκτορα.

Training: Η διαδικασία του *Training* στο *DialogFlow* αναφέρεται στην αυτόματη αλλά και χειροκίνητη προσαρμογή του μοντέλου της τεχνητής νοημοσύνης, που βρίσκεται πίσω από τον διαλογικό πράκτορα. Ουσιαστικά, το μοντέλο μαθαίνει είτε αυτόματα από τις παρεχόμενες φράσεις για εκπαίδευση στο κάθε *Intent* είτε από χειροκίνητο ορισμό κάποιων -ως τότε μη αναγνωρίσιμων - φράσεων σαν ταυτόσημες ήδη γνωστών εννοιών και προσαρμόζεται ώστε να βελτιώσει την ακρίβεια και την αναγνώριση των προθέσεων και των οντοτήτων από τις εισερχόμενες αλληλεπιδράσεις των χρηστών.

Κατά την διάρκεια της διαδικασίας *Training*, το μοντέλο αναλύει τα παραδείγματα εκπαίδευσης που του έχουν δοθεί για τους σκοπούς και τις οντότητες και προσπαθεί να εντοπίσει μοτίβα και συσχετίσεις μεταξύ των διαφορετικών λέξεων, φράσεων και ενεργειών. Στη συνέχεια, το μοντέλο προσαρμόζεται και βελτιώνεται ώστε να κατανοεί και να ανταποκρίνεται καλύτερα στα μοτίβα και τις προθέσεις των χρηστών.

Η διαδικασία του Training είναι συνεχής κατά βάση αυτόματη αλλά και χειροκίνητη. Καθώς λαμβάνονται περισσότερες αλληλεπιδράσεις με τους χρήστες και προστίθενται περισσότερα δεδομένα εκπαίδευσης, το μοντέλο συνεχίζει να βελτιώνεται και να προσαρμόζεται για να εξυπηρετεί αποτελεσματικότερα τις ανάγκες του διαλογικού πράκτορα.

Validation: Στο πλαίσιο του *DialogFlow*, ο όρος *Validation* αναφέρεται στη διαδικασία ελέγχου και επαλήθευσης του διαλογικού πράκτορα και των ρυθμίσεών του προτού τεθεί σε παραγωγική λειτουργία. Ουσιαστικά, η διαδικασία αυτή εξασφαλίζει ότι ο πράκτορας είναι έτοιμος να αλληλεπιδράσει με τους χρήστες με τον τρόπο που έχει καθοριστεί και να παρέχει ακριβείς και συνεπείς απαντήσεις.

Κατά τη διαδικασία του *Validation*, ελέγχεται η ορθότητα και η λειτουργία των εισόδων και των ενεργειών του πράκτορα. Αυτό περιλαμβάνει τον έλεγχο των σκοπών *(intents)* και των οντοτήτων, των απαντήσεων που έχουν καθοριστεί για κάθε σκοπό, καθώς και των διαδικασιών που συνδέουν τον πράκτορα με τυχόν εξωτερικούς πόρους ή υπηρεσίες. Επίσης, ελέγχονται οι αντίστοιχες προγραμματιζόμενες λειτουργίες *(fulfillment)* που ενεργοποιούνται κατά τη διάρκεια των αλληλεπιδράσεων με τους χρήστες.

Ο στόχος του *Validation* είναι να διασφαλίσει ότι ο διαλογικός πράκτορας λειτουργεί σωστά, παρέχει ακριβείς και συνεπείς απαντήσεις και είναι έτοιμος να ανταποκρίνεται στις ανάγκες των χρηστών. Αν κατά τη διαδικασία του *Validation* εντοπιστούν προβλήματα ή ανακρίβειες, τότε πρέπει να προβείτε σε διορθώσεις και βελτιώσεις για να εξασφαλίσετε την αποτελεσματικότητα και την ακρίβεια του πράκτορα πριν τον καθορίσετε ως έτοιμο για παραγωγική λειτουργία.

History: Το *History* στο πλαίσιο του *DialogFlow* αναφέρεται στη δυνατότητα παρακολούθησης και ανάκτησης του ιστορικού των αλληλεπιδράσεων μεταξύ του διαλογικού πράκτορα και των χρηστών. Αυτή η λειτουργία επιτρέπει την προβολή των προηγούμενων διαλόγων και

αλληλεπιδράσεων που έχουν λάβει χώρα με τον πράκτορα, προκειμένου να αναλυθεί η ροή της συνομιλίας, οι απαντήσεις που παρείχε ο πράκτορας και οι επιλογές των χρηστών.

Μέσω της λειτουργίας *History*, μπορεί κάποιος να παρακολουθεί την εξέλιξη των συνομιλιών και να εξετάζει το πώς ο διαλογικός πράκτορας ανταποκρίνεται σε διαφορετικά σενάρια. Επιπλέον, μπορούν να εξεταστούν οι απαντήσεις που παρείχε ο πράκτορας, οι ερωτήσεις των χρηστών και οι επιλογές που παρουσίασαν στο πλαίσιο της συνομιλίας.

Η λειτουργία *History* είναι χρήσιμη για την ανάλυση της απόκρισης του πράκτορα, τον εντοπισμό προτύπων συνομιλίας και τη βελτίωση του πράκτορα βάσει των προηγούμενων διαλόγων. Μπορεί έτσι να διαγραφεί ή να καταστεί διαχειρίσιμο το ιστορικό των αλληλεπιδράσεων ανάλογα με τις ανάγκες που υπάρχουν.

Analytics: Το *Analytics* στο πλαίσιο του *DialogFlow* αναφέρεται στη λειτουργία που προσφέρει αναλυτικές πληροφορίες και στατιστικά δεδομένα σχετικά με τις αλληλεπιδράσεις και τη χρήση του διαλογικού πράκτορα. Αυτή η λειτουργία επιτρέπει στους δημιουργούς να παρακολουθούν και να αναλύουν πληροφορίες σχετικά με το πώς οι χρήστες αλληλεπιδρούν με τον πράκτορα και πώς λειτουργεί ο πράκτορας γενικά.

Μέσω της λειτουργίας *Analytics*, παρακολουθούνται μετρικές όπως ο αριθμός των ερωτήσεων, οι πιο συχνές ερωτήσεις, οι αναγνωρισμένες οντότητες, η επίδοση του πράκτορα σε διαφορετικά σενάρια και πολλά άλλα. Αυτά τα δεδομένα μπορούν να βοηθήσουν να κατανοηθεί η συμπεριφορά των χρηστών, να προσδιοριστούν τυχόν προβλήματα ή αδυναμίες στον πράκτορα και να λυφθούν αποφάσεις για τη βελτιστοποίηση της εμπειρίας των χρηστών.

Επιπλέον, μέσω της λειτουργίας *Analytics*, μπορεί κανείς να παρακολουθεί την απόδοση του πράκτορα σε πραγματικό χρόνο, να ανακαλύπτει ποια μέρη της συνομιλίας είναι πιο επιτυχημένα και να προσαρμόζει τον πράκτορα βάσει των στοιχείων που συλλέγονται.

Prebuilt Agents: Το *Prebuilt Agents* αναφέρεται σε προκατασκευασμένους πράκτορες που παρέχονται από το *DialogFlow* και έχουν δημιουργηθεί εκ των προτέρων για συγκεκριμένους τομείς ή χρήσεις. Αυτοί οι πράκτορες περιέχουν προεργασμένες εντολές, οντότητες, κανόνες και παραμέτρους που σχετίζονται με τον συγκεκριμένο τομέα, προσφέροντας μια βασική βάση για την δημιουργία προκατασκευασμένων διαλογικών πρακτόρων.

Η χρήση των προκατασκευασμένων πρακτόρων μπορεί να επιταχύνει τη διαδικασία ανάπτυξης, καθώς δεν απαιτείται η δημιουργία κάθε στοιχείου από την αρχή. Οι προκατασκευασμένοι πράκτορες προσφέρουν ήδη τη δομή και τα βασικά στοιχεία που απαιτούνται για να ξεκινήσει η δημιουργία του εκάστοτε διαλογικού πράκτορα, ενώ μπορεί επίσης να γίνει προσαρμογή και προσθήκη νέων λειτουργιών ανάλογα με τις απαιτήσεις του κατασκευαστή.

Το *Prebuilt Agents* παρέχει μια ευκολότερη αρχική ανάπτυξη για συγκεκριμένους τομείς, όπως αεροπορία, κρατήσεις ξενοδοχείων, πληροφορίες για εστιατόρια και άλλα.

Small Talk: Το *Small Talk* στο πλαίσιο του *DialogFlow* αναφέρεται σε προκαθορισμένες φράσεις και απαντήσεις που μπορούν να χρησιμοποιηθούν για ανοιχτή και φυσική επικοινωνία με τον διαλογικό πράκτορα, πέραν των κανονικών λειτουργιών του.

Συγκεκριμένα, το *Small Talk* περιλαμβάνει συνομιλιακές φράσεις που συχνά εμφανίζονται σε καθημερινές συζητήσεις και αναφορές σε θέματα όπως ο καιρός, ο χαιρετισμός, η ευχαρίστηση, η απάντηση σε ερωτήσεις για τον εαυτό του πράκτορα, η αναγνώριση πληροφοριών για την εφαρμογή και άλλα παρόμοια.

Η προσθήκη του *Small Talk* στον διαλογικό πράκτορα επιτρέπει στον πράκτορα να ανταποκρίνεται με πιο ανθρώπινο τρόπο και να δημιουργεί μια πιο φυσική αίσθηση στην επικοινωνία με τον χρήστη, παρέχοντας του τη δυνατότητα να αντιμετωπίζει πιο ανοιχτές συζητήσεις και ερωτήσεις.

4.2 Ασφάλεια και Προστασία Δεδομένων στο DialogFlow

Η ασφάλεια και η προστασία των δεδομένων αποτελούν θεμελιώδες ζήτημα κατά την ανάπτυξη διαλογικών πρακτόρων με χρήση της πλατφόρμας *DialogFlow* της *Google*. Στο πλαίσιο αυτής της ενότητας, θα εξετάσουμε τις διάφορες πτυχές της ασφάλειας και της προστασίας δεδομένων στο *DialogFlow* και πώς η πλατφόρμα διασφαλίζει την ασφαλή και αξιόπιστη λειτουργία των διαλογικών πρακτόρων.

4.2.1 Διαχείριση Δεδομένων Χρηστών

Η διαχείριση των δεδομένων χρηστών αποτελεί βασικό κομμάτι της ασφάλειας και προστασίας δεδομένων στην πλατφόρμα *DialogFlow* της *Google*. Καθώς οι χρήστες αλληλεπιδρούν με τους διαλογικούς πράκτορες, παρέχοντας πληροφορίες και δεδομένα, η αναγνώριση και εξασφάλιση της ασφάλειάς τους αποκτούν κρίσιμη σημασία.

Ο *DialogFlow* υιοθετεί μια σειρά από προληπτικά μέτρα για να διασφαλίσει ότι τα προσωπικά δεδομένα των χρηστών είναι ασφαλή. Οι πληροφορίες που συλλέγονται κατά την διάρκεια των διαλόγων διαχειρίζονται με αυστηρότητα και χρησιμοποιούνται αποκλειστικά για τους προκαθορισμένους σκοπούς, ενώ δεν διατίθενται σε τρίτους χωρίς τη συγκατάθεση του χρήστη.

Οι υπηρεσίες του *DialogFlow* διασφαλίζουν την ευελιξία στον τρόπο που χειρίζονται τα δεδομένα. Οι διαχειριστές μπορούν να προσαρμόσουν τις ρυθμίσεις προκειμένου να πληρούνται τα απαιτούμενα πρότυπα ασφαλείας και προστασίας δεδομένων για κάθε συγκεκριμένη εφαρμογή.

Επιπλέον, η πλατφόρμα εφαρμόζει μέτρα κρυπτογράφησης δεδομένων προκειμένου να εξασφαλίσει ότι οι πληροφορίες παραμένουν ασφαλείς κατά τη διάρκεια της αποθήκευσης και της μετάδοσής τους. Αυτό διασφαλίζει ότι ακόμη και σε περίπτωση διαρροής, τα δεδομένα παραμένουν αδύνατα στην αποκρυπτογράφηση.

Στον τομέα της διαχείρισης πρόσβασης, ο *DialogFlow* επιτρέπει την ανάθεση διαφορετικών επιπέδων πρόσβασης σε διαφορετικούς χρήστες. Αυτό εξασφαλίζει ότι μόνο οι εξουσιοδοτημένοι χρήστες έχουν πρόσβαση σε συγκεκριμένα δεδομένα, διασφαλίζοντας έτσι την εμπιστευτικότητα και την ασφάλεια των πληροφοριών.

Συνολικά, η διαχείριση δεδομένων χρηστών στο *DialogFlow* αντιπροσωπεύει έναν συνεκτικό και προηγμένο τρόπο διασφάλισης της ασφάλειας και της προστασίας των προσωπικών δεδομένων των

χρηστών, συμβάλλοντας στην αναβαθμισμένη εμπειρία αλληλεπίδρασης με τους διαλογικούς πράκτορες.

4.2.2 Προστασία Προσωπικών Δεδομένων

Η προστασία των προσωπικών δεδομένων αποτελεί προτεραιότητα ζωτικής σημασίας για το *DialogFlow*, την πρωτοπόρο πλατφόρμα επικοινωνίας με τους διαλογικούς πράκτορες. Καθώς η τεχνολογία ενσωματώνεται ολοένα και περισσότερο στην καθημερινή μας ζωή, η ανάγκη προστασίας των προσωπικών μας δεδομένων από παραβιάσεις και κατάχρηση γίνεται εξίσου επιτακτική.

Στο εσωτερικό της πλατφόρμας *DialoFlow*, εφαρμόζονται αυστηρά πρότυπα ασφάλειας για την διασφάλιση της εμπιστοσύνης των χρηστών. Ενώ η ανταλλαγή πληροφοριών απαιτείται για την απρόσκοπτη λειτουργία των διαλογικών πρακτόρων, η πλατφόρμα διασφαλίζει ότι η πρόσβαση και χρήση αυτών των δεδομένων περιορίζονται αυστηρά στους οριοθετημένους σκοπούς που έχουν δηλωθεί.

Η τεχνική υποδομή προστασίας περιλαμβάνει την κρυπτογράφηση των δεδομένων, την εφαρμογή προηγμένων μηχανισμών αυθεντικοποίησης και τον περιοδικό έλεγχο της πρόσβασης. Επιπλέον, διατηρείται συνεχής εποπτεία για τον εντοπισμό και την αντιμετώπιση απειλών ασφαλείας.

Είναι σημαντικό να αναγνωρίσουμε ότι η προστασία των προσωπικών δεδομένων είναι μια συνεχής διαδικασία βελτίωσης και προσαρμογής.

Η πλατφόρμα *DialogFlow* δεσμεύεται να διατηρεί υψηλά πρότυπα ασφαλείας και να αντιμετωπίζει τις εξελισσόμενες απειλές της ψηφιακής εποχής. Με αυτόν τον τρόπο, η πλατφόρμα *DialogFlow* προωθεί την ασφάλεια, την εμπιστοσύνη και τον σεβασμό προς τα προσωπικά δεδομένα, διασφαλίζοντας την απορρητότητα και την ασφάλεια κάθε αλληλεπίδρασης.

4.2.3 Κρυπτογράφηση Δεδομένων

Η κρυπτογράφηση δεδομένων αναδύεται ως ένα ζωτικής σημασίας μέσο που διασφαλίζει την ασφάλεια και το απόρρητο των πληροφοριών στο εικοστό πρώτο αιώνα. Σε έναν ψηφιακό κόσμο όπου η διακίνηση των δεδομένων είναι ανεξέλεγκτη και οι απειλές κυριαρχούν, η κρυπτογράφηση αναδύεται ως ασπίδα προστασίας.

