Transforming Student Services: Generative AI-Powered Chatbot for Dar Al-Hekma University

BSCS 4392: Capstone II

By

Fatma Alamoudi	2110348
Rahaf Alshabrawi	2110060
Rihab Asif	2110201
Leena Althekair	2110016

Supervised By

Prof. Imed Ben Dhaou

Department of Computer Science

School of Engineering, Computing, and Design

Dar Al-Hekma University

Jeddah, Saudi Arabia

AY 2024-2025

Acknowledgements

First and foremost, we praise Allah the Most Gracious, the Most Merciful, for aiding us with strength, knowledge, and perseverance that enabled us to complete this research. It would not have been possible without His guidance and blessings.

We would like to express special thanks to Professor Imed ben Dhaou for his great help, continuous support, and wise advice throughout the course of this project. His knowledge and patience with us have contributed a great deal to making our ideas clear and our capstone project a success.

Special thanks to Dar Al-Hekma University for providing the opportunity and avenue to take a deeper interest in this ever-growing topic of AI chatbots. We extend our appreciation to all the staff, faculty, and students who participated in our study and supported us with enlightening criticism. Your input has really helped us in improving and bringing our project up to life.

We acknowledge and sincerely thank the role of our team members, whose collaboration, hard work, and mutual encouragement have made this journey both productive and enjoyable. In the shared exchange of ideas and collective efforts, we now celebrate the long-awaited result.

Our biggest gratitude and thanks go to our beloved parents, families, and friends, who continue to support us with patience and inspiration. The first thing that helped us see this project through is your belief in us. May Allah preserve you for us and strengthen our relationships for many years to come.

This is indeed a testimony of everyone's combined efforts; thank you very much.

Abstract

In modern university settings, students often face challenges accessing essential information and resources efficiently. This project proposes the development of a generative-ai powered chatbot as a part of a university app using Flutter to address these issues. The app includes the interactive chatbot for answering FAQs, providing personalized support in addition to a centralized event calendar to reduce email load, delays, and confusion. By leveraging Flutter's cross-platform capabilities and OpenAI's ChatGPT, this app aims to create a user-friendly, reliable solution to streamline student experiences. This project highlights the potential for technology to improve educational environments and enhance student productivity.

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter 1
Introduction

Chapter 1: Introduction

1.1 Project Background

With the rapid growth at Dar Al-Hekma University, demands for students' services are increasing, which up to now have been handled conventionally through email and personal contact. The traditional system often results in delays, increased administrative burden, and student dissatisfaction. This project responds to these challenges by offering a chatbot that supports student support and administrative efficiencies and, therefore, elevates the student experience.

1.2 Problem Definition

With universities changing, they become complex and require quicker and speedier student support. Traditional ways of offering student services rely so much on emails and physical visits to offices, making them slow to respond to student queries. This is particularly evident in the case of students asking about course registration, university events, and other related university services. The need for automated support solutions has been heightened by the transition to digital systems (Følstad & Brandtzaeg, 2017).

The main problem is that the existing student support systems rely too much on email and are not effective in addressing the different needs of students. On the other hand, relying on human staff for routine inquiries burdens them with more work, hence limiting their attention to more complex matters (Hill et al., 2015). As a result, students are often frustrated with the slowness of responses, especially at times of registration and examination periods (Robles et al., 2019).

Dar Al-Hekma University faces similar problems. Many students said that they are facing some difficulties in acquiring information at the right time, made worse by the lack of a single point that would be able to handle queries, announcements of events, or administrative support. Furthermore, the current use of dependent methods of announcements via Outlook email is overwhelming because students neglect their emails due to the high capacity of emails received and sent daily. This is the gap between the needs of the students and the services offered, which affects both student satisfaction and academic performance.

1.3 Project Objective

The purpose of this capstone project is to meet the expectations of the students with direct support from the university, using generative AI chatbot technology. This study will inform the creation of this chatbot in such a way that it directly solves real pain points for students and generates an easy-to-use and seamless platform to access university services.

1.4 Project Scope

This project focuses on the development of a university-based mobile application for Dar Al-Hekma University that integrates an AI-driven chatbot, event calendar, and administrative services. Core functionalities will involve answering routine student queries on finances, registration, and university events, along with real-time event updates. The project will focus on MVP development and therefore not include advanced features involving multi-lingual, living interaction, or advanced data synchronization. The integration with existing databases of students is out of scope at this point and enabling voice interaction in such systems. More details regarding the potential future work are detailed in chapter 8.

1.5 Suggested Solution

The development of an AI chatbot mobile application for solving these issues of Dar Al-Hekma University will be the solution. The proposed chatbot development should be such that it deals with routine inquiries and timely, relevant support to the students. Qualitative feedback from the university community and quantitative data from surveys will help in customizing the chatbot to the specific challenges faced by students. It has the general objective of enriching student experience while easing the administrative burden. The application shall be embedded with a static calendar comprising all the events that are to take place at Dar Al-Hekma. These will be categorized according to school, community service, location, and others. The application would bring an overall relief to the email burden while introducing a unique and pleasant student experience.

1.6 Project Feasibility Study

1.6.1 Success Criteria

The project will be successful:

1. If the user will be able to ask the chatbot financial, registration, or administrative questions and get a correct answer.
2. If the user will be able to ask the chatbot specific questions and, using the user data, get a correct response based on their personal details.
3. If the project achieves its target to effectively declutter email traffic and reduce the number of emails sent daily and weekly.
4. If the user can see all events happening all around Dar Al-Hekma, effectively with their dates, locations, and relevant information.

1.6.2 Completion Criteria

The project is expected to deliver a centralized mobile application for Dar Al-Hekma University that integrates academic, administrative, and extracurricular services into one user-friendly platform.

- Access a **centralized event calendar** with real-time updates on academic and extracurricular activities.
- Use an **AI chatbot** for administrative assistance and personalized question answering.

1.6.3 Limitations

The primary challenges of this approach include a dependency on pre-trained embeddings that may not capture niche knowledge perfectly, potential latency and performance issues with Firebase when storing embeddings at scale, and significant costs associated with OpenAI's API usage and other third-party tools.

The initial minimum viable product will not include advanced natural language features such as multi-turn conversations, real-time database synchronization or live chatbot-human integration, nor will it support multilingual or voice interaction capabilities.

Technical Limitations

Embedding and Search

- Accuracy of Embedding Models:
    - Pre-trained embedding models (like OpenAI's) may not perfectly understand highly specific or niche university terms, jargon, or uncommon abbreviations.

    - Fine-tuning this model could help, but this requires significant resources and is out of the scope for this project due to time and budget constraints.
- Search Challenges:
    - If the university documents are poorly structured or contain overlapping information, even an excellent similarity search might retrieve irrelevant or incomplete chunks. There would have to be documentation revisions by each department to ensure correct embedding.

Authentication and Data Security

- Microsoft Authentication:
    - Configuring secure authentication through Microsoft can be complex, particularly handling OAuth2 tokens and session management, which requires management from the university's Azure center. Since we do not have access to the control panel, real authentication cannot be employed.
    - Ensuring proper protection of user data (e.g., encryption) is essential to avoid vulnerabilities. Employing high-quality cybersecurity practices is limited by the budget.

Integration Challenges

- Embedding Size and Storage:
    - Firebase is not optimized for handling vector searches or very large datasets. If the dataset grows or the university expands, storing embeddings in Firebase and using FAISS indexing could lead to latency or performance bottlenecks.
- Dependency on APIs:
    - The OpenAI API has token usage limits and costs associated with heavy usage.
    - Downtime or rate limits on APIs (e.g., OpenAI or Microsoft authentication) could affect the app's performance.

User Experience Limitations

Chatbot Understanding

- Limited Knowledge Scope:

    - The chatbot's accuracy is limited by the quality and comprehensiveness of the documents provided by the various departments in the university. If the dataset is incomplete or outdated, the chatbot will provide incomplete or incorrect answers.
- Ambiguity in User Queries:
    - Users may phrase questions in unexpected ways. Without advanced techniques like rephrasing or clarification, both of which require extensive testing time and budget, the chatbot might fail to understand the intent.

Response Generation

- Lack of Real-Time Updates:
    - If university policies or calendars change, there might be a delay in the backend engineers updating the dataset and retraining the embeddings.
- Generic Responses:
    - Even with embeddings, GPT-4 may sometimes give vague or generic answers if the retrieved document chunk doesn't fully match the user's intent. The first-sent instructions text file can mitigate this issue mostly but not completely.

Resource Limitations

Team Expertise

- This is our first chatbot implementation project, and learning curves for tools like LangChain, Firebase, and Flutter might slow progress.
- Complex tasks like setting up embedding pipelines or handling vector similarity searches could require more time than expected.

Time Constraints

- Training, testing, and refining the chatbot, as well as integrating all the components, exceeds the time allotted for the capstone project, which is a semester for implementation.

Budget Constraints

- Using OpenAI APIs for embedding and response generation can become costly depending on the volume of queries. This will limit the amount of testing and queries we can make.
- If the project needs to scale beyond Firebase, transitioning to a paid service like Pinecone or a dedicated server might exceed the budget.

1.6.4 Deliverables

The application will produce the following deliverables to address the identified challenges and meet the requirements:

1. Centralized Mobile Application: A fully operational app that integrates administrative and extracurricular services into a single platform with real-time data synchronization.
2. AI Chatbot: A built-in chatbot capable of answering administrative queries, assisting with campus navigation, and providing personalized academic support.
3. Event Calendar: A centralized calendar with real-time updates on extracurricular activities.
4. User Interface and Multi-Language Support: A user-friendly interface accessible in both English and Arabic to accommodate the university's diverse community.
5. Documentation: Comprehensive user guides and training materials to help staff transition into using the app and user-friendly videos for students, faculty, and staff to understand how to utilize the app effectively.

Some of these deliverables are not included in the demo due to time and resource constraints and are included in the future work recommendations of this project.

1.7 Project Plan

The project follows a structured plan divided into two main phases: capstone 1 and capstone 2. Capstone 1 includes initiation, requirements gathering, and design, while Capstone 2 includes development, testing, and deployment.

As such research was done in two main stages, work began in the first stage “Capstone I” through the systemic collection of data, which directly informed the design and functionality of the chatbot in the second stage “Capstone II”. The selected research design mirrored the mechanistic nature of our work, since we are operating under the waterfall model. Here, each step of the research and development process is undertaken in sequence, ensuring that each stage is based on the findings of the previous one.

1.7.1 Gantt Chart

The Gantt table below outlines the project timeline, highlighting key phases and milestones from initiation to deployment.

TASK	ASSIGNED TO	START	END
Initiation			
Choosing a project idea	all	8/25/24	9/8/24
define departments and faculties to speak to	Rahaf	9/9/24	9/16/24
design interview questions	Leena, Rihab	9/9/24	9/16/24
literature review	Fatma	9/9/24	9/23/24
create short-form distributable survey	Leena	9/9/24	9/16/24
schedule interviews with department assistants (email)	Leena	9/16/24	9/20/24
schedule interviews with administration	Rahaf, Fatma	9/16/24	9/23/24
interview students and departments, distribute survey	all	9/16/24	9/30/24
problem definition	Fatma	9/30/24	10/7/24
staeholder analysis	Rihab	9/30/24	10/7/24
methodology	Rahaf	9/30/24	10/7/24
organize + analyze survey results	Leena	9/30/24	10/7/24
project pitch presentation	All	10/5/24	10/8/24
Planning and Design			
functional/nonfunctional requirements table	Leena	10/7/24	10/14/24
Investigate OpenAI keys, database, security, entire chatbot framework	Leena, Rahaf, Rihab	10/14/24	10/28/24
UX/UI design figma (Tabs: Events, Chat, Settings)	Fatma	10/14/24	12/2/24

Midway follow-up for tutorial and prototype	All	10/21/24	10/21/24
introduction, title page, exec summary …	Rahaf	11/1/24	11/18/24
system design and modeling diagrams (ER, class, use-case, activity, system sequence)	Rihab, Rahaf	11/20/24	12/2/24
define limitations + out of scope ideas and services	Leena	11/20/24	11/30/24
cost/price estimation and breakdown	Leena	11/20/24	11/30/24
assumptions made (Continuous)	Fatma	11/20/24	11/30/24
final report	All	11/20/24	12/14/24
final presentation	All	12/1/24	12/14/24
Execution			
Explore calendar database options	Leena, Rahaf	1/19/25	1/31/25
Experiment with LangChain and ChatGPT API	Rihab	1/19/25	1/31/25
Test different chatbot types (LangChain, ChatGPT, Copilot …)	Rihab, Fatma	2/16/25	3/9/25
Define script to save document vector forms to local folder + link to chatbot	Rahaf	3/11/25	3/19/25
Create prototype (demo) using Flutter (chatbot + calendar)	Rahaf	3/12/25	4/20/25
Test calendar Python + JSON setup and link to Flutter	Rahaf	2/16/25	3/19/25
Document implementation steps and trials	Leena	1/19/25	3/19/25
CCTS Conference Poster	Rahaf	3/12/25	4/2/25
UoJ Digital Innovation Hackathon Presentation	All (Leena)	5/7/25	5/8/25
Continuous unit and section testing	Rahaf	4/1/25	5/11/25
Evaluation			
Evaluation and analysis of testing processes	Leena	4/8/25	5/8/25
Create final capstone report	Leena	1/19/25	5/26/25
Gather feedback via user testing	Leena, Rahaf	4/20/25	4/24/25
Create final capstone presentation	All	5/11/25	5/22/25
Exhibition Poster Creation	Rahaf, Leena	5/18/25	5/24/25

Table 1: Gantt Chart Content

1.8 Conclusion

This chapter provided a comprehensive overview of the project, including its background, problem definition, objectives, scope, and the suggested solution. It outlined the challenges faced by Dar Al-Hekma University in providing timely and efficient student support and emphasized the necessity of transitioning to a modern, AI-driven platform. The proposed AI chatbot mobile application aims to streamline student services, reduce administrative burdens, and enhance overall student satisfaction.

By detailing the project's feasibility, success and completion criteria, limitations, and deliverables, this chapter establishes a clear foundation for the capstone project. It highlights the critical need for a centralized and responsive support system while acknowledging technical, financial, and time-related constraints. The structured project plan ensures a methodical approach to research, development, and deployment. This introduction sets the stage for the subsequent chapters, which will delve deeper into the research, design, and implementation of the solution.

Chapter 2

Literature Survey

Chapter 2: Literature Survey

2.1 Introduction

The growing complexity of the university system as well as the growing expectations for quality student support, continues to render the administrative heavy-handed model unsustainable. Generative chatbots come to the rescue in this regard as they are able to automate low value tasks, enable such tasks as responding to frequently asked questions, enhancing communication, and improving the student experience. In this chapter, the authors of the paper review existing literature in order to investigate the impact and effectiveness of chatbots in education and specifically seek to understand the benefits they bring, their limitations as well as ethical considerations.

2.2 Background

Chatbots that are generative utilize natural language processing (NLP) and machine learning techniques to provide human-like conversations. These chatbots have been developed to meet different functions in educational settings such as providing extra academic help and performing administrative tasks. In this way, reading of existing sources is aimed at understanding the basic concepts and modern chatbot approaches in education that focus on enhancing the ability to address issues of access and inclusivity.

Generative chatbots stand to broaden the notion of accessibility to students who have some form of disability. As an example, for a student who is visually impaired, he or she can interact using audio, while for a student with a learning disability, he can use other forms of communication. Certain applications such as providing text-to-speech for lecture notes and providing course materials in different forms have been helpful in increasing accessibility.

In a similar fashion, chatbots create a value-added approach to education so that the student's needs can be addressed cooperatively. They offer the materials, resources, directions, and links to articles tailored to the learning type, pace, preference and requirements of the student. Evidence and case studies as expounded by Ifelebuegu et al. (2023), provide an insight into how such tailored approaches benefit students in terms of focusing on their academics and enhancing the engagement in class.

Apart from these academic and administrative functions, chatbots are also useful for patients with immediate access to mental problems. For instance, they may help in providing

solutions through behavioral techniques, recommending services such as counseling, or providing self-help tools. That said, such support needs to be reconsidered for its ethical aspects including concerns about privacy and for the capability of AI to reasonably address advanced mental problems (Williams, 2024)

2.3 Related Works

The application of AI-driven chatbots is gaining traction, especially within educational contexts. A substantial body of research highlights how these tools can revolutionize student experiences by providing on-demand support and automating administrative tasks. For example, studies have demonstrated that AI chatbots facilitate communication by promptly addressing frequently asked questions, resulting in significant time savings for both students and staff (Alavi et al., 2022; Jia &Xu, 2021). Moreover, chatbots contribute to student engagement by sending reminders regarding important events, deadlines, and available services (Woolf et al., 2018).

Central to the functionality of these chatbots is Natural Language Processing (NLP), which enables machines to comprehend human language (Hirschberg & Manning, 2015). The increasing adoption of NLP-based chatbots in educational institutions signifies their importance in managing student inquiries and providing academic guidance (Zhou et al., 2020). Major tech companies, such as IBM Watson and Microsoft Azure AI, have developed chatbots to address administrative tasks, including university admissions and course registration, significantly relieving the administrative burden on staff (Følstad & Brandtzaeg, 2017; Hill et al., 2015).

In addition to NLP, the integration of machine learning (ML) allows chatbots to improve over time by learning from previous interactions, thereby becoming more intuitive and capable of delivering enhanced responses (Shum et al., 2018). Recent research has focused on enhancing chatbot intelligence through context-awareness, enabling them to understand not only the words in a query but also, the intent behind it (Colombo et al., 2020). This capability is especially critical in academic environments, where inquiries can range from simple administrative questions to complex academic-related issues (Koch et al., 2021).

Empirical studies consistently indicate that AI chatbots positively impact student services. Woolf et al. (2018) found that students utilizing chatbots experienced faster response times and higher satisfaction levels when interacting with university services. Similarly, Robles et al. (2019) demonstrated that universities implementing AI chatbots experienced a notable reduction in staff

workload and improved student retention. However, while chatbots are advancing in sophistication, they still occasionally struggle to fully comprehend context, leading to inaccurate or frustrating responses (Colombo et al., 2020).

Ethical concerns regarding data privacy also arise with the deployment of AI chatbots, as they often collect sensitive information from students (Kim et al., 2021). Addressing these concerns necessitates further research into privacy-preserving techniques and enhanced data governance.

The potential of AI chatbots in education is substantial; however, their success hinges on how well they are designed to meet student needs. Research indicates that employing both qualitative and quantitative methods can assist in tailoring chatbot functionalities to align with students' actual requirements (Alavi et al., 2022; Shum et al., 2018). As Koch et al. (2021) suggest, chatbots that evolve based on user feedback are significantly more effective at addressing student concerns.

In Saudi Arabia, there are universities that have implemented either chatbot services or provided student applications comparable to the proposed project, but none cover the same scope or use the same technology.

King Fahd University of Petroleum and Minerals has shared via their official news portal in February 2025 a planned collaboration with OpenAI to launch “ChatGPT Edu, a dedicated AI platform designed to enhance learning, research, and innovation within the university’s academic ecosystem”. While utilizing ChatGPT, this collaboration does not produce a chatbot that helps student inquiries. It is more focused on research and academic supercharging. Another project at King Saud University shares the idea of a Blackboard support chatbot that can assist students in navigating the website as well as offering personalized reminders and support (Alqahtani & Alrwais, 2023). While closer in scope, it is more focused on the service and on tailored student homework and assignment support.

Princess Noura University and Effat University both have a “student life” themed app, where students can log in and see all their information, browse campus services, and register in courses, among other things. Both mobile applications offer a comprehensive view of university and campus life and while catering to students; however, they do not offer a chatbot or a single point of truth [27, 28].

In the table below, the differences between the existing solutions and the proposed project are discernable.

	Effat University	KFUPM	PNU	KSU	AskDAH (This Project)
Generative AI	❌	✅	❌	❌	✅
User Context Aware	✅	❌	✅	✅	✅
App	✅	❌	✅	❌	✅
Event Calendar	✅	❌	✅	❌	✅
Academic Guidance	❌	❌	❌	✅	✅
Integration Potential	✅	Limited	✅	✅	✅

Table 2: Comparison Between This Project and Other Universities

2.4 Conclusion

The results of the literature review indicate that the use of AI chatbots, customized according to the needs of university settings, will lead to improved and more satisfying experiences for students while boosting their academic performance. However, ethical risks related to privacy, bias, and the shifting dynamics in teacher-student relationships should also be considered. By adhering to ethical guidelines, continuous evaluation, and incorporation of all stakeholders, an institution can leverage the benefits associated with chatbot technology and, at the same time, reduce challenges. The proper balance is required to achieve such responsible development and effective integration of generative chatbots into higher education.

Chapter 3

Data Collection and Analysis

Chapter 3: Data Collection and Analysis

3.1 Introduction

The purpose of this chapter is to describe how data collection and analysis is done as part of our research methodology. Our approach in the research study involved a mixed methodology that incorporates qualitative and quantitative data to ensure that we had a functional and user-oriented AI chatbot for Dar Al-Hekma University. It explains the justification for this strategy, the specific data gathering methods used, and the analytical approaches taken to interpret the collected data. Our study's participants are varied and include academics, administrative workers, and students who reveal important insights into the needs and difficulties facing the university community. The results would serve as a foundation for the subsequent phase's design and for the operation of the suggested chatbot.

The choice of a mixed-methods approach results from the complexity of our research questions which require the understanding of both the participants' qualitative experiences and the statistical needs of the broader student body.

By means of discussions with chairpersons of the various departments of the university, faculty, administrative staff, and students, the main goal was to achieve an in-depth analysis of the problems which trouble the university community. These talks yielded rich, contextualized insights into the issues that the chatbot needs to solve, such as frequently asked questions, recurrent administrative bottlenecks, and student pain points regarding the timely access of information. By employing semi-structured interviews, it was ensured that the main themes were delved into and simultaneously permitted the subjects to introduce new insights that otherwise would not have been discovered.

Moreover, a survey was conducted with a larger number of students. The survey disclosed the quantitative data of student demands for a chatbot, types of agreements they commonly end up with, and their satisfaction degree with current university services. The survey offered us valuable, valid information about the university's needs, as well as corroborating the qualitative outcomes of our interviews.

3.2 Data Collection

The two primary methods of data collection in this research involve interviews and online surveys. These tools were selected to ensure depth in the exploration of the wide range of needs and perceptions within the university community.

Interviews were conducted with department heads, staff, and students. The interviews were set up to determine the areas where the participants require assistance and the hypothetical questions that they frequently have. The questions were crafted to find out the most common problems the participants face, what methods they are using to get guidance, and how additional help could improve their process.

3.2.1 Interview Analysis

Key informants, namely departmental chairs and administrative personnel, were subjected to semi-structured interviews (available in appendix A.) The interview protocol was aimed at exploring qualitative insights into challenges that are characteristic of university communications and related technological needs. The total number of interviewees was 20.

The key findings include:

1. Fragmentation of Communication:
    - Poor information dissemination.
    - Student-advisor connections were sufficient via email.
2. Navigation and Accessibility Issues:
    - Locating faculty offices and lecture halls.
3. Student Barriers to Accessing Information:
    - Little to no knowledge from students about services offered.
    - Confusion in terms of POS, scheduling, and courses.

The interview data points toward a compelling need for an integrated, intelligent solution for communication that can centralize access to information, give real-time personalized advice and ease administrative transactions

3.2.2 Questionnaire Analysis

A survey was also conducted online for a more relaxed way of gathering information to collect quantitative data from a larger student pool. The survey, available in appendix A, was conducted with a total of 94 respondents. The survey proceeded with questions relating to the kinds

of inquiries made by the students, their inability to get instant responses, and which application features (e.g., FAQs, event notifications) they find most useful. Key findings revealed that students maintained a diverse set of expectations from the AI chatbot. The key findings include:

1. Email Overload: Students report that critical information is often lost in the clutter of emails, resulting in missed events and administrative deadlines.
2. Fragmented Platforms: Multiple systems (SIS, Blackboard, Outlook) force students to check several platforms for updates, leading to confusion and inefficiency.
3. Navigation Challenges: Both new and existing students face difficulties navigating the campus, particularly in finding classrooms and offices.
4. Lack of Real-Time Updates: Outdated information and poor notification systems create uncertainty about academic schedules, event updates, and administrative changes.

Students have expressed a strong desire for a centralized platform that contains all essential information. This includes event updates, student handbook rules, and details on administrative processes. Additionally, users expect the AI chatbot to offer a wide range of features aimed at improving their academic experience. These include answering administrative queries related to registration and schedule issues, assisting with campus navigation, providing event notifications, and managing personal academic data such as GPA tracking.

In all, the results show a clear appetite for an all-encompassing and intelligent digital assistant which will make university life easier and smoother in terms of accessing information and services. Many students view the AI chatbot as a "university assistant" that can save time, reduce administrative confusion, provide instant, efficient responses, and help manage academic and campus life.

3.2.3 Ethics

Regarding the ethical aspects of the research, all subjects were given a detailed explanation of the study's target, the procedures, the possible risks and benefits before the study was undertaken so they could make an informed decision on whether they wanted to participate. Every piece of data that was gathered within the research process was kept confidential. No names or (kinds of) data that could make participants identifiable were used or attached in this paper.

Consequently, the users' privacy is looked after, and their data is entirely secure. The act of becoming involved in the study is explicitly a voluntary action, and the subjects had the

opportunity to discontinue at any time without any negative consequences or problems. The research group adhered to the proper data protection rules and protocols to ensure the privacy of the participants which is secure and the data that is saved during the research is also secure.

