Arab American University

Faculty of Engineering

Computer Systems Engineering

SENIOR PROJECT

Chatbot Assistant and Schedule Generator Mobile App for University Students

2024

Students Statement

We, the undersigned students, certify and confirm that the work submitted in this project report is entirely our own and has not been copied from any other source. Any material that has been used from other sources has been properly cited and acknowledged in the report.

We are fully aware that any copying or improper citation of references/sources used in this report will be considered plagiarism, which is a clear violation of the Code of Ethics of the Arab American University.

Estabraq Abualrob 201911113

Mosab Eneirat 201910713 Ahmed Ghannam 201910686

Supervised by: *Prof. Mohammed Awad*

Computer Systems Engineering Dept.
Submitted in partial fulfillment of the requirements of B.Sc. Degree in Computer Systems Engineering

2024

Supervisor Certification

This is to certify that the work presented in this senior year project manuscript was carried out under my supervision, which is entitled:

"***Chatbot Assistant and Schedule Generator Mobile App for University Students***"

Estabraq Abualrob 201911113

Mosab Eneirat 201910713 Ahmed Ghannam 201910686

I hereby that the aforementioned students have successfully finished their senior year project and by submitting this report they have fulfilled in partial the requirements of B.Sc. Degree in Computer Systems Engineering.

I also hereby that I have **read, reviewed, and corrected the technical content** of this report and I believe that it is adequate in scope, quality, and content and it is in alignment with the ABET requirements and the department guidelines.

Prof. Mohammed Awad

ACKNOWLEDGMENT

We extend our heartfelt appreciation to everyone who supported and contributed to the success of this senior project. Your guidance, encouragement, and understanding have played a pivotal role in making this journey both rewarding and fulfilling.

ABSTRACT

The reliance of students on university staff for information proves time-consuming for both parties, and the increasing student population exacerbates the issue. This project addresses the challenges faced by university students, particularly in navigating university resources and efficiently managing course registration.

Our provided solution is a mobile application featuring a chatbot that answers frequently asked questions, providing instant support to students. Additionally, the app includes a schedule generation tool to streamline the registration process and allow students to select a suitable semester schedule.

The results included heightened student awareness of university resources and a simplified registration experience, ultimately enhancing overall satisfaction with the university journey. The project's deliverables encompass a user-friendly mobile app for both iOS and Android, equipped with a knowledgeable chatbot that achieved 89% accuracy and a conflict-free schedule generator that was tested using various real-world test cases. This project successfully achieved significant milestones in meeting the needs of university students.

Keywords: Chatbot, Schedule Generator, Artificial Intelligence, Natural Language Processing

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

LIST OF ABBREVIATIONS	
AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
CSE	Computer Systems Engineering
ER	Entity Relationship
GUI	Graphical User Interface
ML	Machine learning
NLP	Natural Language Processing
NN	Neural Networks
UI & UX	User Interface and User Experience
SAAS	Software as a Service

1 CHAPTER 1: INTRODUCTION

In the field of education, traditional systems face great challenges and obstacles that prevent access and the ability to adapt. One of the most prominent problems facing students is the difficulty of registering for courses and the time conflicts that occur between them, as well as the problems of obtaining the accurate information necessary for registration and the inquiries that the student needs at the time. Fittingly, these challenges require innovative solutions, so technology stands out by enhancing the ability to help students solve their problems. In this context, the integration between chatbot and schedule generator technology becomes pivotal, because it reduces administrative burdens, helps students solve their registration problems quickly and accurately without any conflict between courses through the schedule generator, and answers students' inquiries through the chatbot. Without having to go to university or go to the Registration and Admissions Department, the following chapters will go into detail about these innovative solutions and provide a comprehensive exploration of their impact on university education.

1.1 PROBLEM STATEMENT AND PURPOSE

University students face challenges in getting information and receiving answers to their questions about the university, such as knowing how to navigate the university website or understanding course details. Relying on the Deanship of Admissions and Registration staff can be stressful and time-consuming for employees, especially since they are not always available. The continuous increase in new student numbers also creates additional challenges, especially for students who may be shy to ask questions or may receive misinformation from fellow students.

Registering for university courses is a common challenge for students, as it requires them to

manually select courses, which may be hard because they must avoid time conflicts between different courses, the thing that distracts students from their academic tasks, wastes their time, and frustrates them, especially new students. Adjusting schedules can also be difficult, due to the same reasons mentioned previously.

To solve these challenges, we propose a mobile application with two key features:

- Chatbot: This program will provide students with the necessary support via chat, answering their inquiries related to the university and solving related problems.
- Schedule Generation Tool: This tool will help students create their schedules easily and quickly while avoiding conflicts between courses.

The proposed mobile application aims to improve the course registration experience for computer engineering students at Arab American University. The app will provide a more efficient and user-friendly system, with features such as a chatbot and scheduling generator.

1.2 Project and Design Objectives

The main goal of this project is to design a mobile application that auto-generates a free-conflict schedule based on the number of credit hours the student wants to register and the preferred time, in addition to creating a chatbot that can answer students' questions and give needed information.

The following objectives are produced for the project:

- Create a cross-platform Flutter mobile application that runs on both iOS and Android environments.
- Developing an intelligent chatbot that provides accurate responses to students' inquiries in

Arabic, ensuring fast response times of less than 5 seconds per message and aiming for an accuracy level of around 80%

- Developing a Schedule Generator that can produce a conflict-free schedule in under 20 seconds.

1.3 Intended Outcomes and Deliverables

At the end of this project, we aim to enhance students' university experience. We expect two main outcomes: Firstly, to increase students' awareness of available resources, services, and extracurricular activities in the university, secondly, to simplify the registration process by saving time and effort. With our project, we hope to improve students' satisfaction with their university journey.

The key deliverables of the project are in deploying a user-friendly mobile app for both iOS and Android; packed with a knowledgeable chatbot that answers questions, and a schedule generator that can generate a conflict-free schedule. We believe this documented project can make students' university life easier.

1.4 Summary of Report Structure

The senior project design document is structured into several chapters to ensure readability for technical experts.

Chapter 1 includes an introduction to the project, the problem statement, and the purpose of the project, with an overview of intended outcomes and deliverables.

Chapter 2 provides background information and previous research and projects that have explored similar ideas.

In Chapter 3, the document describes the project's main components, including the system design, design specifications, related standards, and any constraints. The chapter also covers system analysis, and optimization, and concludes with simulation or experimental tests.

Chapter 4 highlights the project's findings and conclusions, while Chapter 5 focuses on project management, including tasks, schedules, milestones, resource management, and cost management. This chapter also includes any lessons learned during the project.

Chapter 6 discusses the impact of the engineering solution on the environment, economy, society, and other relevant issues.

Finally, in Chapter 7, the document concludes with the team's concluding thoughts on the project, skills learned during the process, and recommendations for future work.

2 CHAPTER 2: BACKGROUND

2.1 OVERVIEW

This chapter provides a theoretical background on chatbots and schedule generators, highlighting their relevance to university students and the technologies used in their development. Based on our university experience, we as students faced several problems, especially the difficulty of registering or the need to obtain the correct information easily and quickly. To confirm this, we conducted a questionnaire for the students to ask them about the problems they face at the university.

The first question was about the problems and challenges faced by freshmen at the university, and the answers were about issues of registration and understanding administrative procedures in dealing with the department head and professors, difficulty in finding the location of lectures and moving between colleges, challenges in communicating with doctors and understanding teaching methods, difficulty in Organizing time and understanding the subject of hours and grades, in addition to cognitive challenges such as not knowing the major well and difficulty accessing halls and university facilities. These challenges highlight the importance of supporting students at the beginning of their university journey.

As for the second question about their difficulties in the rest of the university years, the problems that students faced continued in various fields, including challenges in registration processes and study schedules, difficulty in opening closed sections, problems in communicating with the Deanship of Admission and Registration and difficulties in obtaining educational resources. Furthermore, the challenges in organizing exam schedules and conflicting subjects in the study

plan. Despite some benefits and adaptation to the university environment, some problems persist, which highlights the importance of following up and helping students even after they finish the first year.

As for the question of what to do when they encounter difficulties and who to turn to, the students relied on various sources of support and assistance. Many of them reported that they did not find support from anyone and relied on themselves, while others reported that they turned to their older colleagues, department heads, and the Student Union Council for help. Some students indicated using social media and student groups to obtain information and support. This indicates that sometimes there are hardships or the risk of obtaining wrong answers from non-specialists, in addition to the fact that two-thirds of these students find it difficult to obtain this help electronically, as shown in the following figure.

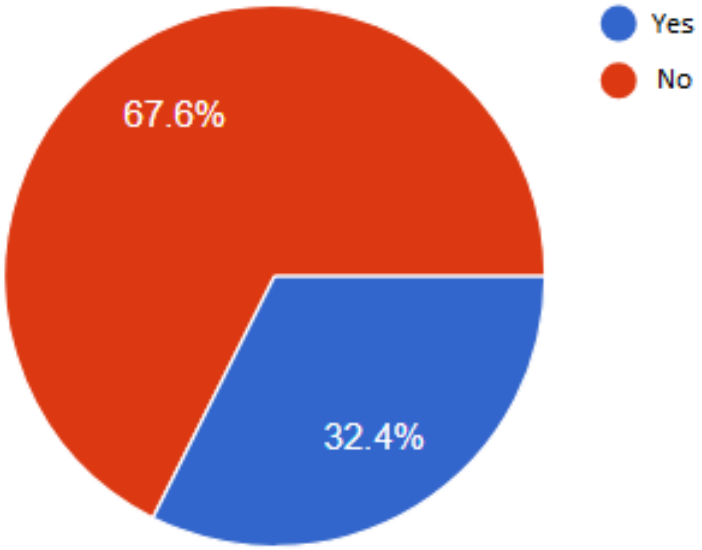


Figure 1 Landscape of Remote Assistance Challenges Among Students

2.1.1 Chatbots

Chatbots are computer programs designed to simulate conversations with human users. These chatbots can provide personal assistance, answer questions, and complete tasks 24/7. From automating customer service and simplifying online transactions to providing companionship and entertainment. In recent years, it has gained increasing popularity, especially in education. Education chatbots can provide students with access to information, answer questions, and provide support, which can improve learning outcomes and student satisfaction.