Στο πλαίσιο του *DialogFlow*, η κρυπτογράφηση δεδομένων είναι ένας πολύπλοκος και προηγμένος μηχανισμός που εξασφαλίζει την προστασία των ευαίσθητων πληροφοριών των χρηστών. Όταν δεδομένα αποστέλλονται από τον χρήστη στον πράκτορα ή αντίστροφα, αυτά κρυπτογραφούνται με προηγμένες τεχνικές. Αυτό σημαίνει ότι τα δεδομένα μετατρέπονται σε μια μορφή που είναι κατανοητή μόνο με τη χρήση ενός κλειδιού κρυπτογράφησης.

Οι κρυπτογραφημένες πληροφορίες παραμένουν ασφαλείς κατά τη μεταφορά τους στο δίκτυο, μειώνοντας τον κίνδυνο της διαρροής και της παραβίασης. Κατά την αποθήκευση, τα δεδομένα εξακολουθούν να παραμένουν κρυπτογραφημένα, επιτρέποντας μόνο στους εξουσιοδοτημένους χρήστες να αποκρυπτογραφήσουν και να έχουν πρόσβαση σε αυτά.

Η επιλογή ισχυρών κρυπτογραφικών αλγορίθμων και η διαρκής αναβάθμιση των μηχανισμών κρυπτογράφησης αποτελούν κύρια αρχή για το *DialogFlow*. Με αυτόν τον τρόπο, εξασφαλίζεται όχι μόνο η εμπιστοσύνη των χρηστών, αλλά και η προστασία των προσωπικών και ευαίσθητων πληροφοριών τους από εξωτερικές απειλές.

Στο σημείο αυτό αποδεικνύεται πόσο σημαντική είναι η κρυπτογράφηση δεδομένων για το *DialogFlow* και πώς αποτελεί αναπόσπαστο κομμάτι της φιλοσοφίας του για την προστασία της ιδιωτικότητας και της ασφάλειας των χρηστών.

4.3 Επίλογος

Το τρίτο κεφάλαιο αποτελεί ένα εμβαθυμένο ταξίδι στην πλατφόρμα ανάπτυξης *DialogFlow* της *Google*. Μέσα από την ενότητα 4.1, εξετάσαμε διεξοδικά τα βασικά στοιχεία του *DialogFlow*, από την εισαγωγή σε αυτόν μέχρι την κατανόηση των βασικών εννοιών που το χαρακτηρίζουν. Εξετάσαμε επίσης τις ευέλικτες δυνατότητες που προσφέρει ο *DialogFlow* μέσα από την ενότητα 4.1.2.

Στη συνέχεια, στο 4.2, εστιάσαμε στην ασφάλεια και την προστασία των δεδομένων στο πλαίσιο του *DialogFlow*. Αναλύσαμε τη διαχείριση των δεδομένων χρηστών, όπου το *DialogFlow* διασφαλίζει την ιδιωτικότητα και την ασφάλεια των προσωπικών πληροφοριών που συλλέγονται κατά την αλληλεπίδραση. Στη συνέχεια, εξετάσαμε την προστασία των προσωπικών δεδομένων ως προτεραιότητα, με την πλατφόρμα να εφαρμόζει αυστηρά μέτρα για την διασφάλιση της απορρήτου και της ασφάλειας των ευαίσθητων πληροφοριών των χρηστών. Τέλος, ασχοληθήκαμε με την κρυπτογράφηση δεδομένων ως ένα ισχυρό μέσο προστασίας, εξηγώντας πώς η εφαρμογή προηγμένων τεχνικών κρυπτογράφησης εξασφαλίζει την ασφαλή μεταφορά και αποθήκευση των δεδομένων.

Η ανάλυση αυτή αποδεικνύει ότι η πλατφόρμα *DialogFlow* δεν εξασφαλίζει μόνον μια εξαιρετική εμπειρία αλληλεπίδρασης με τους χρήστες, αλλά επίσης δείχνει τη σοβαρότητα της ως προς την ασφάλεια και την προστασία των προσωπικών δεδομένων. Αυτό συμβάλλει στη δημιουργία μιας αξιόπιστης πλατφόρμας που μπορεί να χρησιμοποιηθεί με εμπιστοσύνη για την ανάπτυξη προηγμένων διαλογικών πρακτόρων. Αυτό το κεφάλαιο θέτει το θεμέλιο για την κατανόηση και την επιλογή του *DialogFlow* ως πλατφόρμας ανάπτυξης, προσφέροντας την απαραίτητη προοπτική για την δημιουργία ασφαλών και αξιόπιστων εφαρμογών.

Κεφάλαιο 5ο: Σχεδιασμός της Δομής Διαλόγου

Το παρόν κεφάλαιο σηματοδοτεί την απαρχή της ανάπτυξης του διαλογικού πράκτορα-βοηθού μαθήματος, ο οποίος έχει ως κύριο στόχο την παροχή πληροφοριών σε φοιτητές σχετικά με μαθήματα και τον οδηγό σπουδών. Ειδικότερα, το κεφάλαιο περιλαμβάνει δύο βασικές ενότητες που αφορούν τον ορισμό των λειτουργικών απαιτήσεων για τον πράκτορα και τον σχεδιασμό του διαλόγου, παρέχοντας ένα ολοκληρωμένο πλαίσιο για την ανάπτυξη του διαλογικού συστήματος.

5.1 Ορισμός των Λειτουργικών Απαιτήσεων για τον Διαλογικό Πράκτορα-Βοηθό Μαθήματος

Η παρούσα ενότητα επικεντρώνεται στη λεπτομερή ανάλυση των λειτουργικών απαιτήσεων που πρέπει να πληρεί ο διαλογικός πράκτορας-βοηθός μαθήματος, προκειμένου να παρέχει εξειδικευμένη και αποτελεσματική υποστήριξη στους χρήστες του, δηλαδή τους φοιτητές. Οι παρακάτω λειτουργικές απαιτήσεις διασφαλίζουν την πληρότητα της πρότασης του πράκτορα και την απρόσκοπτη αλληλεπίδρασή του με τους χρήστες:

- **Αναγνώριση και Κατανόηση Ερωτημάτων Χρηστών:** Ο πράκτορας πρέπει να διαθέτει εξελιγμένη τεχνολογία αναγνώρισης φυσικής γλώσσας, που του επιτρέπει να αντιλαμβάνεται με ακρίβεια τα ερωτήματα και τις εντολές που του απευθύνονται από τους φοιτητές, ανεξάρτητα από την δομή τους ή την εκφραστική τους μορφή.

- **Παροχή Εξειδικευμένων Πληροφοριών για τα Μαθήματα:** Ο πράκτορας πρέπει να είναι σε θέση να παρέχει λεπτομερείς και ακριβείς πληροφορίες σχετικά με τα μαθήματα που προσφέρονται στο πρόγραμμα σπουδών, στην περίπτωσή μας τα μαθήματα θα αφορούν το *τμήμα Μηχανικών Πληροφορικής του Αλεξάνδρειο ΤΕΙ Θεσσαλονίκης*. Αυτό περιλαμβάνει τις ώρες διδασκαλίας, το περιεχόμενο του μαθήματος και τις απαιτήσεις για επιτυχή ολοκλήρωση.

- **Παροχή Οδηγού Σπουδών:** Ο πράκτορας πρέπει να διαθέτει μια εκτενή βάση δεδομένων που περιλαμβάνει τις διαθέσιμες επιλογές μαθημάτων που προσφέρονται στο πρόγραμμα σπουδών. Μέσω του πράκτορα, οι φοιτητές μπορούν να πλοηγηθούν στο πρόγραμμα σπουδών του τμήματος *Μηχανικών Πληροφορικής του Αλεξάνδρειο ΤΕΙ Θεσσαλονίκης* και να πάρουν ενημερωμένες αποφάσεις σχετικά με την επιλογή των μαθημάτων τους.

- **Διαχείριση Πολλαπλών Ερωτημάτων:** Ο πράκτορας πρέπει να είναι ικανός να ανταποκρίνεται σε πολλαπλά ερωτήματα και αιτήματα από πολλούς χρήστες ταυτόχρονα. Αυτό απαιτεί την επιθεώρηση της αντίστοιχης τεχνολογίας και την αποδοτική διαχείριση των αιτημάτων, ώστε να εξασφαλίζεται η συνεχής και ομαλή αλληλεπίδραση με τους χρήστες.

Ο ορισμός αυτών των λειτουργικών απαιτήσεων καθορίζει το ρόλο και τη λειτουργικότητα του διαλογικού πράκτορα-βοηθού μαθήματος. Με βάση αυτόν τον ορισμό, η περαιτέρω σχεδίαση και υλοποίηση του πράκτορα θα προσανατολίζονται προς την επίτευξη αυτών των απαιτήσεων, με στόχο την δημιουργία ενός αποτελεσματικού και αξιόπιστου εργαλείου υποστήριξης των φοιτητών κατά τη διάρκεια των σπουδών τους.

5.2 Σχεδιασμός του Διαλόγου και των Πιθανών Περιπτώσεων Χρήσης

Στη διαδικασία αυτής της ενότητας, επιδιώκεται ο λεπτομερής σχεδιασμός του διαλόγου που θα υλοποιήσει ο διαλογικός πράκτορας-βοηθός μαθήματος. Κεντρικό σημείο της εργασίας αυτής αποτελεί η ανάλυση των πιθανών περιπτώσεων χρήσης, με σκοπό να διαμορφωθεί ένας δομημένος και εύληπτος διάλογος που θα καλύπτει τις ανάγκες των φοιτητών.

Στο πρώτο στάδιο, ο διάλογος θα διαμορφωθεί με βάση τις πιο συχνές ερωτήσεις και αιτήματα των φοιτητών. Για παράδειγμα:

- Πληροφορίες για μάθημα:
    - **Χρήστης:** "Μπορείς να μου δώσεις πληροφορίες για το μάθημα 'Εισαγωγή στην Πληροφορική';"
    - **Πράκτορας:** "Φυσικά! Τι πληροφορία θα ήθελες να μάθεις;"
    - **Χρήστης:** "Τον κωδικό του μαθήματος"
    - **Πράκτορας:** "Σας παραθέτω τον κωδικό του μαθήματος"
- Οδηγίες επιλογής προγράμματος σπουδών:
    - **Χρήστης:** "Πόσα εξάμηνα έχει η σχολή;"
    - **Πράκτορας:** "Η σχολή έχει Χ εξάμηνα".

Σε αυτό το παράδειγμα, ο διαλογικός πράκτορας αντιλαμβάνεται την ερώτηση του φοιτητή, αναγνωρίζει το μάθημα που αναφέρεται και παρέχει λεπτομερείς πληροφορίες για τον κωδικό του μαθήματος "Εισαγωγή στην Πληροφορική".

Ένας διαλογικός πράκτορας πρέπει να είναι εξοικειωμένος με την ποικιλία των εκφράσεων που μπορεί να χρησιμοποιήσουν οι φοιτητές, αλλά και με τον τρόπο που αυτές οι εκφράσεις μπορούν να αλληλεπιδρούν με το περιεχόμενο του μαθήματος. Συνεπώς, κατά το σχεδιασμό του διαλόγου, θα πρέπει να ληφθούν υπόψη οι διάφορες πτυχές της πληροφορίας που οι φοιτητές ενδέχεται να αναζητούν.

5.3 Επίλογος

Συνοψίζοντας, ο λεπτομερής σχεδιασμός του διαλόγου για τον διαλογικό πράκτορα-βοηθό μαθήματος αποτελεί βασικό στοιχείο για τη δημιουργία μιας πλούσιας και ενδιαφέρουσας ακαδημαϊκής εμπειρίας για τους φοιτητές. Μέσω του επιμελούς σχεδιασμού του διαλόγου, οι φοιτητές θα μπορούν να αποκτήσουν εύκολη πρόσβαση σε απαραίτητες πληροφορίες για τα μαθήματα, το πρόγραμμα σπουδών και άλλες ακαδημαϊκές λεπτομέρειες.

Ο διαλογικός πράκτορας, χάρη στην χρήση προηγμένων τεχνολογιών, θα παρέχει εξατομικευμένες απαντήσεις και οδηγίες, συμβάλλοντας στην περαιτέρω εξέλιξη των φοιτητών. Μέσω της επικοινωνίας με πολυμέσα, οι φοιτητές θα αποκομίσουν πιο συγκεκριμένες και οπτικοποιημένες γνώσεις, ενώ η ενίσχυση της ατομικής εκπαίδευσης θα επιτρέπει την προσαρμογή του μαθήματος στις ανάγκες και τις αντιλήψεις του κάθε φοιτητή.

Τέλος, ο διαλογικός πράκτορας θα αποτελεί έναν ευέλικτο και διοορατικό εκπαιδευτικό σύμβουλο, ικανό να προσαρμοστεί σε νέες ερωτήσεις και ανάγκες των φοιτητών. Μέσα από αυτή τη διαδικασία, ο διαλογικός πράκτορας δεν παρέχει απλώς απαντήσεις, αλλά συμβάλλει στην ανάπτυξη των δεξιοτήτων και της αυτονομίας των φοιτητών, δημιουργώντας μια ενδιαφέρουσα και ενδελεχή ακαδημαϊκή εμπειρία.

Με τον σχεδιασμό αυτό, ο διαλογικός πράκτορας αναδεικνύεται ως ένας πολύτιμος σύμμαχος στην πορεία των φοιτητών προς την ακαδημαϊκή τους επιτυχία.

Κεφάλαιο 6ο: Υλοποίηση της Εφαρμογής

Σε αυτό το κεφάλαιο, παρουσιάζεται λεπτομερώς η διαδικασία ανάπτυξης της εφαρμογής του φοιτητικού διαλογικού πράκτορα, από τη δημιουργία της βάσης δεδομένων μέχρι την ανάπτυξη του *DialogFlow*. Παρακάτω παρουσιάζονται τα βήματα που ακολουθήθηκαν για την υλοποίηση της εφαρμογής.

6.1 Δημιουργία Βάσης Δεδομένων

Σε αυτή την ενότητα, παρουσιάζουμε τη διαδικασία δημιουργίας της βάσης δεδομένων που χρησιμοποιείται για την αποθήκευση πληροφοριών σχετικά με τα μαθήματα της σχολής μας. Για τον σκοπό αυτό, επιλέξαμε την χρήση του συστήματος διαχείρισης βάσης δεδομένων *MySQL*, καθώς προσφέρει μια αξιόπιστη και αποδοτική λύση για την αποθήκευση και οργάνωση των δεδομένων μας.

Η βάση δεδομένων μας αποτελείται από ένα πίνακα, με όνομα course. Ας δούμε πιο αναλυτικά το σχήμα αυτής της βάσης δεδομένων:

Πίνακας **courses:**

- **course_title:** Αυτό το πεδίο περιέχει το όνομα του μαθήματος. Είναι ένα πεδίο κειμένου με μέγιστο μήκος 255 χαρακτήρων για τυχόν προβλήματα που μπορεί να συναντηθούν σε μαθήματα με μεγάλο μήκος τίτλου.

- **id:** Αυτός είναι ένας αυξανόμενος αριθμητικός αναγνωριστικός αριθμός που χρησιμοποιείται για να αναγνωρίσει μοναδικά κάθε μάθημα. Είναι το κύριο πεδίο που το προσδιορίζει.

- **course_code:** Αυτό το πεδίο περιέχει τον κωδικό του μαθήματος. Κάθε μάθημα έχει ένα μοναδικό κωδικό στον πίνακα σπουδών.

- **semester:** Εδώ καταχωρείται το εξάμηνο κατά το οποίο διδάσκεται το μάθημα.

- **sourse_type:** Εδώ καταχωρείται ο τύπος του μαθήματος για το κάθε μάθημα. Για παράδειγμα αν το μάθημα είναι γενικού τύπου, δηλαδή για όλες τις κατευθύνσεις ή αν είναι ειδικού τύπου δηλαδή για ορισμένες κατευθύνσεις.