3.2.4 Limitations

Even though the research strives for a comprehensive point of view, there are still some restrictions that come with it.

One of these limitations includes the chance that participants might display response biases in the interviews and surveys conducted. The respondents may feel the need to give the most socially acceptable answers or may not fully expose their problems.

Since this paper is specific to Dar Al-Hekma University, the study may not apply to other educational institutions. Nevertheless, self-critical discussions will enrich our university's particular situation that we are aiming at.

Due to security and privacy purposes, full access to the university's Microsoft Azure service is not possible. This prevents the project from fully integrating with the university ecosystem as intended.

3.3 Requirements

Requirements specification involves describing the functionalities of the system in detail, providing a structured framework for development. These are divided into functional and non-functional requirements for clarity and comprehensiveness of the system’s goals. Functional requirements define what the system is supposed to do in terms of user needs, such as chatbot responses, class navigation, and event notifications. Non-functional requirements define performance, scalability, usability, and reliability, ensuring the system operates efficiently under all circumstances. These requirements were developed based on insights gained from the survey and interview analysis outlined in the previous section, providing a foundation for designing a system that meets both user needs and implementation feasibility.

3.3.1 Functional Requirements

The functional requirements of the university app outline the core services and features it must provide to meet user needs effectively. At the heart of the system is the AI-powered chatbot, designed to address a wide range of administrative and academic queries. This includes providing

information on registration dates, financial aid, professor details, and academic processes like adding or dropping courses, program of study (POS) inquiries, and addressing graduation delays.

Another key feature is the centralized event calendar, which provides real-time updates on extracurricular and academic events. Events are categorized by either department, school, or community service hours opportunities, allowing users to easily locate relevant activities. This ensures that students and staff remain informed about happenings across the university.

Finally, to enhance personalization and accessibility, the app supports a dark/light mode feature. While optional, this customization allows users to adjust the interface's brightness and color scheme to suit their preferences, improving the overall user experience.

The table below details the main functional requirements of the mobile application.

Function	Operations	Mandatory (M) or Optional (O)	Comments & Assumptions
AI Chatbot	Ability to answer common administrative queries (e.g., registration, financial aid, professor information). Ability to answer common academic queries (e.g., dropping and adding courses, POS, graduation delay). Study schedules and GPA calculator.	M	The chatbot will have access to the general information of the user via the Microsoft login feature.
Centralized Event Calendar	Real-time event updates and notifications. Centralized display of extracurricular and academic events. Divided by department, school, community service	M	Application admins will be diligent in updating this weekly.
Dark/Light Mode	Allow customization of app interface color and brightness.	O	

Table 3: Functional Requirements

3.3.2 Non-Functional Requirements

The non-functional requirements focus on ensuring the app's usability, performance, scalability, and security. The app interface is designed to be intuitive and user-friendly, ensuring that students and staff, regardless of their technical proficiency, can navigate it easily.

Security is paramount, with measures in place to safeguard user data through a secure communication tunnel to the chatbot servers. This ensures compliance with university policies and

protects sensitive student and staff information. The app also incorporates Microsoft login functionality, allowing users to sign in with their university email address (@dah.edu.sa). This not only personalizes the user experience but also provides contextual data for the chatbot to respond effectively.

Scalability is another key consideration, with the app designed to support future enhancements. Furthermore, the app is optimized for reliable performance across various devices and network environments, ensuring a seamless experience under different operating conditions.

The mobile application's non-functional requirements are shown in the table below.

Function	Operations	Mandatory (M) or Optional (O)
Usability	The interface should be user-friendly and accessible to non-technical users, providing easy navigation. Multi-language support (English and Arabic) to accommodate the university's diverse user base.	M
Microsoft Login	Allow users to sign into the application via their "@dah.edu.sa" email address to provide context to chatbot.	M
Cybersecurity	The app must have a secure tunnel for the chatbot servers to commit to student and employee data privacy.	M
Scalability	The app should be scalable to accommodate future expansions, such as additional features for career support or student collaboration tools.	M
Performance	Ensure satisfactory application performance on all devices and networks.	M

Table 4: Non-functional Requirements

3.3.3 Use Case Diagram

The primary use case diagram in Figure 1 depicts the main elements and flow of interactions of the proposed chatbot system. At a high level, it shows the user interface where users would interact with the AI chatbot and the various functions and capabilities that the application is intended to have. The actors include the application's user, administrator, the ChatGPT API, and Microsoft Azure for authentication.

The user can login to the app's chatbot and have a conversation with the bot using natural language. With the app's chatbot, the user can access AI-generated university information and answers from the document vectors accessible. In addition, the user can adjust settings and

preferences, report any issues, and view and favorite calendar events happening in or with the university.

The administrator can access the application directly from the backend. The main functions allowed include editing or updating calendar events and deleting or replacing the vector form source documents for the chatbot.

Surrounding the core chatbot functionality, there are two important external integrations: Microsoft Azure and OpenAI API. The purpose of Microsoft Azure is to log in and authenticate the users. Rather than the application have its own user accounts and credentials, the app depends on the Microsoft Azure platform to verify the users and access the system for personalized context.

The chatbot will be integrated with OpenAI via the ChatGPT API. The API will go through the natural language processing, refer to the context vector, as well as any other AI-driven functionality that is available to it through OpenAI to deliver its interaction and response.

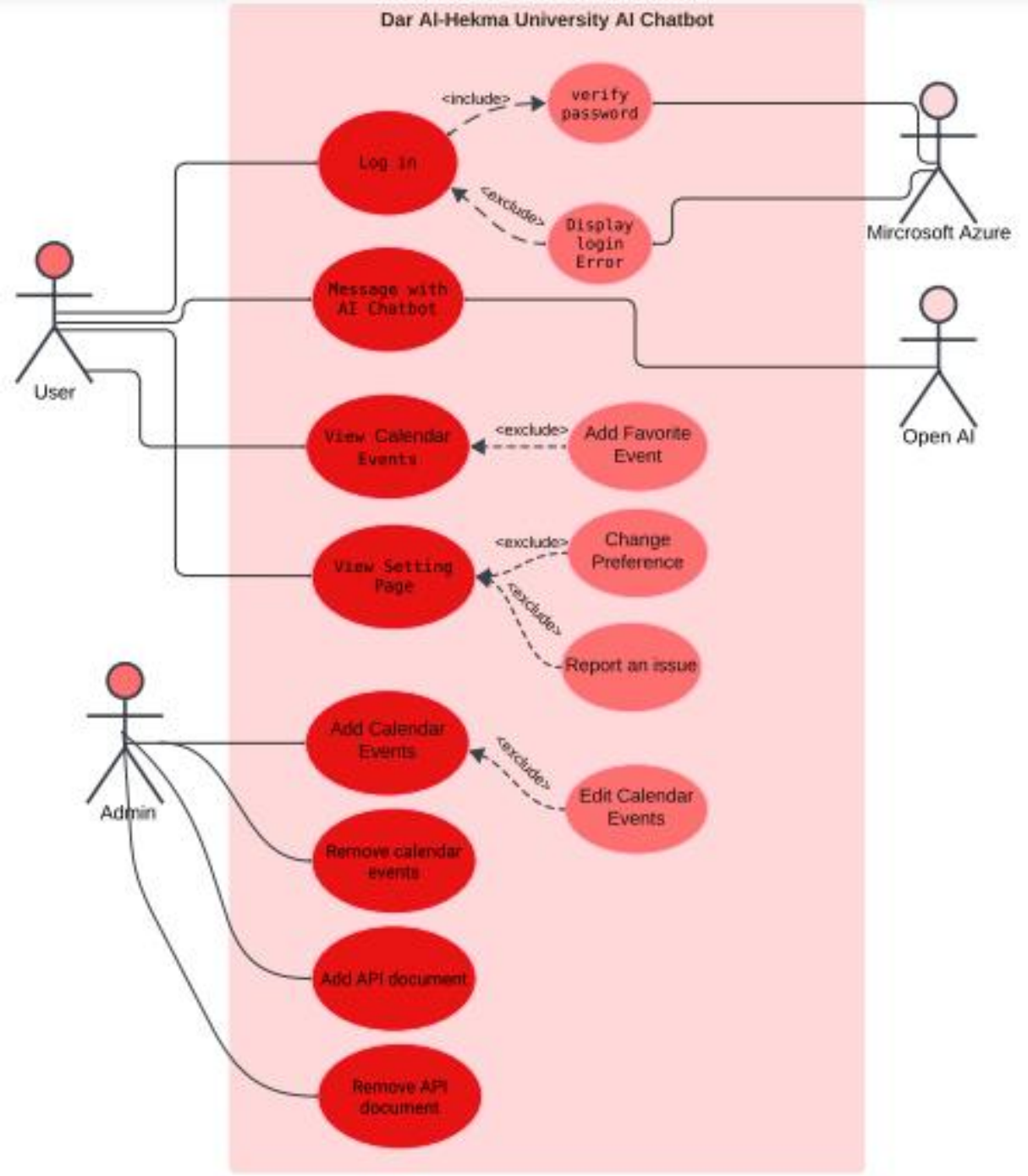


Figure 1: Dar Al-Hekma University AI Chatbot Use Case Diagram

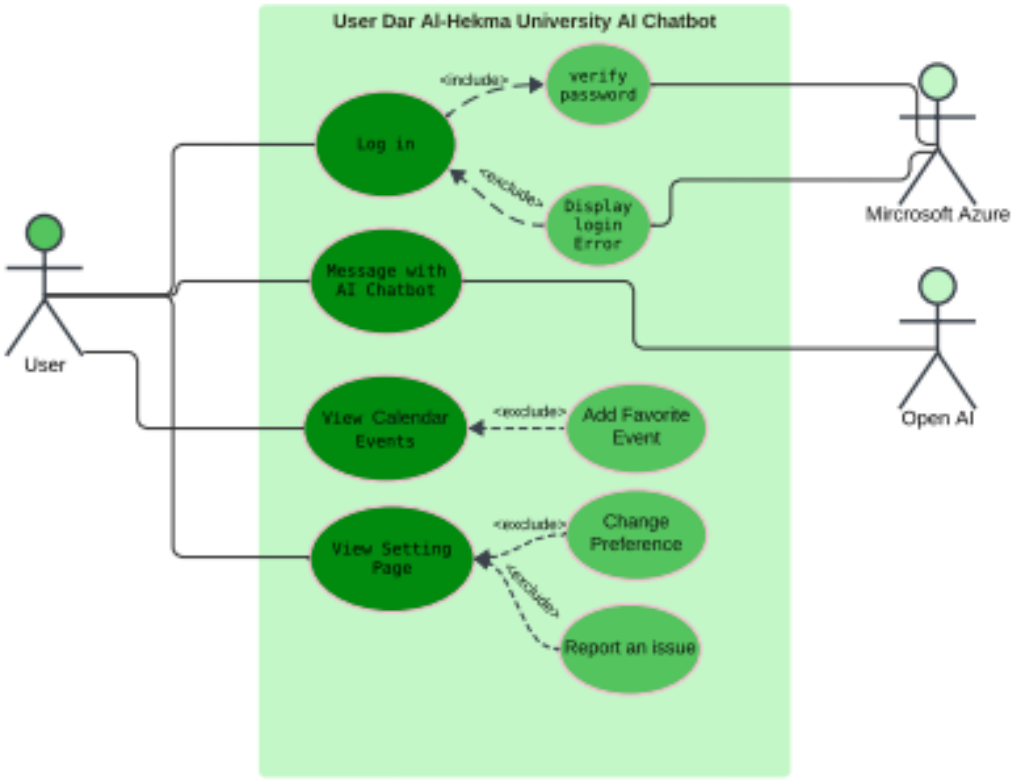


Figure 2: User Actor Use Case Diagram

The detailed use case depicted in figure 2 is for the user actor. This use case describes the interactions a user will have with Microsoft Azure, including login, and interact with OpenAI to send messages to the chatbot, and navigate the application to view calendar events, manage preferences, and report issues.

Include: Log in, Message with AI chatbot, View Calander Events, View Settings Page

Extend: Verify Password, Display Login Error, Add Favorite, change Preference, Report an Issue

Precondition: The user should have an account with valid credentials. The application needs to be connected both to Microsoft Azure and OpenAI services. The chatbot window should be operable and accessible for the user.

Flow of events:

1. Log-in: User logs in with their credentials. Credentials transferred to Microsoft Azure for the confirmation of a password. If credentials matched, access granted; else, a login error should be displayed.
2. Message Chatting with Chatbot: User sends query to a chatbot. The entered query alongside the relevant context vector will be forwarded by the system to Open AI. Open AI processes the query and sends back a generated response for the system to display in the user's interface.
3. Events Overview: The user navigates to the calendar events tab to browse the scheduled events. Possibly, the user stars events as favorites (extended use case).
4. View Settings Page: The user navigates to the settings section to manage the preferences or report an issue (extended use case).

Post condition: The system reflects the user's actions (e.g., favorite events saved, preferences updated). The chatbot responds accurately to the user's queries based on OpenAI processing.

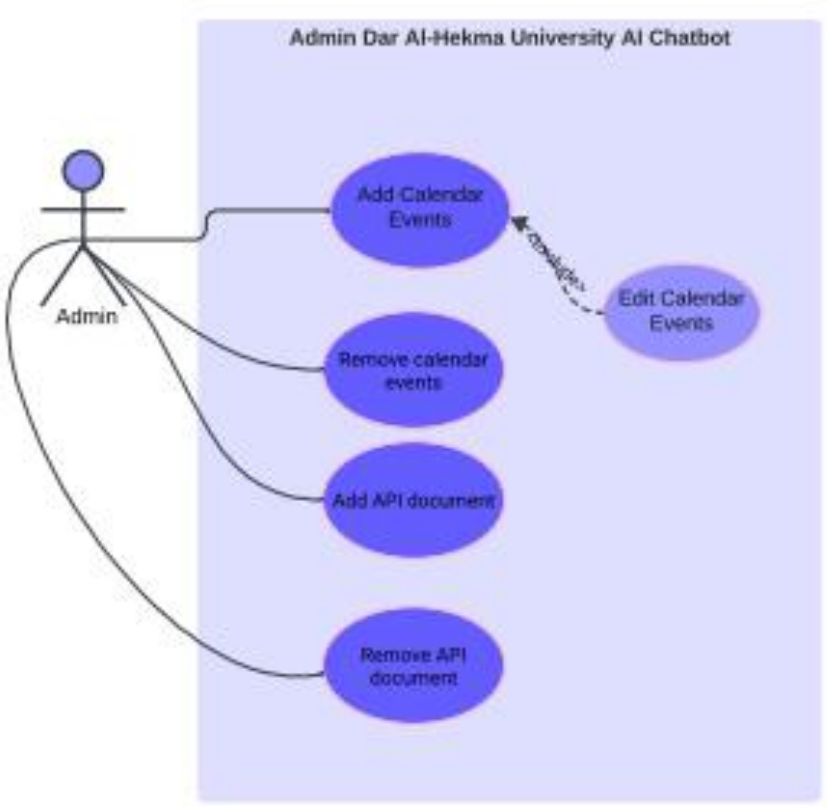


Figure 3: Admin Actor Use Case Diagram

The use case depicted in figure 3 is named “Admin Dar Al-Hakma University AI Chatbot” and details the administrative actions possible. This case describes the actions that an administrator can perform to manage the application system for Dar Al-Hekma University. The administrator is the sole actor.

Include: Add Calendar Events, Remove Calendar Events, Add API Document, Remove API Document.

Extend: Edit Calendar Events

Precondition: The admin should log into the backend system with authentic privileges. The execution environment of the chatbot system is up and communicated to the database where all calendar events and API documents are stored.

Flow of events:

1. Admin logs into the application backend system.
2. Admin chooses action to take such as adding a calendar event.
3. The system will ask the admin to enter/upload the required details, such as event name, date, or category.
4. The system verifies the input or file.
5. The system accordingly updates the functionality of the application, such as adding the event to the calendar.
6. The system confirms that the task has been completed to the admin.

Post condition: The app chatbot's calendar or API functionality reflects the admin's changes.

3.3.4 Requirement Traceability Matrix (RTM)

To ensure comprehensive system development and alignment with user needs, a Requirement Traceability Matrix (RTM) was created. The RTM links the functional requirements of the application to corresponding use cases, providing a clear mapping of system features to user interactions. This approach ensures that every identified requirement is addressed, facilitating a structured development process. The table below outlines the functional requirements and their mapped use cases, along with their priorities and relevant assumptions.

ID	Type	Description	Mapped Use Case	Priority (M/O)	Comments & Assumptions

FR1	Functional	Provide chatbot navigation assistance to specific rooms.	"Message with AI Chatbot"	M	Requires integration with an accurate and up-to-date campus map.
FR2	Functional	Answer common administrative queries (e.g., registration, financial aid).	"Message with AI Chatbot"	M	Chatbot must have access to university-provided data through Microsoft login.
FR3	Functional	Answer common academic queries (e.g., adding/dropping courses).	"Message with AI Chatbot"	M	Microsoft login will contextualize chatbot responses.
FR4	Functional	Provide tools like study schedules, GPA calculators, and course help.	"Message with AI Chatbot"	M	Users will provide grades to the chatbot for calculation.
FR5	Functional	Real-time event updates in a centralized calendar.	"View Calendar Events"	M	Event data categorized by department, school, and community service.
FR6	Functional	Allow customization of app interface with dark/light mode.	"View Settings Page"	O	Optional for enhancing user experience.

Table 5: Requirement Traceability Matrix

3.4 Stakeholders

Stakeholders are all the people involved in the project or impacted by its outcomes. Table 5 shows a list of key stakeholders and their roles. The following figure shows the stakeholder analysis matrix.

Type	Stakeholder Name	Role
Internal to the business	University Admins	Provide strategic direction, approve scope and requirements.
	IT Department	Oversee technical implementation and ensure system operability.

	Students	Act as both primary customers and key stakeholders, providing feedback.
	Faculty	Serve as both content providers and secondary users of the system.
	Software Development Team	Develop the app and ensure the implementation of all requirements.
External to the business	Technology vendors	Provide AI tools and assist with system integration.
	Cybersecurity experts	Ensure compliance with security standards and protect user data.
End-users	Students	Engage with the app for administrative and academic needs.
	Faculty	Use the app for navigation, event tracking, and administrative purposes.

Table 6: Stakeholders

Figure 4: Stakeholder Analysis

3.5 Conclusion

This methodology outlines the process used to gather insights from students, faculty, and administration at Dar Al-Hekma University. By employing both qualitative methods, such as interviews, and quantitative tools like surveys, the research aimed to gain a comprehensive understanding of student needs and preferences. These findings shaped the functional and non-functional requirements of the app, such as providing chatbot assistance for academic queries, campus navigation, and event notifications, as well as ensuring usability, security, and scalability. Stakeholders, including students, faculty, and the IT department, played a central role in defining these requirements by contributing their insights and feedback. These results form the foundation for the chatbot application's design and development and guided the creation of the application solution.

Chapter 4

Design and Methodology

Chapter 4: Design and Methodology

4.1 Introduction

This chapter outlines the design and methodology employed in the development of the university app, focusing on its features: the interactive chatbot and event calendar. The primary aim of this project is to create an intuitive and efficient platform to enhance the student's experience by addressing common challenges faced in university settings.

The technical approach, tools, and frameworks chosen for development are discussed, highlighting their relevance to the project's objectives. By providing a comprehensive overview of the design and methodology, this chapter establishes the foundation for understanding how the app will be developed to achieve its intended outcomes.

4.2 Methodology

The mobile application aims at simplicity and efficiency, delivering a minimal and enjoyable experience for student and employee users alike.

4.2.1 Development Language

Flutter was chosen for building the app because of its versatility, simplicity, and lightweight framework. As a cross-platform tool, Flutter will allow the development team to create a single codebase that works seamlessly on both iOS and Android, reducing development time and effort. Its hot-reload feature makes testing and debugging faster, streamlining the development process. The framework provides a rich set of pre-designed widgets that ensure consistent and visually appealing designs across platforms, enhancing user experience.

Additionally, Flutter's performance is nearly native level because of the way it handles app execution and rendering. It uses the Dart programming language, which compiles directly into native machine code, eliminating the need for a bridge between the app and the device's operating system. This results in faster execution and reduced delays. Flutter's high-performance rendering engine, Skia, interacts directly with the device's graphics hardware to draw the user interface. This allows for smooth animations and fast rendering, ensuring the app feels responsive and fluid, like an app built with native tools like Swift or Kotlin.

These features make Flutter the ideal choice for delivering the responsive, efficient, and visually engaging application intended.

4.2.2 Chatbot

The main backend language for the app is Python. Using the LangChain package, documents are converted into vector representations, allowing for easy comparison and retrieval. These vectors, in theory, would be stored in a Firebase database to facilitate efficient querying.

When a user submits a question or query, the submission is also converted into a vector. BAAI BGE-large package is used to compare the query vector with the university's document and resource vectors and the user's contextual data to identify the most similar match. This process is nearly identical to the Retrieval Augmented Generation (RAG) architecture outlined by OpenAI.

Once the closest match is found, both the inquiry vector and the context vector are sent to OpenAI via the ChatGPT API. A default first-time message with the user context and chatbot rules is sent with the vectors. The API key, stored securely within the app's configuration files rather than the source code to enhance security, is used to authenticate the subscription and calculate the associated costs. OpenAI encrypts all outgoing responses, ensuring privacy and security. Chat history is saved for up to 30 days for each user. After that period is up, the chats are deleted forever.

4.2.3 Event Calendar

The event calendar is a simple, static database that allows users to favorite events for easy access. Event registration is facilitated through an external link displayed at the bottom of each event page. Only the administrator has the authority to add, remove, or edit events, ensuring centralized management. All events would be stored in the Firebase database, and once an event's end date has passed, it is automatically archived for IT and record-keeping purposes and removed from display in the app.

4.2.4 Context

One of the key features of the application is its exclusivity to Dar Al-Hekma internal users and its ability to draw context about each user. This is achieved through integration with Microsoft Azure, as every user already has a custom university email containing basic information, such as user status, ID, class level, and major (for students). Using Flutter's OAuth method, the app delegates login responsibilities to the university's Azure center. Once the user's credentials are authenticated, the Azure center sends the app the user's information, enabling it to store relevant data, such as chat history and favorited events, linked to the user's university ID.

To ensure secure communication, Microsoft Azure authenticates the application by storing the app ID or key within the Azure center managed by the university's IT department. This allows secure information exchange between the app and Azure. Additionally, Microsoft provides robust encryption to maintain the safety of all communications, ensuring a secure data tunnel between the app and Azure.

4.3 High-level system design

Figure 5 provides a high-level view of the proposed system architecture. It illustrates the key components and their interactions, offering a clear visualization of the system's workflow and data flow. This architectural diagram serves as a foundational reference for understanding the system's design and functionality.

The key components and interactions of the items in the figure are:

- Students & Staff: This represents the end-users who will interact with the system.
- Frontend Layer (Flutter): This layer handles the user interface (UI) and user experience (UX) aspects of the application. It provides a platform for students and faculty to interact with the system.
- Backend Layer (Firebase): This layer provides the backend services for the application, including data storage, authentication, and functions. It stores events, archives, user context, user chat history, and performs supportive chatbot operations.
- Chatbot Engine: This component powers the chatbot functionality, linking the application to ChatGPT and enabling users to interact with the system through natural language queries.
- Authentication: This component handles user authentication and authorization using Microsoft Azure.

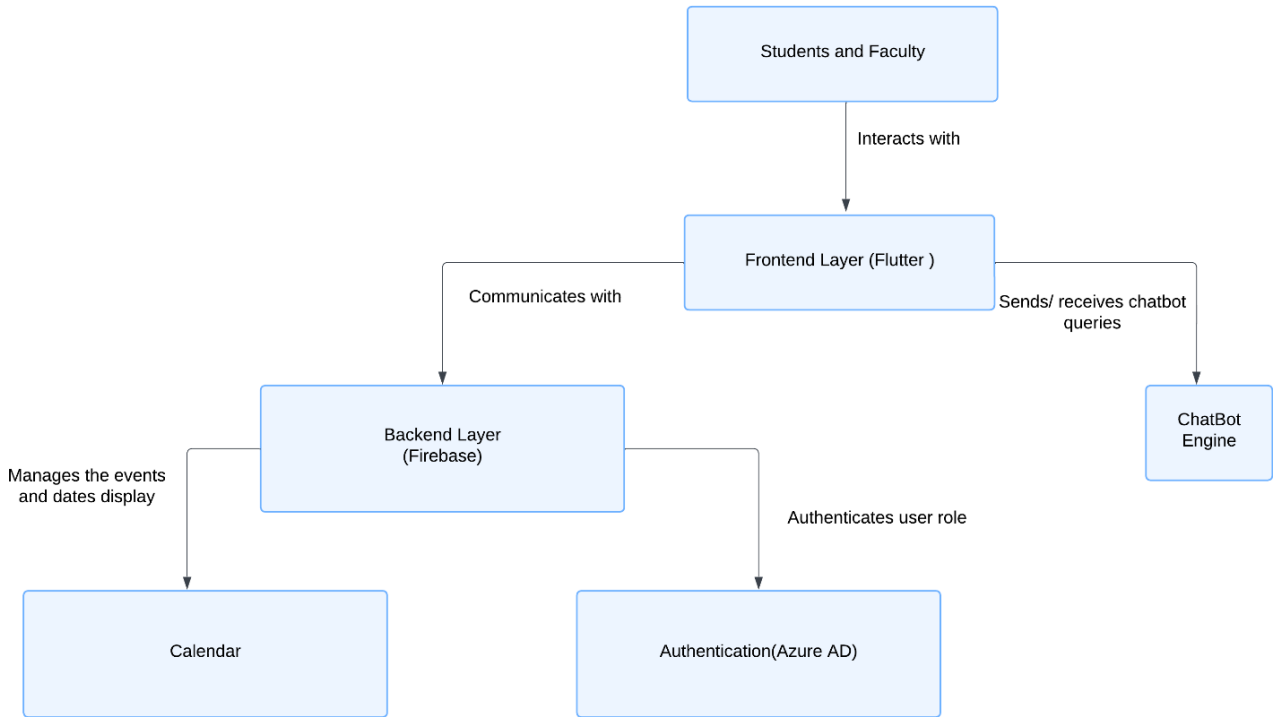


Figure 5: High-Level System Design

Figure 6 presents a more detailed overview of the system's workings. It details the methodology and serves as the starting point for explaining the rest of the low-level diagrams in the coming sections.