2.1.2 Natural Language Processing (NLP)

Natural language processing, the known field of artificial intelligence, gives computers the ability to decode and process human language. This field combines linguistics, machine learning, and deep learning to enable computers to understand text and speech, in addition to contributing to the rise of a range of fields such as machine translation, chatbots, and sentiment analysis. NLP plays a crucial role in the development of chatbots, which allows them to understand user queries and respond to them naturally and engagingly.

2.1.3 Schedule Generator

Schedule Generator is a software application that automates the process of creating student schedules. They apply some algorithms to avoid conflicts, recommend courses, and optimize timetables based on user preferences, reducing the time and effort needed for students to navigate the registration process.

2.1.4 Mobile App Development

Mobile app development requires a suitable framework, which is the reason why we use Flutter. The Flutter framework is a popular choice nowadays for creating mobile apps for both iOS and Android versions with native performance and an easy-to-use development environment. This framework, which is developed by Google, helps developers create amazing, beautiful, and expressive apps for multiple platforms using a single code base written in Dart programming language. Flutter's architecture provides excellent speed and interactive development applications that respond effectively to user interactions.

2.2 RELATED WORK

Because we have two distinct primary features in our application, we will split the related work into two parts, the first for related educational chatbot, and the second for related work about the schedule generator.

2.2.1 Chatbot

2.2.1.1 An Intelligent Arabic Chatbot System Proposed Framework [1]

The paper introduces a rule-based Arabic Chatbot System developed through the COVID-19 pandemic, aiming to reduce the workload of admissions employees and assist students at Al-Zaytoonah Private University of Jordan. The chatbot, designed as a web application, addresses diverse topics such as admissions, academics, attendance, GPA, placement, and other events. The paper outlines the phases involved in the system, including pre-processing, keyword extraction, and response generation based on **A.I.M.L**. files.

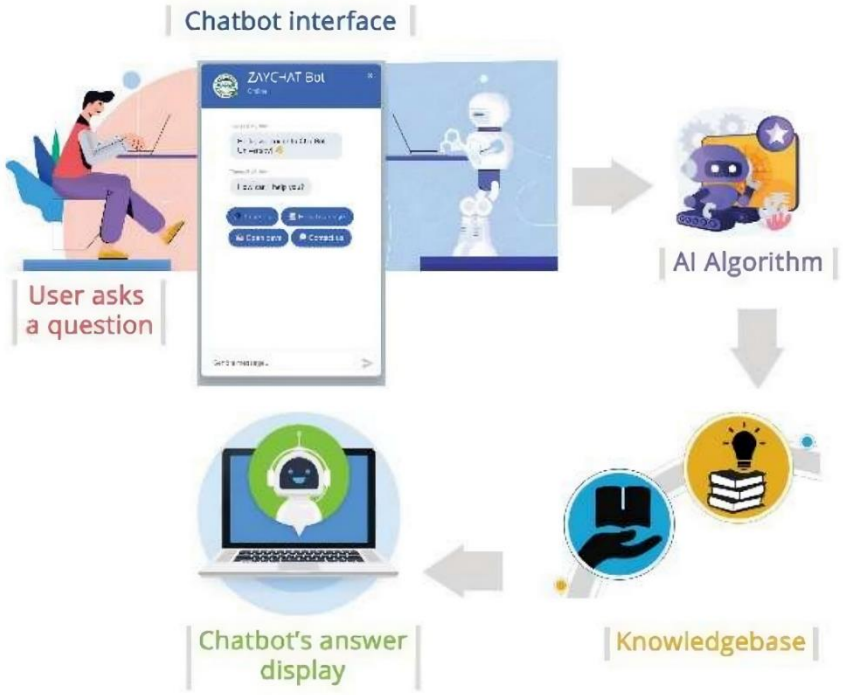


Figure 2 Zaytoonah Private University chatbot main components [1]

2.2.1.2 Arabic Educational Neural Network Chatbot [2]

The paper underscores the significance of developing Arabic chatbots for academic purposes in the United Arab Emirates, with a focus on educational regulations. The authors collected datasets from Arabic educational websites and employed Natural Language Processing (NLP) techniques for data preprocessing. The system was trained using a neural network, specifically a **BiLSTM** architecture with four layers, which is suitable for analyzing both short and long sentences as sequence data. The ultimate objective is to provide 24/7 access to Ministry of Education information for users, eliminating the need for physical visits or emails.

2.2.1.3 Bilingual AI-Driven Chatbot for Academic Advising [3]

The paper outlines the creation of a bilingual chatbot for university students, offering academic advice in both English and Arabic languages. Employing Natural Language Processing (NLP) and a **neural network (NN)** deep learning algorithm, the chatbot is designed to deliver real-time responses to academic inquiries. Two chatbot models are developed through supervised deep learning, and training on English and Arabic corpora, involving pre-processing steps like tokenization, lemmatization/stemming, and word vectorization for improved efficiency. The graphical user interface (GUI) for the advising chatbot is constructed using Python's Tkinter library, creating a user-friendly and language-sensitive interface. The paper concludes with performance evaluations, including a confidence score assessment achieving 80% accuracy in English and 75% in Arabic, along with a user evaluation yielding equivalent results.

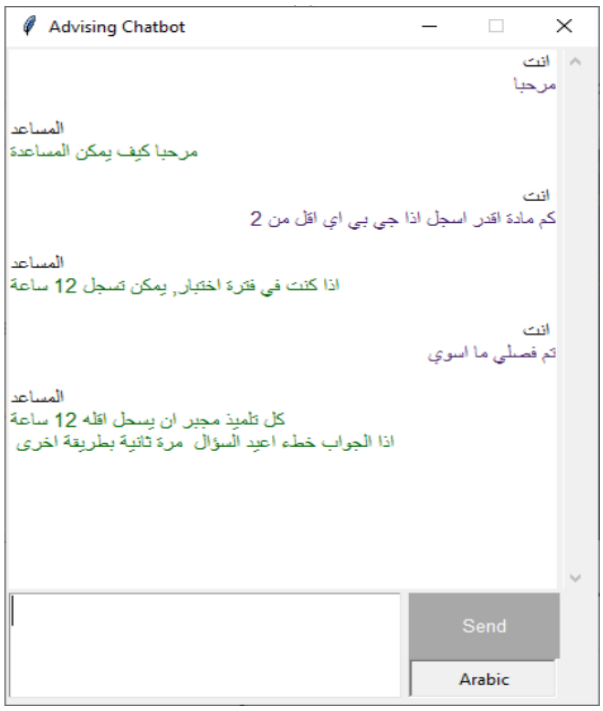


Figure 3 Screenshot of an Arabic Conversation [3]

2.2.1.4 NEU-chatbot: Chatbot for admission of National Economics University [4]

The paper introduces an AI-based chatbot developed to provide daily updates on the curriculum, admission processes, tuition fees, and other information for National Economics University (NEU) in Vietnam. The chatbot, integrated into the **Rasa framework**, employs Deep Learning models and features a rational pipeline for optimal accuracy and prevention of model overfitting. It successfully detects over fifty types of user questions with a 97.1% accuracy on the test set. The chatbot is applied on NEU's official admission Fanpage on Facebook.

Rasa platform can handle conversation flow, understand, and classify intents, and extract entities. Despite achieving accuracy, the paper notes the need for manual updates to the nlu.yml file annually and ongoing training to adapt to new academic year information. The authors foresee

improvements in the chatbot's performance over time.

2.2.1.5 EduChat: An AI-Based Chatbot for University-Related Information Using a Hybrid Approach [5]

The paper introduces EduChat, a chatbot system designed for addressing university-related queries at Ho Chi Minh City University in Vietnam. The chatbot aims to automatically answer queries about universities, academic programs, admission procedures, and student life. EduChat uses a hybrid approach, combining rule-based methods, an **improved random forest (IRF) algorithm, and ChatGPT.** The two-step approach involves text classification to categorize questions into predefined topics and immediate responses based on existing information within the knowledge base. If the query falls outside these bounds, EduChat utilizes the ChatGPT API to generate contextually relevant responses, ensuring the effective handling of a diverse array of inquiries.

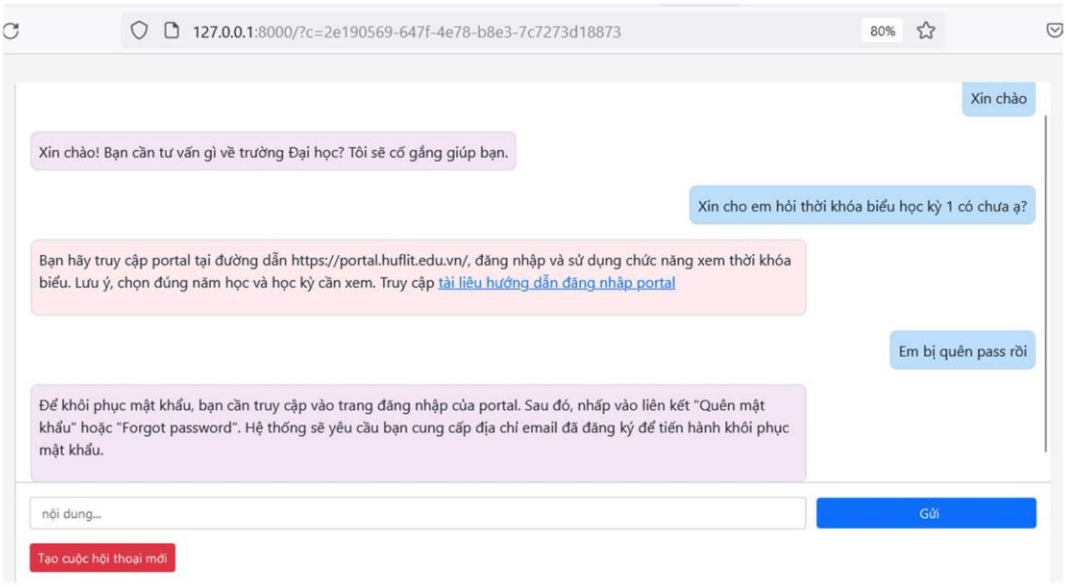


Figure 4 Example of EduChat that directed the student to the portal and reset password [5]

2.2.1.6 Summary for Chatbot

We have seen some related chatbots in the education sector, and the Arabic education sector too, and we aim to use BERT Embedding with similarity score to find the most relevant answer in our dataset for the user input query.

2.2.2 Schedule Generator

2.2.2.1 Automatic Timetable Generator [6]

Automatic Timetable Generator provides an easy-to-use system that can create timetables faster compared to manual methods by taking different inputs such as semester details, topics, faculty details, and workload as inputs and using Genetic Algorithm and N Queen algorithms to create the timetable. The system was developed using technologies such as PHP, HTML, and MySQL database. It contains modules for user registration, login, and generation of the final schedule. Such projects provide a starting point for developing innovative, student-centered scheduling tools using appropriate algorithms.

