



**University of Management and Technology,
C-II, Johar Town, Lahore - Pakistan**

Final Year Project Report

UniBot: An AI Teaching Assistant for Higher Education based on Large Language Model

Project Advisor: Mr. Mahmood Hussain

Submitted By:

Saad Bin Abid	(F2020332029)
Summaya Ayaz	(F2020332009)
Sadaf Younes	(F2020332041)
Ahmad Shamayl	(F2020332021)

Session

F2020 – F2024

Dedication

My sincere dedication is given to my companions and fellow pupils, who have given more than mere participation in this difficult but worthwhile journey—you have been more than friends.

Final Approval

- **Chairperson of Department**
Dr. Muhammad Shoaib Farooq
Department of Artificial Intelligence,
School of Systems & Technology,
UMT, Lahore.

- **Director (Final Year Projects-AI)**
Basit Sattar
Department of Artificial Intelligence,
School of Systems & Technology,
UMT, Lahore.

- **Supervisor**
Mahmood Hussain
Department of Artificial Intelligence,
School of Systems & Technology,
UMT, Lahore.

- **Co-Supervisor**
Muhammad Nadeem
Department of Artificial Intelligence,
School of Systems & Technology,
UMT, Lahore.

Acknowledgment

First and foremost, I would like to appreciate all the support and contributions that made the existence of Unibot come true. Starting off with our project advisor, for their tireless efforts and immense amount of mentorship and guidance. Their abundance of experience, constructive criticism and assistance has been crucial during the entire course of development. I am greatly in dept of their constant motivation and unwavering support.

Secondly, we will also love to honor the most talented faculty members, whose knowledge and opinions have greatly enhanced the scope and complexity of our project. Their keen desire to provide information and participating in stimulating dialogues has been important in determining the project's course of action. The learning environment has really helped us to enhance the entirety of our project.

Thirdly, I would appreciate the assistance of my fellow group members for the dynamic conversations, cooperation, and valuable sharing of information that have contributed to the overall enrichment of this endeavor. Without a doubt, your varied viewpoints and combined contributions have taken this project to a whole new level. Working with people that are so exceptionally bright and driven has been a delight.

We would especially like to thank the University for providing the expert guidance and resources to ensure the Unibot system went into effect successfully. The structures and services have been essential in turning the project's abstract concepts brought to life.

Finally, but just as importantly, I would like to acknowledge the efforts of my family because their understanding and relentless loyalty during the highs and lows of this educational experience made all of this possible. Their belief in us and the project's capability has helped us stay consistent and focused on the end results. To conclude, this project is the amalgamation of all the efforts and hard work of our team and the unwavering support of our mentors. It would have not been possible without any of you, and I am forever grateful to have been a part of this group of exceptional people

Their belief in the project's potential has kept me driven at all times, and their advice has been a constant source of creativity. In conclusion, this project stands as proof of the teamwork and firm support of a remarkable group, including fellow students, instructors, guides, and relatives. Each of you has been instrumental in making the Unibot project a reality at all, and I am deeply appreciative of the chance to work with a group of such remarkable people.

Project Title: UniBot: An AI Teaching Assistant for Higher Education based on Large Language Model

Objective: Unibot is a web-based LLM chatbot designed for academic use.

Undertaken by: Saad Bin Abid (F2020332029), Summaya Ayaz (F2020332009), Sadaf Younes (F2020332041), Ahmad Shamayl (F2020332021)

Supervised by: Mr. Mahmood Hussain

Starting Date: 20-10-2023

Completion Date: 19-07-2024

Tools Used: Pycharm, Colab, React JS, Python, Fast API, AIML, MongoDB

OS: Windows 10

Documentation report: Microsoft Word

Plagairism Report

University of Management and Technology, Lahore

Similarity Report

Turnitin Originality Report

UniBot: An AI Teaching Assistant for Higher Education based on Large Language Model by
Saad Bin Abid, Summaya Ayaz, Sadaf Younes, and Ahmad Shamayl

From Quick Submit (Quick Submit)

- Processed on 12-Aug-2024 15:24 PKT
- ID: 2430975010
- Word Count: 5690

Similarity Index

10%

Similarity by Source

Internet Sources:

5%

Publications:

2%

Student Papers:

9%

Sources:

1. 4% match (student papers from 01-Dec-2016)
Submitted to UT, Dallas on 2016-12-01
2. 1% match (Internet from 13-Apr-2024)
<https://arxiv.org/html/2404.07503v1>


Checked by


Verified by CLO

Note:

- Sometimes the overall similarity index may be a smaller than the repository percentages combined. This would be due to overlapping text within the repositories.

Declaration Form

I have carefully examined the documentation of the Final Year Project titled “*UniBot- An Assistant tool based on LLM*”; and I endorse that this documentation complies with the standards of an undergraduate level Final Year Project report.

The document has been checked for plagiarism through Turnitin software available in UMT Library. The similarities of the document are within acceptable range.

Moreover, the accompanying CDs contain PDF of the documentation, as well as the source code and binaries with user manual and installation guide.

FYP Advisor Name: Sir Mahmood Hussain

Signature: _____

Date: _____

Abstract

Our final year project presents "Unibot," a web-based Large Language Model (LLM) chatbot designed especially for academic use within our learning institution, in response to the constantly shifting world of educational technology. The project aims to meet the requirement for a smart, optimized system that helps teachers and students with many aspects of the learning process.

Creative proposals are required due to the rapid progress of technology in education. As a reply to this popular demand, Unibot was developed to utilize the strength of massive language models that have been adjusted to fit our university's curriculum. With an emphasis on helping teachers create assignments, create quizzes, and manage tasks while giving students access to immediate, useful data, Unibot is positioned to completely transform the way that education is delivered.

The main goal of Unibot is to create an advanced, yet approachable chatbot that will improve overall efficiency and interaction in our university's classroom. Unibot uses modern LLM models—possibly GPT 3.5 Turbo which is calibrated to the curriculum to optimize activities for teachers and give students timely and accurate information. Developing a smooth, engaging framework that improves education and instruction is the aim.

There are three primary parts to the project. React JS was used in the front-end design to create an aesthetically pleasing and user-friendly experience. A refined LLM model is integrated with data vectors, API connectivity, transformers, and embeddings in the backend. Important phases in the development process include model training and deployment. A MongoDB database is used in the last part to effectively store user prompts and model-generated replies.

After we've finished our work, Unibot should give teachers a helpful resource for organizing their workload and make it easier for students to obtain personalized information. By using powerful language models that have been customized to our educational program, we hope to produce responses that are appropriate for the given context and enhance the learning experience for each student.

The future of educational technology will be greatly impacted by the effective roll-out of Unibot. Through the integration of state-of-the-art language models and user-friendly design, Unibot shows promise for improving learning assignment completion, fostering a lively educational setting, and improving engagement. The project's findings might guide future research into similar uses in different areas of education.

EDITING CHART

Edition	Author(s)	Description	Completion
First	Sadaf Younes, Summaya Ayaz, Ahmad Shamayl, Saad Bin Abid	First version made for the first half of our project and evaluator comments	12 January,2024
Second	Sadaf Younes, Summaya Ayaz, Ahmad Shamayl, Saad Bin Abid	Following edition in compliance to the above evaluator comments and set for final checking.	18 February,2024
Final	Sadaf Younes, Summaya Ayaz, Ahmad Shamayl, Saad Bin Abid	The final edition.	29 June, 2024

CONTENTS

CONTENTS	1
ABBREVIATION WITH DESCRIPTIONS	3
LIST OF FIGURES	4
LIST OF TABLES	5
1. INTRODUCTION	6
1.1 MOTIVATIONS.....	6
1.2 PROJECT OUTLINE.....	6
1.3 PROBLEM STATEMENT	7
1.4 OBJECTIVES	7
2. DOMAIN ANALYSIS.....	8
2.1 CUSTOMER.....	8
2.2 STAKEHOLDERS	8
2.3 IMPACTED GROUPS WITH SOCIOECONOMIC IMPACT	9
2.4 DEPENDENCIES/ EXTERNAL SYSTEMS.....	9
2.5 REFERENCE DOCUMENTS.....	10
2.5.1 <i>Similar Projects</i>	10
2.5.2 <i>Feature Comparison</i>	11
3. REQUIREMENTS ANALYSIS	13
3.1 REQUIREMENTS	13
3.2 LIST OF ACTORS.....	14
3.3 LIST OF USE CASES	14
3.4 SYSTEM USE CASE DIAGRAM.....	14
3.5 EXTENDED USE CASES.....	15
3.6 USER INTERFACES (MOCK SCREENS)	17
4. DATA FLOW DIAGRAM (OPTIONAL)	20
4.1 DATA FLOW DIAGRAM (LEVEL 2).....	20
5. SYSTEM ARCHITECTURE DIAGRAM	21
5.1 CLASS DIAGRAM.....	22
5.2 SEQUENCE DIAGRAM	23
5.3 COLLABORATION DIAGRAM	24
5.4 ERD.....	24
5.5 DATA DICTIONARY	25
6. IMPLEMENTATION DETAILS.....	26
6.1 DEVELOPMENT SETUP.....	26
6.2 DEPLOYMENT SETUP	26
6.3 ALGORITHMS	27
6.3.1 <i>Assumptions</i>	29
6.3.2 <i>System constraints</i>	30
6.3.3 <i>Restrictions</i>	30
6.3.4 <i>Limitations</i>	30
7. TESTING	32
7.1 EXTENDED TEST CASES	32
7.2 DECISION TABLE.....	34
7.2.1 <i>Code snippet</i>	34
□ <i>Decision coverage table</i>	34