- **lectures:** Ο αριθμός των ωρών διδασκαλίας ανά εβδομάδα.

- **ects:** Οι *ECTS* μονάδες που αντιστοιχούν στο μάθημα. Είναι ο βαθμός συντελεστή του κάθε μαθήματος.

- **instructors:** Το όνομα του διδάσκοντα του μαθήματος.

- **course_webpage:** Ο σύνδεσμος του αντίστοιχου μαθήματος αν υπάρχει.

- **educational_goals:** Τι ακριβώς πρόκειται να πετύχει κάποιος μετά το πέρας διδασκαλίας του αντίστοιχου μαθήματος.

- **course_contents:** Αυτό το πεδίο περιέχει το περιεχόμενο του μαθήματος.

- **teaching_method:** Οι μέθοδοι διδασκαλίας που χρησιμοποιούνται. Για παράδειγμα κάποιο μάθημα μπορεί να διδάσκεται διαδικτυακά.

- **students_evaluation:** Εδώ καταγράφονται οι τρόποι με τους οποίους ένα φοιτητής θα εξεταστεί σε ένα μάθημα. Για παράδειγμα, με γραπτές ή προφορικές εξετάσεις, με ερωτήσεις πολλαπλής επιλογής.

Αυτή η βάση δεδομένων είναι κατάλληλη για την αποθήκευση λεπτομερών πληροφοριών για τα μαθήματα της σχολής. Κάθε εγγραφή στον πίνακα *courses* αντιστοιχεί σε ένα μάθημα και περιλαμβάνει βασικές πληροφορίες για ένα μάθημα. Έπειτα, σε συνεργασία με τον scrapper που δημιουργήθηκε για την αυτόματη άντληση των δεδομένων από την επίσημη ιστοσελίδα της σχολής, θα έρθει το *DialogFlow* έτσι ώστε να πλαισιώσει όλη αυτή την δουλειά απαντώντας σε ερωτήματα φοιτητών σχετικά με το μάθημα που τους ενδιαφέρει.

6.1.1 Αυτοματοποιημένη Εισαγωγή Δεδομένων

Εισαγωγή

Η αυτοματοποιημένη εισαγωγή δεδομένων μέσω *web scraping* αποτελεί ισχυρό εργαλείο για τη συλλογή πληροφοριών από ιστοσελίδες και την αποθήκευσή τους σε μια βάση δεδομένων. Σε αυτήν την υποενότητα, θα εξετάσουμε τον δικό μας *web scraper* που συλλέγει πληροφορίες για μαθήματα από την ιστοσελίδα της σχολής και τις αποθηκεύει σε μια βάση δεδομένων. Θα αναλύσουμε τον κώδικα και θα παρουσιάσουμε πιθανές επεκτάσεις και βελτιώσεις.

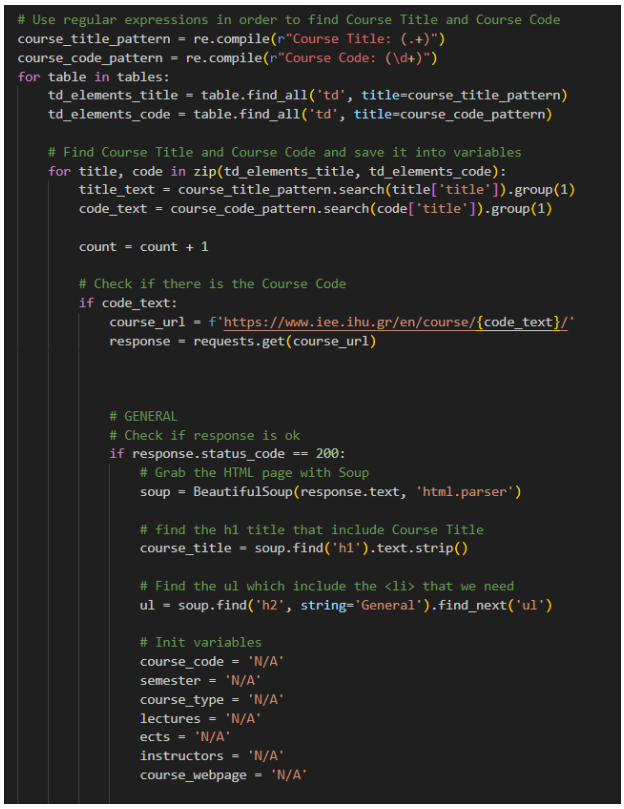
Ο κώδικάς μας πραγματοποιεί *web scraping* από την ιστοσελίδα της σχολής https://www.iee.ihu.gr/en/udg_courses/, συλλέγοντας πληροφορίες για διάφορα μαθήματα και αποθηκεύοντάς τες σε μια βάση δεδομένων *MySQL*. Ας αναλύσουμε τα βήματα που ακολουθήθηκαν περαιτέρω.

```python
from bs4 import BeautifulSoup
import requests
import re
import mysql.connector

# For testing
# import pandas as pd



# Connect with DB
db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="root",
    database="uniBot"
)

# Create a cursor
cursor = db.cursor()


# MATHS
url = 'https://www.iee.ihu.gr/en/udg_courses/'

pages = requests.get(url)

soup = BeautifulSoup(pages.text, 'html.parser')

# find the <div> that include the table with Subjects
div_element = soup.find('div', style="overflow-x: auto;")

# Target the (table) into <div>
tables = div_element.find_all('table')

count = 0

# For testing
# data_list = []

total_count_courses = 0
```

Εικόνα 6.1.1.1 Python Scrapping (1)

Αρχικά, ο κώδικας όπως φαίνεται και στην φωτογραφία κάνει *Import* όλες τις απαραίτητες βιβλιοθήκες που χρειάζεται. Στην συνέχεια πραγματοποιεί σύνδεση με μια τοπική βάση δεδομένων *MySQL* στην οποία και αναλύσαμε προηγουμένος. Οι παράμετροι για τη σύνδεση (όπως το όνομα χρήστη και ο κωδικός πρόσβασης) ανήκουν στην δική μου περίπτωση τους οποίους και έχω ορίσει.

Έπειτα ο κώδικας χρησιμοποιεί το *module requests* για να ανακτήσει το *HTML* της σελίδας https://www.iee.ihu.gr/en/udg_courses/ μαζί με την βοήθεια της βιβλιοθήκης *BeautifulSoup*. Στην συνέχεια γίνεται προσπάθεια εύρεσης του * * tag που μας ενδιαφέρει το οποίο περιέχει και τον πίνακα *<table>* tag με τα μαθήματα της σχολής.

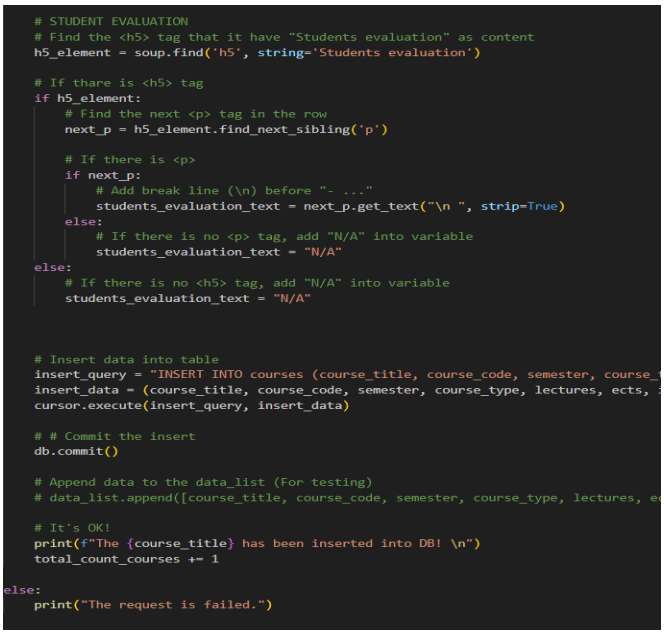


```
# Use regular expressions in order to find Course Title and Course Code
course_title_pattern = re.compile(r"Course Title: (.+)")
course_code_pattern = re.compile(r"Course Code: (\d+)")
for table in tables:
    td_elements_title = table.find_all('td', title=course_title_pattern)
    td_elements_code = table.find_all('td', title=course_code_pattern)

    # Find Course Title and Course Code and save it into variables
    for title, code in zip(td_elements_title, td_elements_code):
        title_text = course_title_pattern.search(title['title']).group(1)
        code_text = course_code_pattern.search(code['title']).group(1)

        count = count + 1

        # Check if there is the Course Code
        if code_text:
            course_url = f'https://www.iee.ihu.gr/en/course/{code_text}/'
            response = requests.get(course_url)



            # GENERAL
            # Check if response is ok
            if response.status_code == 200:
                # Grab the HTML page with Soup
                soup = BeautifulSoup(response.text, 'html.parser')

                # find the h1 title that include Course Title
                course_title = soup.find('h1').text.strip()

                # Find the ul which include the <li> that we need
                ul = soup.find('h2', string='General').find_next('ul')

                # Init variables
                course_code = 'N/A'
                semester = 'N/A'
                course_type = 'N/A'
                lectures = 'N/A'
                ects = 'N/A'
                instructors = 'N/A'
                course_webpage = 'N/A'
```

Εικόνα 6.1.1.2 Python Scrapping (2)

Εν συνεχεία, Ο κώδικας προσπαθεί να εξάγει τα δεδομένα (όνομα μαθήματος και κωδικός μαθήματος) που βρίσκονται στον πίνακα του *HTML* ο οποίος και περιέχει τα μαθήματα της σχολής. Για να πραγματοποιηθεί αυτό γίνεται χρήση *regular expressions* (με τα *course_title_pattern* και *course_code_pattern*) για να εντοπίσει τον τίτλο και τον κωδικό του κάθε μαθήματος.

Αφού εντοπιστούν, για κάθε μάθημα, ο κώδικας επισκέπτεται την αντίστοιχη σελίδα του μαθήματος και συλλέγει πληροφορίες όπως τον τίτλο, τον κωδικό, το εξάμηνο, τον τύπο, τις διαλέξεις, τις ECTS μονάδες, τους διδάσκοντες και άλλες πληροφορίες, έτσι ώστε αργότερα να τις τοποθετήσει στην βάση δεδομένων. Όλα αυτά πραγματοποιούνται με αντίστοιχο τρόπο όπως για την εντόπιση του πίνακα με τα μαθήματα, δηλαδή για την εντόπιση του κωδικού μαθήματος στην *HTML* σελίδα έχει γίνει σχετική έρευνα έτσι ώστε να εντοπίζεται με ακρίβεια στην δομή του *HTML* δένδρου για κάθε μάθημα.



```
    # STUDENT EVALUATION
    # Find the <h5> tag that it have "Students evaluation" as content
    h5_element = soup.find('h5', string='Students evaluation')

    # If thare is <h5> tag
    if h5_element:
        # Find the next <p> tag in the row
        next_p = h5_element.find_next_sibling('p')

        # If there is <p>
        if next_p:
            # Add break line (\n) before "- ..."
            students_evaluation_text = next_p.get_text("\n ", strip=True)
        else:
            # If there is no <p> tag, add "N/A" into variable
            students_evaluation_text = "N/A"
    else:
        # If there is no <h5> tag, add "N/A" into variable
        students_evaluation_text = "N/A"



    # Insert data into table
    insert_query = "INSERT INTO courses (course_title, course_code, semester, course_
    insert_data = (course_title, course_code, semester, course_type, lectures, ects, 
    cursor.execute(insert_query, insert_data)

    # # Commit the insert
    db.commit()

    # Append data to the data_list (For testing)
    # data_list.append([course_title, course_code, semester, course_type, lectures, e

    # It's OK!
    print(f"The {course_title} has been inserted into DB! \n")
    total_count_courses += 1

else:
    print("The request is failed.")
```

Εικόνα 6.1.1.3 Python Scrapping (3)

Κλείνοντας, τα δεδομένα που συλλέγονται αποθηκεύονται σε μια βάση δεδομένων *MySQL*. Χρησιμοποιείται η γλώσσα *SQL* για την εκτέλεση εντολών *INSERT*. Στο Παράρτημα Α φαίνεται και ολόκληρος ο κώδικας του *scrapper*.

Βελτιώσεις και Επεκτάσεις

Κάποιες μελλοντικές βελτιώσεις του *scrapper* έτσι ώστε να είναι πιο αυτόνομος αλλά και πλήρως έγκυρος ως προς τα δεδομένα που συλλέγει, θα ήταν:

- Ο έλεγχος των δεδομένων που συλλέγονται από την ιστοσελίδα για πιθανές ασυνέπειες ή ανωμαλίες και η εύρεση πιο σίγουρων και πιο αποτελεσματικών προτύπων έτσι ώστε η συλλογή να είναι πλήρως έγκυρη.

- Η κατασκευή κάποιας διεπαφής (*API*) έτσι ώστε η πρόσβαση στα δεδομένα να μην γίνεται απευθείας με την βάση δεδομένων για λόγους ασφαλείας, καθώς να είναι και πιο εύκολη στην χρήση.

- Η συλλογή περισσότερων δεδομένων από την σχολή έτσι ώστε ο διαλογικός πράκτορας να έχει προοπτικές ακόμη και για την εδραίωση του στην σχολή.

Παρακάτω θα δούμε πως επιτυγχάνεται η σύνδεση της βάση δεδομένων με τον τοπικό server που αναπτύχθηκε με *Python*.

6.2 Ανάπτυξη του Server με FastAPI και Python

Για την υλοποίηση του βασικού λειτουργικού μέρους της εφαρμογής, επιλέξαμε να χρησιμοποιήσουμε το *FastAPI framework* σε συνδυασμό με την *Python*. Η επιλογή αυτή μας προσέφερε μια γρήγορη και αποτελεσματική μέθοδο για την δημιουργία των απαραίτητων διαδρομών *(endpoints)* που απαιτεί η εφαρμογή μας. Χρησιμοποιήσαμε τις δυνατότητες του *FastAPI* για την επικοινωνία με τη βάση δεδομένων και τη διαχείριση των *HTTP* αιτημάτων.

6.2.1 FastAPI

Το *FastAPI* είναι ένα υψηλής απόδοσης πλαίσιο ανάπτυξης *(framework)* για τη δημιουργία διαδικτυακών εφαρμογών *(web applications)* με τη χρήση της γλώσσας προγραμματισμού *Python*. Το πλαίσιο αυτό έχει σχεδιαστεί με έμφαση στην ταχύτητα, την ευκολία χρήσης, την αυτόματη τεκμηρίωση και την υποστήριξη για τη δημιουργία *API* (Διεπαφές Προγραμματισμού Εφαρμογής).

Ορισμένα σημαντικά χαρακτηριστικά και έννοιες του **FastAPI** περιλαμβάνουν:

- **Δηλωτική Δρομολόγηση (Declarative Routing):** Το *FastAPI* επιτρέπει τη δημιουργία δρομολογητών *(routers)* και διαδρομών *(routes)* χρησιμοποιώντας δηλωτική σύνταξη. Αυτό σημαίνει ότι μπορούν να οριστούν οι λειτουργίες που αντιστοιχούν σε κάθε διαδρομή χρησιμοποιώντας απλή σύνταξη *Python*.

- **Υποστήριξη για Τύπους (Type Annotations):** Οι τύποι δεδομένων μπορούν να χρησιμοποιηθούν για να καθορίσουν τα είδη των δεδομένων που περνούν σε μια διαδρομή και τα είδη των δεδομένων που επιστρέφονται από αυτήν. Αυτό βοηθά στην αυτόματη επικύρωση των δεδομένων και στη βελτίωση της ασφάλειας της εφαρμογής.