2.2.2.2 A Review of Optimization Algorithms for University Timetable Scheduling [7]

This project presents an automated solution for timetabling university courses. The goal is to reduce the time taken to complete this task efficiently and meet the needs of all parties involved. It uses Particle Swarm Optimization (PSO) and Genetic Algorithms, to address the limitations and complexities of the timetabling problem. It considers factors such as teacher allocation and student preferences.

2.2.2.3 An SHO-based Approach to Timetable Scheduling: A Case Study [8]

This project schedules the timetable at Nong Lan University. This process ensures that appropriate time slots and semesters are allocated to various courses, considering specific constraints. Spotted Hyena Optimizer (SHO) algorithms have been used, where the SHO-based approach is effective in finding timetables. The project also explores a combination of Simulated Annealing (SA) and SHO algorithms to improve the overall performance of the scheduling process.

2.2.2.4 Summary for Schedule Generator

In our proposed project, we get valuable insights taken from the mentioned papers on schedule generation, but we use a Recursive Backtracking algorithm.

3 CHAPTER 3: METHODS AND MATERIALS

In this chapter, we will talk about the software components used in our system. In addition, we are going to discuss the algorithms and the flowcharts that our system will follow to implement its functions.

3.1 System Design and Components

3.1.1 System Design

The reason behind this mobile application is to enable students to obtain some educational services more easily and with less effort and time. Our application has two features, a chatbot to answer student questions and a schedule generator that can generate a free-conflict time schedule for the student based on the best non-passed courses yet. The chatbot uses an NLP model, and the schedule generator uses a backtracking algorithm above the courses' database, more details appear later. In figure 5, we present the architectural design of our application.

Figure 5 Architectural Design

3.1.2 Chatbot Flowchart

In this chatbot, we use BERT embedding as a semantic search between the student query and the dataset of the FAQs. BERT (Bidirectional Encoded Representations from Transformers) is a large language model created by Google. It was built on Transformer architecture, which is a powerful neural network model that can capture long-term dependencies and relationships between words. BERT is trained on a huge corpus of text in a bidirectional manner, meaning it processes text from both directions. This bidirectionality enhances their ability to capture semantic relationships between words and understand the context of a sentence.

Unlike traditional word embeddings that assign a fixed vector to each word, BERT embeddings represent words dynamically based on their context within a sentence, leading to a more accurate understanding of language, as depicted in Figure 6.

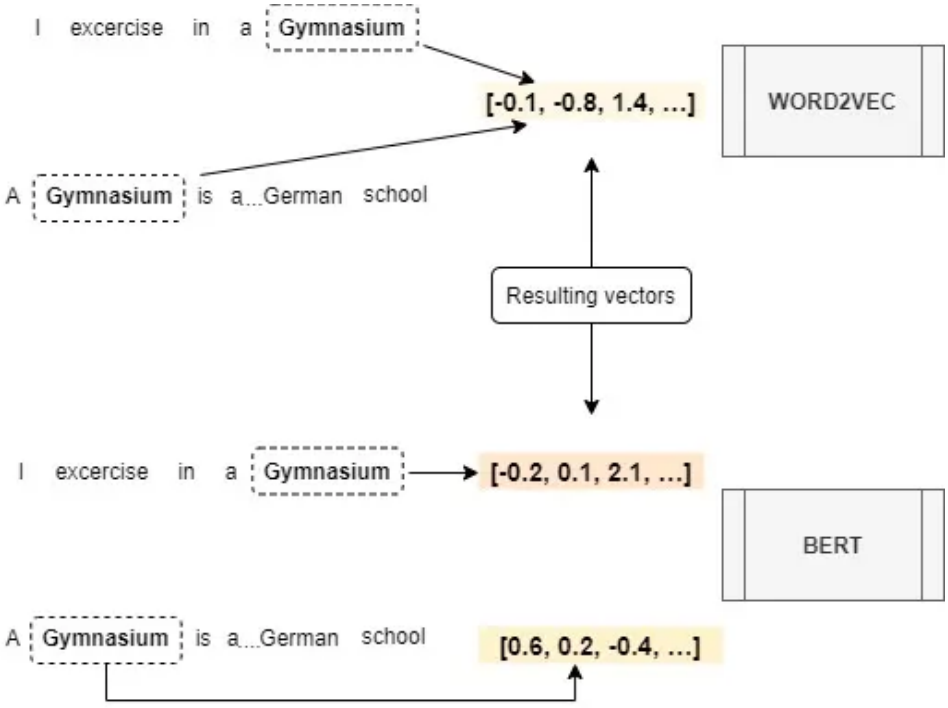


Figure 6 BERT adapts word representations based on context.[9]

As the chatbot flowchart shown in figure 7, we will first initialize the document at the beginning of the application, which includes loading BERT and the tokenizer, to encode and tokenize the dataset. Next, we will create BERT embeddings for each sentence in the dataset and save them.

Then, when a student asks a question to our chatbot, the question will be coded and coded, and then BERT will create an embedding for the question. The question will be checked for inclusion with the dataset, and the chatbot will give the appropriate answer if it gets a similarity of over 70%, otherwise the chatbot will display an error message.

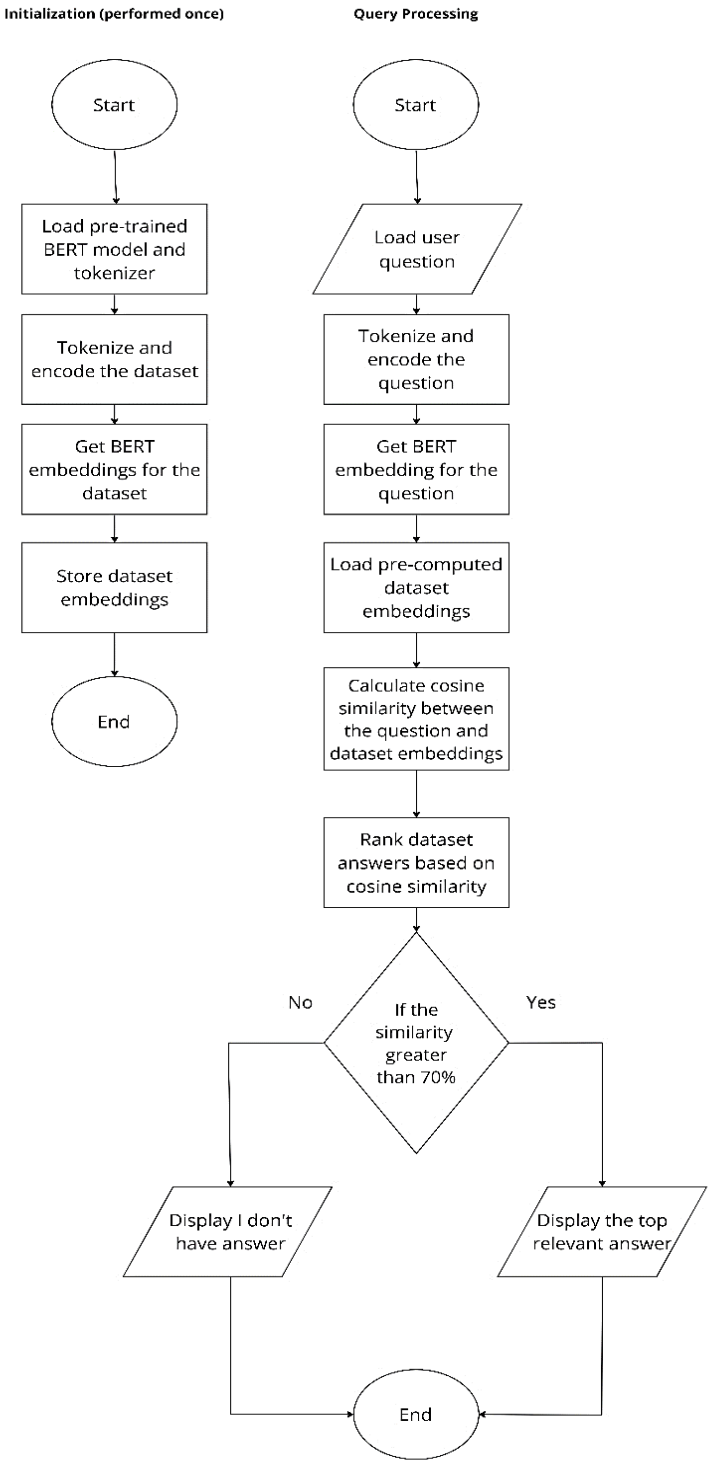


Figure 7 Chatbot Flowchart

3.1.3 Schedule Generator Flowchart

The scheduling generator algorithm goals to generate an appropriate schedule for a student depending on his passed courses, and current available courses with their sections. The algorithm includes recursively exploring the possible set of course sections, each selected section should be open and does not conflict with previously selected sections. The algorithm uses a **backtracking** mechanism to handle the situation where choosing a section leads to scheduling conflict.

1. Initialization:

- Initializing a list of the student courses that he did not pass yet, and an empty schedule.

2. Sorting Courses:

- The application has the not passed courses of the student yet because he entered the passed courses when he signed up for the app.
- Each course has a *defaultSemester* variable, which is an integer indicating the semester number to which the course is assigned in the CSE major tree.
- Each course has a *PreRequisitesCourses* variable, which is an integer that refers to the number of prerequisite courses in this course.
- The non-passed courses, located at the first of the major tree, have a higher weight.
- Courses with more prerequisite courses have a higher weight because this may affect registration for future semesters.
- Some courses must be taught in the same semester as the main course, such as English language courses and labs, or physics with its tutorial section.

- Labs and some important courses like technical writing will have higher weight.

Now, the algorithm will have a list of courses, and each course will have its sum of weights, then the algorithm will sort them inversely using the combined sorting algorithm, to get the best courses at the top of the list.

3. Recursive Course Selection:

The student provides the number of hours he wants to register for this semester, and he may provide other information like the favorite time slots or days for his schedule.