7.3	TRACEABILITY MATRIX	34
7.3.1	<i>RID vs UCID (requirements vs use cases)</i>	36
7.3.2	<i>Prototypes (RID vs PID)</i>	37
7.3.3	<i>Test Cases (RID vs TID)</i>	37
7.3.4	<i>Coverage (UCID vs TID)</i>	38
8.	RESULTS/OUTPUT	39
8.1	%COMPLETION	39
8.2	%ACCURACY.....	39
8.3	%CORRECTNESS.....	39
9.	CONCLUSION	40
10.	FUTURE WORK	41
11.	BIBLIOGRAPHY	42
12.	APPENDIX	43
12.1	GLOSSARY OF TERMS	43

Abbreviation with Descriptions

Abbreviation	Description
LLM	Large Language Model
IT	Information Technology
API	Application Programming Interface
NLP	Natural Language Processing
AI	Artificial Intelligence
GPT	Generative Pre-Trained Transformer

Table 1: Table of abbreviation and descriptions

List of Figures

Figure 1: Use case diagram with explanation.....	14
Figure 2: Data Flow Diagram.....	20
Figure 3: System Architecture.....	21
Figure 4: Class Diagram.....	22
Figure 5: Sequence Diagram	23
Figure 6: Collaboration Diagram.....	24
Figure 7: Entity Relationship Diagram (ERD).....	24

List of Tables

Table 1: Table of abbreviation and descriptions	3
Table 2: List of stakeholders	9
Table 3: Feature Comparison	11
Table 4: Requirements.....	13
Table 5: Data Dictionary	25
Table 6: Decision coverage table	34
Table 7: RID vs UCID.....	36
Table 8: Prototypes (RID vs PID)	37
Table 9: Test Cases (RID vs TID).....	37
Table 10: Coverage (UCID vs TID)	38

1. INTRODUCTION

1.1 Motivations

The main reason to pursue this development of the Unibot project was made with a dedication to developing technological resources for education and tackling modern-day problems in learning as well as instruction. The initiative uses curriculum-tuning a chatbot to offer individualized instruction at the university. By utilizing the latest LLM (Large Language Model) for contextually appropriate responses, which is GPT 3.5 Turbo the objective is to simplify managing workloads for instructors. To one day have an impact on upcoming advancements in the field of education, the project also aims to improve student-teacher interaction and teamwork. The catalyst is a sincere desire to improve education and make it more engaging, efficient, and suited to our institution's specific requirements.

1.2 Project Outline

Problem Statement: The Unibot project creates a web-based Large Language Model (LLM) chatbot that is made for any university to deal with the changing difficulties in the educational scene. This project is based on the present demand for effective task management, tailored lessons, and communication.

Customer: Within the bounds of our institution, teachers and pupils are our main stakeholders. Teachers are looking for a tool that can help them handle tasks more efficiently, while students want quick access to useful data. Unibot provides a platform that is centered around the demands of those who use it.

Goals: Improving the quality of teaching and learning at our university is the main objective of the work being done. Among the specific goals is the development of an easy-to-use chatbot that helps teachers create exams and tests while giving students timely, personalized information.

System Functions: Front-end Interface: Create a user-friendly UI for educators to create quizzes and assign assignments by utilizing React JS.

Backend Platform: To provide specifically appropriate answers for client requests, develop a backend platform that incorporates an LLM model.

Database management: To ensure the successful handling of data, create a MongoDB database to hold user prompts and model-generated actions.

System Attributes: Personalization: By adapting the LLM models to a university's curriculum, the system seeks to give students a customized educational experience.

Adaptability: The system must be able to stay unaltered in the face of growing or changing curriculum and teaching demands.

User-Centric Design: The primary concern of this project is the availability of a user-friendly interface to both the teachers and students.

1.3 Problem Statement

The reason for the Unibot project to be a reality was the ongoing problems in the education sector. Instructors are presented with great difficulty when it comes to managing their workload, mainly when the workload involves the time-consuming process of creating endless assignments and quizzes. It also impacts the students as they must wait to get timely and relevant responses from their instructors. The lack of communication and limited access to educational resources challenged us to make an educational tool. Unibot, a web-based large language model educational chatbot is our approach to the ongoing problem. The purpose of our project is to improve the current educational complications for teachers by allowing them to seamlessly integrate with a university's curriculum and also provide students with appropriate responses. Unibot aspires to revolutionize the field of higher education to develop a more fine-tuned approach towards learning for an institution.

1.4 Objectives

Efficiency of Assignments: Teachers can create assignments and quizzes with ease and in an efficient manner saving time on repetitive tasks.

Personalized Responses: We have enhanced Unibot's Large Language Models (LLMs) to help students and teachers by providing them with customized responses with great accuracy in reduced time.

Efficient Data Management: We have employed MongoDB database to store the Chat Sessions, Prompts, Responses, and Embedding vectors. This helps us to efficiently manage data for uninterrupted data flow.

User-Friendly Interface: Give the most importance to user-oriented design concepts in the front-end interface to create a simple platform that improves access for educators as well as students alike.

Adaptability and Reliability: To guarantee relevance for years and efficacy, we will build Unibot with the capacity to adjust to changes in the course material and to grow or modify as needed in future updates.

Enhanced Academic Productivity: The project's goal is to produce an improved Unibot system that maximizes interaction, organization of tasks, as well as educational opportunities. This will improve academic output overall and make our university's learning environment more effective and captivating.

2. DOMAIN ANALYSIS

2.1 Customer

Educator Representative:

Role: The Educator Representative is a person who is responsible for the educational faculty's requirements in implementing Unibot.

Duty: They provide educators with an understanding of how Unibot might improve the quality of task management and assignment creation. They make sure Unibot meets the needs of educators on university premises.

Student Representative:

Role: The Student Representative represents the needs of the student council in receiving Unibot as a potential customer.

Duty: Their duties include confirming that Unibot complies with the information and communication of the entire student body. Their observations help to improve Unibot's functionality and user experience.

University Administration:

Role: The University Administration chooses to select cutting-edge products such as Unibot for their educational institute.

Duty: Their duties include checking if Unibot aligns with the objectives of the university, measuring its compatibility with the recent processes, and examining whether it has an impact on the efficiency of previous systems.

2.2 Stakeholders

Stakeholder	Role in System
Instructors	Instructors, heads of departments, and staff members who oversee assigning and supervising work.
Students	End users engage with tasks, examinations, and coursework made by Unibot in search of current and relevant information.
University Administration	Administrators in charge of the university's entire goals, such as the vice-chancellor, the chief information officer, or other appropriate administrative positions.
IT Department	The Department of Information Technology is in charge of Unibot's technical elements, preservation, and incorporation.
Unibot Team	The programmers, designers, and team leaders are in charge of Unibot's development, execution, and management.

Future Partners	People or teams who will work together on Unibot's upcoming improvements.
-----------------	---

Table 2: List of stakeholders

2.3 Impacted Groups having socioeconomic influence

- **Local Agencies Offering Educational Materials:**
Local agencies have to face the rising demand for cutting-edge technologies providing the same services as they are for efficient information transmission.
Freelance Developers:
Chances for independent contractors to support cutting-edge educational technology.
- **Startups in Educational Technology:**
Influencing companies in the field and establishing an example for using modern technology in education.
- **Platforms for Online Education:**
Raising expectations and norms in the context of digital education.
- **Student Tutoring Providers:**
A shift in service offerings away from the distribution of basic information and towards customized counseling.

2.4 Dependencies/ External Systems

- **Large Language Model:**
The ability of Unibot to analyze natural language and produce contextually relevant responses depends on the successful incorporation and optimization of GPT-3.5 Turbo.
MongoDB Database:
Used in order to store user queries, model-generated solutions, and appropriate data for optimizing the LLM models, Unibot depends upon the MongoDB database. The effectiveness of data handling is crucial to Unibot's ongoing development.
- **Front-end framework React JS:**
Creating an easy-to-use front-end interface with React JS is essential to giving teachers and students a platform to easily create activities, tests, and assignments.
- **The University's IT Infrastructure:**
To guarantee seamless integration and operation, Unibot's deployment is reliant on the university's current IT framework, which includes network connections, servers, and safety protocols.

- External Resources:

Possible cooperation with third-party organizations such as Futurizm which is offering extra assistance, resources, or knowledge, boosting Unibot's total performance.

2.5 Reference Documents

The documents that we consulted during our analysis phase are as follows:

- ChatGPT for good? On opportunities and challenges of large language models for education
- E. Kasneci, K. Sessler, S. Küchemann, and M. Bannert, "ChatGPT for good? On opportunities and challenges of large language models for education," *Learn. Individ. Differ.*, vol. 103, Art. no. 102274, 2023, doi: 10.1016/j.lindif.2023.102274.
- Large language models encode clinical knowledge
- K. Singhal, S. Azizi, T. Tu, et al., "Large language models encode clinical knowledge," *Nature*, vol. 620, pp. 172–180, 2023, doi: 10.1038/s41586-023-06291-2.