- **Ενσωματωμένη Τεκμηρίωση (Automatic Documentation):** Το *FastAPI* παρέχει αυτόματη δημιουργία τεκμηρίωσης για τα API που δημιουργούνται. Η τεκμηρίωση παρουσιάζει τις διαδρομές, τους τύπους δεδομένων, τις περιγραφές και άλλες πληροφορίες, καθιστώντας ευκολότερη τη χρήση και την ανάπτυξη των *API*.

- **Υποστήριξη Επαναχρησιμοποίησης Κώδικα (Code Reusability):** Το *FastAPI* διαχωρίζει τον κώδικα σε μικρούς δρομολογητές, προσφέροντας έτσι τη δυνατότητα εύκολης επαναχρησιμοποίησης και οργάνωσης του κώδικα.

- **Επεκτασιμότητα:** Το *FastAPI* επιτρέπει τη δημιουργία διαφόρων δρομολογητών για διαφορετικά τμήματα της εφαρμογής σας, βοηθώντας στην επεκτασιμότητα και τη διαχείριση μεγάλων εφαρμογών.

- **Ενσωματωμένη Υποστήριξη WebSocket:** Το *FastAPI* παρέχει υποστήριξη για την ανάπτυξη εφαρμογών που χρησιμοποιούν το πρωτόκολλο *WebSocket*, το οποίο επιτρέπει τη διεξαγωγή πραγματικού χρόνου επικοινωνίας μεταξύ του διακομιστή και του πελάτη.

- **Ενσωματωμένη Υποστήριξη Αποστολής Αρχείων:** Το *FastAPI* καθιστά εύκολο τον χειρισμό και την αποστολή αρχείων μέσω των *API* σας.

Συνολικά, το *FastAPI* είναι ένα πανίσχυρο πλαίσιο ανάπτυξης που επιτρέπει τη δημιουργία γρήγορων, αποδοτικών και καλά τεκμηριωμένων διαδικτυακών εφαρμογών με τη χρήση της *Python*.

6.2.2 Εξωτερική Πρόσβαση με το ngrok

Το *ngrok* είναι ένα εργαλείο που χρησιμοποιείται για να δημιουργήσει ασφαλή δημόσια προσπέλαση *(public access)* προς ένα τοπικό διακομιστή. Συχνά χρησιμοποιείται κατά την ανάπτυξη και τον έλεγχο εφαρμογών, καθώς επιτρέπει σε προγραμματιστές να κάνουν διαθέσιμες τις τοπικές εφαρμογές τους στο διαδίκτυο για δοκιμή ή για κοινή χρήση, χωρίς να χρειάζεται να ανοίγουν διάφορες θύρες στο δρομολογητή τους ή να διαχειριστούν πολύπλοκες ρυθμίσεις ασφαλείας.

Πώς λειτουργεί το **ngrok**:

- **Εκκίνηση του Τοπικού Διακομιστή:** Πρώτα, θα πρέπει να έχει κανείς έναν τοπικό διακομιστή (π.χ. έναν διακομιστή ιστού) που τρέχει στην τοπική του μηχανή.

- **Εκτέλεση του ngrok:** Στη συνέχεια, εκτελείται το *ngrok* από τη γραμμή εντολών, παρέχοντας τη θύρα του τοπικού διακομιστή. Το *ngrok* θα δημιουργήσει ένα δημόσιο URL που προωθεί τις αιτήσεις προς τον τοπικό διακομιστή.

- **Δημόσια Προσπέλαση:** Το *ngrok* παρέχει ένα μοναδικό *URL* το οποίο μπορεί να χρησιμοποιηθεί για πρόσβαση στον τοπικό διακομιστή από οπουδήποτε στο διαδίκτυο.

Το *ngrok* προσφέρει επίσης προηγμένες δυνατότητες όπως προσαρμοσμένα *subdomains*, ασφάλεια με χρήση κλειδιών και πιστοποιητικών *SSL*, καταγραφή των αιτήσεων που κατευθύνονται στον τοπικό διακομιστή κ.ά.

Συνοψίζοντας, το *ngrok* είναι ένα εξαιρετικά χρήσιμο εργαλείο για την επιτροπή δημόσιας προσπέλασης σε τοπικούς διακομιστές, κάνοντας την ανάπτυξη, τη δοκιμή και την κοινή χρήση εφαρμογών πιο εύκολη.

Στο πλαίσιο του *Dialogflow*, η δυνατότητα εξωτερικής πρόσβασης σε μια εφαρμογή είναι κρίσιμη για την αξιολόγηση, γιαυτό γίνεται περισσότερο περίπλοκη λόγω της ανάγκης για ασφαλή σύνδεση *(https)* για την χρήση του *webhook*. Εδώ έρχεται σε βοήθεια το *ngrok*, παρέχοντας μια απλή και αποτελεσματική λύση για την εξωτερική πρόσβαση με ασφάλεια.

Το *DialogFlow* επιτρέπει την ενσωμάτωση των λεγόμενων *webhooks* για την επεξεργασία των αιτημάτων των χρηστών. Αυτά τα *webhooks* είναι λειτουργίες που εκτελούνται από τον διακομιστή και

εξυπηρετούν τα αιτήματα των χρηστών που προέρχονται από το *DialogFlow*. Ωστόσο, για λόγους ασφαλείας, το *DialogFlow* απαιτεί τη σύνδεση με τον διακομιστή μέσω ασφαλούς σύνδεσης *(https)*.

Λόγω των προαναφερθέντων το *ngrok* έχει σημασία. Καθώς η εφαρμογή λειτουργεί τοπικά στον υπολογιστή, το *ngrok* δημιουργεί μια ασφαλή δημόσια διεύθυνση *URL* που αντιστοιχεί στη διεύθυνση *IP* του υπολογιστή και τη θύρα που εκτελείται η εφαρμογή. Αυτό δίνει τη δυνατότητα στο *DialogFlow* να στέλνει αιτήματα στον διακομιστή μέσω ασφαλούς σύνδεσης.

Χωρίς το *ngrok*, θα ήταν δύσκολο να αναπτύξετε και να δοκιμάσετε τον διάλογο σας στο *DialogFlow*, καθώς το *DialogFlow* απαιτεί ασφαλή σύνδεση μέσω *https*. Το *ngrok* απλοποιεί αυτήν τη διαδικασία παρέχοντας μια δημόσια διεύθυνση *URL* που επιτρέπει να αλληλεπιδρά κανείς με τον διάλογο στο *DialogFlow* από οποιαδήποτε συσκευή και τοποθεσία.

Ο συνδυασμός του *ngrok* με το *DialogFlow* απλοποιεί τη διαδικασία ανάπτυξης, δοκιμής και βελτίωσης των διαλόγων, καθιστώντας την πιο ευέλικτη και προσβάσιμη για τον προγραμματιστή.

6.2.3 Python

```python
from fastapi import FastAPI
from fastapi import Request
from fastapi.responses import JSONResponse

import mysql.connector

app = FastAPI()

# Configure your MySQL database connection
db_config = {
    "host": "localhost",
    "user": "root",
    "password": "root",
    "database": "uniBot"
}


# This List match the input user information to database field
field_mapping = {
    "code": "course_code",
    "semester": "semester",
    "type": "course_type",
    "hours": "lectures",
    "ects": "ects",
    "instructor": "instructors",
    "teacher": "instructors",
    "url": "course_webpage",
    "link": "course_webpage",
    "webpage": "course_webpage",
    "goals": "educational_goals",
    "learns": "educational_goals",
    "content": "course_contents",
    "info": "course_contents",
    "teaching method": "teaching_method",
    "method": "teaching_method",
    "evaluation": "students_evaluation"
}
```

Εικόνα 6.2.3.1 Python Main (1)

Εισαγωγή

Στην αρχή της υλοποίησης του διακομιστή *FastAPI*, κάναμε την αναγκαία εισαγωγή των βιβλιοθηκών, όπως βλέπουμε και στον κώδικα παραπάνω, που θα χρησιμοποιηθούν στο έργο μας. Συγκεκριμένα, χρησιμοποιήσαμε τη *βιβλιοθήκη FastAPI*, η οποία είναι ένα εξαιρετικά ισχυρό εργαλείο για τη δημιουργία γρήγορων και αποτελεσματικών διακομιστών. Η βιβλιοθήκη *FastAPI* παρέχει πληθώρα δυνατοτήτων για τη διαχείριση των *HTTP* αιτημάτων και απαντήσεων.

Επιπλέον, χρησιμοποιήσαμε τις κλάσεις *Request* και *JSONResponse* για να διαχειριστούμε τις λειτουργίες που σχετίζονται με τα αιτήματα *HTTP* και τις απαντήσεις *JSON* αντίστοιχα. Η εισαγωγή αυτών των βιβλιοθηκών είναι κρίσιμη για την κατανόηση του τρόπου με τον οποίο θα αλληλεπιδράσει ο διακομιστής μας με τις αιτήσεις από το *DialogFlow* και πώς θα παρέχει απαντήσεις σε αυτές. Μέσω αυτής της εισαγωγής, δημιουργήσαμε τον βασικό θεμέλιο για την υλοποίηση του διακομιστή μας, ετοιμάζοντας το έδαφος για τις επόμενες ενότητες όπου θα αναλύσουμε τις λειτουργίες που θα παρέχει ο διακομιστής μας και πώς αυτές συνδέονται με τη βάση δεδομένων μας.

Επίσης βλέπουμε ότι καθορίζουμε ένα λεξικό με το όνομα *field_mapping* έτσι ώστε να υπάρχει μια αντιστοίχιση μεταξύ των πεδίων που μπορεί να εισάγει ο χρήστης και των πεδίων στη βάση δεδομένων. Αυτό θα χρησιμοποιηθεί αργότερα για τον προσδιορισμό ποιου πεδίου πρέπει να ανακτηθούν πληροφορίες.

Ρυθμίσεις Βάσης Δεδομένων

Προτού προχωρήσουμε στη δημιουργία των λειτουργιών του διακομιστή, ήταν απαραίτητο να ρυθμίσουμε τη σύνδεση με τη βάση δεδομένων *MySQL*. Αυτό επιτεύχθηκε μέσω της δημιουργίας ενός λεξικού με το όνομα *db_config*, το οποίο περιείχε τις απαραίτητες πληροφορίες για την σύνδεση στον διακομιστή βάσης δεδομένων.

Το λεξικό αυτό περιλάμβανε τα εξής στοιχεία:

- **host:** Το όνομα του διακομιστή στον οποίο είναι φιλοξενημένη η βάση δεδομένων.
- **user:** Το όνομα του χρήστη που έχει πρόσβαση στη βάση δεδομένων.
- **password:** Ο κωδικός πρόσβασης του χρήστη για την πρόσβαση στη βάση δεδομένων.
- **database:** Το όνομα της βάσης δεδομένων που θέλουμε να συνδεθούμε.

Οι παραπάνω πληροφορίες είναι αναγκαίες για τη δημιουργία της σύνδεσης με τη βάση δεδομένων. Ενσωματώνοντας αυτές τις πληροφορίες στο λεξικό *db_config*, δημιουργήσαμε ένα εύκολο και οργανωμένο τρόπο να αποθηκεύσουμε τα δεδομένα αυτά.

Η ρύθμιση της σύνδεσης προηγήθηκε της δημιουργίας των λειτουργιών του διακομιστή, διασφαλίζοντας ότι ο διακομιστής θα έχει πρόσβαση στη βάση δεδομένων για την εκτέλεση των απαραίτητων ερωτημάτων. Αυτή η προετοιμασία είναι απαραίτητη για να εξασφαλίσουμε την ομαλή λειτουργία του διακομιστή μας και τη σωστή επικοινωνία με τη βάση δεδομένων.

Επικοινωνία με το Dialogflow

```python
@app.post("/")
async def handle_request(request: Request):

    # Retrieve the JSON data from the request
    payload = await request.json()

    # Extract the necessary information from the payload
    intent = payload['queryResult']['intent']['displayName']
    user_input = payload['queryResult']['queryText']

    if intent == "subject.search-info" or intent == "info.search-subject":
        if intent == "subject.search-info":
            # Variable from JSON
            information = payload['queryResult']['parameters']['information']
            subject = payload['queryResult']['outputContexts'][0]['parameters']['subject']
        elif intent == "info.search-subject":
            # Variable from JSON
            subject = payload['queryResult']['parameters']['subject']
            information = payload['queryResult']['outputContexts'][0]['parameters']['information']
```

Εικόνα 6.2.3.2 Python Main (2)

Αρχικά βλέπουμε την συνάρτηση *handle_request* η οποία ενεργοποιείται όταν λαμβάνεται ένα αίτημα *POST*. Στην συνέχεια ανακτούμε τα δεδομένα *JSON* του αιτήματος και τα αποθηκεύουμε στη μεταβλητή *payload*. Από τα δεδομένα του αιτήματος *payload*, εξάγουμε την πρόθεση *(intent)* της ερώτησης από το *DialogFlow* και το κείμενο της ερώτησης που έκανε ο χρήστης.

Ελέγχουμε την πρόθεση της ερώτησης που κατανοεί το *DialogFlow* αν είναι "*subject.search-info*" ή "*info.search-subject*", τότε ελέγχει αν είναι "*subject.search-info*" ή "*info.search-subject*" και εξάγει πληροφορίες από τα δεδομένα *JSON*. Επίσης ελέγχεται και η περίπτωση να είναι “ *multiple.search*” καθώς και η περίπτωση να είναι “*info.study-program*” και “*info.semester*”.

Έπειτα, ο κώδικας ελέγχει αν η είσοδος, ως προς την πληροφορία, του χρήστη αντιστοιχεί σε κάποιο πεδίο της βάσης δεδομένων (χρησιμοποιώντας το *field_mapping όπου είναι απαραίτητο*). Αν υπάρχει αντιστοιχία, καθορίζεται το πεδίο που αντιστοιχεί στην είσοδο του χρήστη.

Στη συνέχεια, ο κώδικας συνδέεται στη βάση δεδομένων *MySQL* με τη χρήση των παραμέτρων που καθορίστηκαν στη μεταβλητή *db_config*, δημιουργεί έναν δρομέα *(cursor)* για την εκτέλεση SQL εντολών στη βάση δεδομένων. Έπειτα εκτελεί ένα SQL ερώτημα που ανακτά πληροφορίες από τον πίνακα *courses* με βάση το όνομα του μαθήματος *(course_title)* και το πεδίο της πληροφορίας που ζητά ο χρήστης *(information)*.

Ανακτά τα αποτελέσματα του ερωτήματος και επεξεργάζεται το κείμενο απάντησης ανάλογα με τα αποτελέσματα. Αυτό το κείμενο απάντησης αποθηκεύεται στη μεταβλητή *response_text*. Ο κώδικας επιστρέφει το αποτέλεσμα σε μορφή *JSON* χρησιμοποιώντας τη συνάρτηση *JSONResponse*.

Όπως θα δούμε και στο Παράρτημα Β, στο τέλος έχουμε προσθέσει και την λειτουργία αυτόματης εκτέλεσης του server όταν ξεκινάει η εκτέλεση του αρχείου.

6.3 Υλοποίηση των Λειτουργιών του Πράκτορα-Βοηθού Μαθήματος

Για να επιτύχει η απρόσκοπτη επικοινωνία με τον πράκτορα-βοηθό μαθήματος μέσω του *DialogFlow*, πραγματοποιήσαμε τις απαραίτητες ρυθμίσεις στο *DialogFlow* λογισμικό, καθορίζοντας *intents, training phrases*, *entities* και διάφορα σενάρια σχετικά με ερωτήσεις και απαντήσεις που θα μπορούσε να κάνει ένας χρήστης σχετικά με τα μαθήματα. Ο προγραμματισμός του διαλογικού πράκτορα έχει γίνει στα Αγγλικά για λόγους αστοχίας των Ελληνικών.