- Now, the core of the algorithm is the *generateSchedule* function, which will iterate and select sections for each course, starting from the first important course which is the first location at the sorted not passed yet courses list, using the *currentIndex* variable.
- While iterating through courses, any course with an open section is checked for conflicts with the current schedule using the *hasConflict* function.
    - If there is no conflict, the section will be added to the schedule, and then the algorithm moves on to the next course and increases the *currentIndex* variable.
    - If a conflict is found, the algorithm backtracks by removing the last added section and checking for alternative sections for the current course.
- The process continues until all the schedules reach the number of wanted hours. At this point, a successful schedule is achieved, and the algorithm will end and print the schedule to the student. The algorithm may also sometimes give a schedule with one or two hours more than the required hours.

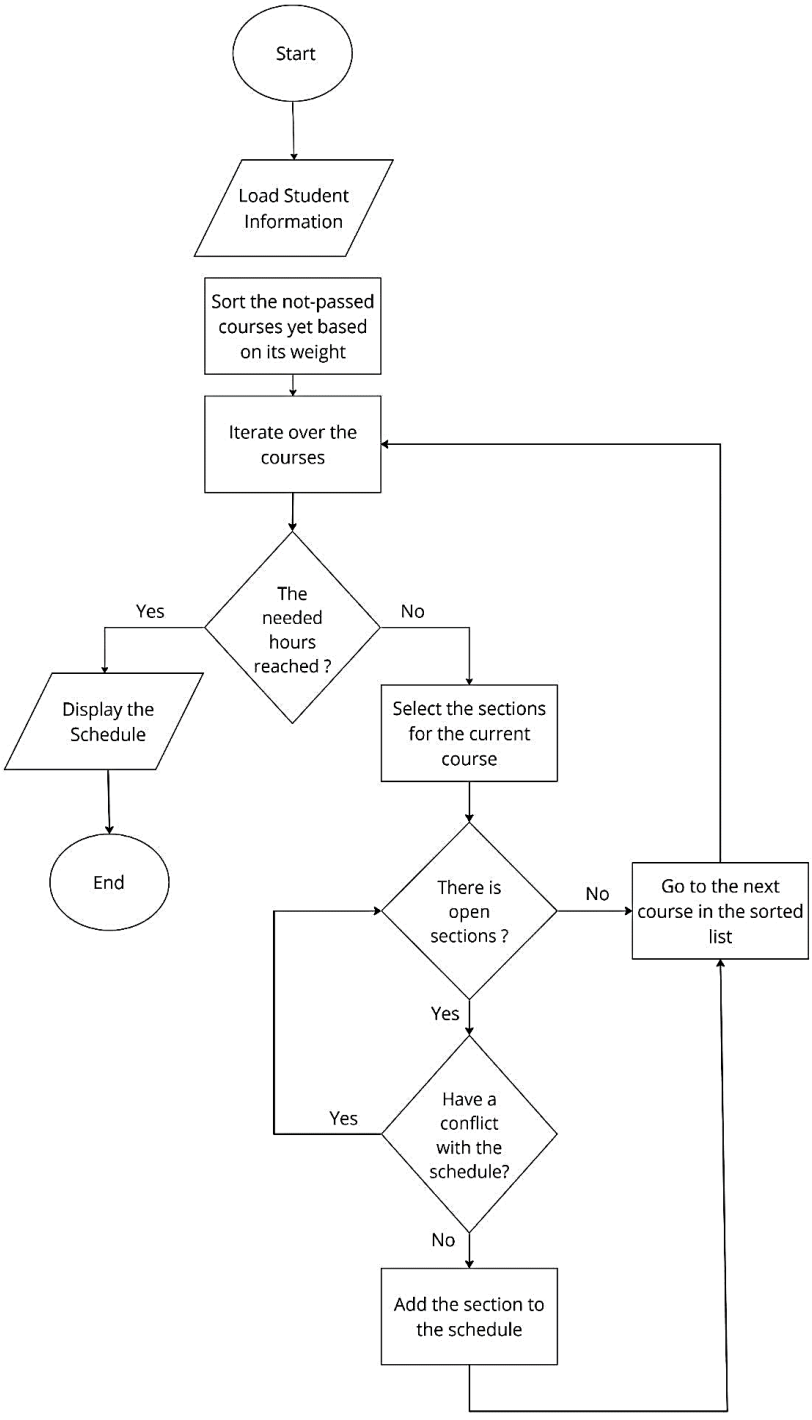


Figure 8 Schedule Generator Flowchart

3.1.4 Database Diagram

Schema is important for organizing data related to courses and departments, and it also helps automate the process of creating a schedule. It allows the system to understand the relationships between different entities and apply the necessary rules and algorithms to create improved schedules.

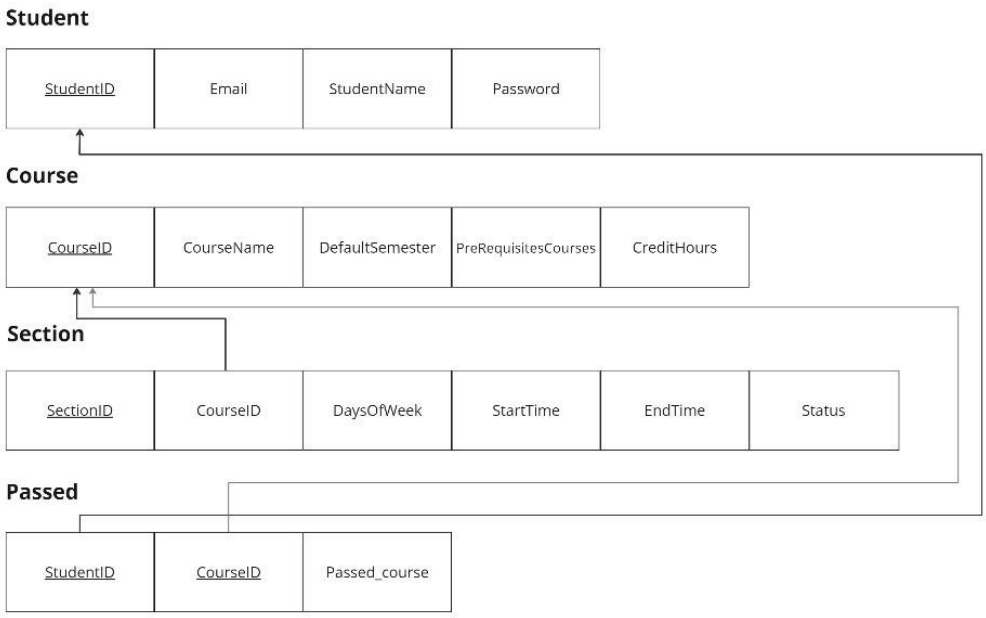


Figure 9 Database Diagram

3.1.5 Use Case Diagram

This use case diagram represents the system and features it serves to the users. Like signing up, chatting with the chatbot, creating tables, and editing user account data. The following services are shown too. Like showing an error message when the email is invalid.

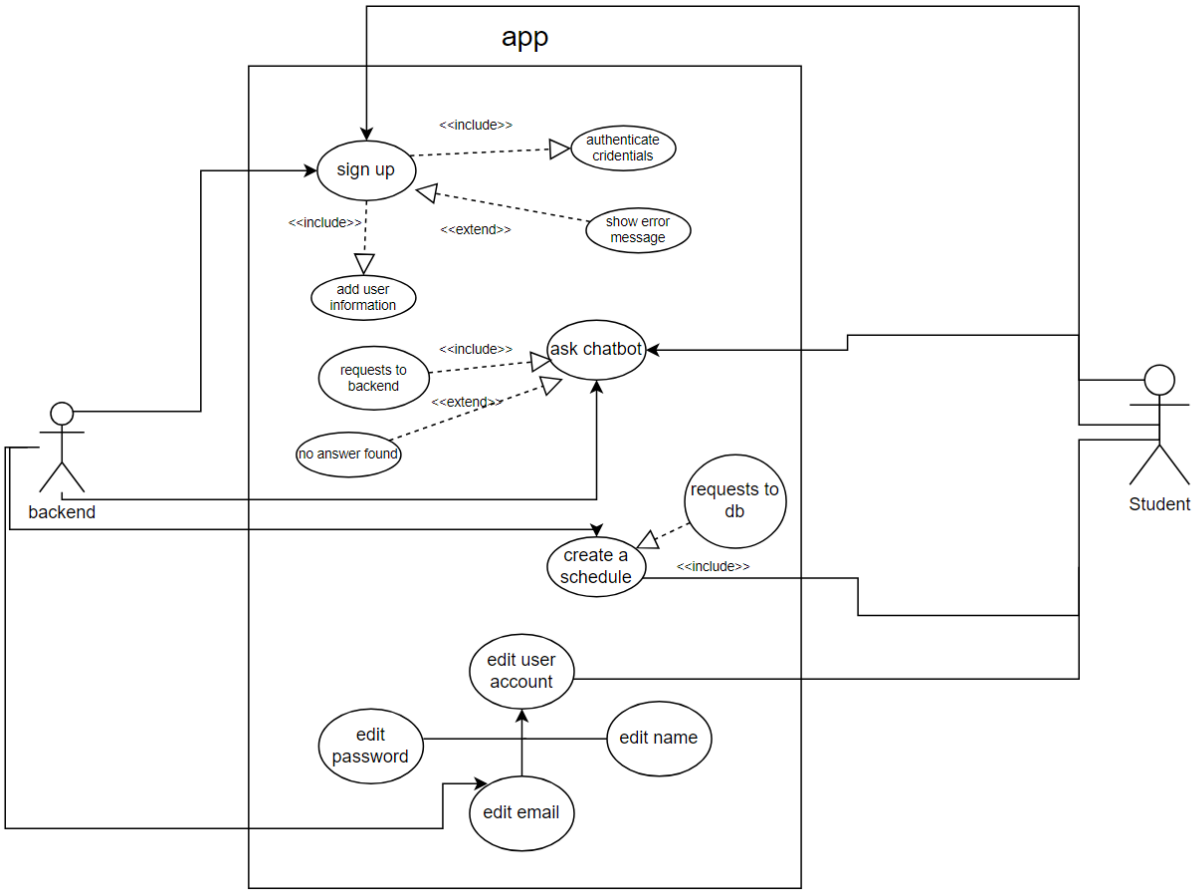


Figure 10 Use Case Diagram

3.1.6 Sequence Diagram

This diagram is more detailed and shows the interactions with the system as sequences or steps so here we consider the timing factor.