2.5.1 Similar Projects

During the creation of Unibot, we researched multiple systems that had similarity to our project as follows:

1. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models

L. Yu and W. Jiang, "MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models," arXiv: 2309.12284, 2023. [Online]. Available: <https://arxiv.org/abs/2309.12284>.

2. Med-HALT: Medical Domain Hallucination Test for Large Language Models

A. Pal and L. K. Umapathi, "Med-HALT: Medical Domain Hallucination Test for Large Language Models," arXiv: 2307.15343, 2023. [Online]. Available: <https://arxiv.org/abs/2307.15343>.

2.5.2 Feature Comparison

Table 3: Feature Comparison

Serial No.	Model	Exact Launch date	Parameters	Efficiency	Limitation	Size	How it works
1.	LaMDA	May-2022	137B	Conversational AI, advanced	Can sometimes generate inaccurate or misleading information	1.56TB	Designed for dialogue, it generates natural conversational responses using advanced language modeling.
2.	PaLM	April-2022	540B	Large-scale, performant	Can sometimes generate inaccurate or misleading information	614GB	Processes large-scale data for various NLP tasks with high performance and accuracy.
3.	PaLM 2	April-2022.	340 billion	Multilingual, versatile, robust	Limited Knowledge: Contextual Understanding Lack of Real-Time Information:	1.56 TB	Excels in multilingual understanding and reasoning by leveraging extensive training data.
4.	GPT-3.5	March-2022	175B	Powerful, flexible, general	Can sometimes generate inaccurate or misleading information	1.35TB	Utilizes a large-scale transformer model to generate contextually appropriate text.

5.	GPT-3.5(turbo)	March-2023	154b	Faster, efficient, scalable	No Critical Thinking: Legal and Ethical Concerns Long-Term Contextual Understanding	660 billion parameters.	An optimized version of GPT-3.5 for faster and more efficient text generation.
6.	GPT-4	March-2023	1.76 trillion.	Advanced, comprehensive, superior	Lack of Real-World Understanding Sensitivity to Input Phrasing Sensitivity	175B-280B	Combines extensive data with advanced algorithms to produce highly accurate text.
8.	Falcon 40B	May-2023	40B	Large, diverse, efficient	Limited access	290GB	A large-scale language model that provides diverse text generation capabilities.
9.	Llama	Feb-2023	65.2B	Compact, optimized, foundational	Can sometimes generate inaccurate or misleading information	7B and 70B	A foundational language model focused on efficiency and optimization for various NLP tasks.
10.	Llama2	July-2023	70B	Improved, refined, accessible	Can sometimes generate inaccurate or misleading information	7B, 13B and 70B	An improved version of Llama with better performance and accessibility for broader applications.

3. REQUIREMENTS ANALYSIS

3.1 Requirements

Table 4: Requirements

RID	Definition	Type	Attribute	Description
R1.1	Individual Authentication	Functional Requirements	Security	By using user authentication, you can guarantee safe system access.
R1.2	Upload Curriculum Document	Functional Requirements	Efficiency	Give teachers the ability to upload documents to construct assignments from them.
R1.3	Assignment Creation	Functional Requirements	Efficiency	Give teachers the ability to easily construct assignments.
R1.4	Quiz Generation	Functional Requirements	Efficiency	Make it easy for teachers to create quizzes.
R1.5	Information Retrieval	Functional Requirements	Accessibility	Let pupils access to the necessary data conveniently.
R1.6	System Performance	Non-Functional Requirements	Performance	Makes sure that the system operates smoothly and quickly.
R1.7	Data Encryption	Non-Functional Requirements	Security	Use encryption to ensure safe storage of data.
R1.8	Database Structure	Data Requirements	Storage	Establish the framework for keeping data, answers, and prompts.
R1.9	Programming Language	Non-Functional Requirements	Technology	For development, pick a programming language that works well.
R1.10	API Connectivity	External Interface Requirements	Integration	Create interfaces to allow for easy connectivity with other systems
R1.11	User Interface Design	Non-Functional Requirements	Design	Make a user interface that is beautiful and simple to use.

3.2 List of Actors

Instructor:

- Function: Assigns and oversees academic assignments.
- Use Cases: Assignment Creation, Quiz Generation, and Task Management.

Student:

- Function: Looks for scholarly material and offers criticism.
- Use Cases: Information retrieval.

System:

- Function: embodies the non-human component of the system.
- Use Cases: User Authentication, API Connectivity, Language Selection, Data Encryption, and Database Structure.

3.3 List of use cases

Create Assignment; Instructor uses Unibot to create assignments, specifying details such as task description, due dates, and grading criteria.

Generate Quiz; Instructors use Unibot to create quizzes with easily defined questions, answers, and alternatives.

Log In; Instructors provide Unibot with their credentials in order to access its services.

Manage Tasks; Instructor utilizes Unibot's task management system to organize and track assignments, quizzes, and other academic tasks.

Provide Assistance; Allows instructor to easily retrieve relevant information by querying the Unibot for academic resources, guidelines, or course details.

3.4 System use case diagram

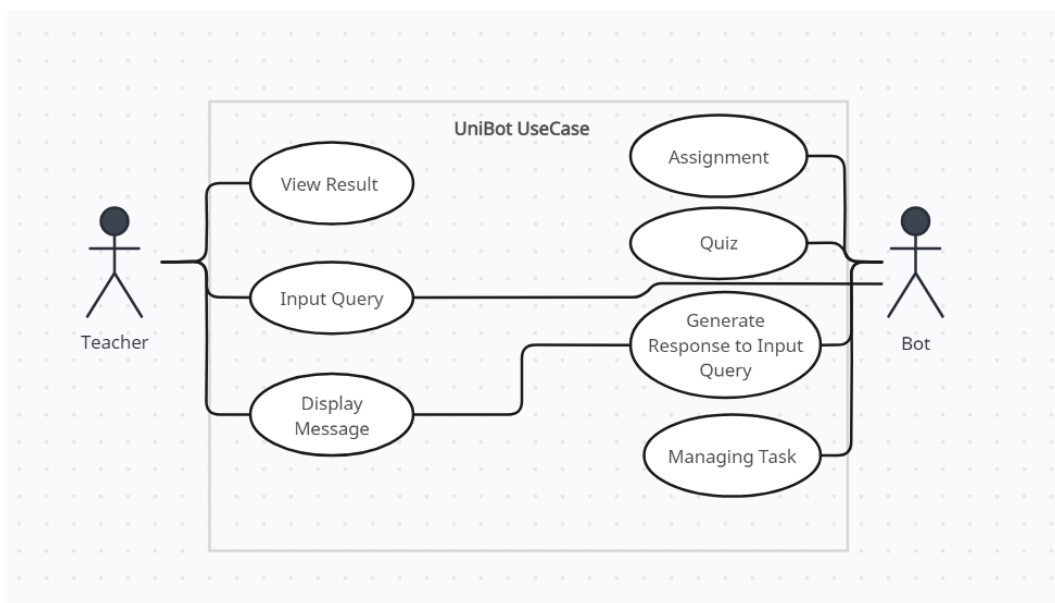


Figure 1: Use case diagram with explanation

3.5 Extended use cases

Use Case ID:	UC-1.1.1		
UseCase Name:	Create Assignment		
Created By:	Summaya Ayaz	Last Updated By:	Summaya Ayaz
Date Created:	11-01-2024	Last Revision Date:	11-01-2024
Actors:	Primary Actor: Instructor Secondary Actor: Unibot (as a tool)		
Description:	In this use case, teachers utilize Unibot to generate homework, assignments and quizzes for their students. It makes the process easier by allowing teachers to provide specifics like the description of the work and upload documents from which we need to create the assignment.		
Trigger:	When a instructor generates a query, that's the use case trigger.		
Preconditions:	Instructor is now signed into the Unibot system. An instructor possesses the necessary rights to use Unibot.		
Post conditions:	The Unibot system was able to accomplish the task. The database contains the document details.		
Normal Flow:	<ol style="list-style-type: none"> 1. Instructor accesses the Unibot system by the LogIn/SignUp module. 2. The instructor goes to the search bar. 3. The system asks the instructor to upload a document. 4. Instructor inputs the necessary information. 5. The data is stored in the system. 6. The query is saved by the system in the database. 7. A message of confirmation is sent to the instructor. 8. The use case comes to an end. 		
Alternative Flows: [Alternative Flow 1 – Not in Network]	Alternative Flow 1 – Invalid Information: If the system finds inaccurate information in step 5. The instructor is prompted by the system to fix the invalid fields. The instructor updates the data. Make use of the step 6 case continues.		
Exceptions:	Exception 1 – Unauthorized Access: In step 1, if the instructor is not logged into the Unibot system. An error message is shown by the system. An instructor receives a login prompt. Make use of the step 1 case resumes.		
Includes:	None		
Frequency of Use:	The number of tasks teachers must make determines how frequently this use case occurs; it may occur once a day or several times a day.		
Special Requirements:	The system should provide a user-friendly interface for the user.		
Assumptions:	The Unibot system is recognizable to instructor. To generate homework, assignments and quizzes instructor is authorized to do so.		
Notes and Issues:	One must ascertain the maximum length for descriptions.		