Τα **entities** βοηθούν στον εντοπισμό φράσεων που έχουν οριστεί σε κάθε entity όπως αναφέραμε και σε προηγούμενη ενότητα. Τα **entities** που δημιουργήσαμε είναι τα εξής:

- **subject**: Το συγκεκριμένο *entity* περιέχει τα μαθήματα που βρίσκονται στην βάση δεδομένων, για τον εύκολο εντοπισμό στις περιπτώσεις που ο χρήστης καλείται να γράψει κάποιο μάθημα που θέλει έτσι ώστε να αναζητηθεί στην βάση δεδομένων. Έχουμε επιλέξει να γίνεται και παρόμοια ταύτιση, για παράδειγμα αν ο χρήστης γράψει *math I* και υπάρχει εγγραφή entity με όνομα *Mathematics I* τότε θα γίνει σωστή αντιστοίχιση. Επίσης να επισημάνουμε ότι η ενημέρωση του της ενότητας *subject entity* γίνεται αυτόματα με την χρήση *Python* δυναμικά και όχι στατικά, όπως θα δούμε και στο Παράρτημα C.

- **information**: Το συγκεκριμένο *entity* περιέχει τις πληροφορίες που ο χρήστης μπορεί να ρωτήσει για ένα μάθημα. Αρχικοποιούνται αυτόματα από ένα αρχείο κειμένου το οποίο περιέχει τα πεδία της βάσης δεδομένων, καθώς περιέχει και συνώνυμα αυτών χωρισμένα με «/». Αυτό υλοποιείται με την χρήση της Python όπως θα δούμε και στο Παράρτημα D.

- **yes**: Επίσης υπάρχει και ένα ακόμη απλό *entity* το οποίο περιέχει φράσεις συμφωνίας όπως για παράδειγμα "*yes, ye, ok*" το οποίο θα μας χρειαστεί στην συνέχεια όπως θα αναλύσουμε.

Επίσης, δημιουργήσαμε τα παρακάτω **intents**:

- **info.lesson:** Αυτό το *intent* αναγνωρίζει όταν ο χρήστης θέλει πληροφορίες για ένα συγκεκριμένο μάθημα. Για αυτό το *intent*, καθορίσαμε *training phrases* όπως για παράδειγμα «*I want to learn information about subject*». Όταν ο χρήστης πληκτρολογήσει κάποια συνώνυμη έκφραση τότε υπάρχουν δύο επιλογές. Η πρώτη προτρέπει τον χρήστη να γράψει τι ακριβώς πληροφορία θέλει να μάθει για το μάθημα, όπως για παράδειγμα τον *κωδικό, το εξάμηνο, τα ects* κτλ. και η δεύτερη προτρέπει τον χρήστη να πληκτρολογήσει το μάθημα για το οποίο θέλει να μάθει πληροφορίες.

- **info.search:** Αυτό το *intent* αναγνωρίζει όταν ο χρήστης πληκτρολογεί κάποιου είδους πληροφορία όπως «*code, url, content*» για την οποία θέλει να μάθει. Υπάρχουν φράσεις για τις οποίες είναι εκπαιδευμένος ο διαλογικός πράκτορας. Όταν ο χρήστης δώσει κάποια πληροφορία και γίνει αντιστοίχιση με την παράμετρο *information* την οποία έχουμε ορίσει με το *entity information* τότε θα εμφανιστεί μήνυμα που θα προτρέψει τον χρήστη να του πληκτρολογήσει για ποιο μάθημα θα ήθελε πληροφορίες. Τώρα για τον έλεγχο του μαθήματος έχουμε δημιουργήσει ένα *follow-up intent* το οποίο ονομάζεται *info.search-subject* το οποίο για να ενεργοποιηθεί πρέπει να προηγηθεί το προηγούμενο *info.search intent.*

- **info.search-subject:** Αυτό το *intent* αναγνωρίζει όταν ο χρήστης πληκτρολογεί κάποιου είδους μάθημα για το οποίο θέλει να μάθει την πληροφορία που πληκτρολόγησε προηγουμένως. Υπάρχουν φράσεις για τις οποίες είναι εκπαιδευμένος ο διαλογικός πράκτορας. Όταν ο χρήστης δώσει κάποιο μάθημα και γίνει αντιστοίχιση με την παράμετρο *subject* την οποία έχουμε ορίσει με το *entity subject* τότε αφού ενεργοποιήσαμε το *Fulfillement* το οποίο θα μας βοηθήσει μαζί με το *webhook* να πάρουμε τον έλεγχο στον *Server* της *Python* τον οποίο έχουμε δημιουργήσει με *FastAPI* και εκεί να γίνει ο κατάλληλος έλεγχος στον οποίο και αναλύσαμε προηγουμένως. Τέλος επιστρέφεται η πληροφορία που ζήτησε ο χρήστης.

- **subject.search:** Αυτό το *intent* αναγνωρίζει όταν ο χρήστης πληκτρολογεί κάποιου είδους μάθημα στο οποίο ενδιαφέρεται να μάθει πληροφορίες. Υπάρχουν φράσεις για τις οποίες είναι εκπαιδευμένος ο διαλογικός πράκτορας. Όταν ο χρήστης δώσει το μάθημα στο οποίο θέλει να ενημερωθεί και γίνει αντιστοίχιση με την παράμετρο *subject* την οποία έχουμε ορίσει με το *entity subject* τότε θα εμφανιστεί μήνυμα που θα προτρέψει τον χρήστη να του πληκτρολογήσει ποια πληροφορία θα ήθελε να μάθει. Τώρα για τον έλεγχο της πληροφορίας έχουμε δημιουργήσει ένα *follow-up intent* το οποίο ονομάζεται *subject.search-info* το οποίο για να ενεργοποιηθεί πρέπει να προηγηθεί το προηγούμενο *subject.search***.**

- **subject.search-info:** Αυτό το *intent* αναγνωρίζει όταν ο χρήστης πληκτρολογεί κάποιου είδους πληροφορία στην οποία ενδιαφέρεται να μάθει για το μάθημα που έδωσε στο προηγούμενο *intent*. Υπάρχουν φράσεις για τις οποίες είναι εκπαιδευμένος ο διαλογικός πράκτορας. Όταν ο χρήστης δώσει κάποια πληροφορία και γίνει αντιστοίχιση με την παράμετρο *information* την οποία έχουμε ορίσει με το *entity information* τότε αφού ενεργοποιήσαμε το *Fulfillement* ο έλεγχος θα περάσει στο *server*. Τέλος επιστρέφεται η πληροφορία που ζήτησε ο χρήστης.

- **multiple.search:** Ένα πολύ σημαντικό και χρήσιμο *intent* είναι το *multiple.search* το οποίο δέχεται ως παράμετρο μια σειρά από μαθήματα και πληροφορίες. Για παράδειγμα ένας χρήστης μπορεί να δώσει περισσότερες από μια παραμέτρους στην πληροφορία καθώς και στα μαθήματα. Έστω ότι ο χρήστης δώσει την εξής πρόταση «*I want to learn the code and web page from Mathematics I and Java*» τότε θα του επιστραφεί και ο κωδικός και το *Url* της σελίδας του μαθήματος και για τα δύο μαθήματα.

- **study.program:** Αυτό το *intent* δέχεται φράσεις όπως «*I want to learn about the study program*». Όταν δεχτεί μια τέτοια είδους πρόταση τότε ο χρήστης θα ερωτηθεί τι ακριβώς θέλει να μάθει, έχουμε προσθέσει δύο επιλογές: Η πρώτη είναι να μάθει πόσα εξάμηνα έχει η σχολή όπου γίνεται με δυναμικό έλεγχο και όχι με στατικό, καθώς και η άλλη ερώτηση είναι να μάθει ποια μαθήματα έχει ένα συγκεκριμένο εξάμηνο. Ανάλογα με το τι θα δώσει θα γίνει αντιστοίχιση και στα *intent* τα οποία υπάρχουν.

- **info.study-program:** Η πρώτη ερώτηση για το πόσα μαθήματα έχει η σχολή, βρίσκεται σε αυτό το *intent* το οποίο περιέχει φράσεις όπως «*How many semesters there are*». Όταν ενεργοποιηθεί τότε γίνεται ξανά σύνδεση με τον *server* όπου και γίνεται ο έλεγχος και επιστρέφεται δυναμικά τα εξάμηνα της σχολής που υπάρχουν.

- **info.semester:** Αυτό το *intent* αντιλαμβάνεται εκφράσεις του τύπου «*Yes I want to learn about 1 semester».* Το *Yes* γίνεται αντιστοίχιση με το *yes entity* που δημιουργήσαμε, και *ο αριθμός του semester* γίνεται αντιστοίχιση με ένα *system entity number*. Όταν ο χρήστης επιλέξει να μάθει τα μαθήματα του 1^{ου} εξαμήνου τότε θα γίνει ενεργοποίηση του αντίστοιχου *intent* και έπειτα σε συνδυασμό με το *fulfillment* θα επιστραφεί στο χρήστη η λίστα με τα μαθήματα του εξαμήνου που επιθυμεί.

Υπάρχουν βέβαια και τα *default* μηνύματα όπως:

- **Default Welcome Intent:** Αυτό το *intent* καλωσορίζει τον χρήστη ¨ακούγοντας¨ στις εκφράσεις «*Γεια».* Ως απάντηση ο χρήστης λαμβάνει ένα τυποποιημένο μήνυμα το οποίο ενημερώνει τον χρήστη για το τι διαλογικός πράκτορας πρόκειται, και τον προτρέπει να κάνει μια ερώτηση.

- **Default Fallback Intent:** Αυτό το *intent* γίνεται κλήση όταν ο χρήστης γράψει κάτι μη αναμενόμενο πέρα από αυτά που έχουν οριστεί στα *intents*. Επιστρέφει ένα μήνυμα σφάλματος κατανόησης του μηνύματος που έλαβε και επανάληψης ερώτησής του.

6.4 Κριτήρια Αξιολόγησης της Απόδοσης του Πράκτορα

Η αξιολόγηση της απόδοσης του πράκτορα-βοηθού μαθήματος είναι απαραίτητη για τη διασφάλιση της ακρίβειας, της λειτουργικότητας και της χρηστικότητάς του.

Προκειμένου να αξιολογήσουμε την απόδοσή του, χρησιμοποιούμε τα παρακάτω **κριτήρια**:

- **Ακρίβεια των Απαντήσεων**:

  Κριτήριο: Η ακρίβεια των απαντήσεων που παρέχει ο πράκτορας στα ερωτήματα των χρηστών είναι κρίσιμη για την αξιολόγησή του. Πρέπει να εξετάσουμε πόσο καλά ο πράκτορας κατανοεί τα ερωτήματα και παρέχει σωστές απαντήσεις.

  Παράδειγμα: Ένας χρήστης ρωτά τον πράκτορα για την περιγραφή ενός μαθήματος με τον όνομα "Μαθηματικά Ι". Ο πράκτορας πρέπει να αναζητήσει στη βάση δεδομένων και να επιστρέψει την περιγραφή για το συγκεκριμένο μάθημα, και αυτή η περιγραφή πρέπει να είναι σωστή και ακριβής.

- **Ανταπόκριση στα Ερωτήματα των Χρηστών**:

  Κριτήριο: Η ανταπόκριση στα ερωτήματα των χρηστών είναι σημαντική. Πρέπει να εξετάσουμε τον χρόνο απόκρισης του πράκτορα και την ικανότητά του να ανταποκρίνεται αποτελεσματικά στα ερωτήματα.

Παράδειγμα: Ένας χρήστης υποβάλλει ένα ερώτημα για το ποιο είναι το πρόγραμμα σπουδών. Ο πράκτορας πρέπει να ανταποκριθεί σε λίγα δευτερόλεπτα με την απαραίτητη πληροφορία.

- **Χρήση Σωστών Δεδομένων**:

  Κριτήριο: Ο πράκτορας πρέπει να χρησιμοποιεί τα σωστά δεδομένα από τη βάση δεδομένων για να παρέχει ακριβείς πληροφορίες. Πρέπει να εξετάσουμε αν ο πράκτορας ανακτά τις σωστές πληροφορίες από τη βάση δεδομένων.

  Παράδειγμα: Αν ένας χρήστης ρωτά για τα μαθήματα του νέου εξαμήνου, ο πράκτορας πρέπει να αναζητήσει τα μαθήματα μόνο για το επόμενο εξάμηνο και όχι για προηγούμενα.

- **Επικοινωνία με τον Χρήστη**:

  Κριτήριο: Η επικοινωνία με τον χρήστη πρέπει να είναι φιλική και κατανοητή. Πρέπει να εξετάσουμε τη σαφήνεια των απαντήσεων και τη δυνατότητα του πράκτορα να απαντά στα ερωτήματα των χρηστών με κατανόηση.

  Παράδειγμα: Αν ένας χρήστης ρωτά για τον προσανατολισμό του εξαμήνου, ο πράκτορας πρέπει να παρέχει μια κατανοητή και σαφή απάντηση χωρίς πολύ τεχνικό ορολογία.

- **Προσαρμοστικότητα**:

  Κριτήριο: Η ικανότητα του πράκτορα να προσαρμόζεται σε νέα δεδομένα και ερωτήματα των χρηστών είναι σημαντική. Πρέπει να εξετάσουμε αν ο πράκτορας μπορεί να ανταποκριθεί σε ερωτήματα εκτός του κανονικού πλαισίου του.

  Παράδειγμα: Αν ένας χρήστης ρωτά για πληροφορίες που δεν είναι προγραμματισμένες στον πράκτορα, αυτός πρέπει να προσπαθήσει να ανταποκριθεί με τον καλύτερο δυνατό τρόπο.

- **Απόδοση σε Συνθήκες Φόρτου**:

  Κριτήριο: Η απόδοση του πράκτορα πρέπει να είναι σταθερή ακόμη και κατά τις περιόδους υψηλού φόρτου. Πρέπει να εξετάσουμε πώς ανταποκρίνεται ο πράκτορας όταν υπάρχει αυξημένη χρήση.

  Παράδειγμα: Κατά την εγγραφή των μαθημάτων, πολλοί χρήστες ενδέχεται να κάνουν ερωτήσεις ταυτόχρονα. Ο πράκτορας πρέπει να διατηρήσει την απόδοσή του χωρίς να κολλήσει ή να υποστεί καθυστερήσεις.

Συνοψίζοντας, η αξιολόγηση της απόδοσης του πράκτορα βασίζεται σε πολλά κριτήρια που καλύπτουν την ακρίβεια, την αποτελεσματικότητα, την ευχρηστία και τη συνολική εμπειρία των χρηστών. Η ανάλυση αυτών των κριτηρίων θα μας επιτρέψει να κατανοήσουμε καλύτερα την απόδοση του πράκτορα και να βελτιώσουμε την εφαρμογή μας με βάση τα ευρήματα.

6.5 Επίλογος

Στο πλαίσιο αυτού του κεφαλαίου, αναλύσαμε λεπτομερώς την υλοποίηση του πράκτορα-βοηθού μαθήματος μέσω του *DialogFlow*. Δημιουργήσαμε *intents*, ορίσαμε *training phrases, entities* και καθορίσαμε απαντήσεις για να επιτρέψουμε στους χρήστες να αποκτήσουν πληροφορίες σχετικά με τα μαθήματα του προγράμματος μελέτης. Επίσης, υλοποιήσαμε την επικοινωνία με τη βάση δεδομένων *MySQL*, επιτρέποντας την ανάκτηση ακριβών πληροφοριών σχετικά με τα μαθήματα που ζήτησαν οι χρήστες.

Το *DialogFlow* μας παρέχει ένα ισχυρό εργαλείο για τη δημιουργία ευέλικτων διαλόγων με τους χρήστες μας. Επιπλέον, η συνδυασμένη χρήση του *FastAPI* με τη βάση δεδομένων *MySQL* μας επέτρεψε να ανταποκριθούμε γρήγορα και αποτελεσματικά στα αιτήματα των χρηστών, παρέχοντάς τους ακριβείς πληροφορίες που χρειάζονται.