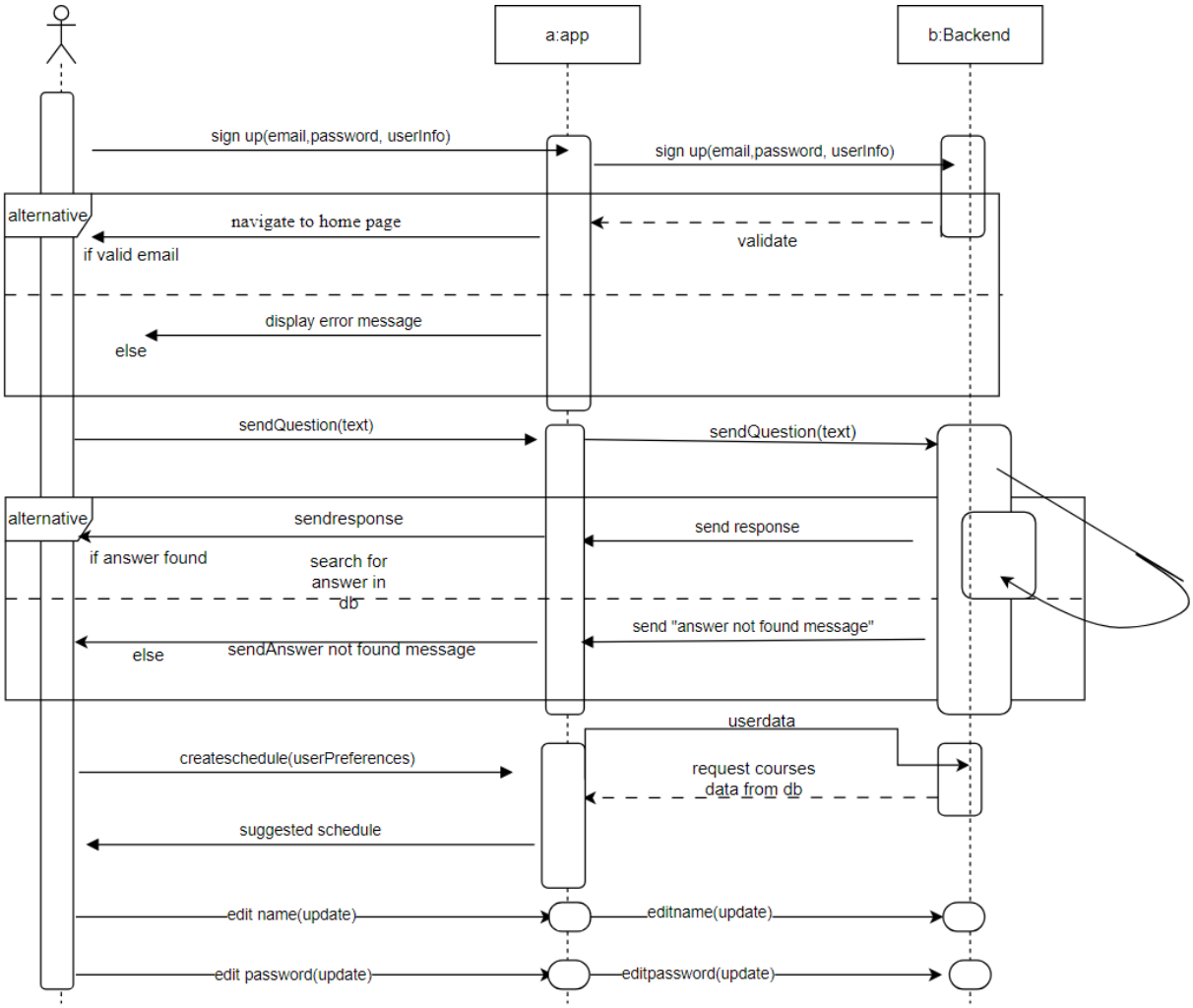


Figure 11 Sequence Diagram

3.1.7 Class Diagram

The class diagram is essential for modeling the relationships, behaviors, and entities that contribute to creating schedules in an efficient, maintainable, and well-organized manner.

The key entities encapsulated within it: the Student class (with attributes such as student ID, name, email, and password), the Passed class (featuring student ID, course ID, and a status indicating

whether the course has been passed or not), the Course class (encompassing course ID, name, number indicates the default semester of the course in the major tree, course credit hours, and the number of prerequisite courses), and the Section class (comprising section ID, course ID, day of the week, starting and ending time, and a status denoting whether the section is open or closed).

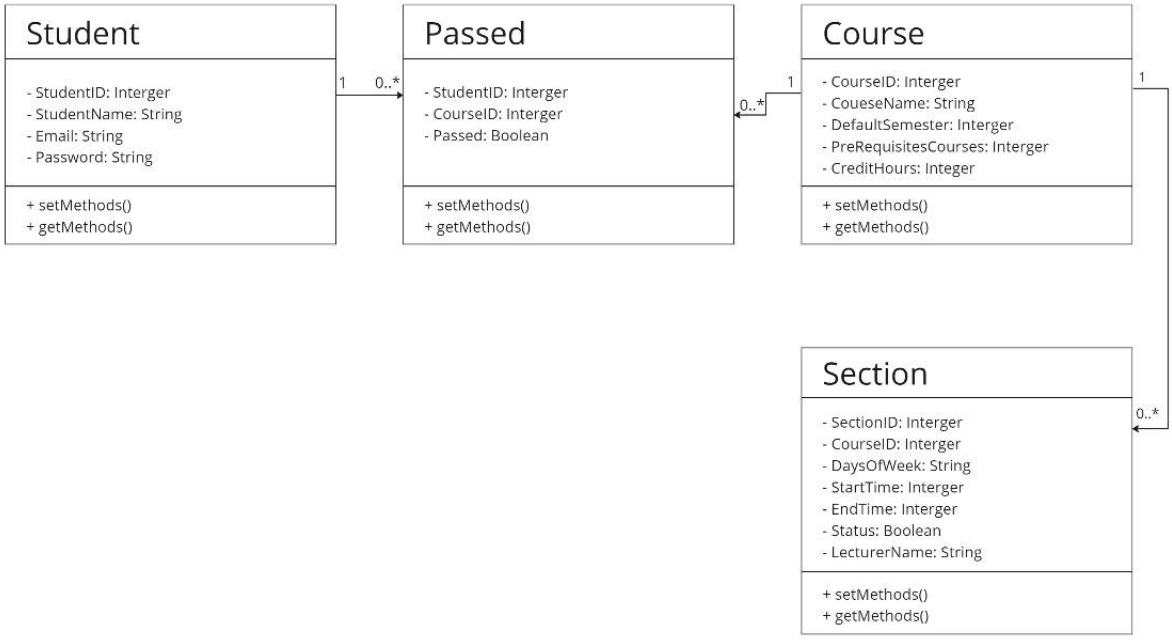


Figure 12 Class Diagram

3.1.8 ER Diagram

The ER diagram is essential for modeling data structure and relationships. It helps visualize the system's structure and provides a basis for designing the database in the schedule generator.

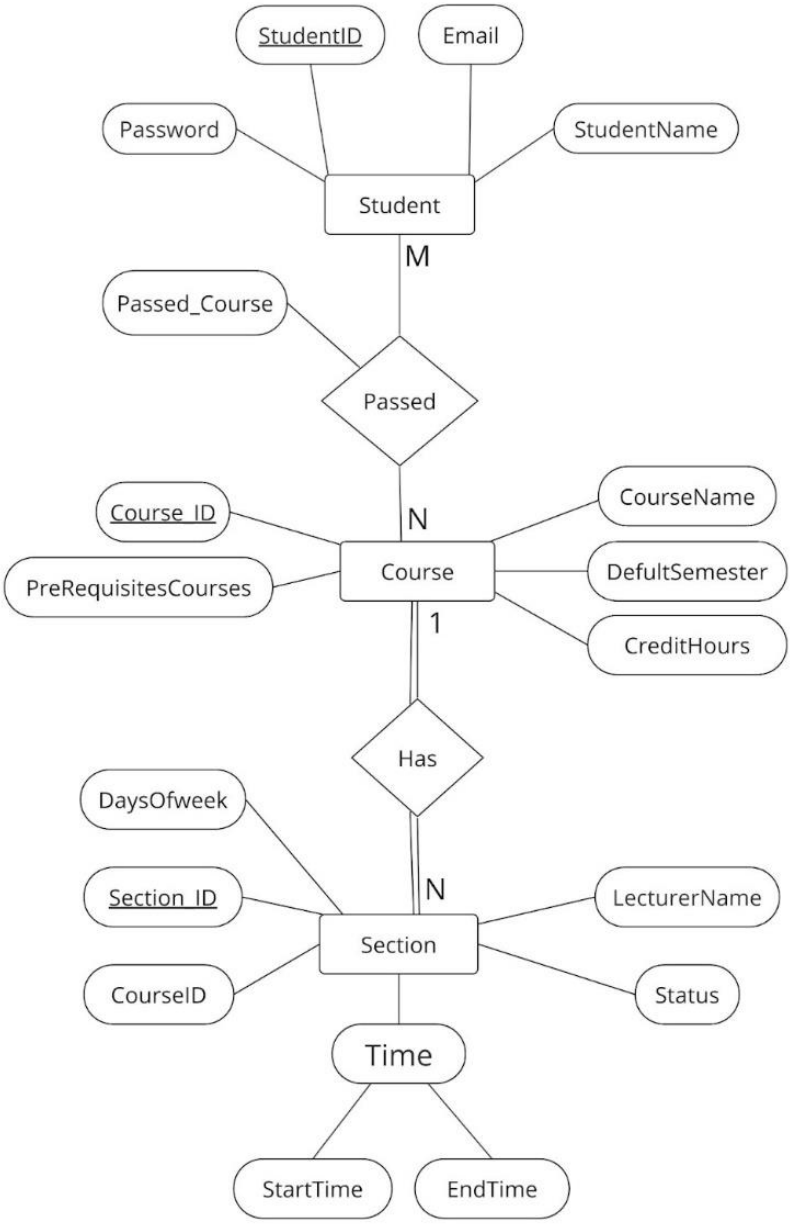


Figure 13 ER Diagram

3.2 Design Specifications, Standards and Constraints

3.2.1 Specification

Our mobile application and the system are intended to make computer-based solutions for two problems:

1. Inquiries helper as a chatbot, this part of the system will analyze messages and questions provided, then try to find the most relative answer to it and send it back to the user.
2. Schedule generator, this part will try to generate the most suitable table of courses for the student in the current semester according to the passed courses and the constraints given by the plan of study and some user preferences.