Use Case ID:	UC-1.1.2		
UseCase Name:	Generate Quiz		
Created By:	Summaya Ayaz	Last Updated By:	Summaya Ayaz
Date Created:	11-01-2024	Last Revision Date:	11-01-2024

Actors:	Primary Actor: Instructor Secondary Actor: Unibot (as a tool)
Description:	In this use case, instructors use Unibot to create quizzes for their students. The instructor provides quiz details and relevant documents, and Unibot generates the quiz.
Trigger:	When an instructor initiates the quiz generation process.
Preconditions:	Instructor is logged into the Unibot system. Instructor has the necessary permissions to create quizzes.
Post conditions:	The quiz is generated and saved in the system. The database contains the quiz details.
Normal Flow:	<ol style="list-style-type: none"> 1. Instructor accesses the Unibot system by the LogIn/SignUp module. 2. The instructor goes to the search bar. 3. System prompts the instructor to upload relevant documents. 4. Instructor provides necessary details for the quiz. 5. Data is validated and stored in the system. 6. The quiz is saved in the database. 7. A confirmation message is sent to the instructor. 8. This use case reaches its ends.
Alternative Flows: [Alternative Flow 1 – Not in Network]	<p>If incorrect information is provided in step 5. The system prompts the instructor to correct the invalid fields. Instructor updates the information. Use case continues from step 6.</p>
Exceptions:	<p>Exception 1 – Unauthorized Access: In step 1, if the instructor is not logged into the system. An error message is displayed. Instructor receives a login prompt. Use case resumes from the first step.</p>
Includes:	Null
Repitition of Use:	Is relative to the exact amount of quizzes the instructor needs to create; could be daily or multiple times a day.
Special Requirements:	The system should provide a user-friendly interface for quiz creation.
Assumptions:	Instructor is familiar with the Unibot system and authorized to create quizzes.
Notes and Issues:	Verify the maximum length for quiz descriptions.

Use Case ID:	UC-1.1.3		
UseCase Name:	Information Retrieval		
Created By:	Summaya Ayaz	Last Updated By:	Summaya Ayaz
Date Created:	11-01-2024	Last Revision Date:	11-01-2024
Actors:	Primary Actor: Instructor Secondary Actor: Unibot (as a tool)		
Description:	In this use case, students use Unibot to retrieve information related to their queries. Students can search for specific information and receive relevant responses.		
Trigger:	When a student submits a query to the system.		
Preconditions:	Student is logged into the Unibot system. Student has the necessary permissions to access information.		
Post conditions:	The relevant information is retrieved and displayed to the student. The query and response details are stored in the database.		

Normal Flow:	<ol style="list-style-type: none"> 1. Pupil accesses the Unibot system by logging in. 2. Pupil navigates the search module. 3. System prompts the pupil for a query. 4. Student submits the query. 5. The query is processed and relevant information is retrieved. 6. The information is displayed to the student. 7. The query and response are saved in the database. 8. This use case comes to a halt.
Alternative Flows: [Alternative Flow 1 – Not in Network]	<p>Alternative Flow 1 – No Results Found:</p> <p>If no relevant information is found in step 5.</p> <p>The system displays a message indicating no results were found.</p> <p>The student is prompted to refine their query.</p> <p>Use case continues from step 3.</p>
Exceptions:	<p>Exception 1 – Unauthorized Access:</p> <p>In step 1, if the student is not logged into the system.</p> <p>An error message is displayed.</p> <p>Student receives a login prompt.</p> <p>Use case resumes from the first step.</p>
Includes:	Null
Frequency of Use:	Is relative to the exact amount of queries the student needs to make; could be daily or multiple times a day.
Special Requirements:	The system should provide a user-friendly interface for information retrieval.
Assumptions:	Student is familiar with the Unibot system and authorized to retrieve information.
Notes and Issues:	Ensure the system can handle a high volume of queries efficiently.

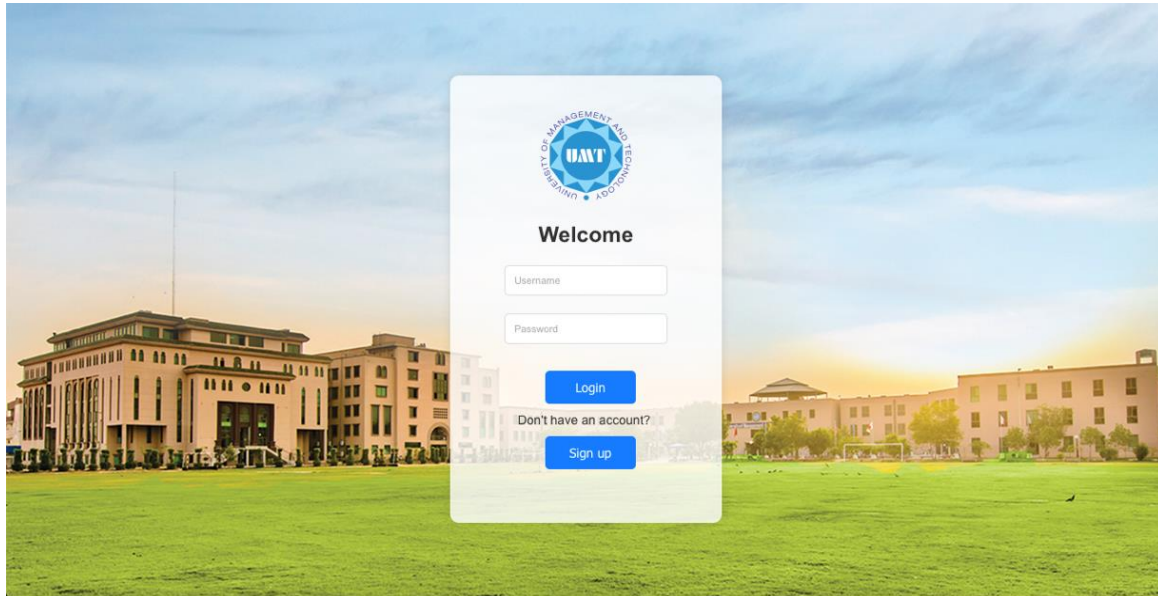
3.6 User interfaces (mock screens)

Prototype1: (P1) Interface of Web based UniBot

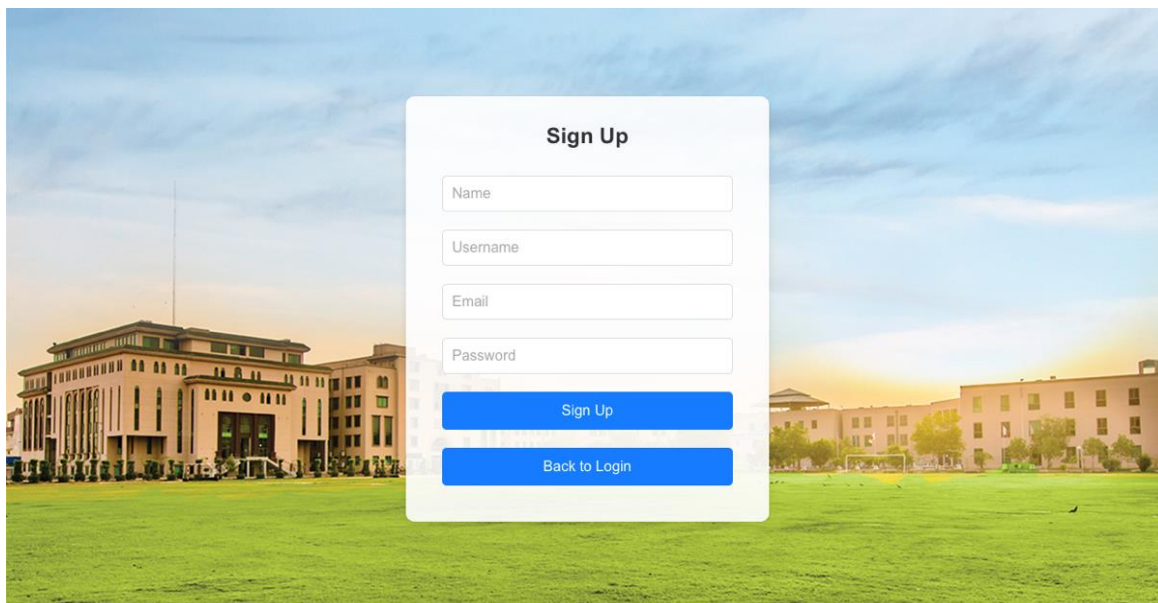


Prototype2: (P2) Interface of Web based UniBot


Login Webpage:



SignUp Webpage:



Chat Interface Webpage:




Chat History

"Generation of Multiple-Choice Questions (MCQs)" 🗑️

The topic could be "Creating quizzes through document generation". 🗑️

New Chat

Conversation Logout



Type your message here...