Αυτό το κεφάλαιο αποτελεί τη βάση για τη σωστή λειτουργία του πράκτορα-βοηθού μαθήματος στην εφαρμογή μας. Ο σχεδιασμός και η σωστή υλοποίηση των *intents*, των φράσεων εκπαίδευσης, και της επικοινωνίας με τη βάση δεδομένων αποτελούν τη βάση για έναν αποτελεσματικό πράκτορα-βοηθό μαθήματος.

Κεφάλαιο 7ο: Συμπεράσματα και Μελλοντική Επέκταση

Το έβδομο κεφάλαιο της πτυχιακής αυτής εργασίας σηματοδοτεί το κλείσιμο της μελέτης και περιλαμβάνει τα συμπεράσματα που προκύπτουν από την έρευνα, καθώς και προτάσεις για μελλοντική επέκταση και βελτίωση της αναπτυχθείσας εφαρμογής.

7.1 Συνοψίζοντας τα Κύρια Αποτελέσματα και τα Συμπεράσματα

Συνοψίζοντας τα κύρια αποτελέσματα και τα συμπεράσματα της παρούσας πτυχιακής εργασίας, προέκυψαν σημαντικά ευρήματα και εκδηλώθηκαν προοπτικές για μελλοντική έρευνα και ανάπτυξη. Η εργασία αυτή επικεντρώθηκε στη βελτιστοποίηση της απόδοσης ενός πράκτορα σε περιβάλλον χρήστη-υπολογιστή, εξετάζοντας διάφορες πτυχές της αρχιτεκτονικής και της λειτουργίας του.

Καταρχάς, πραγματοποιήθηκε μια εκτενής ανασκόπηση της βιβλιογραφίας σχετικά με τους υπάρχοντες πράκτορες και τις σχετικές τεχνολογίες. Αυτό βοήθησε στον καθορισμό των βασικών αρχών και των τεχνικών που απαιτούνται για τη βελτιστοποίηση της απόδοσης.

Έπειτα, αναπτύχθηκε ένας πράκτορας που ενσωμάτωσε τις νέες μεθόδους και τεχνικές που εξετάστηκαν. Οι αλγόριθμοι ανταπόκρισης βελτιστοποιήθηκαν, προσφέροντας ταχείες και ακριβείς απαντήσεις σε ερωτήσεις του χρήστη. Επιπλέον, η διαχείριση ερωτημάτων βελτιώθηκε σημαντικά, ταξινομώντας σε σειρά προτεραιότητας τα ερωτήματα με βάση την επείγουσα ανάγκη.

Σημαντικές προτάσεις για μελλοντική επέκταση και βελτίωση της εφαρμογής συμπεριλαμβάνουν την εξέταση περισσότερων παραμέτρων επίδοσης, όπως την εξέταση της αντοχής του πράκτορα σε

υψηλό φορτίο, την επέκταση του συστήματος για να υποστηρίζει περισσότερες γλώσσες, και την εξερεύνηση νέων μεθόδων βελτιστοποίησης της επικοινωνίας και της απόδοσης.

7.2 Προτάσεις για Μελλοντική Επέκταση και Βελτίωση της Εφαρμογής

Συνεχίζοντας, σε αυτό το κεφάλαιο το οποίο αφορά τις προτάσεις για μελλοντική επέκταση και βελτίωση της εφαρμογής, θα αναλύσουμε λίγο πιο λεπτομερώς κάθε προτεινόμενο βήμα, παρέχοντας παραδείγματα όπου αυτό είναι εφικτό.

Από τα κύρια αποτελέσματα και τα συμπεράσματα της πτυχιακής εργασίας που αναφέρθηκαν και στο προηγούμενο κεφάλαιο, προκύπτει ότι ο πράκτορας χρειάζεται βελτίωση ώστε να αποτελέσει ένα ισχυρό εργαλείο για την επίλυση προβλημάτων και την παροχή πληροφοριών στους χρήστες. Κάποιες προτάσεις μελλοντικής βελτίωσης του διαλογικού πράκτορα είναι οι εξής:

Προσθήκη νέων δεδομένων και ερωτήσεων

Η συνεχής ενημέρωση της βάσης δεδομένων είναι απαραίτητη για τη βελτίωση του πράκτορα. Σκοπός είναι να προστεθούν νέα δεδομένα και ερωτήσεις, επεκτείνοντας το φάσμα των θεμάτων που ήδη καλύπτονται.

Παράδειγμα: Ένας ακόμη τομέας που μπορεί να προστεθεί είναι η ενημέρωση για τα διοικητικά θέματα. Η προσθήκη ερωτήσεων που αφορούν την γραμματεία, όπως για παράδειγμα το τηλέφωνο της σχολής, έγγραφα προς συμπλήρωση προς φοιτητές καθώς και ωράρια λειτουργίας. Ακόμη μια μελλοντική εξέλιξη είναι η προσθήκη στην βάση δεδομένων και των καθηγητών έτσι ώστε ένας φοιτητής να μπορεί να ενημερώνεται για το βιογραφικό του κάθε καθηγητή.

Δημιουργία Φιλικού και Ευανάγνωστου Front-end

Η ανάπτυξη ενός φιλικού και ευανάγνωστου *front-end* αποτελεί βασικό στοιχείο για την επιτυχημένη χρήση του πράκτορα. Ένα καλά σχεδιασμένο *front-end* διευκολύνει τους χρήστες να αλληλεπιδρούν με τον πράκτορα και να λαμβάνουν γρήγορες και σαφείς απαντήσεις.

Παράδειγμα: Ας υποθέσουμε ότι ο χρήστης έχει συνδεθεί στην εφαρμογή και βλέπει το *front-end*.

Το φιλικό και ευανάγνωστο σχεδιασμό θα συμπεριλάμβανε τα εξής:

- **Ευανάγνωστο Κείμενο:** Όλο το κείμενο πρέπει να είναι καλά δομημένο και ευανάγνωστο, χρησιμοποιώντας κατάλληλη γραμματοσειρά, μέγεθος γραμματοσειράς και χρώματα.

- **Οδηγίες και Βοήθεια:** Ένα καλό *front-end* πρέπει να περιλαμβάνει ξεκάθαρες οδηγίες και βοήθεια για τους χρήστες. Για παράδειγμα, μπορεί να υπάρχει ένα κουμπί "Βοήθεια" που εξηγεί πώς να κάνει κάποιος μια ερώτηση ή πώς να αξιοποιήσει τις λειτουργίες του πράκτορα.

- **Συνεχής Επικοινωνία:** Ένα φιλικό *front-end* θα είχε ένα πεδίο εισαγωγής κειμένου που ενθαρρύνει τους χρήστες να κάνουν ερωτήσεις. Όταν ο χρήστης αρχίζει να πληκτρολογεί, μπορεί να εμφανίζονται αυτόματες προτάσεις για να τον βοηθήσουν να διατυπώσει την ερώτησή του.

- **Κατηγοριοποίηση και Αναζήτηση:** Ένα καλό *front-end* θα παρέχει τη δυνατότητα κατηγοριοποίησης των ερωτήσεων και αναζήτησης για να βρεί ο χρήστης γρήγορα τις πληροφορίες που χρειάζεται.

Ο φιλικός σχεδιασμός του *front-end* θα διευκολύνει τους χρήστες στην αλληλεπίδραση με τον πράκτορα και θα τους καθοδηγεί για να λάβουν απαντήσεις στα ερωτήματά τους με ευκολία και αποτελεσματικότητα.

Δημιουργία μιας Πολυγλωσσικής Εμπειρίας

Η προσθήκη πολυγλωσσικής υποστήριξης επεκτείνει την επιρροή της εφαρμογής σας σε παγκόσμιο επίπεδο, δίνοντας τη δυνατότητα σε χρήστες από διάφορες γλωσσικές περιοχές να αλληλεπιδρούν με τον πράκτορά σας στη γλώσσα της προτίμησής τους.

Παράδειγμα: Έστω ότι η εφαρμογή σας υποστηρίζει πολλές γλώσσες, συμπεριλαμβανομένων των Αγγλικών, των Ισπανικών και των Γαλλικών. Τότε, κάποιος φοιτητής Erasmus που μιλά Ισπανικά θα μπορεί να αλληλεπιδράσει με τον πράκτορα χρησιμοποιώντας την Ισπανική του γλώσσα και να λαμβάνει απαντήσεις στην ίδια γλώσσα. Αυτό θα καθιστούσε την εφαρμογή πιο προσιτή και προσαρμοσμένη για κοινό με διάφορες γλωσσικές ανάγκες.

Εξατομικευμένες και Έξυπνες Απαντήσεις

Η ενσωμάτωση προηγμένης τεχνητής νοημοσύνης ανοίγει νέες προοπτικές για τον πράκτορα. Μπορεί να μάθει από τις αλληλεπιδράσεις των χρηστών και να παρέχει εξατομικευμένες απαντήσεις, λαμβάνοντας υπόψη το ιστορικό τους και τις προτιμήσεις τους.

Παράδειγμα: Ένας χρήστης που συχνά θέτει ερωτήσεις σχετικά με το μάθημα της *Java*, μπορεί να λάβει πιο εξειδικευμένες απαντήσεις για το συγκεκριμένο θέμα που τον απασχολεί, καθώς ο πράκτορας θα αντλήσει πληροφορίες από το ιστορικό του.

Δημιουργία Διεπαφής

Η ανάπτυξη μια διεπαφής *API* με την βάση δεδομένων που έχουν γίνει έγκυρα *scrap*, θα μπορέσει να δημιουργήσει μια πιο ασφαλή και πιο ευρέα χρησιμοποιήσιμη εφαρμογή ακόμη και σε μελλοντικά σχέδια.

7.3 Επίλογος

Εν κατακλείδι, οι προτάσεις για τη μελλοντική επέκταση και βελτίωση της εφαρμογής σας ανοίγουν νέους ορίζοντες και ευκαιρίες. Με την συνεχή προσθήκη νέων δεδομένων, την ανάπτυξη ενός φιλικού *front-end*, την υποστήριξη πολλών γλωσσών, την ενσωμάτωση τεχνητής νοημοσύνης και τη βελτίωση της ασφάλειας και του απορρήτου, η εφαρμογή θα αναδεικνύεται μια ακόμα περισσότερο προηγμένη και ευέλικτη λύση για τους χρήστες.

Αυτές οι προτάσεις δεν αποτελούν μόνο εξελίξεις, αλλά και προσφέρουν σημαντικά οφέλη. Επιτρέπουν στους χρήστες να έχουν περισσότερες επιλογές και εξατομικευμένες εμπειρίες, ενώ παράλληλα διασφαλίζουν ότι τα δεδομένα τους είναι προστατευμένα.

Με αυτές τις βελτιώσεις, η εφαρμογή ευελπιστούμε ότι έχει τη δυνατότητα να γίνει πιο ανταγωνιστική, προσελκύοντας νέους χρήστες και διατηρώντας τους ήδη υπάρχοντες.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] “Τι Είναι η Μηχανική Μάθηση; | Ορισμός, Τύποι Και Παραδείγματα | SAP Insights.” *SAP*, https://www.sap.com/greece/products/artificial-intelligence/what-is-machine-learning.html. Accessed 30 Sept. 2023.

[2] “Dialogflow Documentation | Google Cloud.” *Google Cloud*, https://cloud.google.com/dialogflow/docs. Accessed 30 Sept. 2023.

[3] Forogh, Parwiz. *How to Build Chatbot with Google DialogFlow*. https://www.udemy.com/course/how-to-build-chatbot-with-google-dialogflow/. Accessed 30 Sept. 2023.

[4] Burns, et al. “What Is Artificial Intelligence and How Does AI Work? TechTarget.” *Enterprise AI*, TechTarget, 10 July 2023, https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence.

[5] Burns, et al. “What Is Artificial Intelligence and How Does AI Work? TechTarget.” *Enterprise AI*, TechTarget, 10 July 2023, https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence.

[6] Burns, et al. “What Is Artificial Intelligence and How Does AI Work? TechTarget.” *Enterprise AI*, TechTarget, 10 July 2023, https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence.

[7] “What Is Deep Learning? | IBM.” *IBM - United States*, https://www.ibm.com/topics/deep-learning. Accessed 30 Sept. 2023.

[8] “What Are Convolutional Neural Networks? | IBM.” *IBM - United States*, https://www.ibm.com/topics/convolutional-neural-networks. Accessed 1 Oct. 2023.

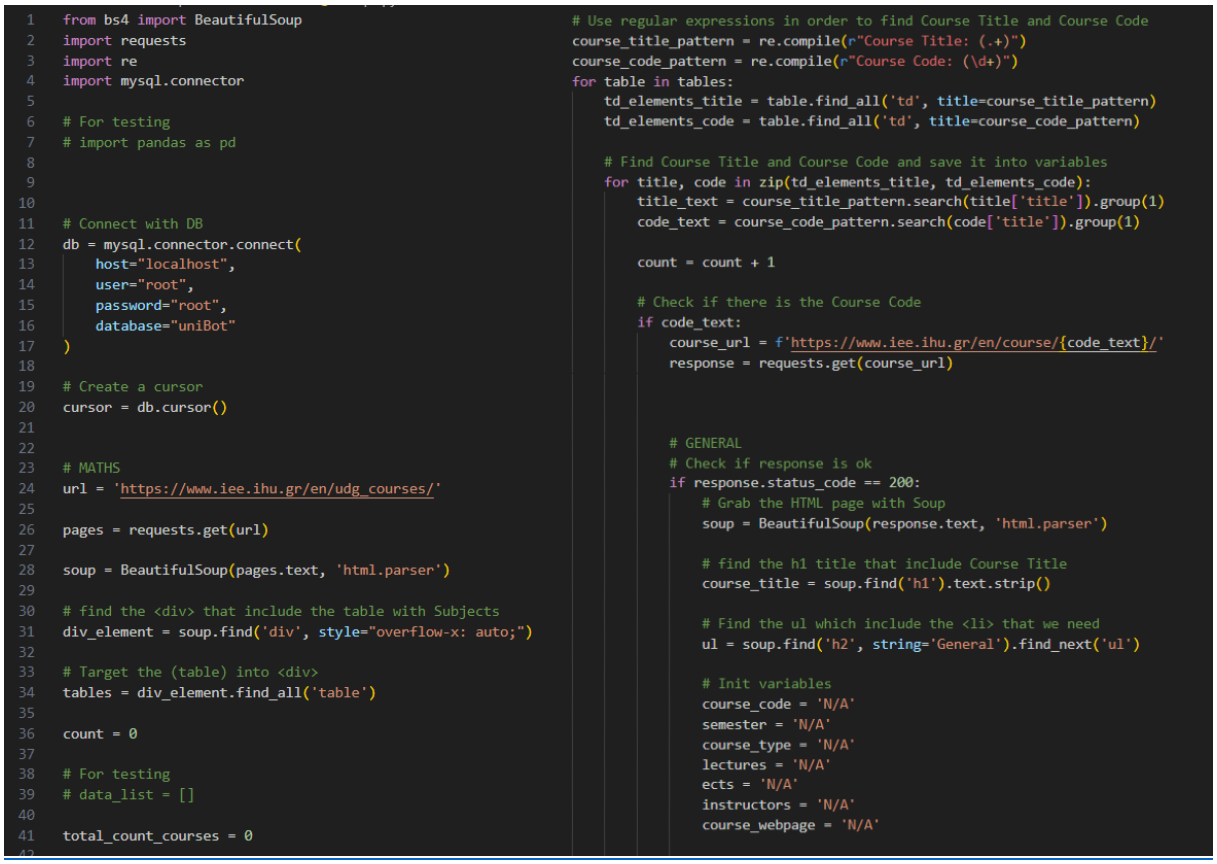
[9] Maxime. “What Is a Transformer?. An Introduction to Transformers And… | by Maxime | Inside Machine Learning | Medium.” *Medium*, Inside Machine learning, 4 Jan. 2019, https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04.