3.2.2 System Requirements

1. Functional Requirements
    - The system shall provide authentication using email and password, which are specific to our system.
    - The system shall allow the user to edit his/her information.
    - The system should allow the student to ask questions about registration and the university system.
    - The system shall allow the user to set constraints to create the intended schedule.
    - The user should be able to enter passed courses at sign-up and edit it periodically.
    - The system shall be able to reply to Arabic inquiries.
    - The system should consider the courses the student has passed and the major plan

when generating the schedule.

- The system shall consider different weights for the courses.

2. Non-Functional requirements
    - The system should be secure, so other students cannot see the student information.
    - The system shall be able to answer the most frequent questions.
    - The response time shall be reasonable and low latency.
    - The system shall be tolerated by a high number of users at the same time.
    - The system should be scalable to more features and majors.
    - The system shall consider the possibility of integration with the university system in the future.
    - The system should have an easy-to-use user interface and good user experience.

3.2.3 Standards

The schedule generator should obey the courses available by the university and the plan determined by the officials. The chatbot should be polite and not accept swearing. The mobile app design will follow the material app design provided by the Flutter framework. The architectural pattern of the app will be Model-View-ViewModel which is considered by Google.

3.2.4 Constraints

1. The system must obey the university's set of rules and obligations.
2. The chatbot should be informative, helpful, efficient, most time available, and polite.
3. The schedule generator will be available only for computer systems engineering students.

4. The schedule generator will generate tables in one time from Sunday, Tuesday, and Thursday lectures which is 50 min, and lectures on Monday and Wednesday will be 75 min.

3.3 DESIGN ALTERNATIVES

While developing this solution, we considered several alternative approaches and options which were rejected, and our proposed design was adopted. Here is a breakdown of those alternatives and why they were ignored:

1. Chatbot Embedding Approach:

Instead of selecting BERT embeddings, we could have used simpler embedding options such as Word2Vec or GloVe. However, these alternatives cannot capture all meanings in the context of language. BERT's understanding of context is crucial to accurately representing user input, especially with complex or ambiguous queries, and it is better able to handle dialects.

2. Schedule Generation Algorithm:

In our current project, a backtracking approach was chosen to generate the schedule, although more complex algorithms could be used. However, given the scope of the current project, the simplicity and transparency of the backtracking method made it the preferred choice given the way data and student tables exist at the university. It was also easy to integrate into the Flutter environment, which contributed to its selection.

3. Data Storage for Scheduling:

In the case of data storage, we chose to store the data in cloud. This decision prioritized efficiency and responsiveness, especially given the limited size and requirements of the tabulation data. By using cloud storage, we expect improved scalability and accessibility, which is a better solution compared to local storage.

In conclusion, the selected solutions were selected based on a careful assessment of their suitability to the specific needs and constraints of this project, while alternative approaches were explored for each aspect of the design. We believe we have arrived at a robust and effective solution for implementing our system, especially prioritizing factors such as accuracy, simplicity, and efficiency.

3.4 SOFTWARE USED IN THE DESIGN PROCESS

3.4.1 FLUTTER

Flutter is a framework that allows producing applications for multiple operating systems with one dart code base like Android and IOS which allows for faster delivery and easier maintenance. And will allow different platform users to reach the newest features at the same time. [10]

Figure 14 Flutter framework logo [10]

3.4.2 FIGMA

It is a SAAS owned by Adobe that allows designers to develop user interfaces with user experience in consideration, which allows the separation of requirements and design from the complicity of the code. We'll consider these designs as prototypes for the finished app. [11]

Figure 15 Figma website logo [11]

3.4.3 DRAW.IO

It is a web-based application that allows engineers and students to draw different kinds of charts to prototype a process or to show the progress of an algorithm. As an example, charts the flow chart, sequence diagram, use-case diagrams, and UML diagrams. [12]

Figure 16 Draw.io website logo [12]

3.4.4 MIRO

It is an online collaboration platform that facilitates visual collaboration between teams and individuals. Miro allows users to create, share, and edit different types of content, including diagrams, mind maps, and wireframes and enables users to collaborate in real-time, making it an invaluable tool for remote teams and creative projects. [13]

Figure 17 Miro website logo [13]

3.4.5 FIREBASE

Firebase is a product of Google that helps developers to build, manage, and grow their apps easily. It helps developers to build their apps faster and more securely. It has a user-friendly interface, which eliminates the need for extensive programming on the Firebase side, thereby enhancing the accessibility of its features. It provides services to Android, iOS, web, and Unity. Notably, it offers various services, with cloud storage standing out as a critical component. So, Firebase stands as a comprehensive solution, empowering developers to navigate the complexities of app development with ease. [14]

Figure 18 Firebase cloud platform logo [14]

3.5 DATASETS

3.5.1 CHATBOT DATASET

In the Chatbot, the data set consists of two main components: first, the questions, which are the user's queries, and second, the answers, which are the bot's responses.

Question	Answer
كم عدد ساعات تخصص هندسة أنظمة الحاسوب؟	عدد الساعات بالمجمل يبلغ 163 ساعة معتمدة

Table 1 Chatbot Dataset

Data is extracted from the university's Frequently Asked Questions (FAQ) section. This provides a set of questions commonly asked by students. By integrating these questions into the dataset, the chatbot is ready to process queries that align with our university-specific topics.

Also, to enhance the breadth of the data set, we surveyed to obtain specific queries and concerns directly from the students themselves. This ensures that the chatbot is not only knowledgeable about the university but is also compatible with students' needs and queries.

Combining these two sources results in a powerful data set that enables the chatbot to provide accurate, relevant, and up-to-date information to students.

3.5.2 SCHEDULE GENERATOR DATASET

To implement this feature, we need the available sections to be registered, coming from the university registration website (Portal).

CourseCode	CourseName	CrdHrs	ActivityDesc	Section	DeptName	ClassTime	Status
100411010	CALCULUS I	3	Lecture	1	Mathematics and Statistics Department	[08:30-09:20 A&S B114 Sunday Tuesday Thursday]	Open

Table 2 Schedule Generator Dataset

This dataset contains all the information about university sections, for example, we added "CALCULUS I" section "1", Here's an explanation of each column:

- CourseCode: The unique identifier for the course, and in this case, it is "100411010"
- CourseName: The name of the course, which is "CALCULUS I" in this instance.
- CrdHrs: Indicates the credit hours associated with the course, and here it is "3".
- ActivityDesc: Indicates the format of the course (e.g., Lecture, Lab).
- Section: Represents a specific section number associated with the course. Here it's "1".
- DeptName: Specifies the department offering the course. In this case, it is the "Mathematics and Statistics Department."
- ClassTime: Provides details about the class time, including the time range ("08:30-09:20"), class's location ("A&S B114"), and class days ("Sunday Tuesday Thursday").

3.6 System Analysis and Optimization

3.6.1 Chatbot:

We measured the response times for various types of queries to ensure the chatbot delivers quick responses.

We also tested the chatbot's ability to provide accurate and relevant answers to user queries. This included comparing the chatbot's responses with the expected correct answers and calculating the accuracy percentage.

3.6.2 Schedule generation:

We evaluated the time required to generate schedules for different levels of complexity. This analysis involved scenarios with varying numbers of courses, time slots, and constraints to ensure the system's efficiency in handling different scheduling demands.

We iteratively improved the scheduling algorithm to reduce generation times and increase the accuracy of the generated schedules.

3.7 Simulation and/or Experimental Test

3.7.1 Chatbot:

The response time of a chatbot is an important and decisive factor in determining its ease of use and user satisfaction.

To evaluate the response time, we designed a series of tests. The tests included querying the chatbot using a set of questions in the dataset, which represent the questions the student needs. The chat program showed high efficiency, with an accuracy rate of 89%.

3.7.2 Schedule generation:

We designed a series of tests to evaluate the schedule generator. These tests involved creating schedules for a wide range of course options, varying in complexity. Each test scenario included different numbers of cycles, time periods, and constraints such as cycle conflicts and personal preferences. The schedule generator demonstrated high efficiency within acceptable limits for all scenarios tested, giving us a reasonable schedules every time.

4 CHAPTER 4: RESULTS AND DISCUSSIONS

4.1 RESULTS

Here we will introduce some UI designs for the mobile application which represent a prototype for the project. These two pages represent the onboarding screens, “welcome pages”.

Figure 19 Welcome Pages

Here the student can sign in or create an account if he doesn't have one yet.

Figure 20 Sign-in and Sign-up Pages

These pages provide the ability to talk and chat with the smart chatbot about university-specific questions and queries in Arabic as the main language.

Figure 21 Chatbot Page

Here the student should fill in the courses he passed, and then the home page show to allow him to navigate through the app.