Key components and interactions include:

- Flutter Frontend: This component handles the user interface (UI) and user experience (UX) aspects of the application. It includes modules for chatbots, events, and navigation.
- State Management (Provider/BLoC): This component manages the application's state, using a combination of the Provider and BLoC patterns.
- Azure Authentication: This component handles user authentication and authorization using Azure services.
- Role Verification: This component verifies the roles of users within the system, ensuring appropriate access control.
- Firebase Backend: This component provides the backend services for the application, including functions, storage, and database.
- Cloud Functions: These functions handle the chatbot logic and server-side processing of the application.

- Firestore Database: This database stores user and event data.
- API Interfaces: These interfaces enable communication between the front-end and back-end components.
- Chatbot Service: This service powers the chatbot functionality using a natural language processing (NLP) package (BERT) for query processing and providing navigation assistance.
- Storage: This component stores images used within the application.

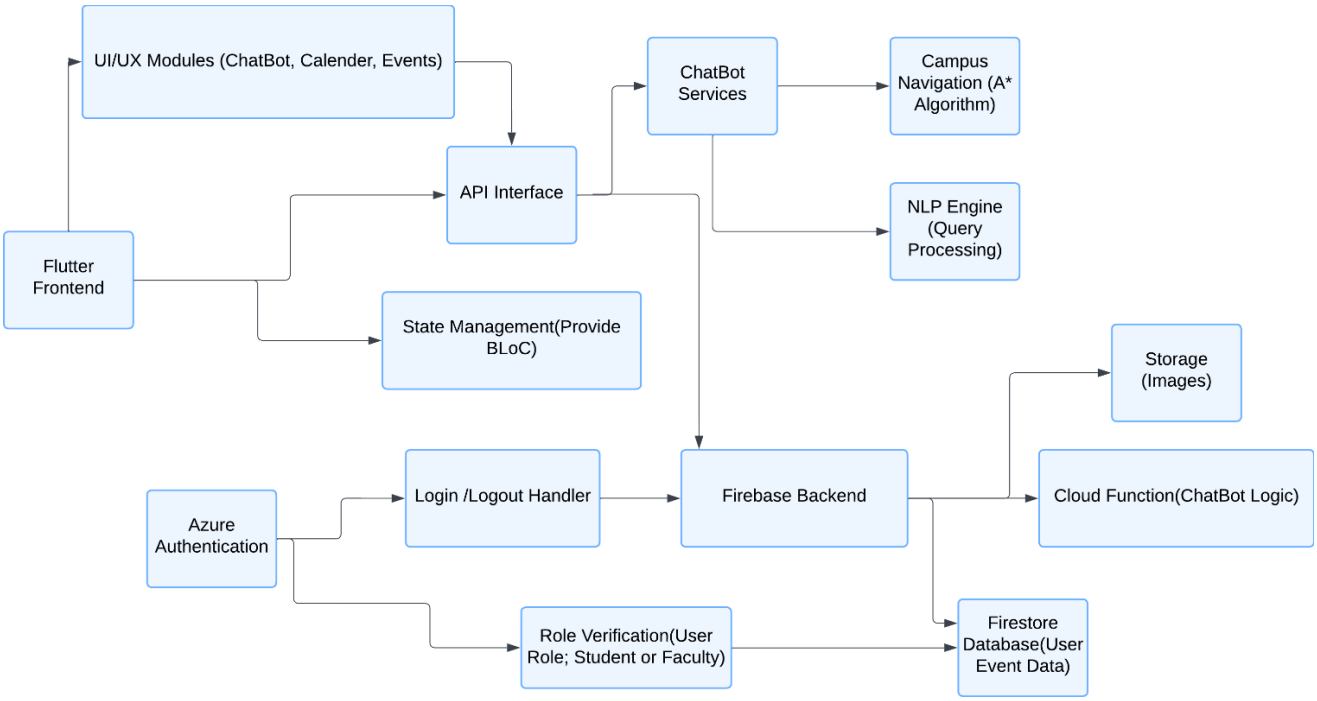


Figure 6: Detailed High-Level System Design

4.4 Low-level system design

The following diagrams detail the development of the implementation. This serves as the theoretical basis upon which Capstone II's program will be built upon.

4.4.1 Sequence Diagram

This is the sequence diagram for visualization of interaction between User and System with Microsoft Azure and OpenAI API for the log-in process, chat usage, page navigation, and event performance. This provides an overview of the user journey through the system and identifies key interactions between components to realize the functionality required.

1. Actors:

   - User: Interacts by logging in, talking to the chatbot, and navigating through the pages.

   - System: Responsible for user interface and interaction with other external services.

   - Microsoft Azure: Handles authentication for sign in.

   - OpenAI API: Process and respond to chatbot queries.

2. Decision Block:

   - The red block indicates the decision point for authenticating the user based on their email and password.

   - If valid, it redirects to the home page.

   - If invalid it displays the error and loops back to requesting credentials.

3. Asynchronous Interactions:

   The dotted lines show asynchronous communication: requests sent to, and responses received from either Open-AI or Microsoft Azure API.

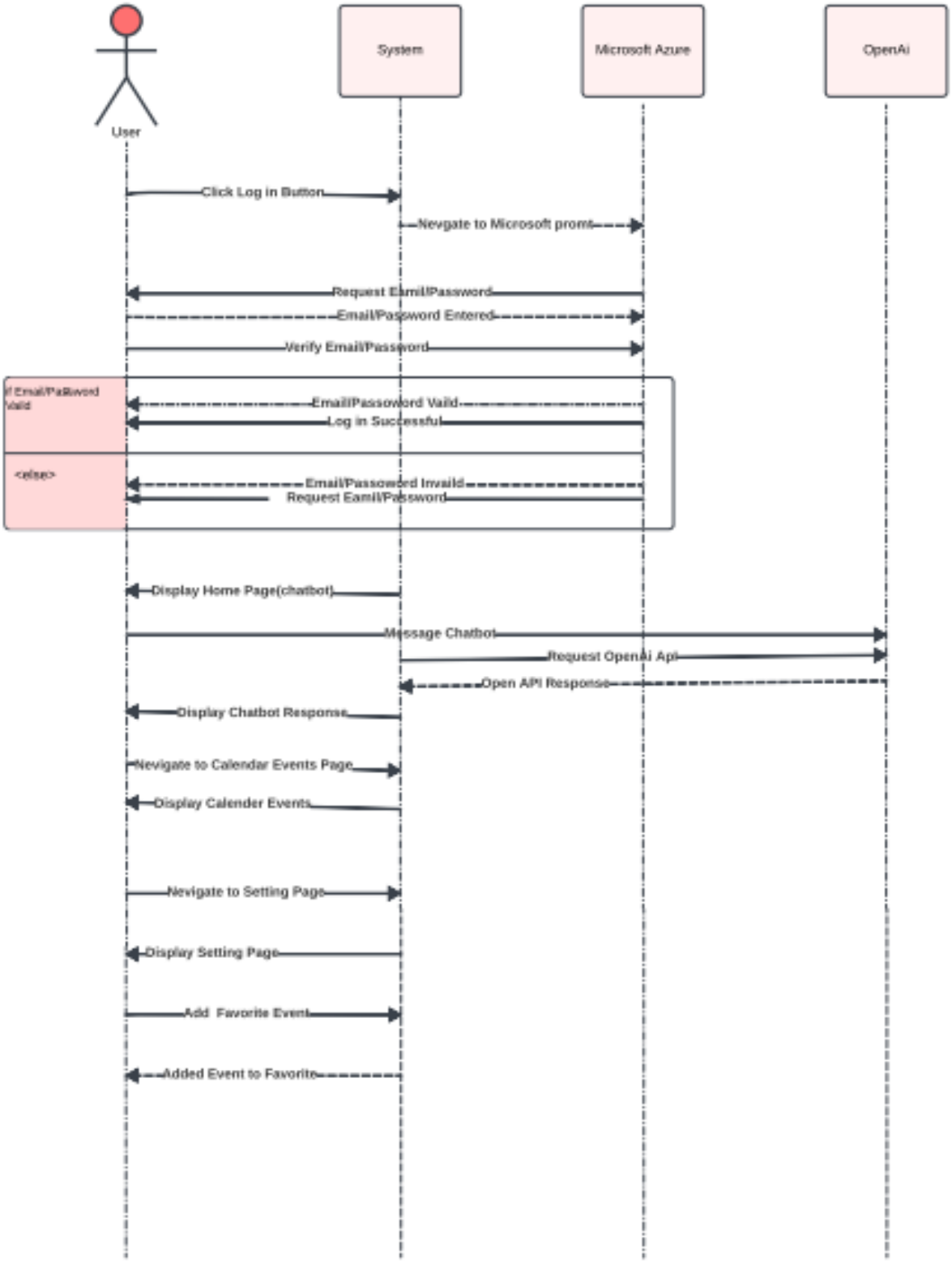


Figure 7: System Sequence Diagram

The complete login process is explained in steps in the table below.

Action No.	Action	Description
1	User Clicks the "Log In" Button	The user starts the login process by clicking the login button on the interface.
2	Redirect to Microsoft Login Page	The system will then redirect the user to the Microsoft login page for processing the authentication of the user.
3	User Enters Email and Password	The system asks for the user's email and password, which he or she inputs.
4	Email and Password Verification	The email and password information are sent to Microsoft Azure for verification
5	Email/Password Valid	If the entered credentials are valid, then the system logs the user in successfully.
6	The Home Page	Chatbot opens.
5	Email/Password Invalid	If the inserted credentials are invalid, an error message will appear, and it will ask to insert the credentials again.

Table 7: Login System Process

Once logged in, the user can now interact with the chatbot, whose process is explained in the table below.

Action No.	Action	Description
1	Display Home Page	After successful login, the system displays the home page with the interface of the chatbot to the user.
2	User Messages Chatbot	The user submits a query or message to the chatbot.
3	Request sent to OpenAI API	System sends the query of the user to the OpenAI API for the response.
4	OpenAI API response	OpenAI processes the request and sends a response back that gets displayed back to the user through the chatbot.

Table 8: Chatbot Interaction System Process

The users also have the option to navigate to the calendar tab, where they can view all events and expand events to view details. The process is detailed in the table below.

Action No.	Action	Description
1	Go to Calendar Events Page	The user clicks on the interface related to calendar events to navigate to Calendar Events tab.
2	Display Events in Calendar	It should list all the events of a given user's calendar and present it to the user.

Table 9: Calendar navigation System Process

In the event a user likes an event and would like to remember to attend it, the user can add it to their favorites. This process is shown in the table below.

Action No.	Action	Description
1	Add Event to Favorites	The user selects an event to add to the calendar as one of their "favorites".
2	Check	System confirms that the event has been added to user's favorites.

Table 10: Favorite Event System Process

Lastly, the user always has the option to adjust the app settings to their preference, shown in the table below.

Action No.	Action	Description
1	Go to Settings Page	The user goes into the interface settings.
2	Display Settings Page	It will then load and open available options of settings to customize interface or report an issue.

Table 11: Settings System Process

4.4.2 Class diagram

The UML class diagram presents a conceptual model of a system related to event management. It shows the classes and their relationships, including inheritance, association, and aggregation.

The system focuses on managing events aspect of the application, allowing users to register for events and view favorites. The admin has additional privileges to create, update, and archive events, as well as manage favorites. The use of categories allows for better organization and filtering of events.

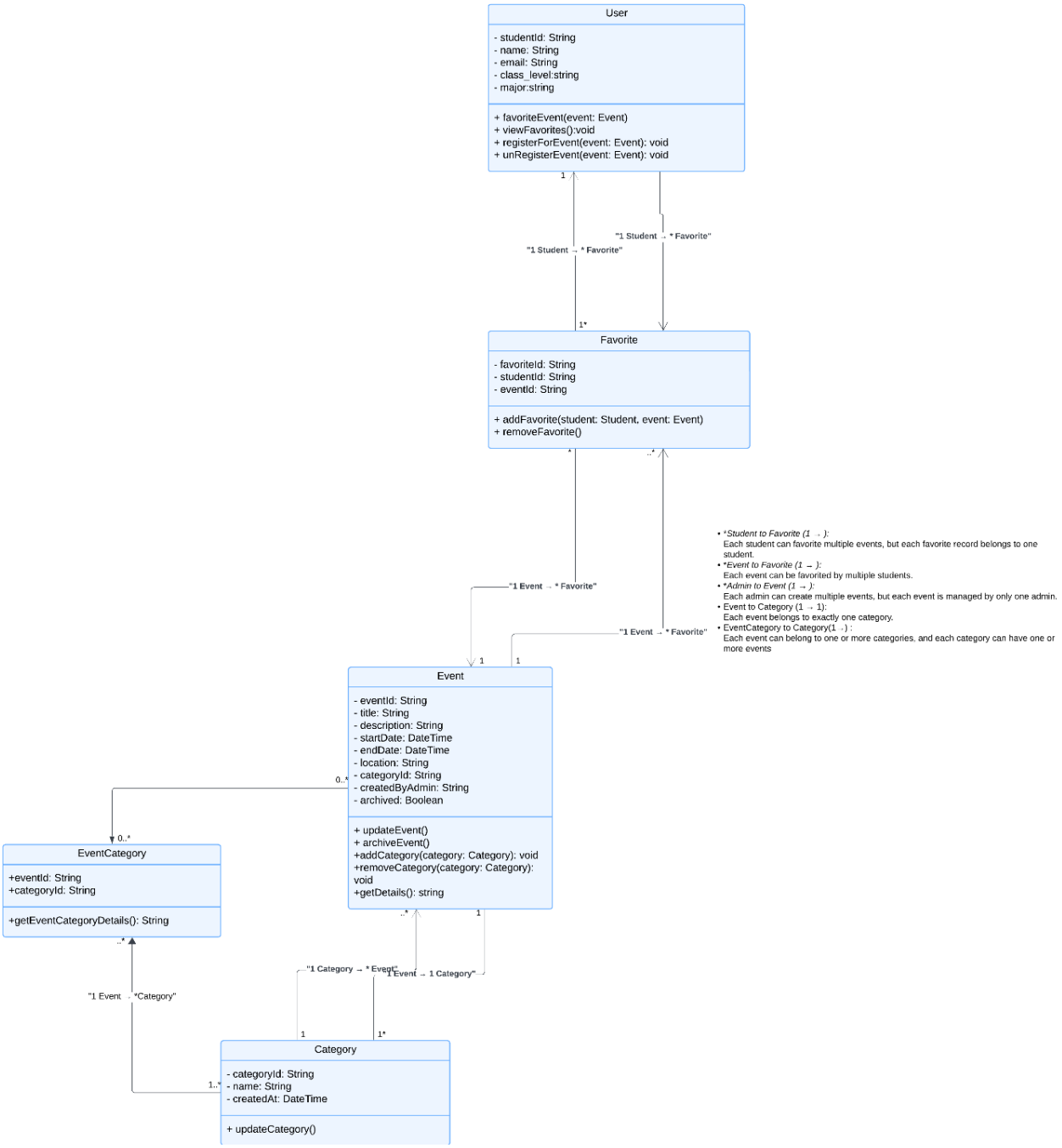


Figure 8: Class Diagram

The calendar feature is a core component of the university app, helping students and faculty track events, add favorites, and navigate important university activities while decreasing the email load. To achieve this, a structured database schema was designed to ensure scalability and efficiency. To enhance clarity and understanding, the UML class diagram is attached above. However, due to its complexity and level of detail, each class is further explained in a tabular

format below, including descriptions and examples for added context. Additionally, the relationships between the classes are outlined individually to ensure a comprehensive understanding of their interactions and dependencies. This approach ensures the information is accessible and clear, regardless of familiarity with class diagrams.

Entities

User: all users of the system, whether students, staff, or faculty (Table 11.)

Attribute	Data Type	Description	Example
userId (PK)	String	Unique identifier for the user	2110123
name	String	Full name of the user	Amna Ahmed
email	String	User's email address	aahmed@dah.edu.sa
role	String	Specifies whether the user is a "Student" or "Faculty"	Student
class_level	String	Applicable only for students; null for faculty	Senior
major	String	Applicable only for students; null for faculty	Computer Science

Table 12: User Entity

Event: all events created by faculty users for the app (Table 12.)

Attribute	Data Type	Description	Example
eventId (PK)	String	Unique identifier for the event	CS001
title	String	Name/title of the event	Tech Talk
description	String	Description of the event	Discussion on AI trends
startDate	DateTime	Start date and time of the event	2024-12-15 10:00 AM
endDate	DateTime	End date and time of the event	2024-12-15 12:00 PM
location	String	Venue of the event	Hall 151
archived	Boolean	Indicates if the event is archived	false

Table 13: Event Entity

Category: the categories possible to assign to events for classification (Table 13.)

Attribute	Data Type	Description	Example
categoryId (PK)	String	Unique identifier for the category	WS01
name	String	Name of the category	Workshop

Table 14: Category Entity

EventCategory: the relationship between events and categories (Table 14.)

Attribute	Data Type	Description	Example
eventId (FK)	String	References the event linked to the category	CS001
categoryId (FK)	String	References the category linked to the event	WS01

Table 15: EventCategory Entity

Favorite: the events favorited by students (Table 15.)

Attribute	Data Type	Description	Example
favoriteId (PK)	String	Unique identifier for the favorite entry	Fav000001
studentId (FK)	String	References the student who favorited the event	2110123
eventId (FK)	String	References the favorited event	CS001

Table 16: Favorite Entity

Relationships

The table below details the relationships between the different entities. The type notations are shortened so that 1:1 means one-to-one, 1:N means one-to-many, N:1 means many-to-one, and N:M means many-to-many.

Relationship	Type	Description	Representation
User to Favorite	N:M	A user can mark multiple events as favorites, and an	`studentId` in the `Favorite` table relates to `userId` in the `User` table.

		event can be favorited by multiple users.	`eventId` in the `Favorite` table relates to `eventId` in the `Event` table.
Event to Category	N:M	An event can belong to multiple categories, and a category can include multiple events.	`eventId` in the `EventCategory` table relates to `eventId` in the `Event` table. `categoryId` in the `EventCategory` table relates to `categoryId` in the `Category` table.
Category to Event	1:N	A category can have multiple events assigned, but each event is assigned to at least one category.	Same as `Event to Category` relationship.
User to User (Role)	1:1	Each user can only be either a faculty or student, determined by their `role`.	`role` attribute in the `User` table distinguishes between student and faculty.

Table 17: Entity Relationships

Event Archival

To efficiently manage expired events and maintain the database, a Firebase Cloud Function will automate the archival process. This function runs daily, checking for events where the endDate is earlier than the current date and the archived field is set to false. For such events, the function updates the archived field to true, ensuring that outdated events are properly marked and removed from view in the app.

The database structure for the calendar is designed for scalability, ensuring seamless event management, user interaction, and automation. With clear entity relationships and automated archival, the system guarantees a robust and user-friendly experience. This procedure is not implemented in the MVP.

4.5 Prototype

This chapter provides the UI design of the application as well as the elementary proof of concept, focusing on user-centric design principles and confirming the feasibility of the program methodology outlined previously.

4.5.1 User Interface Design

The application will have light and dark modes to cater to different users' preferences and increase usability in different environments. Each page is designed to be simple and clear for intuitive user interaction.

Welcome Screen

Starting with the welcome screen depicted in figure 9 below, this is the very first interaction that any user will have with the application, hence the professional yet interesting design. The logo of Dar Al-Hekma University is in the center of the page, with the tentative application name (Ask DAH) and its tagline (Ask, Learn, Engage). Minimalistic design further strengthens clarity and professionalism, putting more emphasis on the identity of the app.

Key features:

- Logo placed in the center for maximum visibility.
- A bold and prominent appearance for the name and slogan of the app.
- A matched color scheme to match university branding.

Figure 9: Welcome Screen

Sign-In Page

The sign-in page in figure 10 ensures secure access to the application by integrating with Microsoft Azure for authentication. This portal guarantees the provision of a safe and seamless avenue for users to sign in through the infamous Microsoft Azure pop-up window. A simple interface is designed to assure maximum accessibility for all users.

Key features:

- An obvious link to the Microsoft Azure Sign-In Portal.
- Clear instructions guiding users on how to log in.
- A consistent design that aligns with the rest of the application.

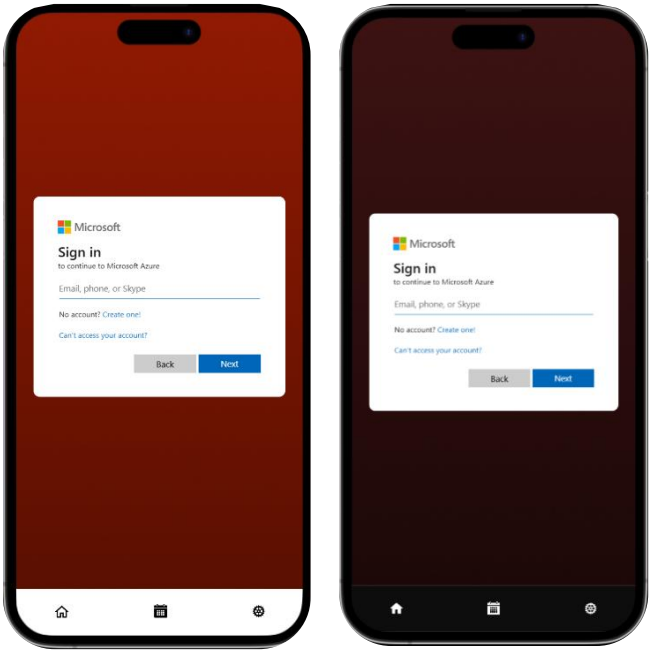


Figure 10: Sign-In Screen

Home Page (Chatbot)

The application opens directly into the chatbot screen in figure 11, to the primary aspect of the application, the chatbot. This design effectively enhances users' interaction with the chatbot. The user can enter his or her queries in text format or select from some quick accesses. The design's simplicity maintains usability.

Key features:

- A responsive chat box located at the bottom of the screen for user input.

- Floating action buttons for potentially adding advanced actions (ex. speech input, file attachment.)
- Uncluttered design to emphasize the chat interface.

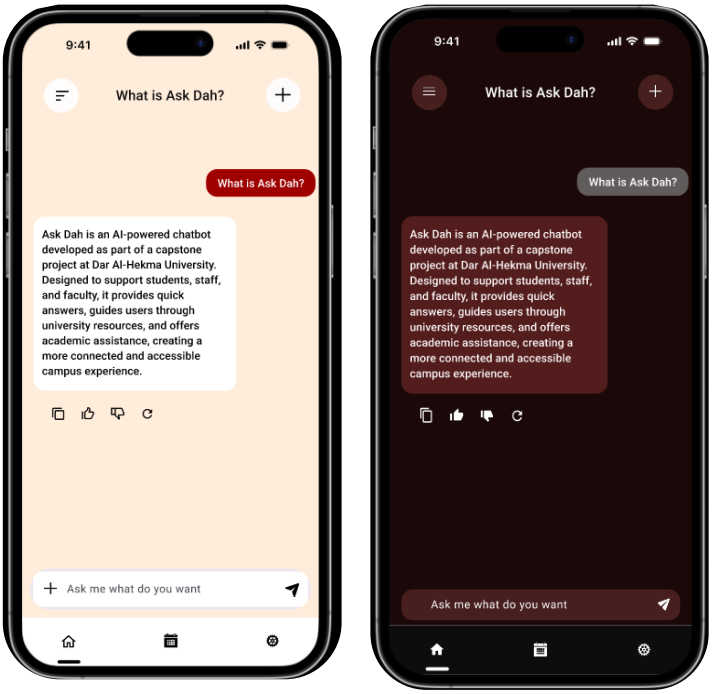


Figure 11: Home Page

The chat history page in the figure below allows users to view their past conversations in an organized manner, making the information easier to reference. This page is designed to show the user's past conversations in chronological order, including timestamps for context. The format makes it easy to see and refer to previous discussions. However, it is important to note that these are transcripts only and the chat cannot be resumed.