[10] “GPT-3 vs. BERT - Which Is Best? This Article Compares Both in Depth.” *SoftTeco - Custom Software Development Company*, SoftTeco, https://softteco.com/blog/bert-vs-chatgpt. Accessed 1 Oct. 2023.

[11] “Choosing a Chatbot Development Tool | IEEE Journals & Magazine | IEEE Xplore.” *IEEE Xplore*, https://ieeexplore.ieee.org/abstract/document/9364349. Accessed 1 Oct. 2023.

ΠΑΡΑΡΤΗΜΑ Α : ΚΩΔΙΚΑΣ SCRAPPER

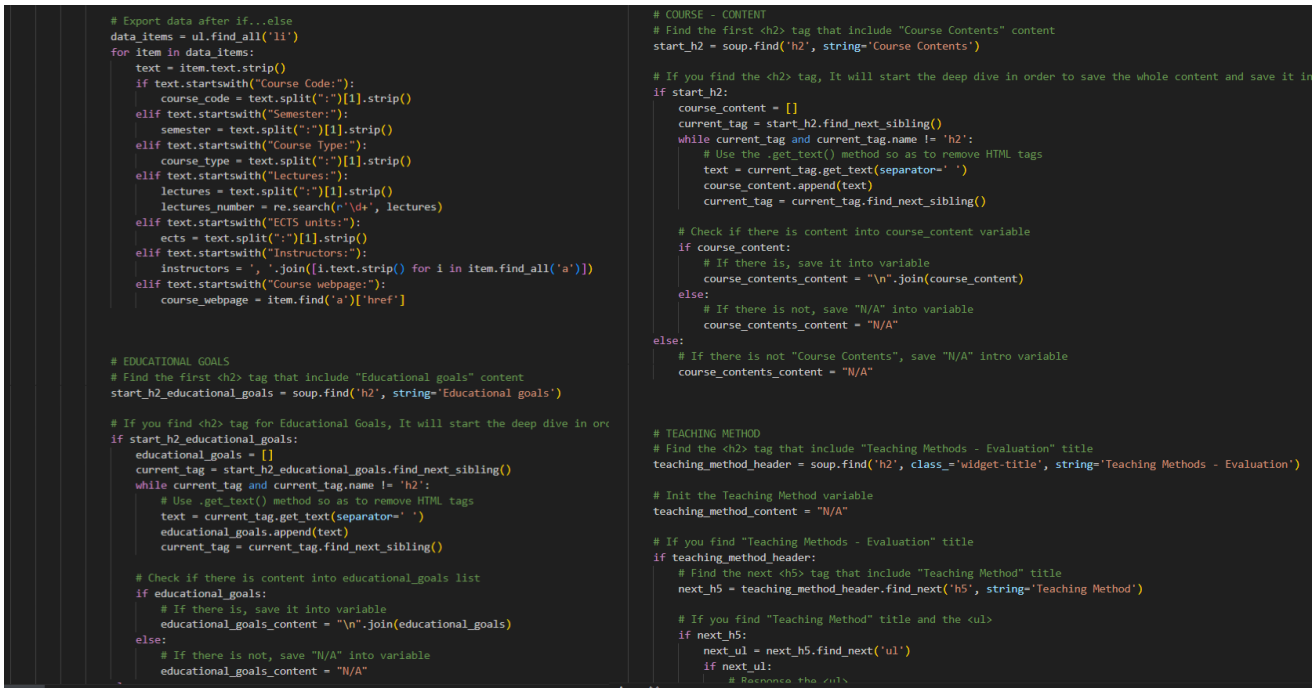
Παρακάτω φαίνεται αναλυτικά ο κώδικας του *scrapper* για την άντληση των πληροφοριών των μαθημάτων της σχολής και η τοποθέτησή τους στην βάση δεδομένων.



```python
from bs4 import BeautifulSoup
import requests
import re
import mysql.connector

# For testing
# import pandas as pd



# Connect with DB
db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="root",
    database="uniBot"
)

# Create a cursor
cursor = db.cursor()


# MATHS
url = 'https://www.iee.ihu.gr/en/udg_courses/'

pages = requests.get(url)

soup = BeautifulSoup(pages.text, 'html.parser')

# find the <div> that include the table with Subjects
div_element = soup.find('div', style="overflow-x: auto;")

# Target the (table) into <div>
tables = div_element.find_all('table')

count = 0

# For testing
# data_list = []

total_count_courses = 0
```

```python
# Use regular expressions in order to find Course Title and Course Code
course_title_pattern = re.compile(r"Course Title: (.+)")
course_code_pattern = re.compile(r"Course Code: (\d+)")
for table in tables:
    td_elements_title = table.find_all('td', title=course_title_pattern)
    td_elements_code = table.find_all('td', title=course_code_pattern)

    # Find Course Title and Course Code and save it into variables
    for title, code in zip(td_elements_title, td_elements_code):
        title_text = course_title_pattern.search(title['title']).group(1)
        code_text = course_code_pattern.search(code['title']).group(1)

        count = count + 1

        # Check if there is the Course Code
        if code_text:
            course_url = f'https://www.iee.ihu.gr/en/course/{code_text}/'
            response = requests.get(course_url)



            # GENERAL
            # Check if response is ok
            if response.status_code == 200:
                # Grab the HTML page with Soup
                soup = BeautifulSoup(response.text, 'html.parser')

                # find the h1 title that include Course Title
                course_title = soup.find('h1').text.strip()

                # Find the ul which include the <li> that we need
                ul = soup.find('h2', string='General').find_next('ul')

                # Init variables
                course_code = 'N/A'
                semester = 'N/A'
                course_type = 'N/A'
                lectures = 'N/A'
                ects = 'N/A'
                instructors = 'N/A'
                course_webpage = 'N/A'
```

Εικόνα Python Scrapper (1)



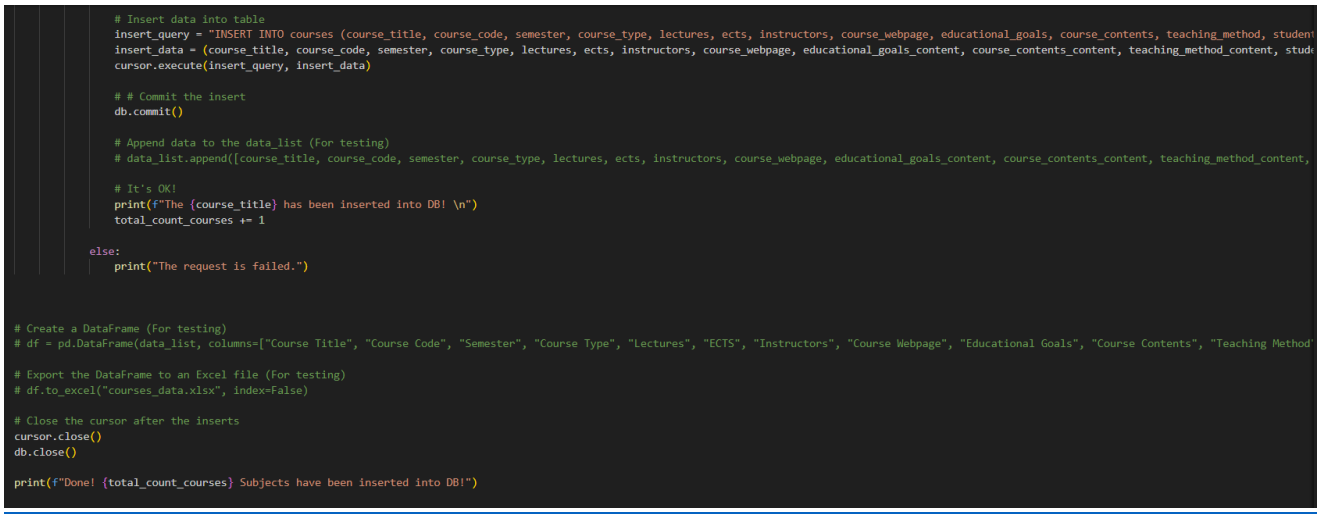
```
# Export data after if...else
data_items = ul.find_all('li')
for item in data_items:
    text = item.text.strip()
    if text.startswith("Course Code:"):
        course_code = text.split(":")[1].strip()
    elif text.startswith("Semester:"):
        semester = text.split(":")[1].strip()
    elif text.startswith("Course Type:"):
        course_type = text.split(":")[1].strip()
    elif text.startswith("Lectures:"):
        lectures = text.split(":")[1].strip()
        lectures_number = re.search(r'\d+', lectures)
    elif text.startswith("ECTS units:"):
        ects = text.split(":")[1].strip()
    elif text.startswith("Instructors:"):
        instructors = ', '.join([i.text.strip() for i in item.find_all('a')])
    elif text.startswith("Course webpage:"):
        course_webpage = item.find('a')['href']



# EDUCATIONAL GOALS
# Find the first <h2> tag that include "Educational goals" content
start_h2_educational_goals = soup.find('h2', string='Educational goals')

# If you find <h2> tag for Educational Goals, It will start the deep dive in or
if start_h2_educational_goals:
    educational_goals = []
    current_tag = start_h2_educational_goals.find_next_sibling()
    while current_tag and current_tag.name != 'h2':
        # Use .get_text() method so as to remove HTML tags
        text = current_tag.get_text(separator=' ')
        educational_goals.append(text)
        current_tag = current_tag.find_next_sibling()

    # Check if there is content into educational_goals list
    if educational_goals:
        # If there is, save it into variable
        educational_goals_content = "\n".join(educational_goals)
    else:
        # If there is not, save "N/A" into variable
        educational_goals_content = "N/A"
```

```
# COURSE - CONTENT
# Find the first <h2> tag that include "Course Contents" content
start_h2 = soup.find('h2', string='Course Contents')

# If you find the <h2> tag, It will start the deep dive in order to save the whole content and save it in
if start_h2:
    course_content = []
    current_tag = start_h2.find_next_sibling()
    while current_tag and current_tag.name != 'h2':
        # Use the .get_text() method so as to remove HTML tags
        text = current_tag.get_text(separator=' ')
        course_content.append(text)
        current_tag = current_tag.find_next_sibling()

    # Check if there is content into course_content variable
    if course_content:
        # If there is, save it into variable
        course_contents_content = "\n".join(course_content)
    else:
        # If there is not, save "N/A" into variable
        course_contents_content = "N/A"
else:
    # If there is not "Course Contents", save "N/A" intro variable
    course_contents_content = "N/A"



# TEACHING METHOD
# Find the <h2> tag that include "Teaching Methods - Evaluation" title
teaching_method_header = soup.find('h2', class_='widget-title', string='Teaching Methods - Evaluation')

# Init the Teaching Method variable
teaching_method_content = "N/A"

# If you find "Teaching Methods - Evaluation" title
if teaching_method_header:
    # Find the next <h5> tag that include "Teaching Method" title
    next_h5 = teaching_method_header.find_next('h5', string='Teaching Method')

    # If you find "Teaching Method" title and the <ul>
    if next_h5:
        next_ul = next_h5.find_next('ul')
        if next_ul:
            # Response the <ul>
```

Εικόνα Python Scrapper (2)

```
# TEACHING METHOD
# Find the <h2> tag that include "Teaching Methods - Evaluation" title
teaching_method_header = soup.find('h2', class_='widget-title', string='Teaching Methods - Evaluation')

# Init the Teaching Method variable
teaching_method_content = "N/A"

# If you find "Teaching Methods - Evaluation" title
if teaching_method_header:
    # Find the next <h5> tag that include "Teaching Method" title
    next_h5 = teaching_method_header.find_next('h5', string='Teaching Method')

    # If you find "Teaching Method" title and the <ul>
    if next_h5:
        next_ul = next_h5.find_next('ul')
        if next_ul:
            # Response the <ul>
            teaching_method_content = next_ul.text.strip()



# STUDENT EVALUATION
# Find the <h5> tag that it have "Students evaluation" as content
h5_element = soup.find('h5', string='Students evaluation')

# If thare is <h5> tag
if h5_element:
    # Find the next <p> tag in the row
    next_p = h5_element.find_next_sibling('p')

    # If there is <p>
    if next_p:
        # Add break line (\n) before "- ..."
        students_evaluation_text = next_p.get_text("\n ", strip=True)
    else:
        # If there is no <p> tag, add "N/A" into variable
        students_evaluation_text = "N/A"
else:
    # If there is no <h5> tag, add "N/A" into variable
    students_evaluation_text = "N/A"
```

Εικόνα Python Scrapper (3)

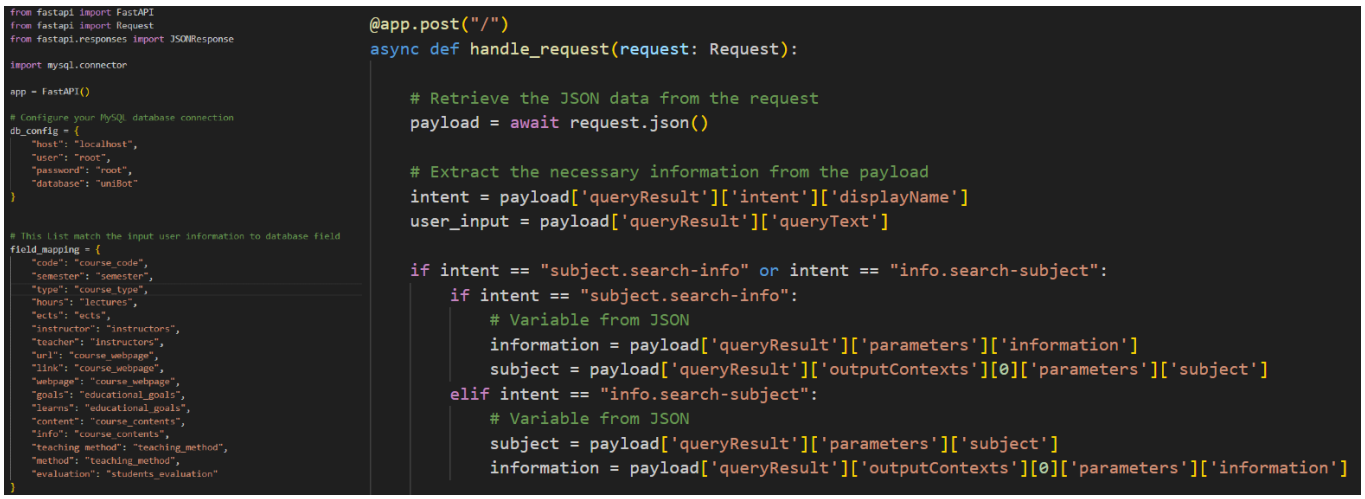


```
            # Insert data into table
            insert_query = "INSERT INTO courses (course_title, course_code, semester, course_type, lectures, ects, instructors, course_webpage, educational_goals, course_contents, teaching_method, student
            insert_data = (course_title, course_code, semester, course_type, lectures, ects, instructors, course_webpage, educational_goals_content, course_contents_content, teaching_method_content, stud
            cursor.execute(insert_query, insert_data)

            # # Commit the insert
            db.commit()

            # Append data to the data_list (For testing)
            # data_list.append([course_title, course_code, semester, course_type, lectures, ects, instructors, course_webpage, educational_goals_content, course_contents_content, teaching_method_content,

            # It's OK!
            print(f"The {course_title} has been inserted into DB! \n")
            total_count_courses += 1

        else:
            print("The request is failed.")



# Create a DataFrame (For testing)
# df = pd.DataFrame(data_list, columns=["Course Title", "Course Code", "Semester", "Course Type", "Lectures", "ECTS", "Instructors", "Course Webpage", "Educational Goals", "Course Contents", "Teaching Method"

# Export the DataFrame to an Excel file (For testing)
# df.to_excel("courses_data.xlsx", index=False)

# Close the cursor after the inserts
cursor.close()
db.close()

print(f"Done! {total_count_courses} Subjects have been inserted into DB!")
```