Send
Choose .txt File
Upload Document

4. DATA FLOW DIAGRAM

4.1 Data Flow Diagram (Level 2)

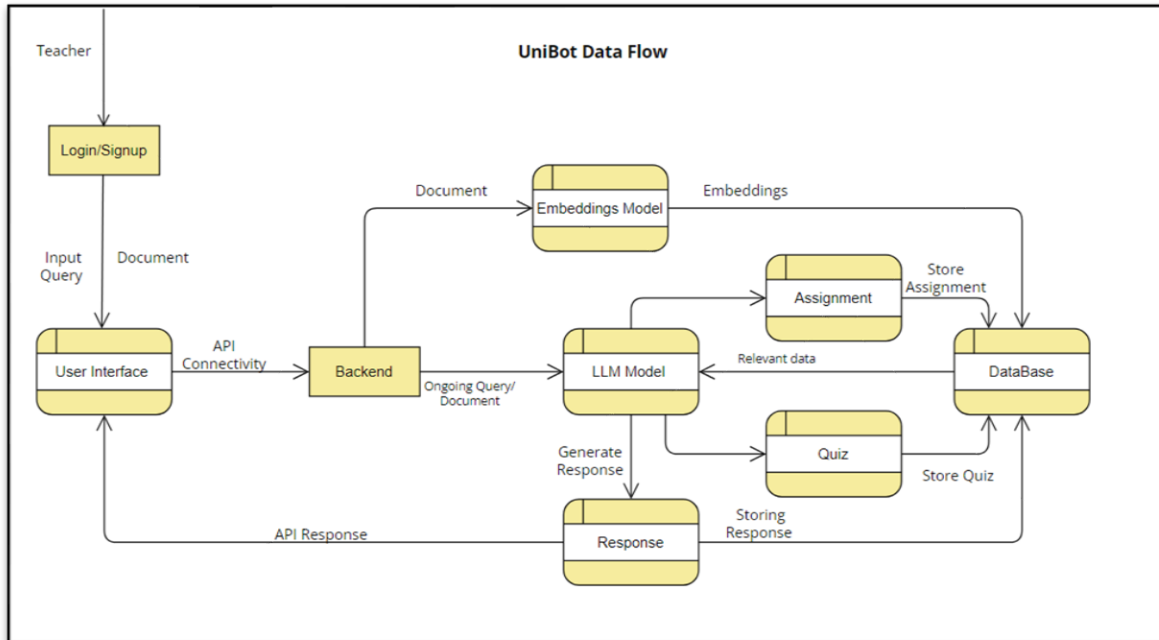


Figure 2: Data Flow Diagram

- Login/Signup: Teacher logs in or signs up.
- User Input: The teacher inputs the query/document through the User Interface.
- Backend Processing: User Interface sends input to Backend via API Connectivity.
- LLM Model: Backend forwards input to LLM Model for processing.
- Embeddings Model: Provides embeddings to the LLM Model in case of need.
- Content Generation: LLM Model generates responses, assignments, or quizzes.
- Database Storage: Assignments and quizzes are stored in the Database by respective modules.
- Response Delivery: The generated response is then sent back to the User Interface via the backend and presented to the teacher.