Key features:

- Categorized view of prior chats
- Delete selected conversation or clear all history options
- Timestamps for each interchange to add clarity

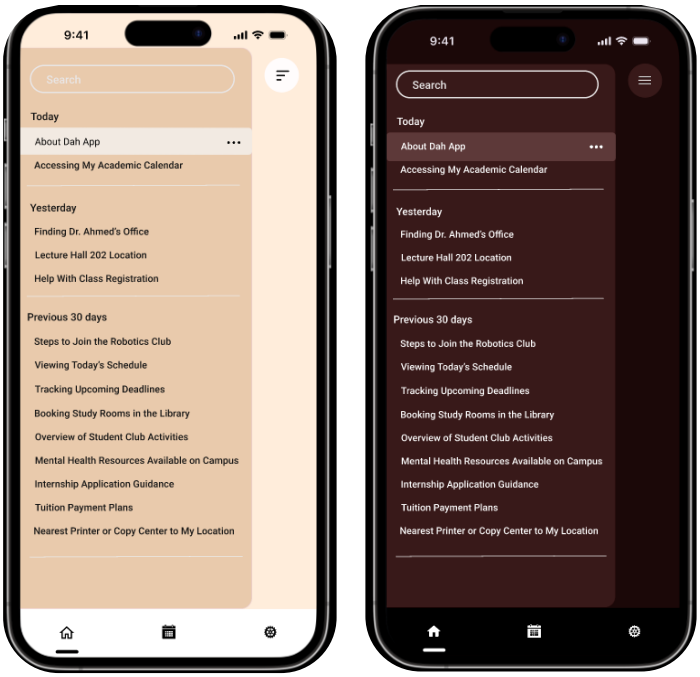


Figure 12: Chat History Page

Calendar

The calendar can be considered a hub for browsing, exploring, and managing favorite events across different categories, such as workshops, Design School, Gaming Club, and more. This page is designed based on functionality and simplicity, as shown in figure 13, allowing users to browse events efficiently via categorical or word search, and adding favorites.

Key features:

- Filterable event categories for ease of access.
- Dedicated search bar.
- Ability to mark events as favorites.

The favorite events page in figure 14 offers the user quick access to marked events, thereby personalizing the experience. The page is designed to clearly list the favorited events with key information available immediately.

Key features:

- Event cards containing title, date, and location of the event.
- Remove events from the favorite list option.
- Notifications when events marked as favorite are updated or upcoming.

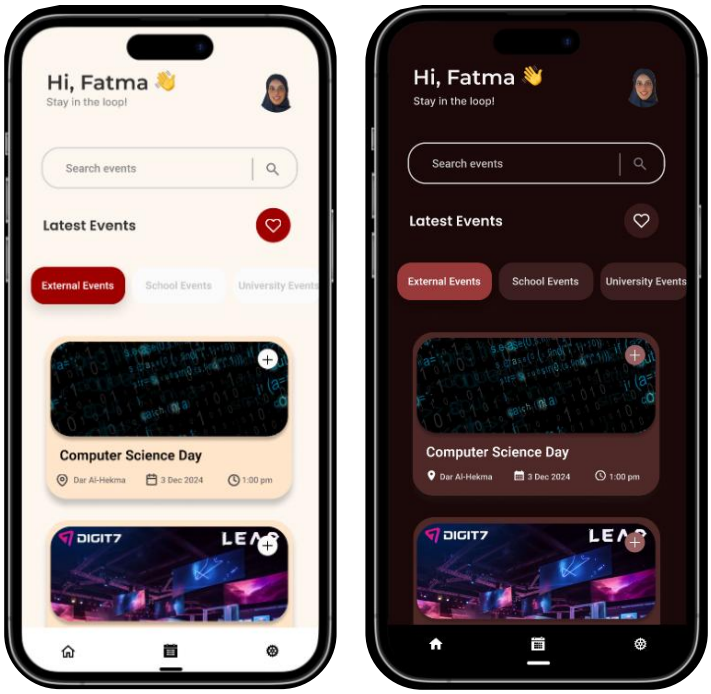


Figure 13: Calendar Page

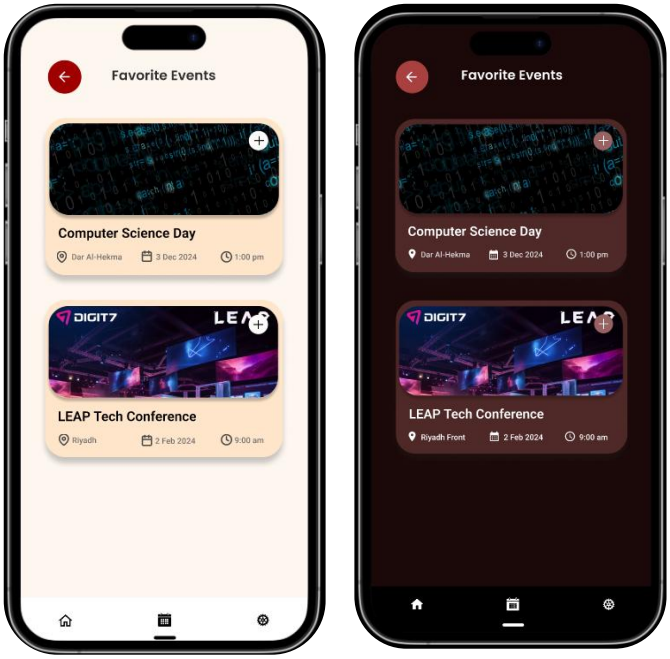


Figure 14: Favorite Events Page

The event description page in figure 15 offers a detailed view of events after tapping on them, allowing users to access comprehensive information. This page highlights all essential

event details in a structured format, complemented by multimedia content to enrich the user experience.

Key features:

- Event information such as title, date, location, and a detailed description.
- Media integration, including images and videos, for visual appeal.
- Registration and sharing options to enhance usability.
- External link icon highlights leacing the app for registration.

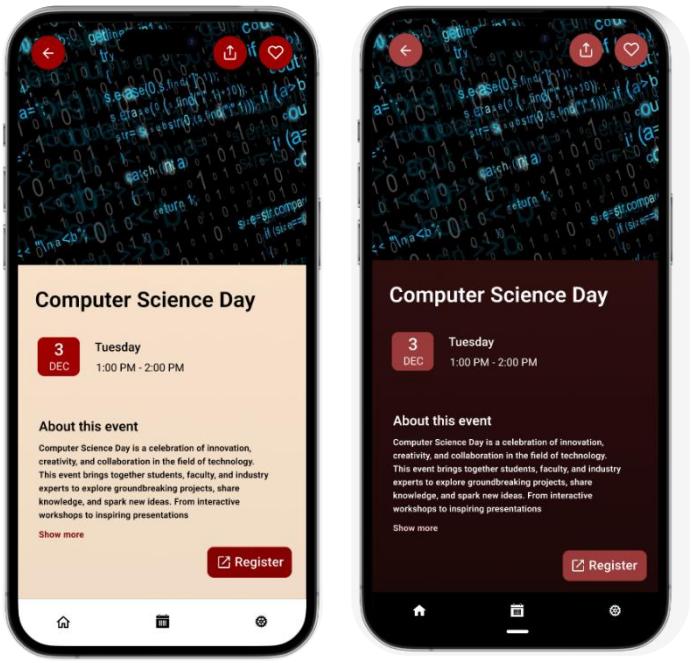


Figure 15: Event Details Page

Settings Page

The settings page shown in figure 16 is where the user can adjust the app's appearance to his or her liking. The intuitive layout of the page presents the necessary customization options and settings for convenience.

Key features:

- Notification preferences regarding events or updates.
- Theme toggle between light and dark mode.
- Viewing frequently asked questions or reporting an issue for easy suggestion and problem support.

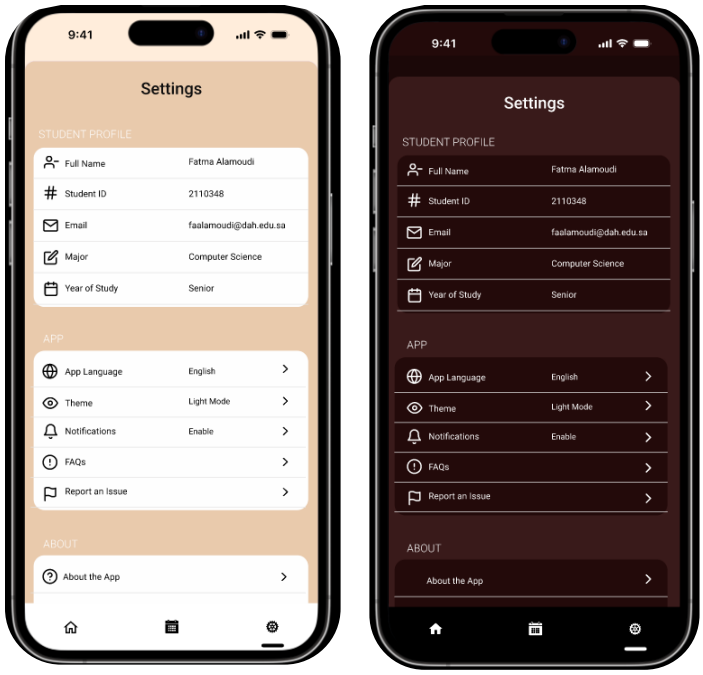


Figure 16: Settings Page

Compatibility with different devices is ensured by the application's responsive design, which improves accessibility due to light/dark modes. The clean modern UI with common affordances promotes a smooth user experience.

4.5.2 Proof of Concept

NLP is an important prerequisite to the development of intelligent systems that can interactively deal with users. Therefore, as proof of concept, the focus of the prototype in this phase was on the design of a chatbot using OpenAI's GPT API integrated with a pre-processing pipeline built using Python's NLTK. This will guarantee that the input data being fed into the AI model is cleaned and optimized for meaningful analysis, and that the plan to use ChatGPT is viable and set to work in the implementation. The source code is available in appendix c.

The underlying principle of any NLP-based system is effective pre-processing. The steps used are as follows:

- Lowercasing: All input text is converted to lower case to avoid case sensitivity in the words; "ChatGPT" and "chatgpt" would be regarded as different words otherwise.

- Removing Special Characters: The usage of regular expressions removes non-alphabetical symbols, punctuation marks, and numerical data to avoid confusion in model understanding. In this way, it will concentrate the data on meaningful language content.
- Tokenization: Then, with the use of NLTK's RegexpTokenizer, a regex pattern, r'\\w+', is used to divide the text into words. This further helps in word handling for many other steps of processing. Stopword Removal: Now it can remove stop words such as "the,", "is,", "and," while using the default stopwords created for NLTK. These terms will mean little about context but do tend to add computational overload in tasks.
- Lemmatization: These above steps will normalize the words to their base form; for example, "running" to "run" by using WordNetLemmatizer. Preprocessing in this way is going to standardize input data; therefore, most of the redundancy in word forms will be removed.

Integration with OpenAI GPT

Preprocessed text is fed into the OpenAI GPT through its API. A Python script held the integration of the key elements, which pre-processing module and GPT 4.0 interacted with smoothly. The OpenAI API client was set up with a secure API key to allow for communication. The method “completions.create” was utilized to send the prompts to, and fetch responses from, the model. As shown in the figure below, the output shows that the connection to ChatGPT was successful.

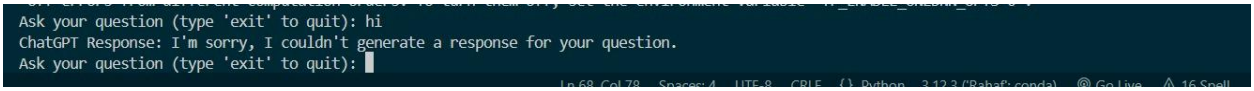


Figure 17: POC Output

The workflow begins with preprocessing the user input using the custom-built pipeline. Now, the cleaned input is fed to the GPT model. The model response is displayed to the user after removing extra whitespaces. To quit the chat, a user can type "quit."

Provisions for handling invalid or vague requests were incorporated. Guidelines for refining unclear queries were also explored as part of improving user experience.

Some enhanced features that can be considered for implementation include:

- Natural Language Prettification: Additional NLP models can clean up ambiguous user inputs before passing them on to GPT.
- Tree-Based Location System: A tree-like structure of locations will help in navigation-related requests, translating programmatic instructions into natural language.

4.6 Conclusion

This chapter provided a detailed overview of the technical design and methodology behind the development of the university app. By leveraging Flutter as the primary development framework, the app ensures cross-platform compatibility, high performance, and a visually cohesive user experience. Flutter's rich widget library and native-level performance have streamlined the development process, allowing for the creation of a responsive and intuitive interface for students and employees alike.

The chatbot functionality, powered by Python and integrated with OpenAI's ChatGPT, offers a robust solution for providing context-aware assistance to users. Using advanced techniques such as vector representations, Firebase for efficient data management, and the A* algorithm for navigation, the app combines innovation with practicality. Security measures, such as encrypting user interactions and securely storing API keys, further enhance user trust and data privacy.

Additional features, including the event calendar and user-specific context retrieval via Microsoft Azure, underscore the app's commitment to exclusivity and personalization. The integration of Azure ensures seamless and secure authentication processes while enabling the app to cater to each user's needs by drawing on their university profile.

The low-fidelity prototype illustrates careful attention to detail in its user interface design, from functional and accessible to aesthetic perspectives. The application includes both Light Mode and Dark Mode to meet different users' preferences with a harmonious balance and visual appeal. The addition of features like chat history, favorite events, and customizable settings further expresses the application's orientation toward user engagement and personalization.

Together, these design choices and methodologies reflect a well-thought-out strategy for delivering a user-centric application that simplifies campus life, fosters communication and engagement, and enhances accessibility within the Dar Al-Hekma University community.

Chapter 5
Implementation

Chapter 5: Implementation

5.1 Introduction

This chapter provides a clear overview of our capstone project implementation process: "AskDAH", an application housing an AI-enabled university support chatbot designed to assist Dar Al-Hekma University students and a centralized event calendar. The implementation process was a collection of tools, frameworks, and methodologies that collectively contributed to a working and usable system. The chapter is divided into sections that discuss excluded features from the design phase, tools utilized, the system development process overall, issues encountered while developing, and how we solved them. This chapter illustrates how technical decisions were made and transformed into functional components in the system development process.

5.2 Demo Scope

The foundation of this project was to develop a student-centered application that is intuitive, accessible, and responsive, with a specific focus on students enrolled at the university. The core objective was to centralize and simplify access to essential resources while maintaining a clean, user-friendly interface.

One of the key features implemented was integration with existing university systems. This included linking year level, major, and student ID through Microsoft authentication, and consolidating academic resources such as the POS, course prerequisites, student handbook, and university regulations into one accessible platform. However, to avoid redundancy and maintain simplicity, dynamic processes like registration, grade checking, and homework submissions were deliberately excluded, as they are already handled effectively by existing services like SIS and Blackboard.

A major component of AskDAH is the chatbot, designed to operate via local API-based computation. This ensures that no user data is stored or learned from beyond the device, prioritizing student privacy. The chatbot is equipped to redirect students to appropriate university resources for issues beyond its scope, such as academic support or mental health services. Additionally, it offers the option for saving a local history of chats for the user's future reference.

Another key feature is the event calendar, which consolidates all university-wide events. Events are added by the IT department upon departmental request. This incurs an administrative

workload, however when looking at the admin time saved by the institution it is manageable. There is also the potential for automating this process further as discussed in chapter 8. Each event includes its essential details like date, time, description, image, and external registration links when applicable.

University documentation for chatbot document and vector sources are requested and added by IT annually at the beginning of the academic year, and events are added weekly every Sunday as per department request.

While several features were conceptualized during the design phase, they were not implemented in the current demo due to constraints related to time, resources, development capacity, and budget. These excluded features include Microsoft Azure login integration, GPA calculation tools, chatbot-driven course assistance, and cybersecurity enhancements.

5.3 Implementation Tools

5.3.1 Graphical User Interface Tools

The primary technology used for front-end development is Flutter, Google's open-source user interface framework for developing natively compiled applications from a single codebase. The Flutter framework allowed us to create a responsive web interface with a similar mobile app experience, fixing the first mobile deployment constraint we were facing. Deploying to the web platform, we ensured that the chatbot would be easily accessible on various devices such as iOS and Android without installation.

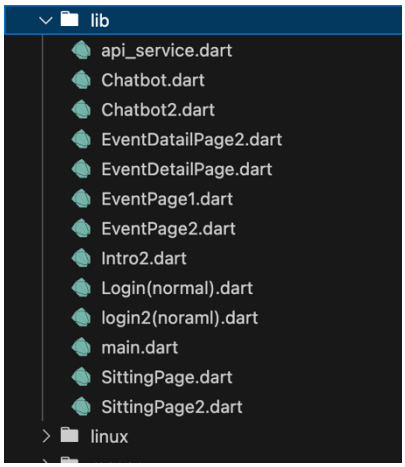


Figure 18: Flutter Front-End Files

```dart
void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(debugShowCheckedModeBanner: false, home: IntroScreen());
  }
}
```

Figure 19: Flutter Main Class Code

For user interface and experience planning, we employed Figma, one of the widely used cloud-based UI/UX design tools, as outlined in chapter 4. Figma allowed our team to work together on interface layouts, prototype user interactions, and receive real-time feedback. With Figma, we created high-quality wireframes and defined accurate user flows to develop an intuitive, accessible experience for all users.

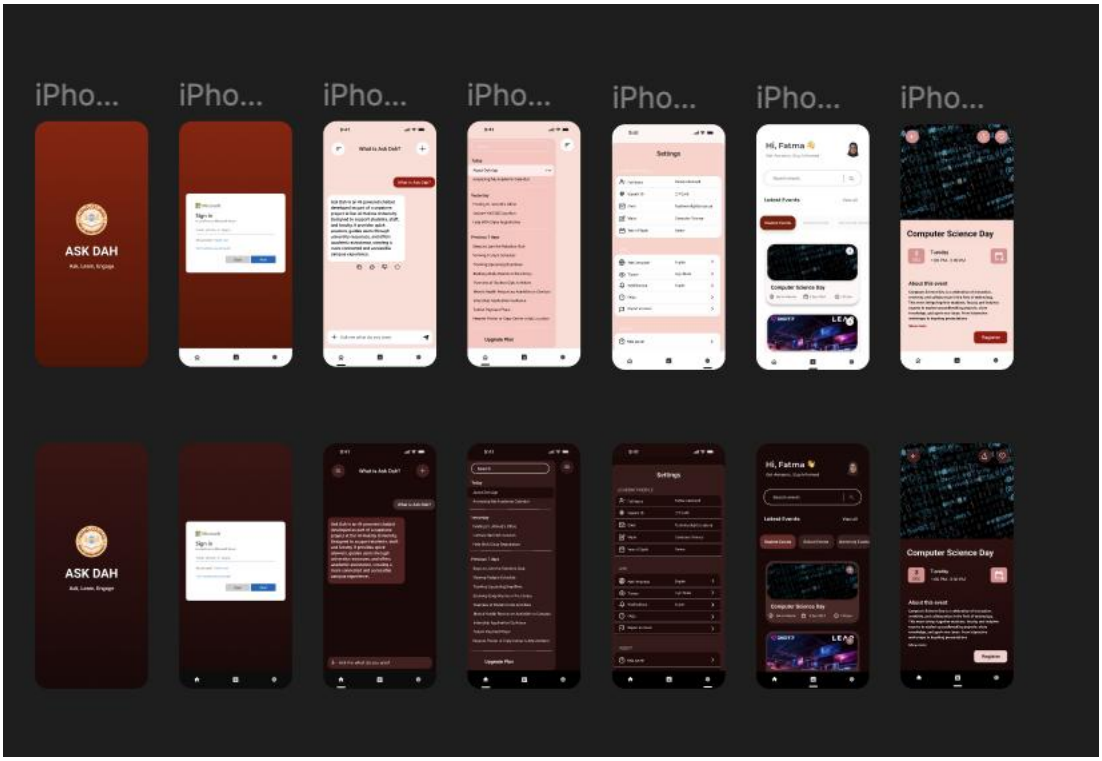


Figure 20: Figma UX/UI Screens

For testing and development across different platforms, we employed a set of tools. Android Studio provided us with an in-built phone emulator for testing builds on Android. Xcode, Apple's official development environment, was used to test for iOS compatibility, as we were able to use it to emulate and debug the interface on iPhones. Additionally, we tested the

web app mostly using Google Chrome, which served as our debugging, layout inspection, and performance analysis default browser. These tools collectively enabled us to simulate the application's behavior on Android, iOS, and web platforms with consistency and reliability across user contexts.

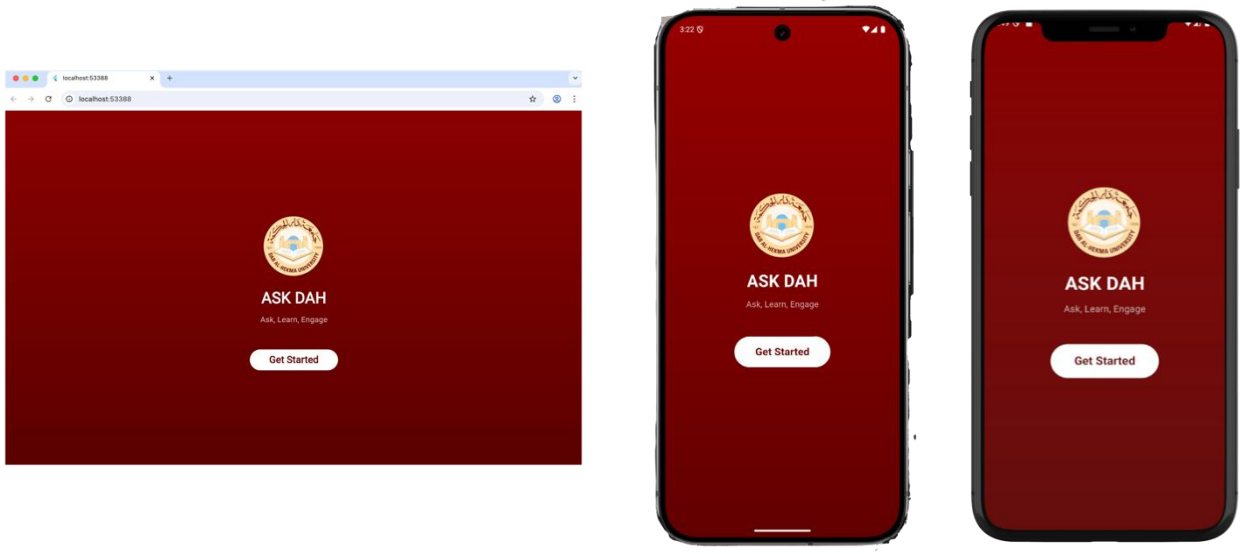


Figure 21: Android Studio Emulator on Web, Android, and iOS

5.3.2 System Tools

The system's backend was developed with a modern and efficient tech stack capable of supporting intelligent response generation, document search, and memory management. These libraries and utilities were chosen carefully to develop an efficient, scalable, and intelligent system capable of supporting Retrieval-Augmented Generation (RAG) operations in real-time.