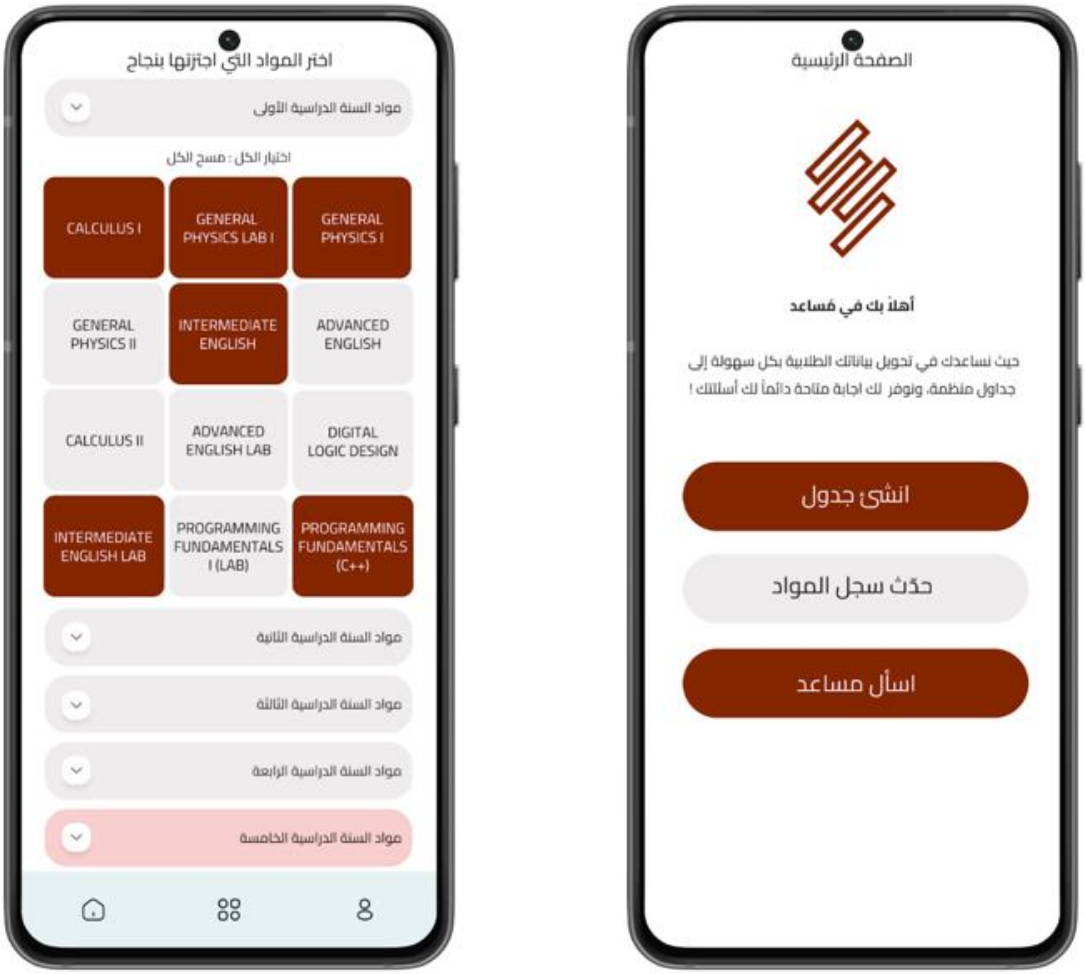


Figure 22 Application's Starting Pages

Here, the student can choose his specific options and configurations to show the table when clicking on the button.

Figure 23 Schedule Generator Pages

This page shows the student's information and allows him to modify it, and he can sign out through this page.

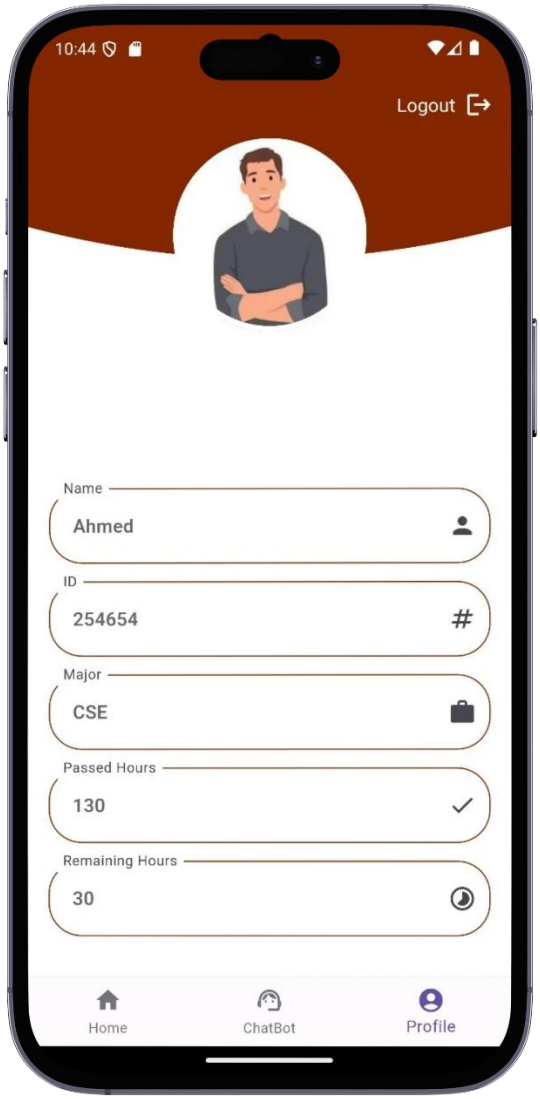


Figure 24 Student Profile Page

4.2 DISCUSSIONS

The results from our simulations and experimental tests show that the "*Chatbot Assistant and Schedule Generator Mobile App for University Students*" performs efficiently and accurately, so this ensures a positive experience for the student.

- As the Chatbot maintained an average response of less than two second, which indicates its efficiency. With the Chatbot's accuracy reaching 89%, indicating reliable responses.
- The schedule generator also handled different levels of complexity efficiently,

These results confirm the strength of the system, and we aspire for further improvement in the future.

5 CHAPTER 5: PROJECT MANAGEMENT

5.1 TASKS, SCHEDULE, AND MILESTONES

The diagrams below illustrate the project schedule, depicting the ongoing tasks and our efficient time allocation. On the right side, a Gantt chart provides a visual representation of the accomplished processes.

Task	Duration	Start Date	End Date
Project Initiation	10	2023-12-02	2023-04-28
1.1 Define Project Scope	3	2023-12-02	2023-12-11
1.2 Establish Project Team	2	2023-12-12	2023-12-13
1.3 Identify Stakeholders	5	2023-12-14	2023-12-18
Requirements Analysis	12	2023-12-19	2024-01-01
2.1 Conduct Stakeholder Interviews	4	2023-12-19	2023-12-22
2.2 Document Functional Requirements	4	2023-12-23	2023-12-28
2.3 Prioritize Requirements	4	2023-12-29	2024-01-01
Collect Data	6	2024-01-02	2024-01-07
3.1 Gather Relevant Data Sources	2	2024-01-02	2024-01-03
3.2 Validate Data Accuracy	2	2024-01-04	2024-01-05
3.3 Organize Collected Data	2	2024-01-06	2024-01-07
Mobile App Development	20	2024-01-08	2024-01-27
4.1 Create UI/UX in Figma	5	2024-01-08	2024-01-12
4.2 Create the app in Flutter	5	2024-01-13	2024-01-17
4.3 Configure backend app (Firebase)	5	2024-01-18	2024-01-22
4.4 Test on different devices	5	2024-01-23	2024-01-27
Building Machine Learning Model	10	2024-01-28	2024-02-06
5.1 Training the model with collected data	5	2024-01-28	2024-02-01
5.2 Testing the model with new data	5	2024-02-02	2024-02-06
Documentation	9	2024-02-07	2024-02-15
6.1 Create Project Plan	3	2024-02-07	2024-02-09
6.2 Document Design Specifications	3	2024-02-10	2024-02-12
6.3 Compile User Manuals	3	2024-02-13	2024-02-15
Testing and QA	9	2024-02-16	2024-02-24
7.1 Develop Test Cases	3	2024-02-16	2024-02-18
7.2 Execute Testing Protocols	3	2024-02-19	2024-02-21
7.3 Address and Resolve Defects	3	2024-02-22	2024-02-24
Project Presentation	6	2024-02-25	2024-03-01
8.1 Prepare Presentation Materials	2	2024-02-25	2024-02-26
8.2 Rehearse Presentation	2	2024-02-27	2024-02-28
8.3 Conduct Project Presentation	2	2024-02-29	2024-03-01
Project Reporting	6	2024-03-02	2024-03-07
9.1 Generate Project Status Reports	2	2024-03-02	2024-03-03
9.2 Communicate Progress to Stakeholders	2	2024-03-04	2024-03-05
9.3 Identify and Mitigate Issues	2	2024-03-06	2024-03-07
Risk Management	9	2024-03-08	2024-03-16
10.1 Identify Potential Risks	3	2024-03-08	2024-03-10
10.2 Assess Risk Impact and Probability	3	2024-03-11	2024-03-13
10.3 Develop Mitigation Strategies	3	2024-03-14	2024-03-16

Figure 25 Project Management Schedule

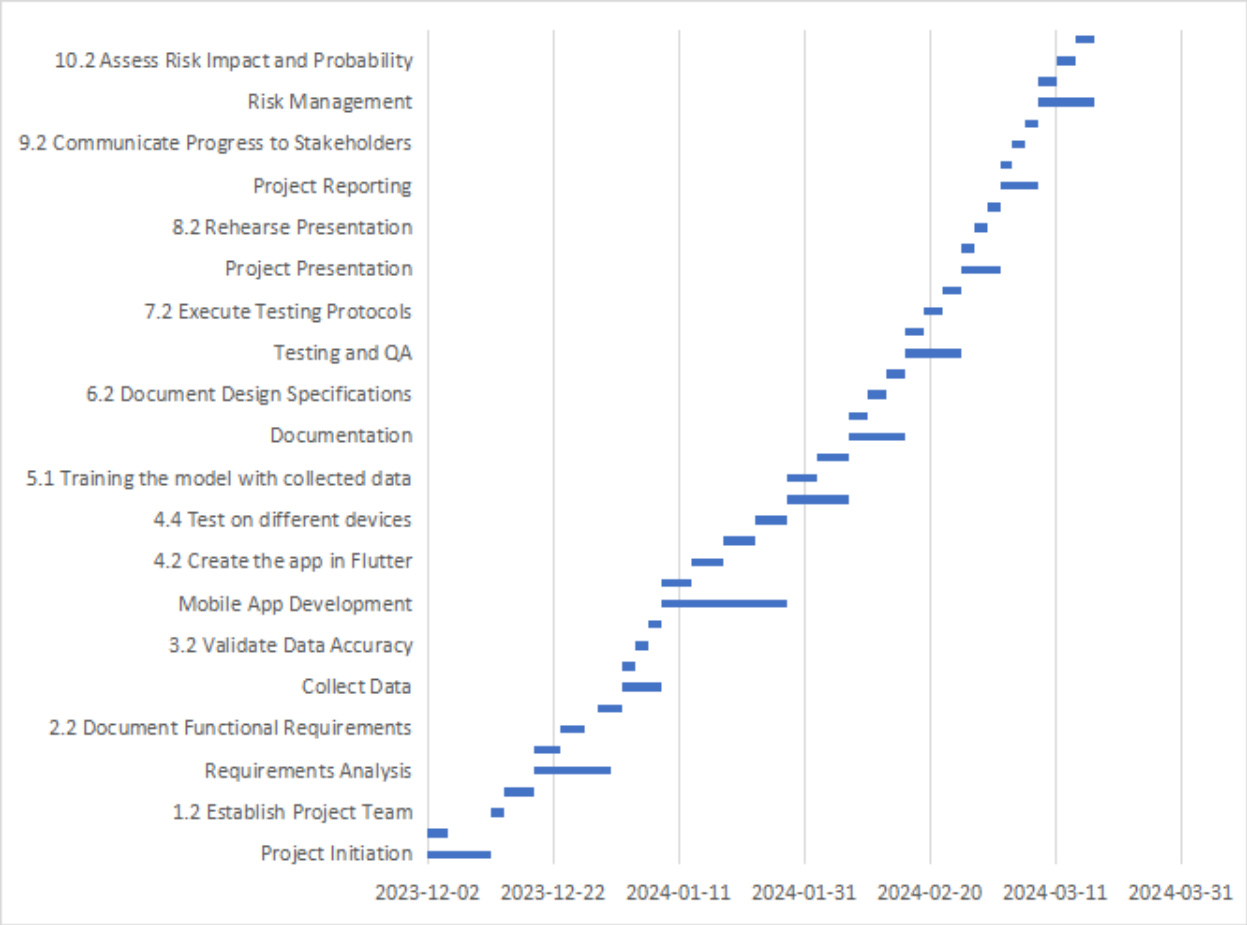


Figure 26 Project Management Gantt Chart

5.2 Resources and Cost Management

Table 3 outlines the projected cost of developing our mobile application, divided into three key stages: design, development, and testing & quality assurance (QA).