5. SYSTEM ARCHITECTURE DIAGRAM

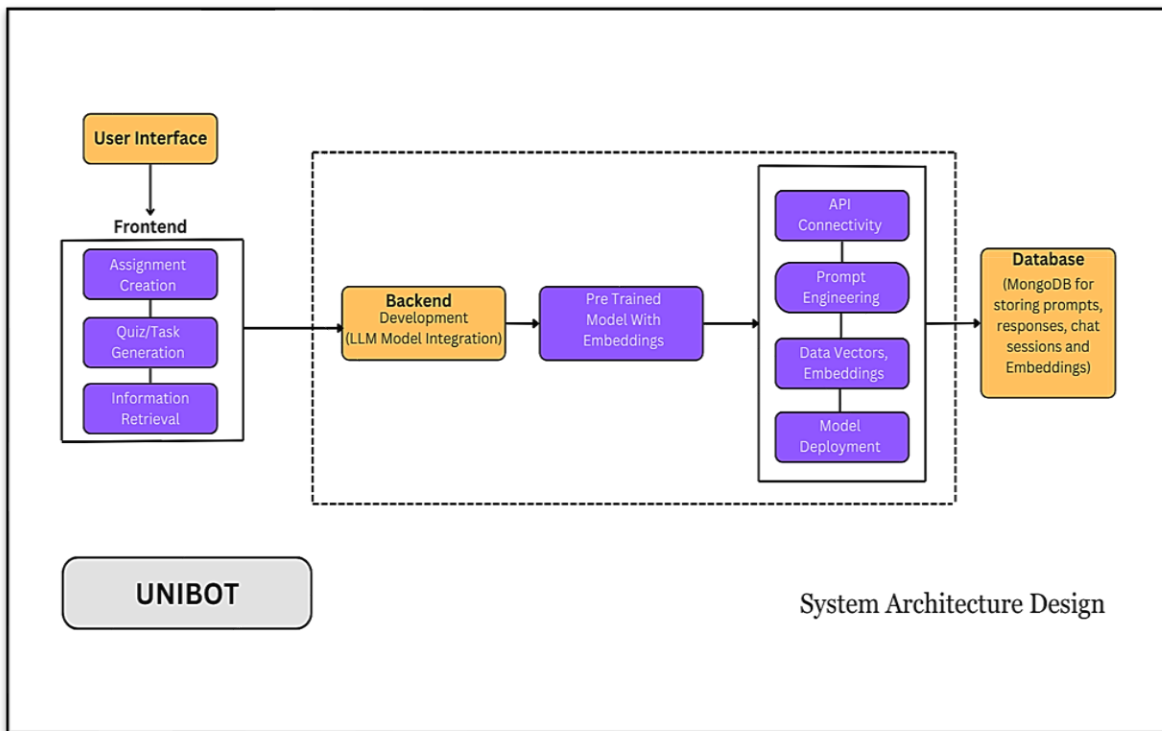


Figure 3: System Architecture

5.1 Class Diagram

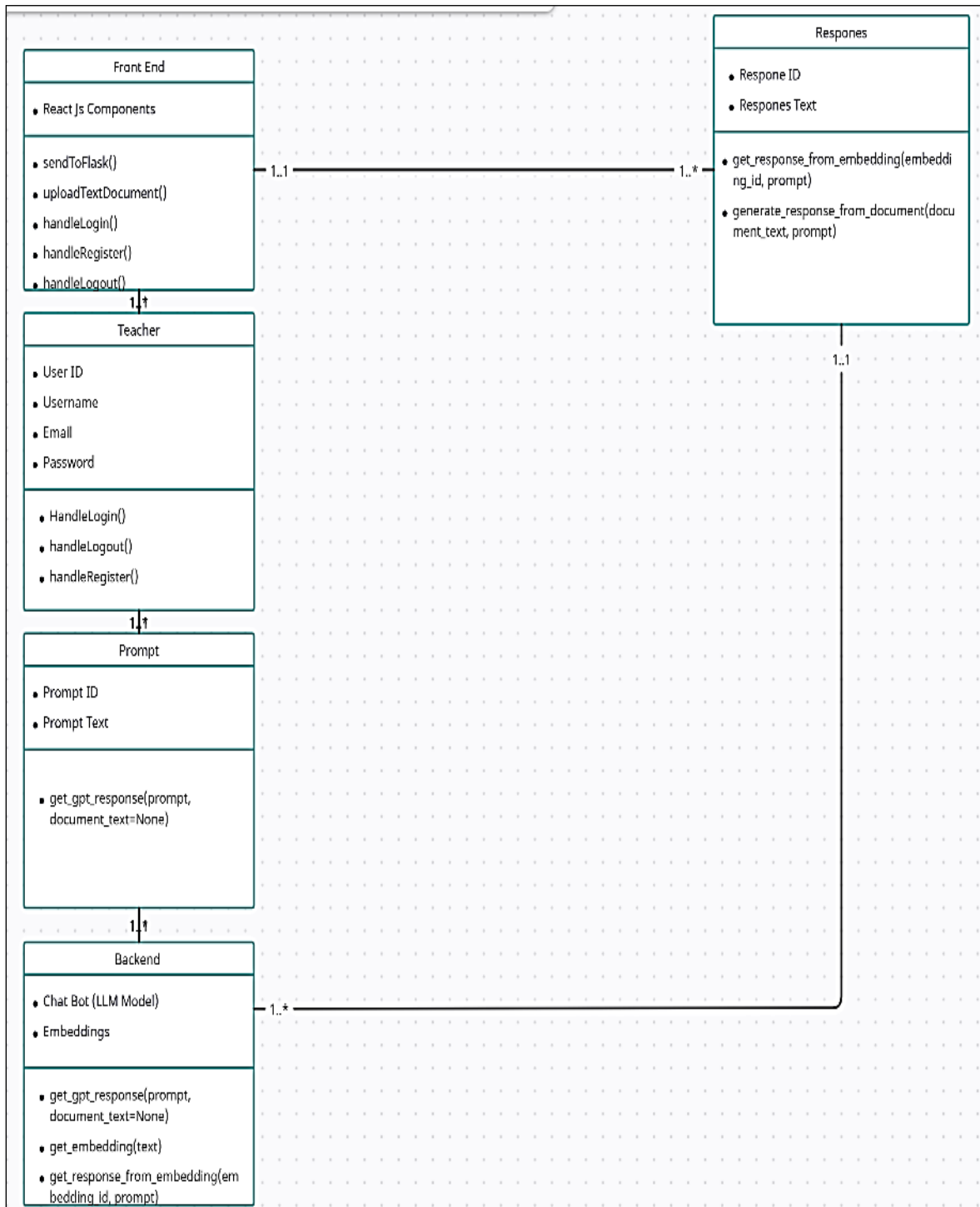


Figure 4: Class Diagram

5.2 Sequence Diagram

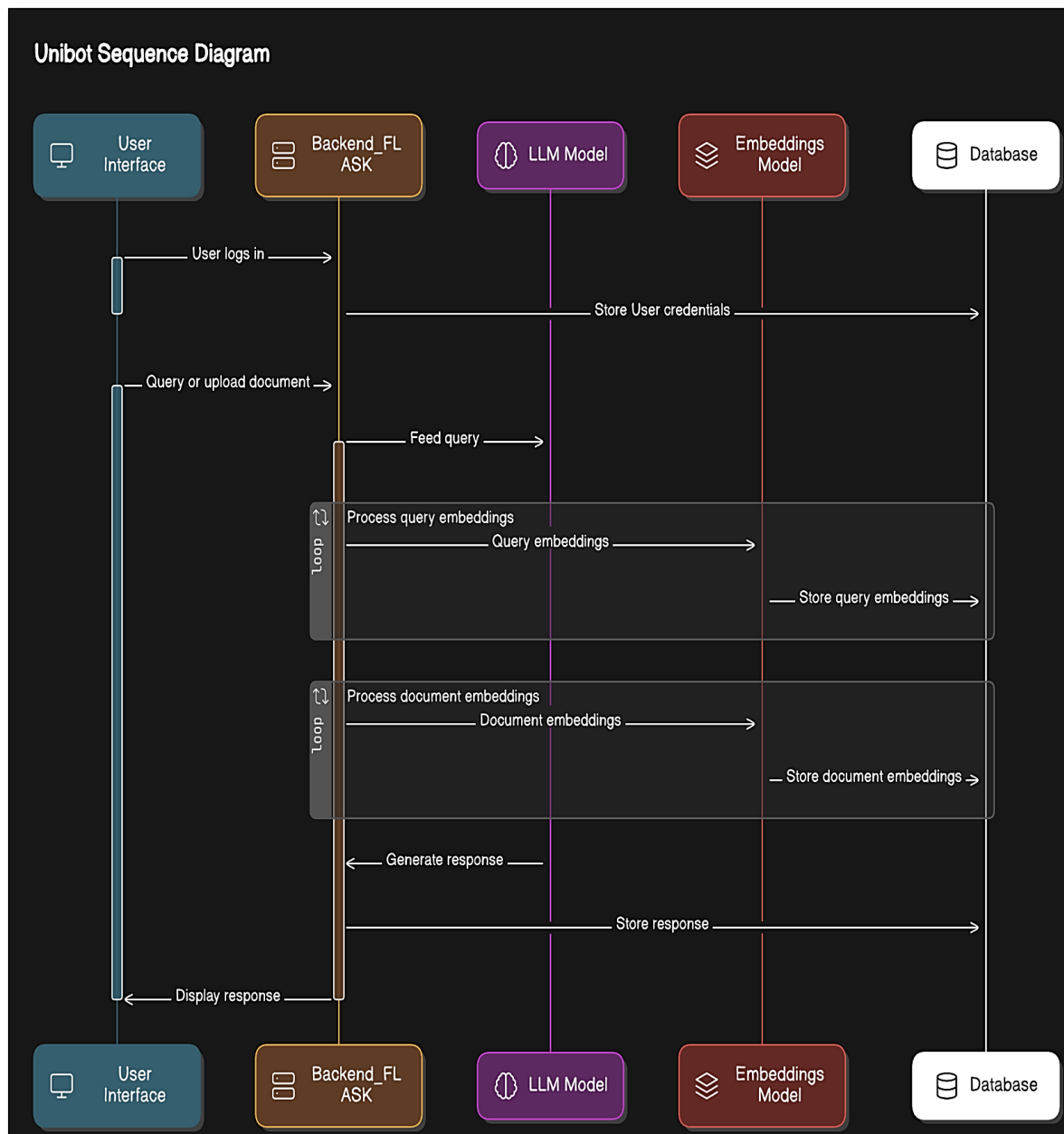


Figure 5: Sequence Diagram

It sends all the queries to the model and displays the response; it also acts as a connection that sends all the data from the front end and back to the backend for effective query resolution.

5.3 Collaboration Diagram

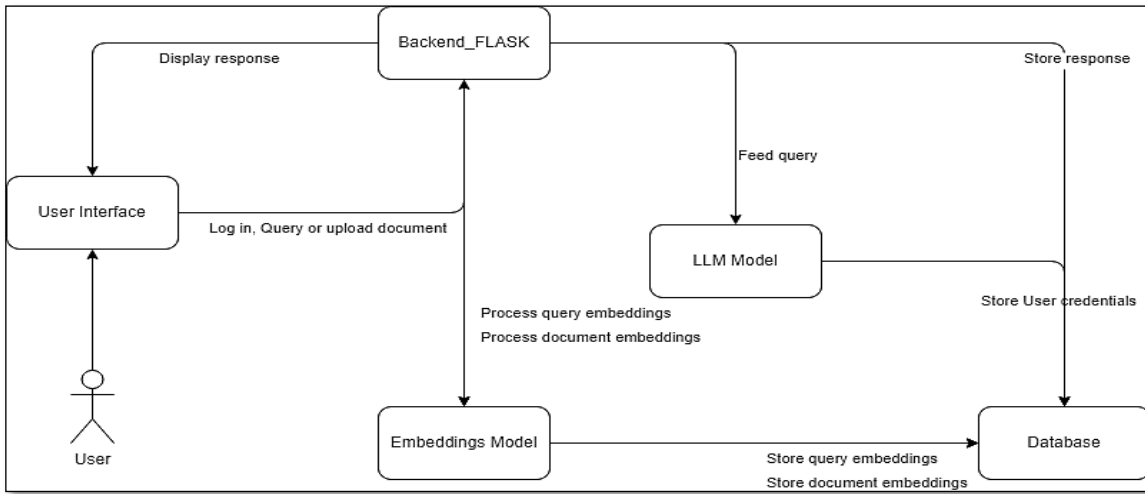


Figure 6: Collaboration Diagram

5.4 ERD

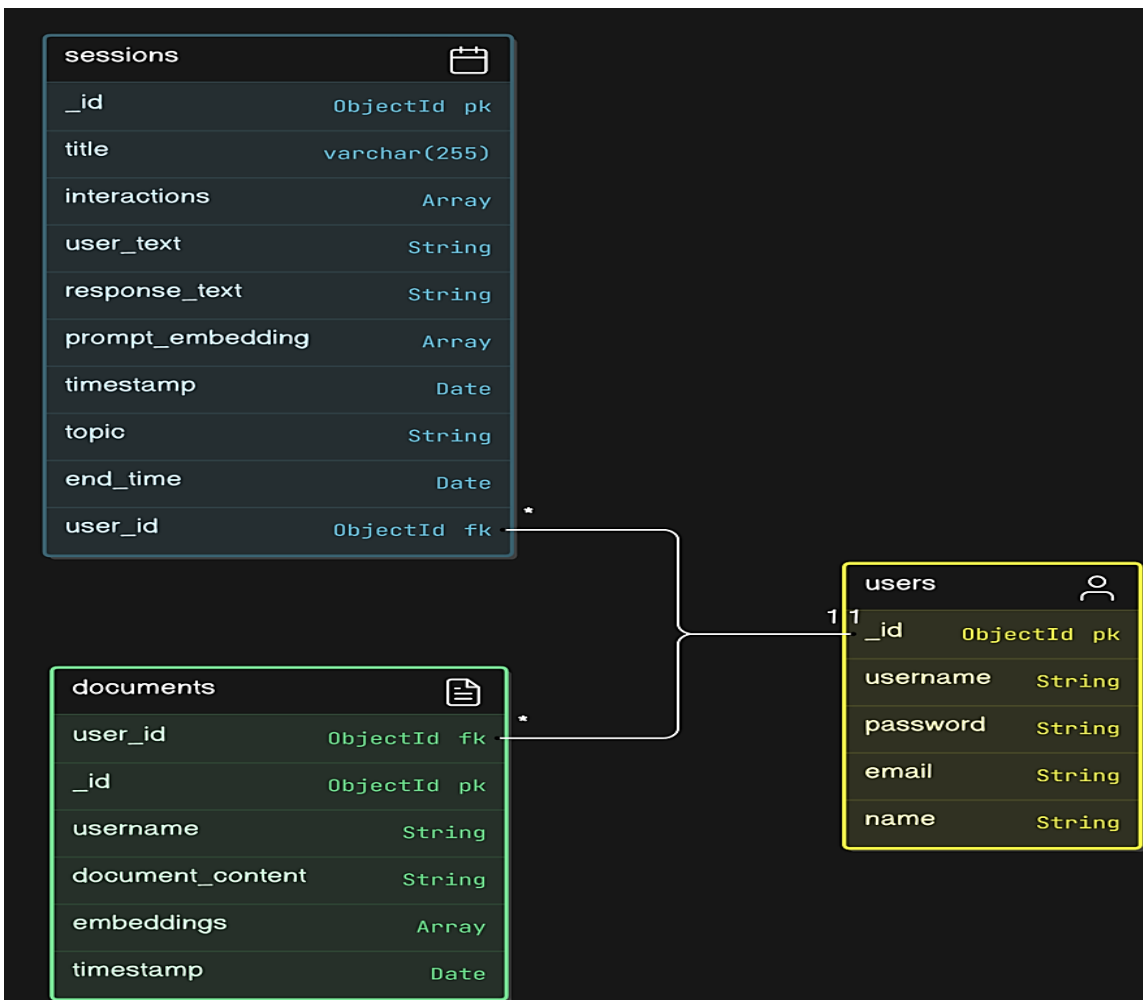


Figure 7: Entity Relationship Diagram (ERD)

5.5 Data Dictionary

Element	Category	Validation	Mandatory	Remarks
ID	Integer	10	Yes	Unique identifier of chat
Prompt	varchar		Yes	Query written by the instructor
Response	varchar		Yes	Response generated by Unibot.
Time Stamp	Date		No	Time when response is generated.
User ID	Integer	10	Yes	Unique identifier for Instructor
Display Name	varchar		Yes	Name of the instructor.