The main backend development programming language is Python due to simplicity, readability, and rich ecosystem of AI and data processing libraries. For the construction of the RESTful API layer, we employed FastAPI, a high-performance web framework renowned for its asynchronous nature, rapid execution, and auto-generated interactive documentation via Swagger and ReDoc. Compared to traditional frameworks like Flask, FastAPI offers a cleaner, more expressive syntax along with being built using natively supported Python type hints to support enhanced code reliability and maintainability.

```python
# Serve static files from events directory
@app.route('/api/events/images/<path:filename>')
def serve_event_image(filename):
    return send_from_directory('events/images', filename)

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5555)
```

Figure 22: Initializing the API Code

```
class ApiService {
  static const String baseUrl =
      'http://172.20.10.2:5555'; // 10.0.2.2 Android Emulator
  //localhost:5555 iOS
  //localhost5000 windows
```

Figure 23: API Service Port Number for the Local Host Code

To implement the AI-driven logic, we used LangChain, a solid framework for simplifying the orchestration of large language models (LLMs) with additional data. LangChain was responsible for managing prompt templates, saving conversation history, and integrating pieces such as memory, retrievers, and LLMs. Such key modules like ChatPromptTemplate, ConversationBufferMemory, and MongoDBChatMessageHistory allowed us to create a pipeline where the chatbot would be able to remember past user inputs, retrieve appropriate knowledge from documents, and respond naturally and contextually properly.

```python
import logging
from typing import Optional, List
import yaml
from pymongo import MongoClient
import numpy as np
from langchain.schema.output_parser import StrOutputParser
from langchain.schema.runnable import RunnablePassthrough
from langchain_core.prompts import ChatPromptTemplate
from langchain_openai import ChatOpenAI
from sentence_transformers import SentenceTransformer
from langchain_community.chat_message_histories import MongoDBChatMessageHistory
from langchain.memory import ConversationBufferMemory

logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

class ChatBot:
    def __init__(self, config_path: str = "config.yaml"):
        """Initialize ChatBot with configuration."""

        with open(config_path, 'r') as file:
            config = yaml.safe_load(file)

        self.client = MongoClient(config['mongo_connection_str'])
        self.db = self.client[config['database_name']]
        self.collection = self.db[config['collection_name']]
        # Add collection for conversation memory
        self.memory_collection = self.db.get_collection('conversation_memory')

        self.embedding_model = SentenceTransformer(config['embedding_model'])

        self.model = ChatOpenAI(
            model_name=config['model_name'],
            openai_api_key=config['openai_api_key'],
            temperature=config['temperature']
        )
```

Figure 24: Chatbot Parameters Code

```
# Updated prompt with explicit instructions to remember previous messages and handle references
self.prompt = ChatPromptTemplate.from_template(
    """
    You are a helpful and friendly chatbot for Dar Al-Hekma University. Your role is to assist users by answering their questions clearly and conci

    IMPORTANT: You have access to the conversation history. ALWAYS refer to this history when the user asks about previous messages.
    If asked about previous messages, check the conversation history and respond with specific details about what was said.
    Never say you don't have the ability to remember previous messages, as you have full access to the conversation history.

    IMPORTANT FOR FOLLOW-UP QUESTIONS: When the user asks a short follow-up question or uses pronouns like "it", "this", "they", etc.,
    always resolve these references based on the previous conversation. For example, if they ask "Is it available?" after discussing
    a program, understand that "it" refers to that program and respond accordingly.

    CRITICAL INSTRUCTION: NEVER start your response with phrases like "Based on our previous conversation", "From our earlier discussion",
    "According to what was mentioned before", or any similar meta-references. Always answer directly as if you inherently understand
    what the user is referring to. Give immediate, direct answers without explaining how you determined what they meant. Directly answer
    without refering what "it", "this", "they", etc., mean.

    Previous conversation:
    {history}

    Context:
    {context}

    User's current question:
    {question}

    Answer:
    """
)
```

Figure 25: Chatbot Instructions with the First Message Code

```
# Use LangChain's MongoDB Chat History
message_history = MongoDBChatMessageHistory(
    connection_string=self.mongo_connection_string,
    session_id=session_id,
    database_name=self.db_name,
    collection_name="langchain_messages"
)

# Create a ConversationBufferMemory from the message history
memory = ConversationBufferMemory(
    memory_key="history",
    chat_memory=message_history,
    return_messages=True  # Changed to True to get the actual message objects
)

# Get history
history_data = memory.load_memory_variables({})
history = history_data.get("history", "")
```

Figure 26: LangChain and MongoDB Interaction for Saving Chats Code

To transform user queries and document content into a machine-readable format, we utilized SentenceTransformers, a library which is built on top of HuggingFace Transformers. We used the BAAI/bge-large-en-v1.5 model to generate high-dimensional embeddings with semantic meaning. These were crucial in performing similarity searches against cached knowledge stored in the database.

```
# Embedding Model Configuration
embedding_model: "BAAI/bge-large-en-v1.5"
```

Figure 27: Vectorization Embedding Model Code

```
self.model = SentenceTransformer(config['embedding_model'])

self.text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=1024,
    chunk_overlap=100
)
```

Figure 28: Vectorization Embedding Model Parameters Code

The chat memory and knowledge base were cached in MongoDB, a low-maintenance NoSQL document-oriented database. MongoDB was chosen because of its ability to support unstructured data handling and how seamlessly it fit into LangChain's memory modules. We were able to store chat logs for each user session using MongoDBChatMessageHistory and enabled the chatbot to recall past interactions. Furthermore, the database stored metadata such as document source, chunk ID, and page number with each vector and enabled traceable and explainable responses.

```
# MongoDB Configuration
mongo_connection_str: "mongodb://localhost:27017"
database_name: "knowledge_base"
collection_name: "documents"
```

Figure 29: MongoDB Configuration Code

```
class PDFProcessor:
    def __init__(self, mongodb_uri: str = "mongodb://localhost:27017/", config_path: str = "config.yaml"):
        """Initialize PDFProcessor with MongoDB connection."""
        # Load config file
        with open(config_path, 'r') as file:
            config = yaml.safe_load(file)

        self.client = MongoClient(mongodb_uri)
        self.db = self.client.knowledge_base
        self.collection = self.db.documents

        self.model = SentenceTransformer(config['embedding_model'])

        self.text_splitter = RecursiveCharacterTextSplitter(
            chunk_size=1024,
            chunk_overlap=100
        )
```

Figure 30: PDF Processor Code

To process and prepare documents for embedding, we used LangChain's PyPDFLoader, which extracts text from PDF files, and RecursiveCharacterTextSplitter, which splits the text into usable, overlapping chunks. Chunks are decent in context and content to preserve the integrity of the employed embeddings in retrieval. Pre-processing was undertaken in a single script called pdf_processor.py.

Logic for retrieval was established within rag_module.py, where embedded user query matched the stored vectors by cosine similarity. The most appropriate document fragments are

retrieved and passed along with recent chat history into the GPT-4 model via LangChain's execution chain. This enables the system to generate responses that are not only accurate but also data grounded.

```python
def compute_cosine_similarity(self, vec1, vec2):
    """Compute cosine similarity between two vectors."""
    vec1 = np.array(vec1)
    vec2 = np.array(vec2)
    dot_product = np.dot(vec1, vec2)
    norm1 = np.linalg.norm(vec1)
    norm2 = np.linalg.norm(vec2)
    return dot_product / (norm1 * norm2)

def get_relevant_chunks(self, query: str, k: int = 5, score_threshold: float = 0.2) -> List[str]:
    """Retrieve relevant chunks from MongoDB using vector similarity."""

    query_embedding = self.embedding_model.encode(query)

    all_docs = list(self.collection.find({}, {'content': 1, 'embedding': 1}))

    similarities = []
    for doc in all_docs:
        similarity = self.compute_cosine_similarity(query_embedding, doc['embedding'])
        if similarity >= score_threshold:
            similarities.append((doc['content'], similarity))

    similarities.sort(key=lambda x: x[1], reverse=True)
    return [content for content, _ in similarities[:k]]
```

Figure 31: Vector Similarity Comparison Code

All the system configurations, including model names, database connections, API keys, and retrieval thresholds, were kept in a single config.yaml file. This provided maintainability and flexibility, where developers could modify without changing core logic.

```yaml
# config.yaml
# MongoDB Configuration
mongo_connection_str: "mongodb://localhost:27017"
database_name: "knowledge_base"
collection_name: "documents"

# OpenAI Configuration
model_name: "gpt-4"  # or "gpt-4" if you have access
openai_api_key: "sk--iIesrgwskbU4s1HxtelBn1sRt3bUvi9Tv2Dpu02M0T3BlbkFJ7aal4rANaVqWXRSI72klvx4Q4Zg7DiLslKikzi_8wA"
temperature: 0.3



# sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
# intfloat/e5-small-v2
# intfloat/multilingual-e5-large

# Embedding Model Configuration
embedding_model: "BAAI/bge-large-en-v1.5"

# Retrieval Configuration
retrieval_k: 8
similarity_threshold: 0.1
```

Figure 32: MongoDB, OpenAI, Embedding Model, and Retrieval Configuration Code

The Python libraries and modules used are detailed in the table below.

Library / Module	Purpose
os	Used to interact with the operating system (e.g., read files, list directories).
logging	Adds logging to track events like errors or status updates.
typing	Adds type hints (List, Dict, Optional) to improve code clarity.
yaml	Loads configuration values from a .yaml file for flexibility and separation of config from code.
numpy	Used for vector operations like cosine similarity.
langchain_community.document_loaders.PyPDFLoader	Extracts text from PDF files as LangChain Document objects.
langchain.text_splitter.RecursiveCharacterTextSplitter	Splits long text into smaller chunks, keeping context with overlaps.
langchain_core.prompts.ChatPromptTemplate	Creates structured prompts for GPT (template with variables like history, context).
langchain_openai.ChatOpenAI	Interface to GPT-4 or GPT-3.5 via OpenAI API.
langchain.schema.runnable.RunnablePassthrough	Passes data through the pipeline without modification (used to build chains).
langchain.schema.output_parser.StrOutputParser	Converts the LLM output into clean, readable text.
langchain.memory.ConversationBufferMemory	Stores and retrieves conversation history so the chatbot can handle follow-ups.
langchain_community.chat_message_histories.MongoDBChatMessageHistory	Saves and loads chat messages from MongoDB for persistent memory.
sentence_transformers.SentenceTransformer	Converts text into embedding vectors for similarity search and retrieval. Used for both document chunks and user queries.
pymongo.MongoClient	Connects to MongoDB. Used to store PDF chunk embeddings, metadata (e.g., page numbers, sources), and chat history.

Table 18: Python Libraries and Packages Used in the Implementation Demo

5.4 Implementation Process

Before finalizing the chatbot for deployment, multiple approaches were explored to ensure the most effective solution was selected. Even after identifying a strong candidate early in Chapter 3, the search continued to evaluate whether a better alternative existed. In total, four distinct trials were conducted, as outlined in the table below. The first two were command-line interface (CLI) based with no front-end component, while the fourth was web-only. All versions utilized LangChain and ChatGPT on the backend; however, the initial two relied solely on Python scripts rather than FastAPI. Throughout the trials, different vector stores and NLP libraries were tested—including Chroma Engine, FAISS (Facebook AI Similarity Search), and SentenceTransformers—to assess their impact on performance and vector retrieval quality.

Feature/Trial	Chatbot 1	Chatbot 2	Chatbot 3	Chatbot 4
Frontend	None (CLI only)	None (CLI only)	Flutter (Mobile + Web)	Streamlit (Web only)
Backend Framework	Python + LangChain + GPT-4	Python + LangChain + GPT-4	FastAPI + LangChain + GPT-4	Python + LangChain + GPT-4
Vector Store	Chroma	FAISS	SentenceTransformers + MongoDB	NA
Memory / Chat History	None	None	Persistent with MongoDB	In-Chat Context Only
Document Processing	PyPDFDirectoryLoader + Recursive Splitter	PyPDF2 + CharacterTextSplitter	PyPDFLoader + Recursive Splitter	PyPDFLoader + Recursive Splitter
API / Deployment	CLI only, no API	CLI only, no API	REST API with FastAPI	NA
UI/UX	None	None	Responsive, native-like with Flutter	Simple Web UI with Streamlit
Metadata Tracking	No	No	Yes	Yes
Key Strengths	Lightweight, real-time streaming	Detailed PDF parsing, fast similarity search	Cross-platform, scalable, memory-rich	Easy to deploy, modular
Main Limitations	No UI, no memory, no web/mobile	Redundant embedding, no UI	More complex setup	Web-only, simpler UI

Table 19: Chatbot Trials Comparison

Ultimately, the third iteration was selected for deployment, as it provided the most balanced solution in terms of front-end accessibility and back-end robustness. It supported both mobile and

web platforms, offered persistent conversation memory to enhance user experience, and utilized FastAPI to enable scalable and maintainable deployment. Additionally, it featured metadata tracking to support transparency and explainability in chatbot responses. Due to time constraints and recurring issues with mobile builds, the implementation was finalized as a web-based application using Flutter, rather than a standalone mobile app.

The decision to use GPT-4o mini was based on a balance of performance, cost-efficiency, and ethical considerations. Among the top AI models currently available, OpenAI's offerings stand out for their commitment to user privacy—an area where alternatives like Google's Gemini raise concerns due to the company's broader data collection practices. GPT-4o mini strikes an ideal balance by delivering much of the robustness and accuracy of GPT-4o while maintaining faster response times and a lower cost than even GPT-3.5.

In addition to its efficiency and affordability, GPT-4o mini is well-suited for real-time applications like chatbots, thanks to its optimized architecture and reduced latency. It handles nuanced language understanding, supports multilingual input, and performs well in low-resource environments. Its ability to integrate smoothly with tools like LangChain and FastAPI also made it a strong fit for our system's backend infrastructure. Overall, GPT-4o mini allowed us to provide intelligent, responsive interactions without compromising on speed, cost, or data ethics.

Model	Price per 1M Tokens (Input / Output)	Privacy & Data Handling	Speed (Tokens/sec)	Hallucination Rate (%)
GPT-4o Mini	$0.15 / $0.60	OpenAI offers option to opt out of training; built-in end-to-end encryption.	~68.4	~10–15% (estimated)
GPT-4o	$2.50 / $10.00		~105.3	~60% (SimpleQA benchmark)
GPT-3.5	$0.50 / $1.50		~27.6	~20–30% (estimated)
Gemini 2.5 Pro	$1.25	Operated by Google data handling policies.	~54.16	~37.1% (SimpleQA benchmark)
DeepSeek R1	$0.55 / $2.19	Potential data sharing with Chinese government, third-party security.	Not specified	Up to 83% failure in accuracy tests
Claude 3.7 Sonnet	$3.00 / $15.00	Anthropic emphasizes privacy	Not specified	Low (estimated)

		and data handling practices.		

Table 20: Most Popular AI Agent Comparison

The decision to use BGE-large as the embedding model was driven by its strong balance between retrieval accuracy and computational efficiency. In retrieval-based systems, the retrieval_k parameter determines how many of the most similar document chunks are fetched from the vector database. A smaller k improves speed but risks omitting relevant information, while a larger k increases the chance of retrieving irrelevant or noisy data. Given the length and complexity of our documents, a larger k was necessary, making it critical to choose an embedding model that could handle this without sacrificing precision.

BGE-large excelled in this context by offering high-quality embeddings that preserved semantic meaning across longer inputs. It provided more accurate retrieval results compared to smaller models while remaining lightweight enough for efficient deployment. Furthermore, its open-source nature made it a flexible and transparent choice, allowing for easier customization and integration into our pipeline. Overall, BGE-large enabled the system to retrieve contextually relevant information at scale, making it a key component of our backend architecture.

Model	**k**	**Query**	**Comment**
BGE-large-v1.5	3	"What is CCTS in details?"	Good, but missed some specific details
	5		Well-balanced, but skipped some important parts like date.
	8		Very detailed and complete, best performance
all-MiniLM-L6-v2	3		Missed core details like organizer and venue
	5		Better, but still missing location
	8		Slightly more accurate, still less depth than BGE
e5-small-v2	3		Short and to the point, lacks background
	5		Relevant but concise; didn't mention publication detail
	8		Improved context, but not as rich as BGE

Table 21: Embedding Vectorization Packages Comparison

5.5 Faced Problems and Solutions

During the execution phase of the AskDAH project, several technical and integration challenges were encountered. These issues had a direct impact on both system performance and the user interface, making their resolution critical to the success of the final solution. The following section provides an overview of the most significant problems faced and the approaches used to address them.

5.5.1 Vectorizing the Documents

Transformation of the university documents to embedding vectors was the first step in enabling the chatbot to produce correct responses. We first experienced sub-optimal vector quality due to inconsistent text chunking and formatting. Long paragraphs were chunked as a whole, and therefore low semantic accuracy and irrelevant retrieval outputs were achieved. To solve this, we used a preprocessing pipeline with PyPDFLoader to extract raw text and RecursiveCharacterTextSplitter to cut the content into tiny pieces that overlap each other. This allowed the embedding model (BAAI/bge-large-en-v1.5) to encode useful context in each piece, significantly improving the chatbot's ability to extract and reason over useful content.

5.5.2 Citing the Sources

The second fundamental challenge was obtaining the chatbot to provide the pages or documents the answers were extrapolated from. Users might doubt the validity of the answers without source attribution. Our initial chatbot responses were purely generative, and no reference was made to the source of information. To fix this, we modified the retrieval-augmented generation (RAG) chain in LangChain. We ensured that metadata such as the source file name and page number were stored with each embedded chunk in MongoDB. These were then appended to the prompt template sent to GPT-4. As a result, the chatbot began referencing its sources naturally in its responses, enhancing both trust and transparency.

5.5.3 Back Referencing JSON Data to Flutter and Rendering it Appropriately

Merging the backend output with the front-end in Flutter, we were facing problems while parsing and rendering the JSON data that the API was providing us. There were fields missing, fields being null, or fields being in other formats we were not anticipating, leading to runtime exceptions and absence of UI rendering. This issue was quite rampant in the events module and rendering of chatbot messages. To remedy this, we enhanced both layers, namely, backend and

frontend. Backend, we standardized response structure with the same key name and defaulting. On Flutter, we provided null-safety checks, placeholder defaults for fields that are not required, as well as better error handling in data parsing. These allowed error-free communication between back and front ends and guaranteed a hassle-free user experience.

5.6 Conclusion

In conclusion, this chapter detailed the execution phase of the AskDAH project, highlighting the technologies and strategies that shaped its development. The use of Flutter for the user interface and LangChain with FastAPI for the AI-powered backend allowed for the creation of a responsive and intelligent system aligned with the project's goals.

A key decision during implementation was the shift from a mobile-first to a web-based deployment, prompted by limitations in Android testing environments. This change ensured a smoother development process without compromising core functionality.

The team addressed a range of technical challenges—such as document-based semantic retrieval, chatbot source citation, and backend-frontend communication—through collaborative problem-solving and iterative development. These efforts not only improved the system's performance but also strengthened competencies in integration and API management.

The modular design of the system, comprising independent yet connected components like the document loader, vector store, memory, and reasoning engine, enables flexibility for future expansion. Potential enhancements include multilingual support, additional data sources, and personalized user experiences.

Overall, while the execution phase posed several challenges, it successfully translated the initial concept into a functioning prototype. This sets a solid foundation for the next stage: evaluating the system's performance, usability, and real-world effectiveness, as discussed in the following chapter.

Chapter 6

Testing

Chapter 6: Testing

6.1Introduction

This chapter presents the comprehensive testing processes conducted to evaluate AskDAH's functionality, reliability, and user experience. The testing phases include unit testing, integration testing, system testing, and usability testing, each designed to validate specific aspects of the chatbot's performance. Unit testing ensures the accuracy of individual components, while integration testing examines how different modules interact. System testing assesses the application's overall stability, and usability testing focuses on user feedback to refine interface accessibility and efficiency. By systematically analyzing these results, this chapter provides insights into the chatbot's effectiveness, identifies key areas for improvement, and lays the foundation for future enhancements.

The app went through multiple levels of testing to ensure quality, performance, and usability. We combined automated and manual testing to validate different components of the app. Feedback was collected from real university students from various majors.

6.2 Testing Method

6.2.1 Unit Testing

In the unit testing phase, each functional component of the AI-powered chatbot application was systematically evaluated to ensure accuracy, reliability, and performance. Unit tests were designed to validate core functionalities, namely response generation and event retrieval. Mock data sets were employed to simulate user interactions, enabling verification of chatbot responses against expected outputs. Additionally, edge cases were examined, including ambiguous queries and incorrect input formatting, to strengthen robustness. MongoDB integration was tested to confirm proper data handling, ensuring that event information, chat history, and user preferences were stored and retrieved seamlessly. In short, unit testing ensured each unit functions correctly in isolation. Individual components were tested, such as:

- Chatbot response function
- Event calendar retrieval
- Save chat feature

6.2.2 Integration Testing

In the integration testing phase, the various components of the application were systematically tested to ensure seamless interoperability. This process involved validating the interactions between the chatbot, MongoDB database, and event management modules. Data flow was examined to confirm that user queries were correctly processed and responses generated based on retrieved contextual information. Special attention was given to verifying the chatbot's ability to accurately pull academic schedules, event details, and administrative support resources. API calls between the chatbot and external service OpenAI for natural language processing were rigorously tested for reliability and latency. Edge cases, such as delayed API responses and data retrieval inconsistencies, were addressed to optimize system robustness. By conducting integration testing across multiple scenarios, the application was refined to ensure smooth operation and a cohesive user experience and verified interactions between modules:

- Chatbot integration with knowledge base
- Calendar sync with backend
- Settings and UI features like dark mode and saved chats

6.2.3 System Testing

In the system testing phase, the entire application was evaluated as a cohesive unit to ensure that all integrated components function correctly in real-world conditions. This phase involved verifying interactions between the chatbot, database, and external APIs, such as OpenAI for natural language processing. System-wide tests were conducted to assess the application's responsiveness, stability, and performance.

It was planned to test the chatbot against a Microsoft Copilot chatbot by feeding both bots the same documents and comparing the answers. Due to budget and administrative constraints on the university student Microsoft account, it was unfortunately not possible.

6.2.4 Performance Testing

Since this project is a baseline prototype and not designed for scalability, performance testing is not applicable. The current implementation focuses on establishing core functionality rather than optimizing high-load environments. As a result, stress testing, load balancing, and response time analysis for large-scale deployment are outside the project's scope. The chatbot and event management features are designed primarily for proof-of-concept validation rather than

handling extensive concurrent usage. Future iterations may incorporate performance evaluation once scalability becomes a priority, but for this phase, the emphasis remains on ensuring correct functionality and user experience within a controlled environment.

6.2.5 Usability Testing

The usability testing phase assessed the chatbot's functionality, efficiency, and overall user experience based on key tasks performed by diverse users. Real-time usability tests were conducted with 9 anonymous students across different departments and majors. The evaluation criteria included the ability to message the chatbot, the accuracy and comprehensiveness of responses, viewing the calendar and settings, saving chats, enabling dark mode, and overall experience.

There are 3 levels of criteria used in usability:

1. **Excellent (1):** This means that the software is easy to use and clear according to the performance of this task.
2. **Acceptable (0.5):** This means that the user is satisfied with the performance level of this task.
3. **Unacceptable (0):** This means that the user is facing some problems in this task.

The following table shows the average time spent on accomplishing the tasks:

Task	Criteria	The amount of time to accomplish task
Task 1:	Can message chatbot	***>0 sec***
Task 2:	Correct and comprehensive answers	***4 sec***
Task 3:	View calendar	***3 sec***
Task 4:	View settings	***3 sec***
Task 5:	Overall experience	***3 min***
Task 6:	Dark mode	***>0 sec***
Task 7:	Save chats	***2 sec***

Table 22: Usability Testing Task Criteria and Duration

Results indicate that fundamental interactions, such as sending messages and retrieving responses, were consistently rated as excellent. However, certain functionalities, such as accessing settings and saving chats, showed variation in user experience, with some users rating them as acceptable. A notable success is that all tasks received a rating "Acceptable" or higher, with no tasks earning an "Unacceptable" rating. Additionally, users provided valuable feedback on areas for improvement that would improve their experience. User suggestions are detailed further in chapter 7.

Task /User	U1	U2	U3	U4	U5	U6	U7	U8	U9	Total	Average	Task Performance
1	1	1	1	1	1	1	1	1	1	9/9	100%	Excellent
2	1	0.5	0.5	1	0.5	1	1	1	0.5	7/9	77.8%	Acceptable
3	0.5	0.5	1	1	1	1	1	1	0.5	7.5/9	83.3%	Acceptable
4	1	1	1	1	0.5	1	1	1	1	8.5/9	94.4%	Acceptable
5	1	0.5	1	1	0.5	1	1	1	0.5	7.5/9	83.3%	Acceptable
6	1	1	1	1	1	1	1	1	1	9/9	100%	Excellent
7	0.5	1	1	1	1	1	1	1	1	8.5/9	94.4%	Acceptable

Table 23: Usability Testing Results

Overall, core usability metrics were met successfully, and the feedback underscores opportunities to refine accessibility, enhance user engagement, and address minor inconsistencies to optimize the chatbot's effectiveness in real-world support scenarios. The attitude of the users showed that they quickly understood, trusted, and benefited from the solution. Notable insights include:

- 100% of tasks were completed successfully by all users.
- Users found the chatbot accurate, helpful, and fast.
- The interface was intuitive, and navigation was seamless showing ease of use.
- Users especially liked the dark mode and saving chat features.

6.3 Conclusion

In this chapter, the testing phases confirmed that the chatbot successfully meets its core requirements, providing users with a reliable and accessible tool facilitating their lives. Unit and integration testing ensured seamless interaction among system components, while system testing

validated its overall stability. Usability testing highlighted strengths in fundamental interactions while identifying opportunities to refine response formatting, expand faculty listings, and enhance accessibility features like language support and personalized notifications. Although performance testing was excluded due to the project's prototype nature, the results reinforce the chatbot's viability as a working minimum value product. Moving forward, these findings provide valuable direction for future improvements, guiding the development of a more refined, scalable, and user-friendly academic assistance system

Chapter 7
Results and Discussion

Chapter 7: Results and Discussion

7.1 Introduction

In this chapter, an overview of the development and real-world impact of AskDAH is shown. It includes key interface snapshots, user testing feedback, system architecture diagrams, and highlights from the University of Jeddah Digital Transformation hackathon. This chapter illustrates how the chatbot addresses critical challenges in university communication, offering a streamlined and intuitive solution for students.

7.2 Project Snapshots

The screenshots of the chatbot below show how users interact with the chatbot, including examples of typical queries, responses, interface features like dark mode, and chat saving.