Εικόνα Python Scrapper (4)

ΠΑΡΑΡΤΗΜΑ Β : ΚΩΔΙΚΑΣ PYTHON

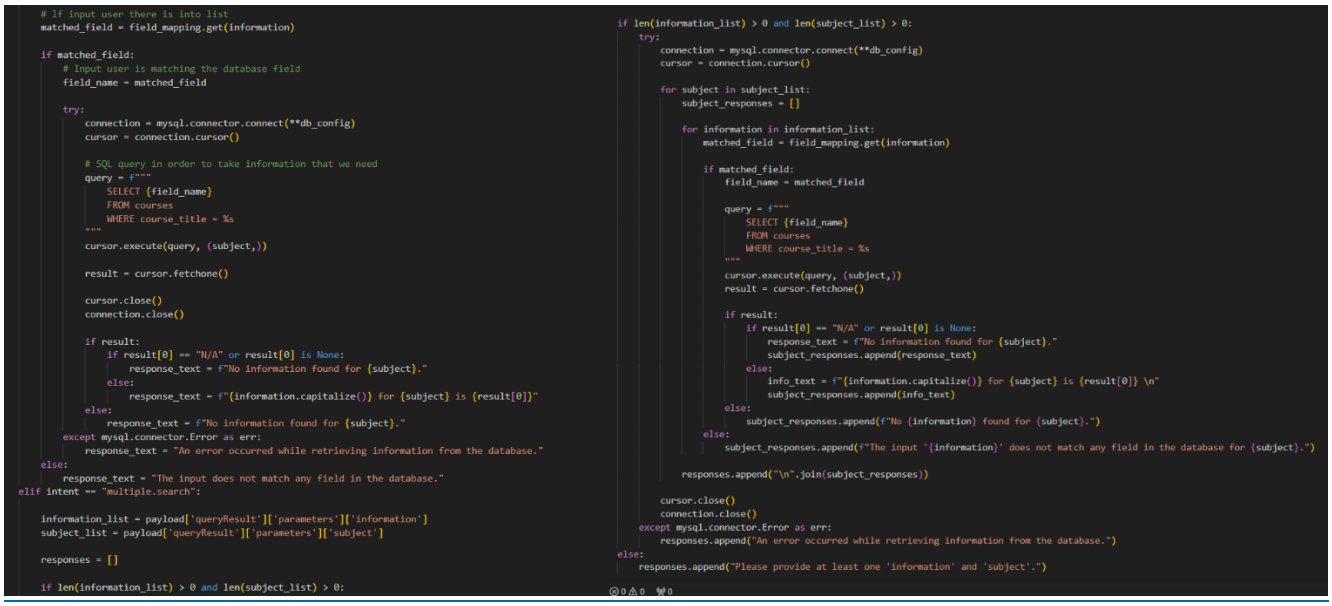
Παρακάτω φαίνεται αναλυτικά ο κώδικας της *Back-End* υλοποίησης για την διαχείριση του διαλογικού πράκτορα με την χρήση της γλώσσας προγραμματισμού *Python*.



```
from fastapi import FastAPI
from fastapi import Request
from fastapi.responses import JSONResponse

import mysql.connector

app = FastAPI()

# Configure your MySQL database connection
db_config = {
    "host": "localhost",
    "user": "root",
    "password": "root",
    "database": "uniBot"
}


# This list match the input user information to database field
field_mapping = {
    "code": "course_code",
    "semester": "semester",
    "type": "course_type",
    "hours": "lectures",
    "ects": "ects",
    "instructor": "instructors",
    "teacher": "instructors",
    "url": "course_webpage",
    "link": "course_webpage",
    "webpage": "course_webpage",
    "goals": "educational_goals",
    "learns": "educational_goals",
    "content": "course_contents",
    "info": "course_contents",
    "teaching method": "teaching_method",
    "method": "teaching_method",
    "evaluation": "students_evaluation"
}
```

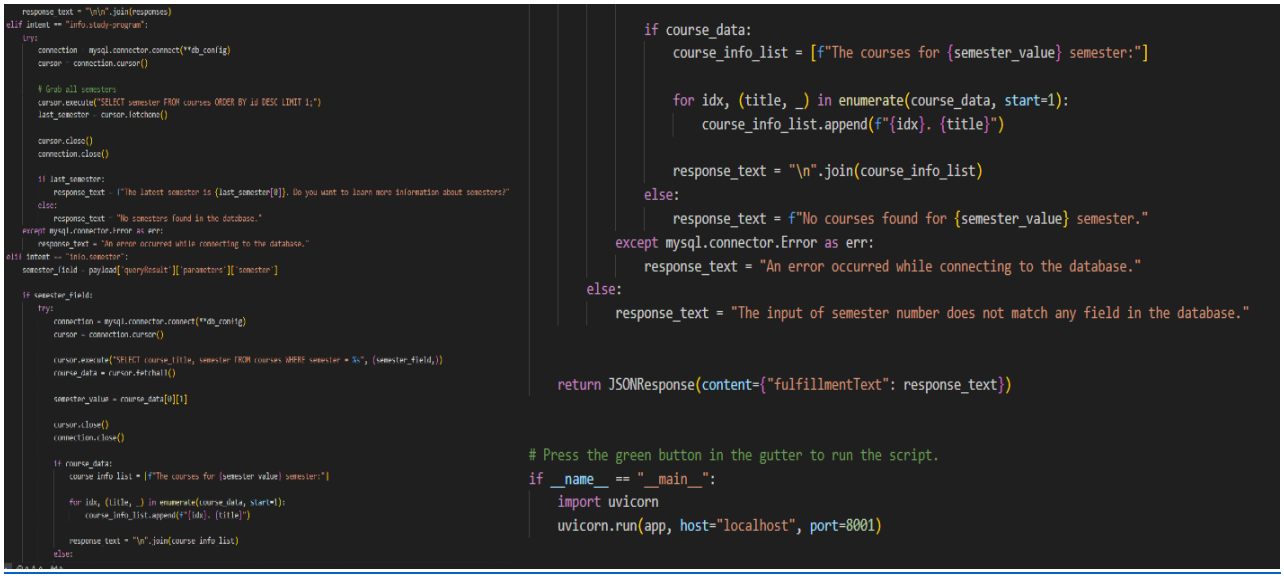
```
@app.post("/")
async def handle_request(request: Request):

    # Retrieve the JSON data from the request
    payload = await request.json()

    # Extract the necessary information from the payload
    intent = payload['queryResult']['intent']['displayName']
    user_input = payload['queryResult']['queryText']

    if intent == "subject.search-info" or intent == "info.search-subject":
        if intent == "subject.search-info":
            # Variable from JSON
            information = payload['queryResult']['parameters']['information']
            subject = payload['queryResult']['outputContexts'][0]['parameters']['subject']
        elif intent == "info.search-subject":
            # Variable from JSON
            subject = payload['queryResult']['parameters']['subject']
            information = payload['queryResult']['outputContexts'][0]['parameters']['information']
```

Εικόνα Python (1)



```
    # If input user there is into list
    matched_field = field_mapping.get(information)

    if matched_field:
        # Input user is matching the database field
        field_name = matched_field

        try:
            connection = mysql.connector.connect(**db_config)
            cursor = connection.cursor()

            # SQL query in order to take information that we need
            query = f"""
                SELECT {field_name}
                FROM courses
                WHERE course_title = %s
            """
            cursor.execute(query, (subject,))

            result = cursor.fetchone()

            cursor.close()
            connection.close()

            if result:
                if result[0] == "N/A" or result[0] is None:
                    response_text = f"No information found for {subject}."
                else:
                    response_text = f"{information.capitalize()} for {subject} is {result[0]}"
            else:
                response_text = f"No information found for {subject}."
        except mysql.connector.Error as err:
            response_text = "An error occurred while retrieving information from the database."
    else:
        response_text = "The input does not match any field in the database."
elif intent == "multiple.search":

    information_list = payload['queryResult']['parameters']['information']
    subject_list = payload['queryResult']['parameters']['subject']

    responses = []

    if len(information_list) > 0 and len(subject_list) > 0:
        try:
            connection = mysql.connector.connect(**db_config)
            cursor = connection.cursor()

            for subject in subject_list:
                subject_responses = []

                for information in information_list:
                    matched_field = field_mapping.get(information)

                    if matched_field:
                        field_name = matched_field

                        query = f"""
                            SELECT {field_name}
                            FROM courses
                            WHERE course_title = %s
                        """
                        cursor.execute(query, (subject,))
                        result = cursor.fetchone()

                        if result:
                            if result[0] == "N/A" or result[0] is None:
                                response_text = f"No information found for {subject}."
                                subject_responses.append(response_text)
                            else:
                                info_text = f"{information.capitalize()} for {subject} is {result[0]} \n"
                                subject_responses.append(info_text)
                        else:
                            subject_responses.append(f"No {information} found for {subject}.")
                    else:
                        subject_responses.append(f"The input '{information}' does not match any field in the database for {subject}.")

                responses.append("\n".join(subject_responses))

            cursor.close()
            connection.close()
        except mysql.connector.Error as err:
            responses.append("An error occurred while retrieving information from the database.")
    else:
        responses.append("Please provide at least one 'information' and 'subject'.")
```

Εικόνα Python (2)



```
    response_text = "\n\n".join(responses)
elif intent == "info.study-program":
    try:
        connection = mysql.connector.connect(**db_config)
        cursor = connection.cursor()

        # Grab all semesters
        cursor.execute("SELECT semester FROM courses ORDER BY id DESC LIMIT 1;")
        last_semester = cursor.fetchone()

        cursor.close()
        connection.close()

        if last_semester:
            response_text = f"The latest semester is {last_semester[0]}. Do you want to learn more information about semesters?"
        else:
            response_text = "No semesters found in the database."
    except mysql.connector.Error as err:
        response_text = "An error occurred while connecting to the database."
elif intent == "info.semester":
    semester_field = payload['queryResult']['parameters']['semester']

    if semester_field:
        try:
            connection = mysql.connector.connect(**db_config)
            cursor = connection.cursor()

            cursor.execute("SELECT course_title, semester FROM courses WHERE semester = %s", (semester_field,))
            course_data = cursor.fetchall()

            semester_value = course_data[0][1]

            cursor.close()
            connection.close()

            if course_data:
                course_info_list = [f"The courses for {semester_value} semester:"]

                for idx, (title, _) in enumerate(course_data, start=1):
                    course_info_list.append(f"{idx}. {title}")

                response_text = "\n".join(course_info_list)
            else:
                response_text = f"No courses found for {semester_value} semester."
        except mysql.connector.Error as err:
            response_text = "An error occurred while connecting to the database."
    else:
        response_text = "The input of semester number does not match any field in the database."


return JSONResponse(content={"fulfillmentText": response_text})


# Press the green button in the gutter to run the script.
if __name__ == "__main__":
    import uvicorn
    uvicorn.run(app, host="localhost", port=8001)
```

Εικόνα Python (3)

ΠΑΡΑΡΤΗΜΑ C : ΚΩΔΙΚΑΣ ENTITY - SUBJECT

Παρακάτω θα δούμε τον κώδικα *Python* ο οποίος αρχικοποιεί αυτόματα το *Subject Entity*. Να επισημάνουμε ότι για να πραγματοποιηθεί κάτι τέτοιο χρειάζεται και η δημιουργία κλειδιού από το *DialogFlow* ώστε να μπορέσουμε να αλληλεπιδράσουμε με αυτό.

```python
import os
import mysql.connector
from google.cloud import dialogflow_v2 as dialogflow

# Set the credentials from Dialogflow.
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "key_uniBot.json"

# Connect with DB.
db = mysql.connector.connect(
    host="localhost",
    user="root",
    passwd="root",
    database="uniBot"
)

cursor = db.cursor()

# Run the SQL query in order to grab the subjects title.
cursor.execute("SELECT course_title FROM courses")
courses = cursor.fetchall()

# Save the previous results into array.
subjects = [course[0] for course in courses]

# Connect with Dialogflow.
client = dialogflow.EntityTypesClient()

# Set the entity name that you want to update.
entity_name = "projects/unibot-prvi/agent/entityTypes/51ba006f-d57f-4ba4-804c-84c643afae97"

# Target the entity that there is
entity_type = client.get_entity_type(name=entity_name)

# Update the entity.
for subject in subjects:
    entity_type.entities.append(dialogflow.EntityType.Entity(value=subject))

client.update_entity_type(entity_type=entity_type)

print("Entity updated:", entity_type.name)

# Close DB connection.
cursor.close()
db.close()
```

Εικόνα Python Entity Subject

ΠΑΡΑΡΤΗΜΑ D : ΚΩΔΙΚΑΣ ENTITY - INFORMATION

Παρακάτω θα δούμε τον κώδικα *Python* ο οποίος αρχικοποιεί αυτόματα το *Information Entity*. Αρχικά λαμβάνει από ένα αρχείο εισόδου τις λέξεις κλειδιά που αντιστοιχούν σε κάθε πεδίο της βάσης δεδομένων, καθώς μπορεί να περιέχει και συνώνυμες λέξεις χωρισμένες με «/» όπου γίνεται και ο αντίστοιχος έλεγχος. Να επισημάνουμε ξανά ότι για να πραγματοποιηθεί κάτι τέτοιο χρειάζεται και η δημιουργία κλειδιού από το *DialogFlow* ώστε να μπορέσουμε να αλληλεπιδράσουμε με αυτό.

```python
import os
from google.cloud import dialogflow_v2 as dialogflow

# Set the credentials from Dialogflow.
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "key_uniBot.json"

# Connect with Dialogflow.
client = dialogflow.EntityTypesClient()

# Define the entity name.
entity_name = "projects/unibot-prvi/agent/entityTypes/21f36b89-d4c7-4931-9e13-bc1156fe221c"

# Read structured words from the text file.
with open("information.txt", "r") as file:
    structured_words = [line.strip() for line in file]

# Get the existing entity.
entity_type = client.get_entity_type(name=entity_name)

# Process and add synonyms to the entity for each structured word.
for word in structured_words:
    parts = word.split("/")

    # Check if the word contains "/".
    if len(parts) > 1:
        main_word = parts[0]
        synonyms = parts[1:]
    else:
        main_word = word
        synonyms = []

    # Add the main word and synonyms to the entity.
    entity_type.entities.append(dialogflow.EntityType.Entity(value=main_word, synonyms=synonyms))

# Update the existing entity.
client.update_entity_type(entity_type=entity_type)

print("Entity 'information' updated with synonyms from the text file.")
```

Εικόνα Python Entity Information (1)

```python
import os
import mysql.connector
from google.cloud import dialogflow_v2 as dialogflow

# Set the credentials from Dialogflow.
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "key_uniBot.json"

# Connect with DB.
db = mysql.connector.connect(
    host="localhost",
    user="root",
    passwd="root",
    database="uniBot"
)

cursor = db.cursor()

# Run the SQL query in order to grab the subjects title.
cursor.execute("SELECT course_title FROM courses")
courses = cursor.fetchall()

# Save the previous results into array.
subjects = [course[0] for course in courses]

# Connect with Dialogflow.
client = dialogflow.EntityTypesClient()

# Set the entity name that you want to update.
entity_name = "projects/unibot-prvi/agent/entityTypes/51ba006f-d57f-4ba4-804c-84c643afae97"

# Target the entity that there is
entity_type = client.get_entity_type(name=entity_name)

# Update the entity.
for subject in subjects:
    entity_type.entities.append(dialogflow.EntityType.Entity(value=subject))

client.update_entity_type(entity_type=entity_type)

print("Entity updated:", entity_type.name)

# Close DB connection.
cursor.close()
db.close()
```

Εικόνα Python Entity Information (2)