Table 5: Data Dictionary

6. IMPLEMENTATION DESCRIPTION

6.1 Development Setup

Resources and Technologies

Frontend Development

React JS

Role: Used to build a dynamic and responsive user interface. It handles the rendering of components such as the search bar, history, and upload document buttons, providing an interactive experience for users.

HTML/CSS

Role: HTML is used for structuring the web pages, while CSS is used for styling and layout, ensuring the frontend is visually appealing and user-friendly.

Backend Development

Node.JS

Role: Provides the runtime environment for executing JavaScript on the server side. It handles API requests and responses, connecting the frontend to the backend logic.

Python

Role: It is used to implement our Large Language Model and to integrate the front end and backend components.

Flask:

Role: Flask is used for backend functions in which it creates API endpoints for LLM model inference and seamless data storage.

Database Management

MongoDB

Role: MongoDB is a NoSQL database which is used for storing user prompts, responses, interaction history, Chat Sessions, and Embedding vectors of each documents uploaded. It is best for dealing with dynamic data which is essential in Unibot.

6.2 Deployment setup

To deploy our model, we are first strengthening its key areas. We have also decided to add more features including User Feedback to take constructive criticism to improve our model's efficiency and to add a new layer of security to protect the education institution's data from cyber-attacks. These enhancements will make sure that our product only gives the best performance, and it becomes a secure system for protective data flow.

6.3 Algorithms

Function to get response from GPT-3.5-turbo

Function `get_gpt_response(prompt, document_text=None)`:

Initialize messages as a list containing:

- A dictionary with "role" as "system" and "content" as "{ \"role\": \"system\", \"content\": \"You are UniBot for University of Management And Technology. Answer the user's questions based on the provided content. Answer correctly and in detail and you should not apologize frequently. Avoid suggesting the user visit the website.\" }"
- A dictionary with "role" as "user" and "content" as prompt

If `document_text` is provided:

Insert a dictionary at index 1 of messages with:

- "role" as "system"
- "content" as "Here is the document content for reference: {document_text}"

Call `openai.ChatCompletion.create` with:

- model set to "gpt-3.5-turbo-0125"
- messages set to messages
- temperature set to 0.1

Retrieve the response content from the first choice of the response

Return the response content

Function to perform LDA for topic modelling

Start the texts as a collection of already-processed conversation texts.

Eliminate empty lists from text

If the text is blank:

return "No topics that are valid were found"

Use corpora to initialise the dictionary. Dictionary with texts

Initialize corpus as a list of bag-of-words representations of texts using dictionary

If corpus is empty:

Return "No valid topics found"

Create an LDA model with:

- corpus
- num_topics set to 1
- id2word set to dictionary
- passes set to 30

Retrieve topics from the LDA model with `num_words` set to 3

Extract topic words from the first topic

Generate context_text by joining topic words with spaces

Initialize prompt as "Summarize the following keywords into a single topic:
{context_text}"

Call get_gpt_response with prompt

Return the summary from get_gpt_response

Function to perform embeddings

Function get_embedding(text, is_prompt=False):

If is_prompt is True:

Set model to "text-embedding-ada-002"

Else:

Set model to "text-embedding-ada-002"

Call openai.Embedding.create with:

- input set to text
- model set to model

Retrieve the embedding from the response

Return the embedding

Function to upload document:

Function uploadTextDocument():

If uploadedFile is not selected:

Print "No file selected for upload"

Return

Initialize formData as a new FormData object

Append uploadedFile to formData with key 'file'

Append username to formData with key 'username'

Try:

Call fetch with:

- URL set to 'http://127.0.0.1:5000/upload_text_document'
- method set to 'POST'
- body set to formData

Parse the response as JSON

Print 'Server response:' and the parsed data

Set upload message to 'Document uploaded successfully'

Set a timeout for 3000 milliseconds to clear the upload message

Catch error:

Print 'Error uploading text document:' and the error

Function to fetch Chat History of user:

Function fetchChatHistory(username):

Try:

Call fetch with:

- URL set to 'http://127.0.0.1:5000/chat_history'
- method set to 'POST'
- headers set to {'Content-Type': 'application/json'}
- body set to JSON.stringify({ username: username })

Parse the response as JSON

If data.success is True:

Set chat history in state to data.chatHistory

Else:

Print 'Error fetching chat history:' and data.message

Catch error:

Print 'Error fetching chat history:' and the errorConstraints

6.3.1 Assumptions

Technical Support:

We are entitled to get all required technological help from university's IT department for integrating the chatbot with existing systems and resolving any technical issues.

Data Availability:

The university will provide access to all relevant curriculum data, including course materials, lecture notes, and textbooks, for fine-tuning the language model.

Database Maintenance:

All database maintenance, including backups and updates, will be handled by the university's IT team to ensure data integrity and availability.

Stable Internet Connection:

Users will have a stable internet connection to interact with the chatbot and access its features without interruptions.

User Training:

Both students and teachers will receive adequate training and support to use the Unibot system effectively.

6.3.2 System constraints

1. Performance Constraints:

The system must handle a large number of concurrent users without significant delays, ensuring response times remain under 2 seconds for 95% of interactions.

3. Compatibility Constraints:

The system should be compatible to all modern web browsers such as Google Chrome, Apple Safari, and Microsoft Edge to ensure broad accessibility.

4. Scalability Constraints:

The backend infrastructure must be scalable to accommodate increasing numbers of users and data, with the ability to add resources dynamically as needed.

5. Availability Constraints:

The system must maintain an uptime of at least 99.5%, ensuring reliable access for students and teachers throughout the academic year.

6.3.3 Restrictions

1. Resource Usage:

The system must operate within the allocated computational and memory resources to avoid overloading the university's infrastructure or incurring excessive costs.

2. Data Input:

Only authorized users (students and teachers) can input data into the system to prevent unauthorized access and data manipulation.

3. Third-Party Services:

Integration with third-party services is limited to approved educational tools and platforms, ensuring compatibility and adherence to security policies.

6.3.4 Limitations

2. Specialized Subject Matter:

The chatbot may have limited capability in handling highly specialized or niche subject matter outside the core university curriculum due to the scope of the training data.

3. Language Support:

Initial deployment supports only English. Support for additional languages may require significant model retraining and data collection.

4. Offline Access:

The system requires an internet connection and does not support offline access, limiting its usability in areas with poor connectivity.

6. Customization by End Users:

End users (students and teachers) cannot customize the chatbot's behavior or training data directly, relying on the development team for updates and improvements.

7. TESTING

7.1 Extended Test Cases

Test Case ID: 1			Test Design By: Ahmad Shamayl			
Test Module Name: User Authentication			Test Design Date: 15 May, 2024			
Test Priority: High			Test Executed By: Summaya Ayaz			
Test Title/Name: To authenticate users			Test Executed Date: 18 May,2024			
Description: To check if only authorized users access Unibot.						
Pre-Condition: The user opens our website, clicks on the signup button and sign up according to the format						
Dependencies						
Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1.	Open website				Pass	
2.	Click SignUp Key					
3.	Fill the SignUp form in accordance with the format	Full Name: Ahmad Shamayl Username: Ahmad Email: F2020332009 Password:12345678				
4.	Click on the SignUp Button		The user creates their account	The user has successfully created their account.		
Post Condition The user now only needs to login to their account to access Unibot.						

dTest Case ID: 2			Test Design By: Saad Bin Abid			
Test Module Name: Quiz Generation			Test Design Date: 15 May, 2024			
Test Priority: Very High			Test Executed By: Sadaf Younes			
Test Title: Quiz Generation			Test Executed Date: 18 May,2024			
Description: To check if the generated quizzes are relevant and accurate.						
Pre-Condition: The user opens our website which is working smoothly and gives a query to Unibot which generates an error-free response.						
Dependencies						
Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1.	Go to the website				Pass	
2.	Signup /Login	Username: sadaf Password:12345				
3.	Write the Query and select the number of questions as well as the course.	Number of Questions: [Specify the desired number of questions for the quiz]				
4.	Review the generated questions.		The quiz generation should execute without errors and give accurate and relevant responses.	The quiz generated executed without errors and gave accurate and relevant responses		
Post Condition Each question in the quiz is accurate, well-formulated, and aligned with the difficulty level appropriate for the course.						