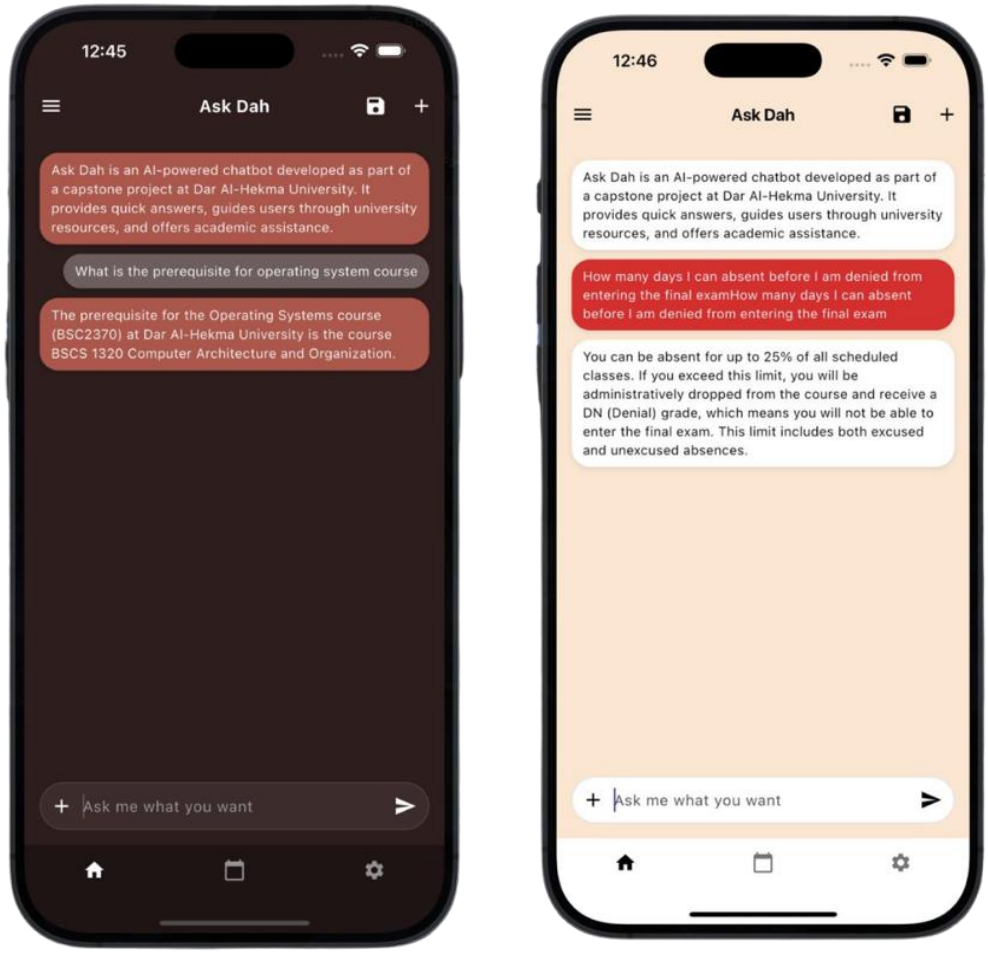


Figure 33: Demo Sample Chatbot Interaction in Dark Mode and Light Mode

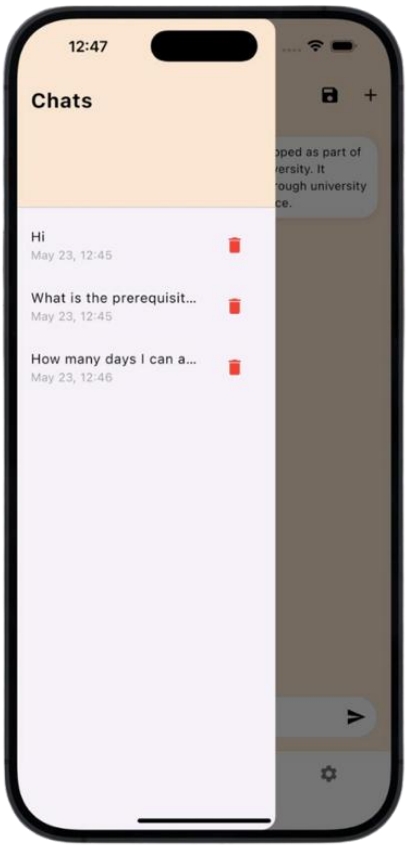


Figure 34: Demo Chatbot Saved Chats Menu

The figures below show screenshots of the event calendar in-app. It shows all university events in chronological order with the option to filter by department or community service hour opportunities.

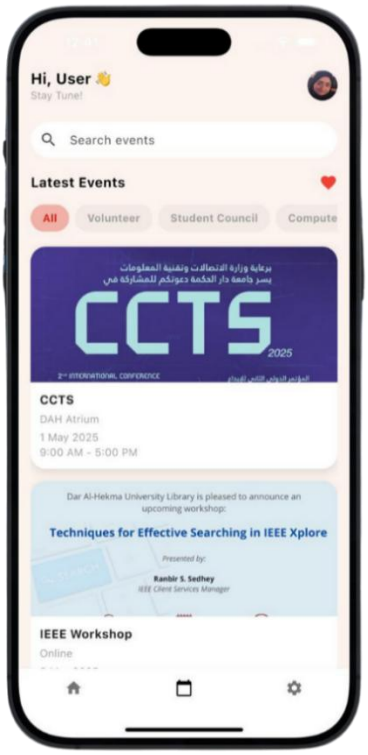


Figure 35: Demo Event Calendar Interface in Light Mode

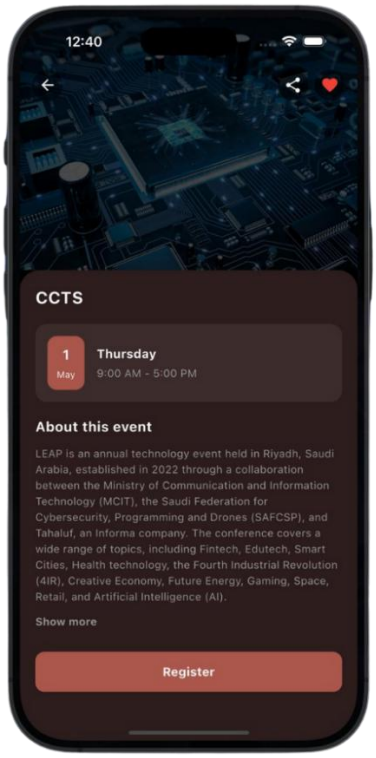


Figure 36: Demo Event Details Interface in Dark Mode

The figure below shows a screenshot of the supplemental excel file was used to record user testing results in the format shown below. The full user testing forms are available in appendix c.

Task	Criteria	The amount of time to accomplish task	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	***>0 sec***	1		
Task 2:	Correct and comprehensive answers	***4 sec***	1		
Task 3:	view calendar	***3 sec***		1	
Task 4:	view settings	***3 sec***	1		
Task 5:	overall experience	***3 min***	1		
Task 6:	Dark mode	***>0 sec***	1		
Task 7:	Save Chats	***2 sec***		1	
Comments	Reminds of due dates without asking (personalized notifications)				
Major	SLHS				

Table 24: Unit Testing Excel Form

The figure below shows the architecture the chatbot follows is retrieval augmented generation for smart, context-aware answers.

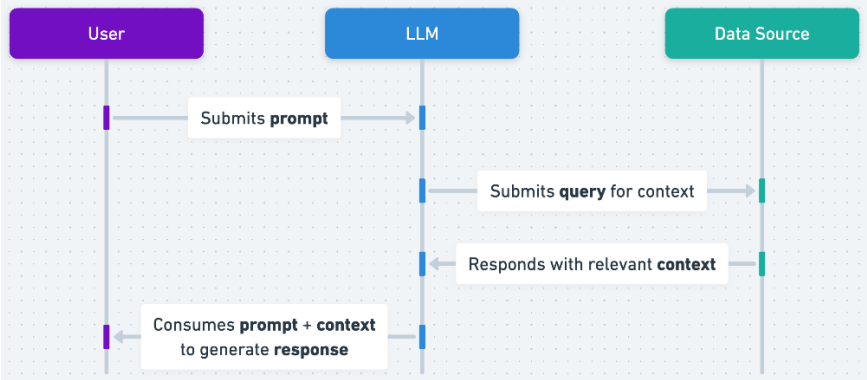


Figure 37: RAG Architecture (Source: OpenAI)

The figure below shows the calendar architecture which is customized to the application at hand and the technologies used.

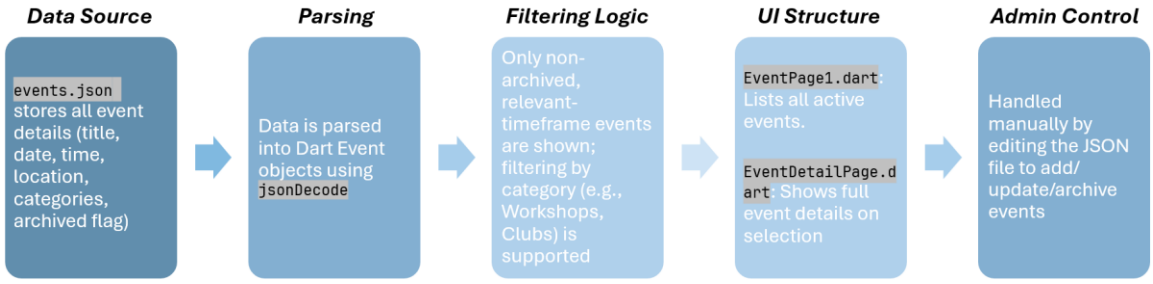


Figure 38: Calendar Architecture

User feedback was collected as well. The figure below displays a compiled list of the key suggestions received. These suggestions highlight key opportunities for future iterations, enhancing accessibility, personalization, and overall user experience.

Personalized Notifications	• Ability to remind users of due dates without needing to ask.
Semester Schedule Integration	• Include academic courses and schedules to enhance usability.
Improved Answer Formatting	• Structure chatbot responses into concise points rather than long paragraphs.
Floor Plan Navigation	• Implement a feature allowing users to ask for directions within university buildings.
Augmented Reality Map	• Provide an interactive, AR-based map for campus navigation.
Faculty Listings	• Expand the database to specify faculty positions and include more departments.
Arabic Language Support	• Offer an interface with Arabic language options to improve accessibility.
University Map Guide	• Implement a built-in map with location-based assistance for navigating campus facilities.
Expanded Event Calendar Filters	• Improve filtering options to accommodate different majors and community service preferences.

Figure 39: Compiled User Feedback Suggestions

We had the immense honor of winning first place in the University of Jeddah Digital Transformation hackathon, part of the prestigious First International Conference on Artificial Intelligence and Internet of Things. Out of 50 applying projects and 11 finalists, we won the title on May 8th, 2025, as shown in the figure below.

Figure 40: The Team Winning First Place at the UoJ Hackathon

7.3 Project Outcomes

Modern universities have become increasingly complex, making it difficult for students to access timely and accurate information. Traditional communication channels, such as email and face-to-face interactions, often lead to delays, confusion, and missed messages. Simple administrative or academic inquiries can take unnecessarily long to resolve, frustrating both students and staff. Additionally, institutional information is scattered across multiple platforms—including Outlook, SIS, Blackboard, university websites, emails, and nondisclosed files—making it challenging to find a single source of truth. This fragmentation contributes to overwhelming email traffic, delayed responses from administrators, and a lack of efficient, centralized communication.

Our project successfully addresses these challenges by providing a streamlined, intuitive mobile application designed to serve as a single point of truth for students. The AI-powered chatbot functions as an immediate, accessible resource for answering administrative and light academic queries, reducing the reliance on slow, inefficient email exchanges. Additionally, the centralized event calendar ensures students can easily view academic and community service events, filtered by major, without needing to search across multiple disconnected platforms. By eliminating unnecessary delays, reducing email burdens, and consolidating crucial information into a single, user-friendly app, this solution significantly improves accessibility, efficiency, and student experience within the university ecosystem.

The development and evaluation of AskDAH, the application solution, yielded significant achievements, solidifying its potential as an innovative academic support tool. Throughout testing, the chatbot successfully demonstrated core functionalities, including seamless user interactions, accurate information retrieval, and intuitive navigation assistance. Despite limitations such as non-scalability and incomplete institutional integration, the project's proof-of-concept validation was overwhelmingly positive.

A major highlight of the project's success was securing first place in the University of Jeddah Digital Transformation hackathon under the First International Conference on Artificial Intelligence and Internet of Things. This recognition underscores the chatbot's impact and alignment with the university's vision for technological advancement. Furthermore, the enthusiastic reception from students, faculty, and staff reflected the real-world demand for such an application. Many who encountered the project passionately advocated for its official implementation, emphasizing its practical benefits and user-friendly approach.

These outcomes affirm the chatbot's viability and establish a strong foundation for future improvements. With additional data sources, refined personalization, and expanded institutional integration, the project has the potential to evolve into a fully scalable, widely adopted academic assistance platform.

7.4 Limitations

While AskDAH successfully demonstrates its core functionalities, several limitations of the minimum viable product demo that was implemented impact its scalability and broader implementation.

1. **Non-Scalable Architecture:** This prototype is designed as a baseline model without scalability considerations, meaning performance optimization for high-volume usage was not a priority. Future iterations would require structural modifications to support larger user bases and concurrent interactions, such as using a service aside from MongoDB and storing the events in a structured SQL format rather than a simple JSON file.
2. **Lack of Azure Integration**: Due to the nature of this project, the chatbot is unable to link with Microsoft Azure services for authentication and cloud-based functionality because the university's IT department is unable to grant access to the

Azure dashboard for students. This limits its ability to leverage enterprise-level security and seamless institutional integration.

3. **Resource Constraints**: Time limitations and the size of the development team restricted extensive refinement and optimization. This affected the depth of testing, debugging efficiency, and additional feature implementation. Features such as the favoriting events, notifications, and GPA calculation were not implemented.
4. **Incomplete Data Sources**: The chatbot currently lacks comprehensive institutional data, as it does not yet pull information from all departments. This results in gaps in academic schedules, faculty listings, and administrative resources. Expanding document sources is critical for improving response accuracy.
5. **Custom Test Case Limitations:** While testing focused on core functionalities, specialized test cases for edge scenarios—such as multi-department interactions, data inconsistencies, and complex navigation requests—remain limited. Broader testing is necessary for more refined AI behavior.
6. **Usability and Personalization Enhancements**: Feedback from usability testing indicates the need for additional features, such as personalized notifications, Arabic language support, refined response formatting, an augmented reality map, and expanded faculty directories. These enhancements require further development efforts.

Although these limitations define the current scope of the project, they provide a valuable roadmap for future development. Addressing these challenges would significantly enhance the chatbot's capability, usability, and institutional relevance.

7.5 Conclusion

This chapter highlighted the strengths, challenges, and overall impact of the AI-powered university chatbot. The project snapshots provide a visual representation of user interactions, system architecture, and recognition at the University of Jeddah Digital Transformation hackathon, affirming its effectiveness. The overview section showcases how the chatbot streamlines university communication, centralizing information and reducing email congestion, significantly improving accessibility and response time for students. Meanwhile, the limitations section outlines

key constraints such as scalability, incomplete data sources, and Azure integration challenges, offering insights into areas for future enhancement.

Despite these limitations, the overwhelmingly positive reception from students and employees—many of whom expressed a strong desire for its official implementation—solidifies the chatbot's potential as a transformative tool in university administration. The discussion points provide a foundation for refining its functionality, addressing institutional gaps, and ultimately expanding its capabilities in future iterations.

Chapter 8

Conclusion and Future work

Chapter 8: Conclusion and Future work

8.1 Project Summary and Benefits

This project introduced AskDAH, a minimum viable product (MVP) that combines a centralized university event calendar with an AI-powered chatbot. The system is designed to enhance event coordination and streamline student support services across campus. Students and staff can create and view events in one unified platform while receiving instant answers to common academic or administrative queries via the chatbot.

The solution aims to improve communication, simplify campus navigation, and elevate the overall user experience. Key benefits of the system include:

- **Enhanced Accessibility**: Support for multiple languages and voice interaction broadens usability for users with varying needs and preferences.
- **Improved User Experience**: AI-driven personalization delivers relevant, timely support tailored to individual students' academic journeys.
- **Centralized Communication**: A unified platform fosters smoother coordination between university departments and students.
- **Data-Driven Development**: Built-in analytics provide insights into user interaction, allowing for iterative improvement over time.
- **Future-Readiness**: The system's modular and scalable design allows it to grow with institutional needs and technological advancements.

8.2 Future Enhancements

While the current version of AskDAH successfully demonstrates core features—including chatbot integration and event management, there are several potential enhancements for future development:

1. **Multilingual Support**: Expanding language options (e.g., Arabic and English) to reflect the diversity of the university community.
2. **Voice Interaction**: Adding voice-to-text and text-to-speech capabilities to improve accessibility and allow hands-free operation.

3. **Real-Time System Integration**: Connecting the platform with live university systems (e.g., course registration or student information systems) to provide up-to-date, context-aware support.
4. **Advanced Personalization**: Leveraging AI to predict student needs based on academic history or engagement, offering proactive assistance.
5. **Analytics and Feedback Loops**: Implementing tools for tracking usage patterns and collecting user feedback to support data-informed refinements.
6. **Scalability**: Preparing for large-scale deployments through high-capacity vector databases and cloud-based infrastructure.

The chatbot also offers potential for expanded tools such as GPA calculators, course-specific help, and interactive study schedules—all aimed at improving academic outcomes.

8.3 Campus Navigation System

One notable future feature is a campus navigation system. This service would convert the university's floor plan into a node map—potentially using a linked list structure to represent rooms, hallways, staircases, and elevators. When a student requests directions from point A to point B, an A* algorithm would identify the optimal path. This route would then be translated into natural language by ChatGPT and presented to the user.

In case a user gets lost or deviates from the suggested route, the chatbot could assist in reorientation using nearby landmarks. This navigation feature could later be enhanced through augmented reality integration, offering visual guidance within the app.

8.4 Alignment with Vision 2030

This project aligns strongly with Saudi Arabia's Vision 2030 in several key areas:

- *Digital Transformation*: By utilizing AI and automation, AskDAH improves the quality and efficiency of digital services within the university.
- *Quality of Life Program*: Enhances campus life by making services more engaging, accessible, and user centric.
- *Education and Lifelong Learning*: Equips students with access to modern tools that support their academic growth and tech proficiency.

- *Local Content and Innovation*: Encourages the development of locally driven AI solutions, promoting student-led innovation and contributing to national research goals.
- *Government Efficiency*: Automates routine inquiries, reducing administrative burden and reflecting principles of smart governance.

8.5 Conclusion

The development of AskDAH marks an important step in reimagining student support systems in higher education. Its modular design, AI-powered functionality, and potential for personalization make it a strong foundation for continued innovation. With further development, this system can evolve into a comprehensive digital assistant—enhancing learning experiences, streamlining communication, and supporting Saudi Arabia's broader goals for digital transformation and educational excellence.

AskDAH transforms the student's experience by offering real-time, personalized support—far beyond what traditional help desks or portals provide. With Microsoft authentication, OpenAI integration, and a centralized event calendar, the application streamlines information sources, reduces email overload, and addresses real student needs. Built for scalability and aligned with the university's digital strategy, this tool isn't just innovative, it's practical, student-centered, and ready to redefine campus life. AskDAH isn't just solving a student problem, it's building the foundation of a smart, student-friendly campus in line with Vision 2030 via a project made by students, for students.

References

[1] S. Alavi, A. Hojjat, and R. Farahani, "The effectiveness of chatbots in higher education: A systematic review," *J. Educ. Technol. Soc.*, vol. 25, no. 3, pp. 1–17, 2022.

[2] K. M. Alraimi, H. Zo, and A. P. Ciganek, "User acceptance of e-learning systems in developing countries: An empirical study," *J. Comput. Inf. Syst.*, vol. 55, no. 3, pp. 46–54, 2015.

[3] M. Colombo and A. Zangari, "Chatbots for higher education: A systematic review," *Int. J. Educ. Manag.*, vol. 34, no. 2, pp. 333–348, 2020.

[4] N. Dabbagh and A. Kitsantas, "Personal learning environments, social media, and self-regulated learning: A natural formula for connecting formal and informal learning," *Internet High. Educ.*, vol. 15, no. 1, pp. 3–8, 2012.

[5] P. D. Eckel and B. King, "An exploratory study of student perceptions of academic support services," *J. Coll. Stud. Dev.*, vol. 55, no. 3, pp. 250–254, 2014.

[6] A. Følstad and P. B. Brandtzaeg, "Chatbots for customer service: A systematic literature review," in *Proc. 20th Int. Conf. Human-Computer Interaction Mobile Devices Services*, pp. 1–12, 2017.

[7] J. Forrester and Z. Pan, "IT support and the student experience: A study of higher education institutions," *J. High. Educ. Policy Manag.*, vol. 43, no. 2, pp. 163–176, 2021.

[8] J. C. Hearn and M. Smith, "Faculty-student interaction and student engagement in higher education: Implications for student success," *Rev. High. Educ.*, vol. 45, no. 4, pp. 735–760, 2022.

[9] J. Hirschberg and C. D. Manning, "Advances in natural language processing," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 112, no. 33, pp. 10130–10132, 2015.

[10] J. Hill, W. Ford, and I. Farreras, "Artificial intelligence for customer service: The role of chatbots in service delivery," *J. Serv. Manag.*, vol. 26, no. 3, pp. 373–393, 2015.

[11] Y. Jia and S. Xu, "Chatbots in education: A review of applications, challenges, and prospects," *Int. J. Educ. Technol. High. Educ.*, vol. 18, no. 1, pp. 1–21, 2021.

[12] S. Kim and H. Kim, "Ethical implications of chatbots in higher education: A framework for research and practice," *J. Comput. Assist. Learn.*, vol. 37, no. 2, pp. 279–290, 2021.

[13] D. Koch and L. Schuster, "The role of AI chatbots in higher education: A case study of student perceptions," *Educ. Technol. Res. Dev.*, vol. 69, no. 3, pp. 1179–1199, 2021.

[14] L. R. McKinney and C. R. Graham, "Toward personalized learning in higher education: The role of academic advising," *Int. J. Educ. Technol. High. Educ.*, vol. 14, no. 1, pp. 1–13, 2017.

[15] A. Murphy and D. Evans, "The impact of campus navigation technologies on student experience," *Stud. High. Educ.*, vol. 44, no. 8, pp. 1452–1468, 2019.

[16] L. Reddy and P. Kalra, "The impact of chatbots on student engagement in higher education: A review of the literature," *Int. J. Innov. Educ.*, vol. 5, no. 4, pp. 317–328, 2018.

[17] R. Robles and F. de Sousa, "Assessing the impact of AI chatbots on student satisfaction and retention: Evidence from higher education institutions," *J. Educ. Comput. Res.*, vol. 57, no. 8, pp. 2007–2026, 2019.

[18] M.-A. Sicilia and M. González, "Information overload in higher education: The role of digital communication channels," *Comput. Educ.*, vol. 161, art. no. 104078, 2021.

[19] A. Shah and M. Hossain, "Smart campus technologies and their impact on student experience: A systematic review," *J. Educ. Technol. Syst.*, vol. 49, no. 2, pp. 134–162, 2020.

[20] S. B. Shum and R. Ferguson, "Social learning and AI: A research agenda," *Int. J. Artif. Intell. Educ.*, vol. 28, no. 4, pp. 517–525, 2018.

[21] J. M. Stewart and L. Padilla, "Improving event management in higher education: A case study," *J. High. Educ. Policy Manag.*, vol. 40, no. 5, pp. 489–502, 2018.

[22] B. P. Woolf and H. C. Lane, "AI and education: A research agenda," *AI Soc.*, vol. 33, no. 2, pp. 195–210, 2018.

[23] L. Zhou and L. Liu, "AI in education: The role of chatbots in enhancing learning," *J. Educ. Technol. Soc.*, vol. 23, no. 3, pp. 11–21, 2020.

[24] OpenAI, “Retrieval-augmented generation (RAG) and semantic search for GPTs,” *OpenAI Help Center*. [Online]. Available: https://help.openai.com/en/articles/8868588-retrieval-augmented-generation-rag-and-semantic-search-for-gpts. [Accessed: Nov. 19, 2024].

[25] KFUPM, “KFUPM becomes first Saudi University to launch ChatGPT Edu,” *KFUPM News*. [Online] Available: news.kfupm.edu.sa/news/kfupm-becomes-first-saudi-university-to-launch-chatgpt-edu/75/ [Accessed: May 18, 2025].

[26] Q. Alqahtani and O. Alrwais, " Building a Machine Learning Powered Chatbot for KSU Blackboard Users," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, pp. 781–789, 2023.

[27] Princess Nourah bint Abdulrahman University - ITC, “طالبات نورة”, *Apple App Store.* [Online] Available: https://apps.apple.com/kn/app/%D8%B7%D8%A7%D9%84%D8%A8%D8%A7%D8%AA-%D9%86%D9%88%D8%B1%D8%A9/id1601087538 [Accessed: May 18, 2025].

[28] Effat University, “Effat Mobile”, *Apple App Store.* [Online] Available: https://apps.apple.com/us/app/effat-mobile/id736974909 [Accessed: May 18, 2025].

Appendix A – Interviews and Questionnaires

A) Staff Interview Questions:

1. Position

- Director/Head (8)
- Manager (2)
- Employee (3)

2. What are the most common types of inquiries your department receives from undergraduate students?

- How to register for courses (new students).
- What courses to take (guidance for new students).
- Prerequisite courses and their importance.
- Differences in tracks by spring semester.
- Section schedules for the next semester.
- Changing course or section, drop/add procedures.
- Questions on the academic calendar:
- Course offerings (schedules, postponement implications on graduation).
- Withdrawal periods.
- Summer course openings.
- Midterm and vacation schedules.
- How to register for courses (new students).
- What courses to take (guidance for new students).
- How to write an attractive CV.
- Reviewing CVs.
- Internship opportunities and database.
- Internship letters for senior students (one-day processing).
- Job opportunities and professional certificates:
- Common jobs after earning the degree.
- Most relevant professional certifications.
- Access to career workshops and competitions (national/international).
- Invitations to webinars and conferences.

- courses and their importance.
- Differences in tracks by spring semester.
- Section schedules for the next semester.
- Changing course or section, drop/add procedures.
- Questions on the academic calendar:
- Course offerings (schedules, postponement implications on graduation).
- Withdrawal periods.
- Summer course openings.
- Midterm and vacation schedules.
- Is Ramadan online?

3. How does your department currently communicate important information (POS change, events, etc.) or announcements to students?

- Advising is done through Blackboard, ensuring students feel supported during shifts.
- Older students should not feel disconnected due to changes, so maintaining a consistent advising approach is important.
- Email is the primary mode of communication:
- Faculty and program directors use email to reach out to students.
- Important course-related information, deadlines, and events are communicated via email.
- Emails are also used for student queries, including requests for personal or academic information.
- Faculty members may also communicate directly with students via email for class-specific needs.
- Blackboard is used by program directors and instructors to contact students.
- It serves as the platform for course registration, announcements, and accessing academic resources.
- Face-to-face interactions are conducted when needed for personalized advising or discussing academic concerns.
- It’s an effective method for in-depth guidance and clarifying any complex issues.
- A help desk system may be utilized for addressing technical or administrative queries.
- It’s a formal system used to manage and resolve student inquiries.
- WhatsApp is sometimes used, but it is considered unofficial.
- The "Big Sis/Lil Sis" program may also be utilized for peer-to-peer support, providing students with additional informal channels to seek advice or information.