App Design stage encompasses the creation of the app's overall visual system, including both the user interface (UI) and user experience (UX) elements, while App Development stage focuses on building the functionality of the app, including developing the individual pages, integrating a chatbot, and implementing a schedule generator.

The final stage, Testing & Quality Assurance, ensures the app functions properly and is free of bugs before launch.

Item	# Unit (hours)	Unit Cost ($)	Subtotals ($)	Comments
App Design	24	20$	480$	Includes designing system and UI/UX
App Development	300	30$	9000$	Includes developing the app pages, Chatbot and Schedule Generator
Testing & QA	45	25$	1$	Includes testing functionality and bugs
Total Estimation	10605$			

Table 3 Resources and Cost Management

5.3 LESSONS LEARNED

During the project, we acquired skills in time management, task delegation among team members, and plan monitoring. We recognized the significance of well-defined requirements to prevent excessive changes and the need for rework, ultimately preventing project delays.

6 CHAPTER 6: IMPACT OF THE ENGINEERING SOLUTION

6.1 ECONOMICAL, SOCIETAL AND GLOBAL

Firstly, it has resulted in time and effort savings for both students and academic staff. The solution has made it easier to schedule courses and retrieve necessary information, thereby enhancing the academic experience for students. It has also led to reduced stress levels during critical periods and increased accessibility to academic courses.

Secondly, the solution provides students with the ability to personalize their schedules based on their preferences and academic requirements. This can be achieved without any conflict in times between different courses, making it easier for students to manage their academic workload. Finally, the engineering solution provides a model for integrating advanced technology into university processes, thereby paving the way for future innovations. Its success shows technology's potential to improve the efficiency and effectiveness of academic institutions globally.

6.2 ENVIRONMENTAL AND ETHICAL

The chatbot's responses are designed to be culturally sensitive, using appropriate language. We prioritize the protection of student privacy, ensuring a secure and trusted environment for all users. To achieve this, we conduct regular security audits to identify and address any vulnerabilities in the system.

6.3 OTHER ISSUES

According to our questionnaire, students encounter difficulties during the course registration process. To address these issues, a chatbot and schedule generator will be implemented to assist students in quickly responding to inquiries related to course registration and details, this will make university life easier for the students and help them utilize their time to achieve academic success.

7 CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS

7.1 Summary of Achievements of the Project Objectives

The chatbot and schedule generator have contributed to creating a more efficient educational environment that is easy to use and helps students get quick replies to their questions. This has eliminated the need for students to waste time and effort on complicated processes such as registration or searching for information. The project's objective was to simplify the educational experience for students, and these tools have successfully achieved that goal.

7.2 New Skills and Experiences Learnt

The project provided a useful educational experience for us, including technical skills and collaborative practices among team members. We also learned to use appropriate algorithms for our project and acquired skills in coordination and writing reports accurately at a high level, which will put the team in a position of continuous growth and the ability to contribute effectively to future projects and challenges.

7.3 Recommendations for Future Work

There are a few areas we can improve on in the future:

Firstly, we can integrate our system with the university's existing system to make it more comprehensive and useful for all university students. By doing this, students will be able to register for their courses easily and rapidly, without feeling overwhelmed. They will also be able to access their inquiries at any time and from any place, without having to wait for the university's registration department or staff.

Secondly, we can improve the algorithm to make it more personalized for each student. This can be done by incorporating the grades of the previously passed courses of each student and predicting better courses based on which subjects are easy or difficult for them. Moreover, we can add more data about the university or general questions about university life, time management, and stress management.

Finally, we can use a more powerful generative models for the chatbot to make it even more effective.

REFERENCES

[1]. Al-Madi, Nagham A., et al. "An intelligent Arabic chatbot system proposed framework." 2021 International Conference on Information Technology (ICIT). IEEE, 2021.

[2]. Alazzam, Bayan A., Manar Alkhatib, and Khaled Shaalan. "Arabic Educational Neural Network Chatbot."

[3]. Bilquise, Ghazala, Samar Ibrahim, and Khaled Shaalan. "Bilingual AI-Driven Chatbot for Academic Advising." International Journal of Advanced Computer Science and Applications 13.8 (2022).

[4]. Nguyen, Trung Thanh, et al. "NEU-chatbot: Chatbot for admission of National Economics University." Computers and Education: Artificial Intelligence 2 (2021): 100036.

[5]. Dinh, Hoa, and Thien Khai Tran. "EduChat: An AI-Based Chatbot for University-Related Information Using a Hybrid Approach." Applied Sciences 13.22 (2023): 12446.

[6]. Ambhore, Shraddha, et al. "Automatic Timetable Generator."

[7]. Alghamdi, Hayat, et al. "A review of optimization algorithms for university timetable scheduling." Engineering, Technology & Applied Science Research 10.6 (2020): 6410-6417.

[8]. Nguyen, Van Du, and Tram Nguyen. "An SHO-based approach to timetable scheduling: a case study." Journal of Information and Telecommunication 5.4 (2021): 421-439.

[9]. Ankit Tomar. Word2Vec vs BERT. Medium. https://medium.com/@ankiit/word2vec-vs-bert-d04ab3ade4c9. Accessed February 10, 2024.

[10]. Flutter. "Flutter: Beautiful native apps in record time." Flutter, https://flutter.dev/

[11]. Figma. "Figma: the collaborative interface design tool." Figma, https://www.figma.com/

[12]. Draw.io. "Draw.io: Online Diagram Software & Visual Solution." Draw.io, https://draw.io/

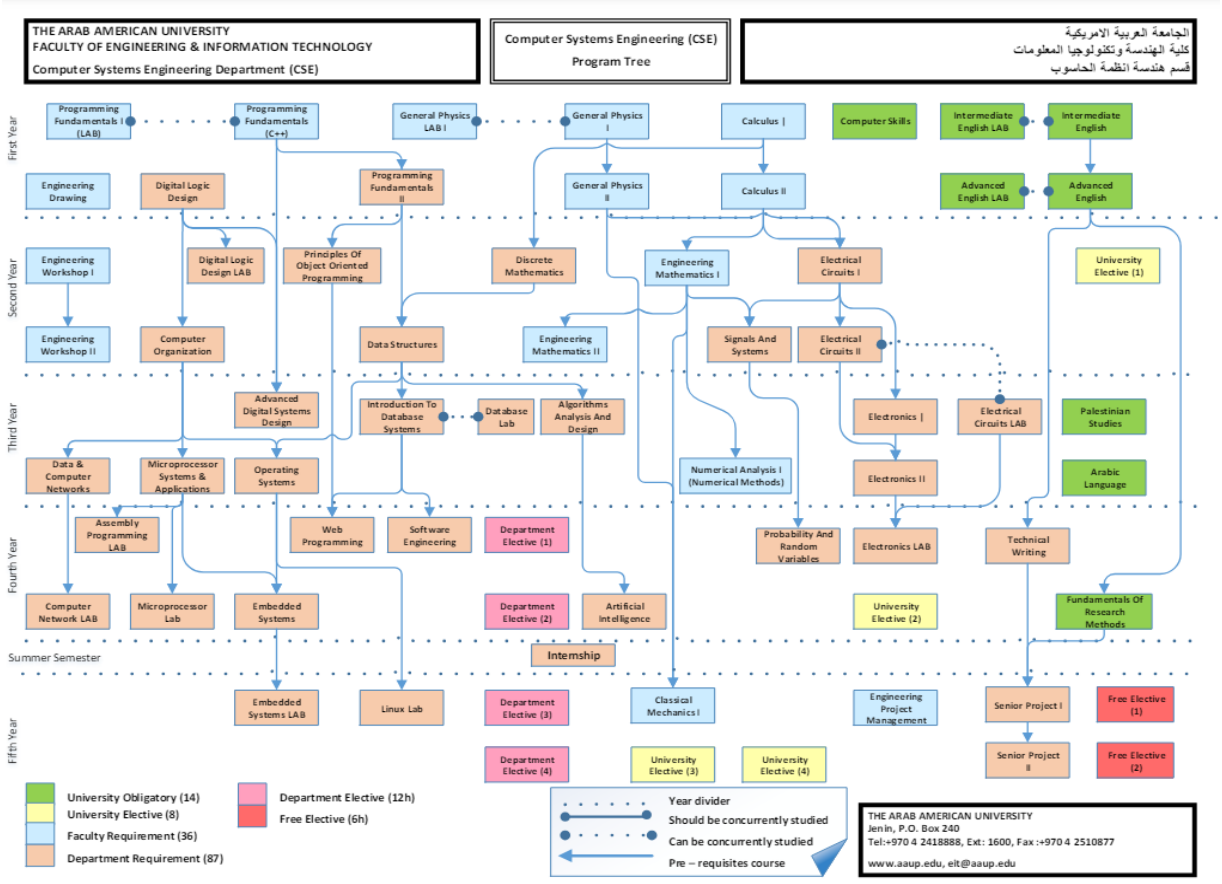
[13]. Miro. "Miro: Online Collaborative Whiteboard Platform." Miro, https://miro.com/

[14]. Google. "Firebase: Build Apps Without Limits." Firebase, https://firebase.google.com/

APPENDICES

Appendix A:

- The Following is the general plan for the Computer System Engineering major, which we used to create and improve the Scheduler Generator for our project:



Appendix B:

- The Following dataset contains questions and answers to assist students majoring in computer systems engineering with their academic and administrative inquiries. The dataset was crucial in developing the chatbot's knowledge and ensuring it could provide accurate and helpful responses to users.

  Musae'd Chatbot Dataset Link