7.2 Decision Table

7.2.1 Code snippet

Condition Stubs	Rule 1/TC1	Rule 2/TC2	Rule 3/TC3	Rule 4/TC4	Rule 5/TC5	Rule 6/TC6	Rule 7/TC7	Rule 8/TC8
User Authentication	T	T	T	T	F	F	F	F
Query Submission	Valid	Invalid	-	-	Valid	Invalid	-	-
Assignment Submission	-	-	Valid	Invalid	-	-	Valid	Invalid
Document Upload	-	-	-	-	-	-	-	-
Action: Process Query	X							
Action: Error Message		X		X		X		X
Action: Create Assignment			X				X	
Action: Access Denied					X			
Action: Upload Document								

Table 6: Decision coverage table

Explanation:

T: True (Condition met)

F: False (Condition not met)

Valid: Valid input

Invalid: Invalid input

X: Action to be taken

7.3 Traceability Matrix

Use Cases (UCID)

UC1: Sign Up

UC2: Log In

UC3: Authentication

UC4: UniBot Dashboard

UC5: General Query

UC6: Assignment Creation
UC7: Quiz Generation
UC8: Chat History
UC9: Chat Deletion
UC10: Chat Synchronization
UC11: Upload Document
UC12: Task Generation from Document
UC13: Logout
UC14: Chat Sessions

Test Cases (TID)

TC1: Test Sign Up
TC2: Test Log In
TC3: Test Authentication
TC4: Test Dashboard
TC5: Test General Query Submission
TC6: Test Assignment Creation
TC7: Test Quiz Generation
TC8: Test Chat History Display
TC9: Test Chat Deletion
TC10: Test Chat Synchronization
TC11: Test Document Upload
TC12: Test Task Generation from Document
TC13: Test Logout

Prototypes (PID)

P1: Sign Up Page
P2: Log In Page
P3: Authentication Mechanism
P4: Dashboard Layout
P5: General Query Interface
P6: Assignment Creation Interface
P7: Quiz Generation Interface
P8: Chat History Display
P9: Chat Deletion Interface
P10: Chat Synchronization Mechanism
P11: Document Upload Interface
P12: Task Generation from Document Interface
P13: Logout Mechanism
P14: Chat Sessions Display

Requirements (RID)

R1: User should be able to sign up
R2: User should be able to log in

- R3: System should authenticate the user
 R4: User should see the UniBot dashboard
 R5: User should be able to submit a general query
 R6: User should be able to create assignments
 R7: User should be able to generate quizzes
 R8: User should see chat history
 R9: User should be able to delete chat history
 R10: User chat should sync across sessions
 R11: User should be able to upload documents
 R12: System should generate tasks from the document
 R13: User should be able to log out
 R14: User should be able to see active chat sessions

7.3.1 RID vs UCID (requirements vs use cases)

UCID/RID	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
UC1	✓	✓												
UC2	✓													
UC3			✓											
UC4				✓										
UC5					✓									
UC6						✓								
UC7							✓							
UC8								✓						
UC9									✓					
UC10										✓				
UC11											✓			
UC12												✓		
UC13													✓	
UC14														✓

Table 7: RID vs UCID

7.3.2 Prototypes (RID vs PID)

PID/RID	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
P1	✓													
P2		✓												
P3			✓											
P4				✓										
P5					✓									
P6						✓								
P7							✓							
P8								✓						
P9									✓					
P10										✓				
P11											✓			
P12												✓		
P13													✓	
P14														✓

Table 8: Prototypes (RID vs PID)

7.3.3 Test Cases (RID vs TID)

TID/RID	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
TC1	✓													
TC2		✓												
TC3			✓											
TC4				✓										
TC5					✓									
TC6							✓							
TC7								✓						
TC8									✓					
TC9										✓				
TC10											✓			
TC11												✓		
TC12													✓	
TC13														✓

Table 9: Test Cases (RID vs TID)

7.3.4 Coverage (UCID vs TID)

TID/UCID	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12
TC1	✓											
TC2		✓										
TC3			✓									
TC4				✓								
TC5					✓							
TC6							✓					
TC7								✓				
TC8									✓			
TC9										✓		
TC10											✓	
TC11												✓
TC12												

Table 10: Coverage (UCID vs TID)

8. RESULTS/OUTPUT

8.1 %completion

Completion is calculated to be the percentage of requirements which are satisfied by the above use cases.

From the above RID vs UCID matrix:

Total requirements (R): 14

Covered requirements: 13

$$\text{Completion Percentage} = \left(\frac{13}{14} \right) \times 100 \approx 92.86\%$$

8.2 %accuracy

Accuracy is the calculation of the percentage of requirements that are fulfilled by the corresponding test cases.

From the above RID vs TID matrix we have,

Total requirements (R): 14

Covered requirements: 13

$$\text{Accuracy Percentage} = \left(\frac{13}{14} \right) \times 100 \approx 92.86\%$$

8.3 %correctness

Correctness is the calculation of the percentage of use cases that have correlating test cases applying to them.

From the above UCID vs TID matrix, we have:

- Total use cases (UC): 12
- Covered use cases: 11

$$\text{Correctness (\%)} = \left(\frac{11}{12} \right) \times 100 \approx 91.67\%$$

9. CONCLUSION

Unibot is an all-inclusive software that is specifically made for educational institutions, which provides teachers and students with a medium of smooth communication and workflow. Its easily accessible interface allows teachers to speed up the whole quiz generation and assignment creation tasks which in turn facilitates them to effectively manage their coursework. Unibot not only makes the workflow easier for teachers it also pushes for collaboration and engagement of various educational resources. In conclusion, Unibot stands firm in transforming the educational sector of Pakistan by bridging the gap between cutting-edge technology and education with its main aim to improve the efficiency and success of students.

10. FUTURE WORK

- **Enhanced Student Interaction:** Introduce features like discussion forums and collaborative project spaces to foster active student engagement and peer-to-peer learning.
- **Personalized Learning Paths:** Implement algorithms to study the data of student performance to enhance learning resources.
- **Integration with Learning Management Systems (LMS):** Explore integration options with existing LMS platforms to extend UniBot's functionality and compatibility with institutional systems.
- **Mobile Application Development:** We can develop a mobile application for Unibot to make it more accessible and convenient for teachers to manage their coursework at any place at any time.
- **Analytics and Reporting:** We can add a dashboard for analytics to show the number of users, and their activity and extract insights from it to enhance our software accordingly
- **Feedback Mechanisms:** Implement feedback mechanisms for both teachers and students to gather input on the platform's usability and drive iterative improvements based on user feedback.

11. BIBLIOGRAPHY

- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998-6008.
- J. Smith and A. Johnson, "Enhancing natural language processing for educational chatbots," *J. Artif. Intell. Educ.*, vol. 15, no. 2, pp. 123-135, 2023.
- Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Improving coursework generation with large language models," *IEEE Trans. Learn. Technol.*, vol. 16, no. 3, pp. 245-258, 2023.
- T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 14665-14677, 2020.
- A. Radford, J. Wu, D. Amodei, D. Amodei, I. Clark, G. Brundage, A. Narasimhan, and I. Sutskever, "Language models are unsupervised multitask learners," OpenAI Blog, 2019. [Online]. Available: <https://openai.com/blog/language-models/>.
- J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annu. Conf. North Am. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2018, pp. 4171-4186.
- GPT-3.5, "GPT-3.5: An enhanced language model for diverse applications," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2021, pp. 567-578.

12. APPENDIX

12.1 Glossary of terms

- **UniBot:** An AI-powered teaching assistant designed for higher education institutions, facilitating tasks such as coursework generation and query handling.
- **Falcon 7B:** A large language model used for the competitive analysis during the selection phase of our project.
- **MongoDB:** A NoSQL database used as the main storage system for our project to store as well as receive data.
- **React JS:** A JavaScript library that is used to create the frontend interface of UniBot, which consists of a responsive user interface.
- **Natural Language Processing (NLP):** Is known to be the division of AI used to help computers understand human language.
- **Transformer Models:** Deep learning systems that process parts of data in a sequence with the help of attention mechanisms to analyze only the needed part of data.
- **APIs (Application Programming Interfaces):** These interfaces make it easier for UniBot's frontend and backend systems to integrate by enabling communication and interaction between various software programs.
- **Embeddings:** Machine Learning models employ vector representations to understand semantic relationships between data.
- **OpenAI:** An Organization that has created AI models like GPT-3.
- **GPT-3.5 Turbo:** OpenAI's new edition of the well-known GPT -3, which can produce writing that resembles a human.
- **LLM (Large Language Model):** AI frameworks which are used to comprehend and create human-like text.
- **Flask:** A lightweight Python web framework.
- **Prompt Engineering:** Crafting inputs to guide a language model's output effectively.
- **Data Vectors:** Numerical representations of data for machine learning.
- **CSS (Cascading Style Sheets):** A language which is used to style and structure web pages.