- Posters in the Arch Wing are used for advertising important events or announcements, particularly around registration periods or deadlines.
- The program director may contact students directly through Blackboard for any program-specific needs or updates.
- Emails are used for sending information about events, conferences, and research-related topics.
- SIS (Student Information System) is sufficient for handling personal information and communicating about academic progress.
- Students have access to their personal data through the system for privacy and security.

4. Are there any challenges in providing timely support to students?

- Many students don't check their emails regularly, leading to important messages being missed.
- The volume of emails is overwhelming, causing key information to get lost among other communications.
- It typically takes up to two days to respond to student inquiries via email. However, if something is urgent, students are encouraged to visit the office directly.
- Students are not showing interest in extracurricular activities, and they often do not attend events, despite their importance for career development.
- Students may not be aware of the university's working hours or need to stay beyond regular hours, leading to confusion around availability for academic activities.
- Students are often unaware of important processes and policies, including requirements like obtaining ethics approval for research, which the instructor is responsible for.
- There is a lack of clarity about course-related procedures, both from students and instructors.
- Students sometimes need more clarification regarding assignment deadlines and expectations. While the advisor tries to offer full support, it can be difficult to give individual attention to every student, especially during high-demand periods like registration time.
- If students face difficulties or issues in communication, they should be encouraged to provide feedback through course evaluations, helping improve future interactions.

5. Do you have any concerns about students not being aware of certain services or resources your department offers?

- Students don't typically read emails, making email communication less effective for delivering important information.
- Universities rely on websites, orientation days, and emails for communicating necessary details to students. However, emails tend to be ignored, and additional channels might be needed.
- A special payment plan is available for students who need it, ensuring they have flexibility in paying their fees.
- The institution offers all the essential information via websites, orientation sessions, and emails, but students may still struggle with engagement or accessing this information effectively.

6. Do you believe that the existing platforms (SIS, Blackboard, Outlook) effectively cover all your needs for communication with students?

- There are challenges in getting a lot of students to attend events, possibly due to poor engagement or communication.
- There is a belief that focusing on one communication platform is more effective than spreading information across multiple platforms. Multiple platforms can lead to neglect or confusion.
- The institution is working on integrating AI via WhatsApp to automatically respond to important questions, streamlining communication and improving response times.

7. What gaps do you see between them?

- Professionally, yes it is sufficient. But main pathway should be enough, Bb and email, whatsapp is optional.
- Important to have all information like bb sis and outlook in one platform
- It is difficult to reach the students
- Student don't Read emails and thats big issue so they use WhatsApp instead
- Suggested: event registration in the event management software. SUPPORT: Consultations in the research studio (common sesne of the research, not subject specific.)

8. Sector?

- **Educational (7)**
- **Administrative (5)**

9. How do you currently inform students of administrative procedures or deadlines?

- **SMS Messages**
- **Outlook Email**

10. What are the most common reasons students contact you outside of class?

- Personal Support

- absence excuses
- Lecture/Assignment/Project Questions
- Personal Support
- Change Grade
- Understand grade assignment

11. Do you find the current methods of communication between faculty and students effective?

- Yes (5)
- Somewhat (2)

12. In your opinion, what kind of technology would help improve student engagement with important department announcements?

- email sort out, filter by sender genre or email content, cc or bcc alone, all faculty, personal emails
- Traditional - verbal sharing, in addition toe ails, have you checked?
- reaching to the office needs appointment but otherwise it is easy, but not very efficient
- social media , he already proposed for each department to have social media account, or encourage student to use.
- Event management calendar

13. Do students respond well to existing methods? How is attendance, interaction? Do you think students are aware of what's happening?

- Unaware, they do not know until too late because of the clutter of emails. Maybe if they knew they would interact. Orientation had good attendance, students are aware.
- Low interaction, unaware of whats happening.
- They don't know, we ask the instructor to put an announcement on bb
- low attendance. unaware or not motivated
- "yes, it’s sufficient but the problem if the student find easy to use . we have to consider research "
- Very low

14. Additional comments?

- check outlook features

- Freshamn: types the name of the request, help me in xyz. Where should i go? Direct to the necessary section? Where is xyz?
- Prospective students (out of our scope? Maybe use the dame model and integrate it into the website, or into the WhatsApp API.) ALL ADMISSION INFORMATION. Events and activites, organizations outside of DAH through us (student affairs invitations) so it includes those as well. Simplify the information the DAH site is trying to convey to students and internal persons.
- High importance marking in Outlook. (Auto flags.) easy UI/UX, stupid login credentials. Push notifications. Kicks user out every time. Notifs tab. Smart app. Easily accessible and flat lurning curve. Customizable fonts and sizes, color contrast.
- What are references I can go to? Major-related FAQs (job prospect, best track for me, why? for my future?) how can i improve my CV? who can i contact (students or faculty or staff) if I need help with xyz?
- he we be happy to provide us with information for our project
- They are working in plan of improving payment link they usually send to students to pay
- Decision making, weighing the pros and cons.
- Open communication for the students to us and for us to contact them. Students are not telling us they published things, so we don't know and cannot celebrate them. Have them tell us. Policies, inform stydents. Internal and external events? Department-specific. What can we do?
- Similar to Effat's, eventually. Box/link for suggestions. News about the university. For complaints, reach out to student affairs. Uploading images, filter by category, seeing friends. Things happening today - at the forefront. Daily feed. Winner/achievements. Make it for more than questions.

B) Student Survey Questions

Seven (7) informal student interviews were taken with general notetaking incorporated into the survey findings. The rest of the 94 respondents took the survey below. The questions are shared first, and the results are attached as figures following the questionnaire.

Section 1 (All Respondents)

University Experience Survey - استبيان التجربة الجامعية

Welcome tothe Survey! We are Computer Science students at Dar Al-Hekma University, working on acapstone project to improve university life. Your insights are essential tous. This brief survey will take just 3-4minutes of your time. Your honest feedback will help us develop atool designed tomake your university experience more seamless and connected. Thank you for your participation!

مرحبًا بكم في الاستبيان! نحن طالبات علم حاسب بجامعة دار الحكمة، ونعمل على مشروع تخرج لتحسين الحياة الجامعية. آراؤكم مهمة جدًا بالنسبة لنا. سيستغرق هذا الاستبيان القصير 3-4 دقائق فقط من وقتكم. ملاحظاتكم الصادقة ستساعدنا في تطوير أداة مصممة لجعل تجربتكم الجامعية أكثر سلاسة وترابطًا. شكرًا لمشاركتكم!

1. Gender - الجنس

- Male – ذكر
- Female – أنثى

2. Level – المستوى

- Freshman - أول سنة
- Sophomore - ثاني سنة
- Junior - ثالث سنة
- Senior - رابع سنة

3. How often do you face challenges navigating the university campus (e.g., finding offices, halls, or your classes)?

كم مرة واجهت تحديات في التنقل داخل حرم الجامعة (مثل العثور على المكاتب أو القاعات أو فصولك الدراسية)؟

- Always – دائمًا
- Sometimes – أحيانًا
- Rarely – نادرًا
- Never – أبدًا

4. Have you ever wished for quicker access to university services or answers to your questions?

هل تمنيت يومًا الوصول بشكل أسرع إلى الخدمات الجامعية أو الحصول على إجابات لأسئلتك؟

- Yes – نعم
- No – لا

5. How easy is it to find information related to your professors (e.g., email, office hours, office location)?

ما مدى سهولة العثور على المعلومات المتعلقة بأساتذتك (مثل البريد الإلكتروني، ساعات المكتب، موقع المكتب)؟

[NUMBER RATING]

- 1 - Very Difficult - صعب للغاية
- 6 - Very Easy - سهل للغاية

6. Which platform(s) do you use to communicate with professors? (Select all that apply)

(ما هي المنصة/المنصات التي تستخدمها للتواصل مع أساتذتك؟ (اختر جميع ما ينطبق

- Email - البريد الإليكتروني
- Blackboard – بلاكبورد
- WhatsApp - واتس أب
- In person – شخصيًا
- Microsoft Teams - مايكروسوفت تيمز
- Other …

7. How effective are the communication methods you use with professors?

ما مدى فعالية قنوات التواصل التي تستخدمها حاليًا؟

[NUMBER RATING]

- 1 - Very Ineffective - غير فعّالة على الإطلاق
- 6 - Very Effective - فعّالة للغاية

8. Do you use any apps to organize your daily tasks (university-related or not)?

هل تستخدم أي تطبيقات لتنظيم مهامك اليومية (سواء كانت متعلقة بالجامعة أم لا)؟

- Yes, I use multiple apps -تعم، أستعمل عدة تطبيقات
- Yes, I use one main app - نعم، أستعمل واحدًا فقط
- No – لا
- Other …

9. How important is it to you that digital tools you use are customizable (ex. color, theme, widget organization)?

ما مدى أهمية أن تكون الأدوات الرقمية التي تستخدمها قابلة للتخصيص (مثل اللون، والثيم، وتنظيم الودجتات) بالنسبة لك؟

[NUMBER RATING]

- 1 - Not Important - ليس مهم
- 6 - Very Important - مهم جدًا

10. Are you a current student of Dar Al-Hekma University?

هل أنت طالبة حالية بجامعة دار الحكمة؟

- Yes – نعم
- No – لا

Section 2 (DAH Students Only)

Dar Al-Hekma Experience - تجربة جامعة دار الحكمة

This project is custom-tailored to our campus, so your answers are extra-valuable to us.

هذا المشروع مصمم خصيصًا لحرمنا الجامعي، لذا فإن إجاباتك ذات قيمة كبيرة بالنسبة لنا.

11. How do you stay informed about university events or extracurricular activities? (select all that apply)

(كيف تبقى على اطلاع بالأحداث الجامعية أو الأنشطة اللامنهجية؟ (اختر جميع ما ينطبق

- Email - البريد الالكتروني
- WhatsApp - واتس أب
- Campus Flyers and Posters - ملصقات ولائحات الحرم الجامعي
- Word of Mouth - من حديث الناس
- Other …

12. Have you ever missed out on events or activities due to the current communication method?

هل فاتتك يومًا حدث أو نشاط بسبب طريقة التواصل الحالية؟

- Yes – نعم
- Sometimes – أحيانًا
- No – لا

13. Do you feel that the university provides enough resources to help you stay informed?

هل تشعر أن الجامعة توفر موارد كافية لمساعدتك على البقاء على اطلاع؟

- Yes – نعم
- Kind of - نوعًا ما
- Not really - ليس كثيرًا
- No – لا

14. How overwhelmed do you feel by the number of platforms (SIS, Blackboard, Outlook) you need to use?

التي تحتاج إلى استخدامها؟ (Outlook، Blackboard، SIS ما درجة شعورك بالإرهاق من عدد المنصات (مثل

[NUMBER RATING]

- 1 - Not at all - ليس على الإطلاق
- 6 - Extremely – بشدة

15. If there was an AI assistant that could help you find class locations or answer basic questions (e.g., academic calendar, schedules, room locations, events, accounting deadlines, registration), how likely would you be to use it?

إذا كان هناك مساعد ذكي يمكنه مساعدتك في العثور على مواقع الفصول الدراسية أو الإجابة عن الأسئلة الأساسية (مثل التقويم الأكاديمي، الجداول، مواقع القاعات، الأحداث، مواعيد المحاسبة، التسجيل)، ما مدى احتمال استخدامك له؟

[NUMBER SCALE]

- 1 - Not at all – أبدًا
- 6 - Always – دائمًا

16. Would you be interested in a feature that lets you see which friends or colleagues are attending events or joining activities?

هل ستكون مهتمًا بميزة تتيح لك رؤية الأصدقاء أو الزملاء الذين يحضرون الفعاليات أو يشاركون في الأنشطة؟

- Yes - نعم
- Maybe – ربما
- No – لا

17. Would you find it helpful if the app recommended events or activities based on your interests or academic major?

هل ستجد من المفيد إذا كان التطبيق يقوم بتوصية على الفعاليات أو الأنشطة بناءً على اهتماماتك أو تخصصك الأكاديمي؟

[NUMBER RATING]

- 1 - Not helpful at all - ليس مفيدًا على الإطلاق
- 6 - Very Helpful - مفيد جدًا

18. Have you used any AI assistants or chatbots (e.g., Siri, Alexa, ChatGPT, Gemini)? If so, how satisfied are you with them?

هل سبق واستخدمت أي مساعد ذكي أو روبوتات محادثة (مثل Gemini، ChatGPT، Alexa، Siri)؟ إذا كان الأمر كذلك، ما مدى رضاك عنها؟

- Satisfied – راضي
- Neither satisfied nor dissatisfied - لست راضي أو غير راضي
- Dissatisfied - غير راضي
- I have not used any AI assistants or chatbots - لم استخدم أي من هذه التقنيات

19. What are some features you would expect the DAH app AI chatbot to have, or expect it to be able to do?

ما هي بعض الميزات التي تتوقع أن يمتلكها روبوت المحادثة الذكي في تطبيق جامعة دار الحكمة، أو ما الذي تتوقع أن يكون قادرًا على فعله؟

[OPEN-ENDED]

Section 3 (Non-DAH Respondents Only)

Other Universities - الجامعات الأخرى

20. Please enter your university's name.

الرجاء مشاركتنا اسم جامعتك.

- KAAU - جامعة الملك عبدالعزيز
- UBT - جامعة الأعمال والتكنولوجيا
- Effat University - جامعة عفت
- Jeddah University - جامعة جدة
- Other …

Survey Results:

1. Gender - الجنس

More Details Insights

Male - ذكر	13
Female - أنثى	77

2. Level - المستوى

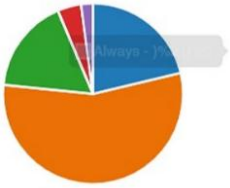
More Details Insights

Freshman - أول سنة	21
Sophomore - ثاني سنة	24
Junior - ثالث سنة	9
Senior - رابع سنة	34
Other	2

3. How often do you face challenges navigating the university campus (e.g., finding offices, halls, or your classes)?
كم مرة واجهت تحديات في التنقل داخل حرم الجامعة (مثل العثور على المكاتب أو القاعات أو فصولك الدراسية)؟

More Details

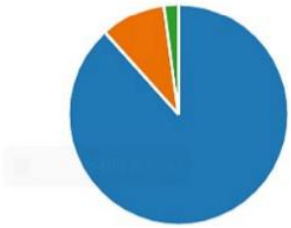
Always - دائمًا	20
Sometimes - أحيانًا	52
Rarely - نادرًا	16
Never - أبدًا	4
Other	2



4. Have you ever wished for quicker access to university services or answers to your questions?
هل تمنيت يومًا الوصول بشكل أسرع إلى الخدمات الجامعية أو الحصول على إجابات لأسئلتك؟

More Details

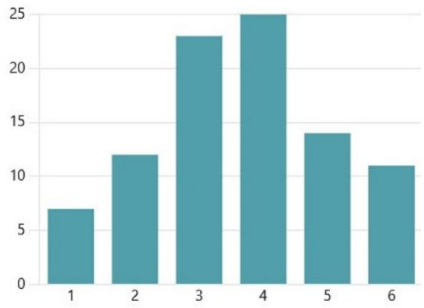
Yes - نعم	83
No - لا	9
Other	2



5. How easy is it to find information related to your professors (e.g., email, office hours, office location)?
ما مدى سهولة العثور على المعلومات المتعلقة بأساتذتك (مثل البريد الإلكتروني، ساعات المكتب، موقع المكتب)؟

More Details Insights

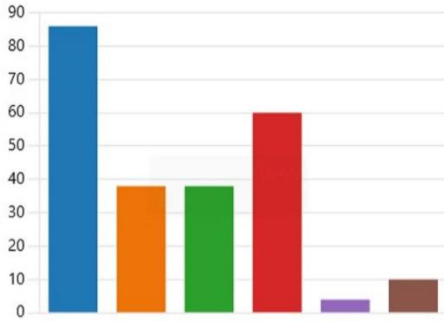
3.65
Average Rating



6. Which platform(s) do you use to communicate with professors? (Select all that apply)
ما هي المنصة/المنصات التي تستخدمها للتواصل مع أساتذتك؟ (اختر جميع ما ينطبق)

More Details

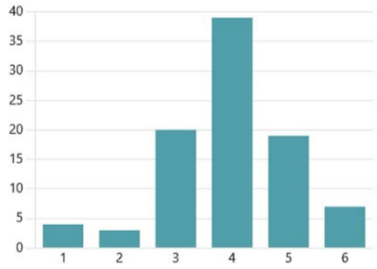
Email - البريد الإليكتروني	86
Blackboard - بلاكبورد	38
WhatsApp - واتس أب	38
In person - شخصيًا	60
Microsoft Teams - كروسوفت تيمز...	4
Other	10



7. How effective are the communication methods you use with professors?
ما مدى فعالية قنوات التواصل التي تستخدمها حاليًا؟

More Details Insights

3.95
Average Rating



8. Do you use any apps to organize your daily tasks (university-related or not)?
هل تستخدم أي تطبيقات لتنظيم مهامك اليومية (سواء كانت متعلقة بالجامعة أم لا)؟

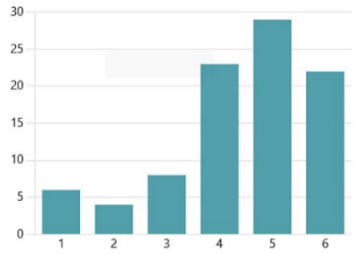
More Details

Yes, I use multiple apps -تطبيقات...	26
Yes, I use one main app - فقط ...دًا	29
No - لا	34
Other	5

9. How important is it to you that digital tools you use are customizable (ex. color, theme, widget organization)?
ما مدى أهمية أن تكون الأدوات الرقمية التي تستخدمها قابلة للتخصيص (مثل اللون، والثيم، وتنظيم الودجتات) بالنسبة لك؟

More Details Insights

4.51
Average Rating



10. Are you a current student of Dar Al-Hekma University?
هل أنت طالبة حالية أو خريجة جامعة دار الحكمة؟

More Details

Yes - نعم	37
No - لا	35
Other	22

11. How do you stay informed about university events or extracurricular activities? (select all that apply)
كيف تبقى على اطلاع بالأحداث الجامعية أو الأنشطة اللامنهجية؟ (اختر جميع ما ينطبق)

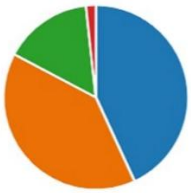
More Details

Email - البريد الالكتروني	54
WhatsApp - واتس أب	24
Campus Flyers and Posters - ...عي	17
Word of Mouth - من حديث الناس	23
Other	4

12. Have you ever missed out on events or activities due to the current communication method?
هل فاتتك يومًا حدث أو نشاط بسبب طريقة التواصل الحالية؟

More Details

Yes - نعم	25
Sometimes - أحيانًا	23
No - لا	9
Other	1



13. Do you feel that the university provides enough resources to help you stay informed?
هل تشعر أن الجامعة توفر موارد كافية لمساعدتك على البقاء على اطلاع؟

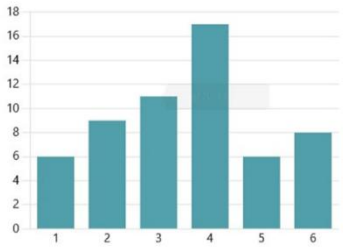
More Details Insights

Yes - نعم	8
Kind of - نوعًا ما	41
Not really - ليس كثيرًا	6
No - لا	2
Other	1

14. How overwhelmed do you feel by the number of platforms (SIS, Blackboard, Outlook) you need to use?
ما درجة شعورك بالإرهاق من عدد المنصات (مثل SIS، Blackboard، Outlook) التي تحتاج إلى استخدامها؟

More Details Insights

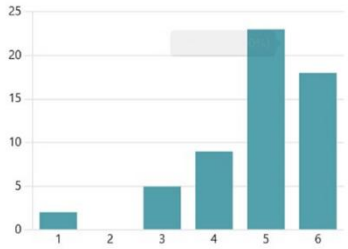
3.56
Average Rating



15. If there was an AI assistant that could help you find class locations or answer basic questions (e.g., academic calendar, schedules, room locations, events, accounting deadlines, registration), how likely would you be to use it?
إذا كان هناك مساعد ذكي يمكنه مساعدتك في العثور على مواقع الفصول الدراسية أو الإجابة عن الأسئلة الأساسية (مثل التقويم الأكاديمي، الجداول، مواقع القاعات، الأحداث، مواعيد المحاسبة، التسجيل)، ما مدى احتمال استخدامك له؟

More Details Insights

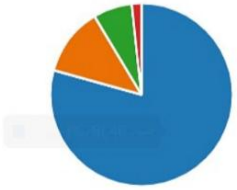
4.93
Average Rating



16. Would you be interested in a feature that lets you see which friends or colleagues are attending events or joining activities?
هل ستكون مهتمًا بميزة تتيح لك رؤية الأصدقاء أو الزملاء الذين يحضرون الفعاليات أو يشاركون في الأنشطة؟

More Details Insights

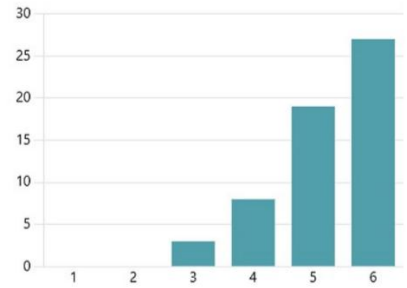
Yes - نعم	46
Maybe - ربما	7
No - لا	4
Other	1



17. Would you find it helpful if the app recommended events or activities based on your interests or academic major?
هل ستجد من المفيد إذا كان التطبيق يقوم بتوصية على الفعاليات أو الأنشطة بناءً على اهتماماتك أو تخصصك الأكاديمي؟

More Details Insights

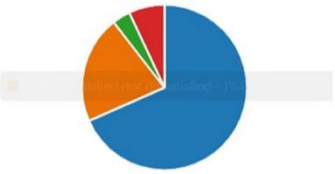
5.23
Average Rating

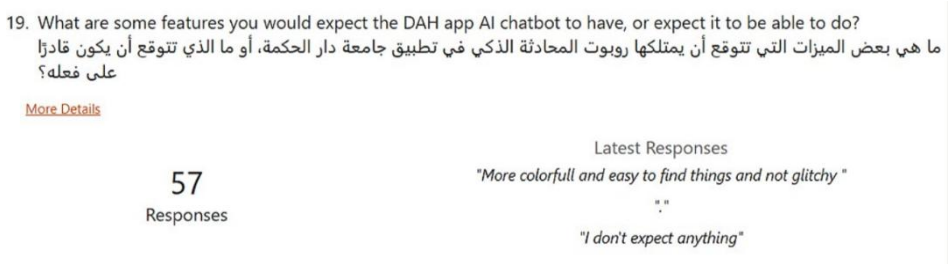


18. Have you used any AI assistants or chatbots (e.g., Siri, Alexa, ChatGPT, Gemini)? If so, how satisfied are you with them?
هل سبق واستخدمت أي مساعد ذكي أو روبوتات محادثة (مثل Siri، Alexa، ChatGPT، Gemini)؟ إذا كان الأمر كذلك، ما مدى رضاك عنها؟

More Details

Satisfied - راضي	39
Neither satisfied nor dissatisfied...	12
Dissatisfied - غير راضي	2
I have not used any AI assistants...	4



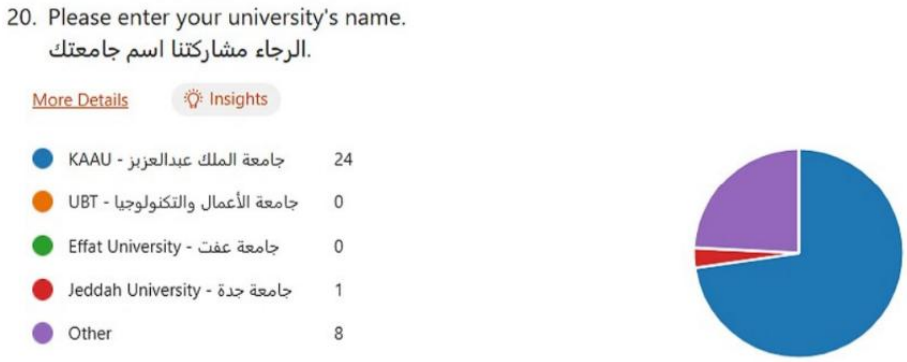


Question 19:

- Know the system well
- Real time information about event updates and announcements
- i can ask it if something is against the unversity rules or not, and it will tell me. also accepts suggestions
- probably help with navigation and make it easy or appear as a map or gps yk and probably have any commonly asked questions
- with answers for them and mainly just help visualize students time and help with time management :)
- Answer administrative queries such as schedule issues, registration questions, finance/accounting questions. Have the working
- hours of the different departments within the university, ie Student development office, registration office, accounting desk etc.
- Direct me to where I can access information, ie library portal, bb, SIS, petition system or be able to centralise and do most of not
- all applications with me
- يخصص و ينظم وقت در اسة الطالب و ايضا يش ر ح له المحاض ر ات بطريقة مختصر ة
- Provide assistance at any time, ensuring users can get help whenever needed.
- If somebody got lost at uni, maybe the user can enter the room/class/lab no. and it generates an easy roadmap to that specific
- class.
- معر فه اسم الدكاتر ه وار قام مكاتبهم وطريقه التواصل معاهم
- توجيه للفعاليات ايضا
- answer specific questions related to the campus or the major and be updated on current activities and changes.
- إجابة على للأسالة
- التواصل
- اشعار بالفعاليات

- Organize tasks and deadline
- Effective Workshops suggestions
- Amazing idea! Good luck ✨
- Job and internship listings
- Info on clubs
- Info on food vendors
- Feedback or suggestions on personal academic journey
- Tutoring collaborations or services
- Make things and access easier for less techy people
- Things keeping me know that events and clubs are going to happen
- يقدر يسوي كل ش ر ي احتاجه
- Navigation
- I have never used it so I don't know
- I think should be the most helpful for new students to introduce to the university and offerings. Also help create in registration
- process of courses in each semester based on each students GPA and interests.
- تنبيهات بالمهمات مواعيد الصلاه مواعيد الفعاليات
- notify events and reminders for events
- Answer my questions quickly and efficiently.
- Help in finding classrooms , DR offices and directions events with dates and email to contact if interested in volunteering or participating
- Calculate my gpa
- I would like it to answer a questions whether a professor is free or in a meeting, where certain rooms are (which floor, north or
- south side of the campus etc), and much more
- Generate automatic emails to my instructors . Remind me about my upcoming quiz , assignments and to dos .
- Keeping track of your grades, assignments to motivate students and even qoutes or advices if they need. Like personal uni
- assistant!
- Fill it with all the details of the professor so that we know if we can find her today in office or no - saves a lot of time!
- I don't know

- Help me with classes locations
- locations of classes and answers of basic questions
- I dont really know
- A map of the campus labelling everything
- Connected to my SIS
- Good memory
- To tell me if there are parking spots available, based on how many parked their cars that day.
- معر فة التقويم الأكاديم ي و اماكن القاعات و معر فة جدول ي اليوم ي
- Idk
- telling when advisors or professors are in their offices or not
- I would like for dar al hekma to allow the use of ai within policy
- something to help with navigation for new ppl using sis for the first time
- put alarms automatically
- helpful guides to campus and student profiles to make friends :)
- Have a calendar for future events
- Provide direct help without twists and turns
- Schedules, help with assignments and where classes are
- I don't expect anything
- More colorfull and easy to find things and not glitch



Appendix B – Minutes of Meeting

The meeting minutes included in this appendix are crucial for understanding the project's development and decision-making processes. They provide a detailed record of discussions, agreements, and action items, promoting transparency and accountability among team members. These minutes were meticulously recorded by team members Leena, Fatma, Rahaf, and Rihab on a rotating basis, ensuring thorough documentation of key decisions and action points. By reviewing these minutes, readers will gain valuable insights into the challenges encountered and the strategies employed to meet the project's objectives.

Capstone I

Meeting #1

Date: 9/9/2024 - Monday
Time: 10:00-11:00am
Venue: Meeting Room 284
Attendees: Dr Imed Ben Dhaou, Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

- Finalize capstone project idea
- Define the requirements
- Lay steps to be competed by next meeting next week

Discussion:

- The final idea is an internal DAH application, complete with an AI assistant and fully curated news/activities. It will draw data from the email announcements, custom to the student's role as a student, staff, or faculty. It will focus on addressing issues internal to DAH by its inhabitants.
- Requirements gathering will likely occur through interviews with the department and select students. A questionnaire needs to be designed to discreetly identify what the users need.
- Potential WhatsApp API? Tentative
- Next steps discussed in meeting later this evening (Meeting #2)

Meeting #2

Date & Time: 9/9/2024 – Monday, 7-8pm
Venue: Microsoft Teams
Attendees: Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

1. decide who will record the MoM every time we meet
2. define the DTA for smooth sailing
3. outline the project timeline (tentatively at least)
4. basically, laying the foundation for the capstone

Discussion:

- The one in charge of recording the meeting minutes in alternating order: Leena, Fatima, Rahaf, Rihab.
- Target audience: internal to DAH employees and students.
- Mandate downloading the app for DAHers.
- Requirements gathering (focus until next week):
    - Interviewing 1) students 2) staff (each department) 3) faculty
    - Non-functional requirements
    - Design questionnaire for interviews.
- Functional:
    - Syncing it with outlook, student handbook, academic calendar
    - AI assistant (low-level intelligence) can answer basic questions
    - Locate hall locations (maybe)
- Non-functional:
    - Color changes according to user's major/school
    - Application security to ensure CIA.
    - Sign in via Microsoft (uni email) and through it access database

Plan of Action:

- Design questionnaire to gauge what the requirements are from the interviewees without outright saying it (2 girls) (L, Ri)
- Define/list the different sections and stakeholders in the university, the different departments and people we will need to approach based on hierarchy. (1 girl) (Ra)
- Research online platforms, social media, other similar apps in other countries, what they have etc. on what students want. (1 girl) (F)

Finish the above tasks by the weekend inshAllah.

Start interviews on Sunday + Monday.

Send emails to all the DAH people we need to interview.

- Interviewing departments (2 girls) (all)
- Interviewing students (1 girl, 1 girl) (all)

Meeting # 3

Date: Monday, 16/9/2024
Time: 10:30-11:00 AM
Venue: University, Second Floor
Attendees: Fatma Alamoudi, Leena Althekair, Rihab Asif, Rahaf Alshabrawi
Discussion Points:
• Progress Review
• Action Items & Next Steps
• Additional Comments & Future Plans
Progress Review:

• Gantt Chart Creation: We finalized the Gantt chart outlining tasks and deadlines for the remainder of the term. This will serve as our roadmap, ensuring we stay on track.
• Departmental Outreach: A comprehensive list of departments to contact for interviews has been compiled. This is essential to gather diverse input for our project.
• Survey Results: Successfully collected 100 student responses, providing valuable insights into their expectations and needs for our project.

Action Items & Next Steps:
• Upcoming Interviews: We will schedule interviews with the following individuals to deepen our research and ensure a comprehensive perspective:
    • Dr. Samar Altarteer (Research Center)
    • Ms. Tasneem Kabli
    • Dr. Malak Abunar
•Research on Generative AI:
    1. Determine the required data volume for AI training.
    2. Investigate various generative AI models, assess their features, and decide which model fits best for our use case.
    3. Explore large language models (LLMs) and evaluate their applicability to our project, particularly for natural language understanding.
3 Additional Comments & Future Plans:
    • Gantt Chart Review: While the Gantt chart has been completed, it will be formally reviewed and approved in the next meeting, after which it will guide our execution.
    • Chatbot Demo: We aim to build a preliminary version of the chatbot. This demo will be used to showcase potential functionalities and gather feedback from stakeholders.
    • Collaboration with Cybersecurity Team: Given that our project includes the development of an app, we will collaborate closely with the cybersecurity team. Ensuring data privacy and protection will be critical, especially since we'll be handling sensitive data.
    • Meeting Room Requests: Going forward, we will coordinate all meeting room requests through Ms. Karima Alkatheri to ensure availability and smooth scheduling.

Meeting #4

Date: Monday 30/9/2024
Time: 10:00-10:30am
Venue: Meeting Room 284
Attendees: Dr Imed Ben Dhaou, Leena Althekair, Rihab Asif, Rahaf Alshabrawi
Discussion:

- What we done so far
- What should we do next
- Additional Comments

What we done so far:

- 100 responses (survey) from students
- Interviewed all departments
- Gantt chart for the rest of the term

What should we do next:

- Interview with Research center Dr.Samar Altarteer
- Interview with Ms.Tasneem Kabli
- Interview with Dr.Malak Abunar
- We need to start with researching about the process to use generative Ai:
  1 - How many data will we use/need
  2 - What the models are there and what we will use
- Explore large model languages

Additional Comments:

- Gantt chart is good (approved)
- We should make demo of chatbot (mini one)
- We should collaborate with cybersecurity team who are developing a security app since we are doing app
- We start to send the meeting room requests from Ms.Karima Alkatheri

Meeting #5

Date: 7/10/2024 (Monday)
Time: 10:30 - 11:00 AM
Venue: Meeting Room 284
Attendees: Dr. Imed Ben Dhaou, Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

- Updates on interviews conducted with various departments.
- Status update on project phases.
- Complete PowerPoint presentation and send for supervisor feedback by tonight.
- Discussion on the rubric for the project proposal presentation.

Discussion:

- Proposal Presentation:
    - The proposal PowerPoint should be finalized and sent to the supervisor for feedback. Necessary changes should be made in a timely manner.
- Interview Updates:
    - All interviews are completed except for Dr. Tasneem.
    - The requirements phase of the project is complete. The team is working on the project idea proposal this week, as the presentation is scheduled for Wednesday.
- Presentation Structure:
    - The presentation should be 5-7 minutes maximum. The following sections were discussed for inclusion:
    - Introduction
    - Problem Statement
    - Proposed Solution
    - Methodology
    - Project Timeline (Gantt chart)
    - Emphasis should be placed on highlighting the strengths of the project idea, particularly why OpenAI GPT-4 is being used for the project.
    - The presentation should include examples of where similar technology is being applied and demonstrate its effectiveness.

Meeting #6

Date: 14/10/2024 - Monday
Time: 10:00-10:30 AM
Venue: Meeting Room 284
Attendees:

1. Dr. Imed Ben Dhaou

2. Leena Althekair
3. Rahaf Alshabrawi
4. Fatima Alamoudi

Agenda:

- officially close initiation phase
- review requirements w dr imed
- establish next week's tasks (prototyping)

Discussion:

- Payment & API Keys: Investigate tutorials, experiment with integration and payment processes.
- Reviewmat Tool: Research paper submission processing using Google credits.
- LLM Fine-Tuning: Emphasize the importance of prompt engineering and its impact on chatbot responses. Focus on LLM scalability (scholarships, training) and reinforcement learning with human feedback.
- Documentation: Ensure the documentation is thorough to allow future teams (even non-technical) to continue the project.
- Arabic & English Focus: Arabic prompt engineering is important, but English remains the primary language focus.
- Exploring Multiple LLMs: Experiment with Llama, ChatGPT, and others to see which fits the project's needs.
- Live Data: Data collection from sources like Twitter or previous questions to improve the chatbot's training and accuracy.
- Version Control: Consider updates for university policy or documents, ensuring the chatbot stays current with new versions (V1, V2, etc.).
- Accuracy: Aim for 90% accuracy in responses. Many understand LLMs, but how they function is less known; this needs to be clarified in documentation.
- Data Analytics: Possibility of adding data analytics features for insights and improvements.

Next Steps:

1. Continue research on LLM models.
2. Start the research portion using YouTube links and other resources.
3. Investigate GitHub for package errors and expert Q&A.
4. Collect more data for training the chatbot and explore fine-tuning methods.
5. Investigate tutorials for payment and key integration
6. Reviewmat research and credits processing
7. Fine-tuning considerations for LLMs
8. Next steps in documentation and chatbot scalability

Resources:

https://www.youtube.com/watch?v=q5HiD5PNuck
https://www.youtube.com/watch?v=yo0qy7xyd3A
https://www.youtube.com/watch?v=YstMV8GM_zU
https://www.spa.gov.sa/en/N2169413
https://www.anthropic.com/api

Meeting #7

Date: 22/10/2024 - Monday Time: 10:00-10:30 AM Venue: Meeting Room M80-82

Attendees: Fatima Alamoudi, Leena Althekair, Rahaf Alshabrawi, Rihab Asif
Agenda:

- •Discuss the functionality of the chatbot
- •Review user authentication process
- •Plan data handling and mapping for class locations
- •Discuss the application design in Figma

Discussion:

1. Chatbot Functionality:
    - o Research conducted on how the chatbot will operate, focusing on user interaction and responsiveness.
    - o Authentication will be handled using MAAHA20, which will access user information such as email, full name, major, and academic level.
2. User Experience:
    - o After signing in, users will have access to the chatbot.
    - o The chatbot will utilize vector databases to enhance functionality, ensuring efficient data retrieval.
    - o Vector data will be split into chunks to optimize processing and performance.
3. Application Design in Figma:
    - o The design of the application is currently being created in Figma, allowing for visual representation and user interface design.
    - o Two design approaches are under consideration, but a final decision has yet to be made. The team is evaluating factors such as user flow, aesthetics, and functionality for each option.
    - o Feedback from team members will be essential in selecting the most effective design.
4. Data Handling with Dio Package:
    - o The Dio package in Flutter will send user questions along with relevant sources, streamlining communication.
    - o This includes a header, rules, and student information.
    - o An alternative package, HTTP, was considered but not preferred due to the need for additional programming; Dio is already configured for use.
5. Microsoft Authentication:
    - o For authentication with Microsoft, location permissions for classes need to be enabled to provide a seamless user experience.
6. Class Location Mapping:
    - o ChatGPT cannot read floor maps directly. A solution involves programming floor layouts in Python, converting them into a usable map format.
    - o An A* algorithm will be implemented to determine the fastest path to a location, sending this information to ChatGPT for visual representation.

Next Steps:

- •Finalize the chatbot's authentication process using MAAHA20.
- •Continue refining the application design in Figma and make a final decision on the design approach.
- •Ensure proper integration of the Dio package for data handling.
- •Develop the Python mapping program to visualize class locations.
- •Implement the A* algorithm for efficient pathfinding.

Meeting #8

Date: 11/11/2024 - Monday
Time: 10:00-10:30 AM
Meeting Room: Behind the Library
Attendees: Leena Althekair , Rahaf Alshabrawi, Fatima Al-amoudi, Rihab Asif
Agenda:

- Review Prove of Concept
- What should we close this week work phase
- Discuss the application design in Figma

Discussion:

- We reviewed the proof of concept for our project and discussed why the code wasn't working as expected. We identified potential issues and brainstormed solutions to address them.
- We also reviewed the first draft of the app design in Figma and proposed a few enhancements. Specifically, we suggested adding a name tag to both the chatbot and user message bubbles for better clarity in conversations. Additionally, we recommended incorporating a settings menu with options like language selection, editing preferences, and other customizable features.
- We also discussed the structure of the research paper and began organizing our work to ensure a clear and cohesive flow. We outlined the main sections, such as the introduction, literature review, methodology, findings, and conclusion.
- We also discussed the Intro to LLM (Large Language Models) course,by https://learn.365datascience.com/courses/preview/intro-to-llms/ , which provided us with useful source codes. These resources could help us understand model implementation better and potentially improve our project.

Next step:

- Our next step will be to start working on the diagrams for the project. These will include flowcharts, system architecture, and any other visuals needed to represent the structure and processes clearly.
- The interface design adjustments will also be ready soon. This will include updates based on our recent feedback, such as adding name tags to message bubbles and enhancing the settings menu with options like language selection and editing preferences. These adjustments will refine the user experience and ensure the design aligns with our project goals before we move forward with development.

Meeting #9

Date & Time: 20/11/2024 – Monday, 7-8pm
Venue: 1st Floor sofa Area
Attendees: Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

1. Update on the prototype of the application
2. Finalize the prove of concept
3. Officially close the requirement/ initiation phase

4. Start working on development phase

Discussion:

- Use longchain for vectorization
- Start on final ppt+report(should be done by 12 dec)
- Security for API key(from chatGPT)

Distributed tasks for the planning/ design phase:

- System design(Rihab+Rahaf)
- Assumptions(Fatma)
- Limitations+scope(Leena+Fatma)
- Cost/price(Leena)

Plan of Action:

- Complete all the tasks assigned
- Start working on the ppt and report after getting the diagrams approved from dr turki
- Get approval of prototype from Dr imed
- Finalize the prototype
- Start with vectorization, use langchain

Meeting #10

Date: Monday 2/12/2024
Time: 7-8pm
Attendees: Leena Althekair, Fatma Alamoudi, Rahaf Alshabrawi, Rihab Asif
Location: Virtual
Agenda

1. Document Review (Cost, Limitations, Assumptions)
2. Diagram Updates
3. UX/UI Review
4. Final Report Template
5. Next Steps

Discussion

1. Document Review:

- Reviewed documents on cost, limitations, and assumptions.
- All documents were completed, and this task was marked as done.

2. Diagram Updates:

- Rihab's Diagrams (ERD, UML Class):
  - Agreed to rename "Student" to "User" with attributes: ID, Full Name, Email, and Profile Picture.
  - A "Student" child class will be added with "Major" and "Class Level" attributes.
  - Rihab will create the object diagram.
- Rahaf's Diagrams (System Sequence, Activity, Use Case):
  - System Sequence Diagram:
    - Combine "Student" and "User" into "User."

        - ♣ Allow admin to bypass login since no admin interface will exist; they will operate directly from the backend.
    - o Activity Diagram:
        - ♣ Add the "Favorite Event" action.

3. UX/UI Review:
- Designs were reviewed and accepted with the following adjustments:
    1. Add a "Favorite Events" page. (Leena)
    2. Add a "Register" button with an external link icon. (Fatma)

4. Final Report Template:
- The final report template was found and shared on Teams.
- Agreed to start compiling all work in the shared document.

Next Steps:
1. Meet with Dr. Turki tomorrow to present diagrams and gather feedback.
2. Compile all completed work into the shared report document.

Capstone II
Meeting #1

Date: 15 January 2025
Time: 10-11am
Attendees: Leena, Rahaf, Rihab, Fatma
Venue: Library
Agenda:
- Review Gantt Chart
- Distribute beginning tasks
- Devise implementation plan

Discussion:
- Gantt chart for now is sufficient. As semester goes on, the larger tasks will be divided into subtasks.
- The entire team will work on the backend, and then after completing it we will move to the front end to maximize the learning benefit for all members.
- Leena and Rahaf: start working on the calendar, python classes and firebase database to store events. Focus on Fall 2024-2025 events.
- Fatma and Rihab: LangChain package to convert documents to vector representations and ChatGPT API implementation. Start with the student handbook.

Meeting #2

Date & Time: 11/2/2025 – Tuesday, 2-2:30 pm
Venue: Online
Attendees: Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi, Dr. Imed
Agenda:
- Feedback on research paper draft for CCTS 2025 Research Paper Competition

Discussion:
- Dr. Imed reviewed the research paper draft and provided constructive feedback.
- Noted that the paper is too brief and needs text reduction in several sections.
- Highlighted structural issues – the abstract, introduction, and literature review need refinement.

- Emphasized better descriptions of the LLM and chatbot components.
- Suggested separating design and implementation and clearly distinguishing functional and non-functional design elements.
- Chapter 3 heading needs to be revised for clarity.
- The design and data collection sections must be expanded and clearly detailed
- Tools used, such as the API chatbot, should be properly described.
- Recommended including specific results obtained from using the chatbot with queries.

Plan of Action:

- Revise the paper according to the feedback shared by Dr. Imed.
- Clarify and expand all relevant sections, specially design and tools.
- Finalize the paper for submission to the CCTS 2025 competition.
-

Meeting #3

Date: 2 Feb 2025
Time: 10-10:30am
Attendees: Rahaf, Fatima, Rahib
Venue: Teams (online)
Agenda:

- Paper Conference Structure
- Deadline time

Discussion:

- We will cover LLM, chatbot technologies, pricing, available tools, and development aspects.
- A comparison table will be included to analyze existing models.
- The methodology section will outline the approach for development.
- Expected results will highlight what the model can provide.
- The literature review will include related works.
- The document should be 6-8 pages long and include findings in the results section.

Meeting #4

Date: February 25, 2025
Time: 9:30- 10:30 am
Location: Microsoft Teams
Attendees: Dr. Imed, Leena, Rahaf, Rihab, Fatma
Agenda:

- Discuss work done so far in the implementation
- Share the expected outcome at the end of the semester and the project rubrics
- Explain the capstone report template

Discussion:

- Shared the progress on the calendar script: creating events, creating categories, and the corresponding .json files. (Need to troubleshoot adding category by ID.)
- LangChain (Fatma) and ChatGPT (Rihab) progress was not shared because they joined late; Dr. Imed pushed to next week.
- Flutter progress was shared that it's being installed and configured, will try and have a basic UX/UI set up by the next meeting.

- Dr. Imed shared the rubrics: the project will be evaluated on the documentation, presentation, poster, and prototype. It will be evaluated by most department faculty as well.
- All members must present the slideshow. Questions will be asked individually.
- Chapters 1-4 in the report can be taken from capstone 1 but will have to be adjusted for updates. Next 4 chapters are implementation, texting, evaluation, and future work.

Action Points:

- Create CoLab file to work on the ChatGPT API and the LangChain backends.
- Bear unit testing in mind when working.
- Document and work on the capstone report alongside the implementation.
- Next week: show a rudimentary interface + show the working LangChain + ChatGPT script.

Meeting #5

Date: March 11, 2025
Time: 2:00- 3:00 pm
Location: Microsoft Teams
Attendees: Dr. Imed, Rahaf, Fatima, Leena, Rihab
Agenda:

- Present and review the work completed so far.
- Poster Award Competition

Discussion:

- Chatbot Spell Check Feature
    - Evaluate whether the chatbot should check for spelling errors.
    - Discuss implementation and potential impact on usability.
- Best Poster Award Criteria: Define key elements for the poster, including:
    - Interface: Visual layout, readability, and design.
    - Title: Clarity, conciseness, and relevance.
    - Abstract & Project Statement: Summarizing the research effectively.
- Scientific Poster Structure: Ensure the poster includes the following essential sections:
    - Abstract – Concise summary of the research.
    - Introduction – Background and research context.
    - Methodology – Explanation of methods and techniques used.
    - Results – Key findings with supporting visuals (charts, graphs, etc.).
    - Discussion/Conclusion – Interpretation of results and overall impact.
    - Recommendations – Suggestions based on findings.
    - Acknowledgments & References – Recognizing contributors and sources.
- Gantt Chart Progress Review: Assess current progress on the giant chart.

Meeting #6

Date & Time: 18/2/2025 – Tuesday, 2-2:30 pm
Venue: Online
Attendees: Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

- Update on project progress
- Review the poster for the CCTS competition

Discussion:

- Reviewed the CCTS competition poster with Dr. Imed; suggested adding the OpenAI logo for credibility.
- Ensured the poster content is appropriately sized for readability.
- Showed the progress of work on the Gantt chart.

Plan of Action:

- Complete the chatbot implementation before returning from the break.
- Begin implementing the chatbot in the app using Flutter.

Meeting #7

Date & Time: 8/4/2025 – Tuesday, 10 -11 am
Venue: Online
Attendees: Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi, Dr. Imed
Agenda:

- Update on project progress
- Confirm participation in the upcoming hackathon with Dr. Imed
- Status update on the CCTS 2025 poster

Discussion:

- The team discussed potential participation in the hackathon with Dr. Imed, and all members agreed to take part.
- No significant updates on the project at this time; nothing new was ready to be present.
- Rahaf implemented the suggested changes to the CCTS 2025 poster and submitted it before the deadline.

Plan of Action:

- The team will begin preparing for the hackathon following their decision to participate.
- They will reach out to Dr. Fadwa from the Innovation Center for further details and registration procedures.

Meeting #8

Date: April 15, 2025
Time: 10:00 – 11:00 AM
Location: Office 280
Attendees: Dr. Imed, Leena, Rahaf, Rihab, Fatma
Agenda:

- - Evaluate progress on chatbot
- - Demonstrate the three chatbot models
- - Discuss testing results
- - Follow up on UJ Hackathon application
- - Follow up on report questions email

Discussion:

- Chatbot Progress
    - Fatma presented a model based on a web service allowing document upload and interaction. It is useful as a reference but the team must build the final version from scratch.

    - Rahaf's chatbot is integrated well with the front end and provides accurate answers. A feature for automatic scrolling when chatbot replies may be added. More documents are needed for better evaluation.
    - Rihab's chatbot answers professionally and with clear formatting. However, some results seem unrelated, likely due to gaps in document coverage. Suggested to attach a `.txt` instruction file with student requests to guide the model.
- Testing Strategy
    - Microsoft Copilot can be used for comparison purposes (not in the final product).
    - Testing will consist of two phases:
    - Part 1: Compare our chatbot with Microsoft Copilot.
    - Part 2: Recruit ~10 students anonymously to evaluate chatbot functionality (front end and back end separately). Rahaf and Rihab will manage this.
- Presentation and Demo Preparation
    - Begin working on presentation and demo as they are due the week before finals.
    - Presentation must follow the provided rubric.
    - Report is expected to be due as listed on the SIS.
    - Posters: Prepare internal and external versions. The focus is to present the app as a product. Should be understandable to a general audience.
    - Include: methodology, data collection, LLM and RAG approach, problem-solving techniques, comparisons of solutions, and a brief literature review showing what differentiates this project.
- Report Progress
    - Report progress is steady but slow.
    - Follow up on the report-related email.
    - Dr. Imed will review the draft soon, pending his availability (CCTS preparations ongoing).
    - Report should briefly mention other attempted models and extra work as part of our general approach.

Action Points:

- - Test Microsoft Copilot for comparison.
- - Decide on the final chatbot code base.
- - Recruit students for testing.
- - Begin working on presentation slides and demo.
- - Dr. Imed will provide resources and share the presentation rubric.
- - Email Ms. Karima requesting needed documents.

Meeting #9

Date: April 30, 2025
Time: 10:00 – 10:30 AM
Location: Office 280
Attendees: Dr. Imed, Rahaf,Leena, Rihab, Fatma
Agenda:

- - CCTS poster competition
- - Latest update of the Capstone project

Discussion:

- The poster of CCTS and what should we present about

- Jeddah Hackthon preparation (presentation)
- The need of the presentation the next week for potential capstone project presenting
- Dr.Imed wanted to see a demo but due to low resolution we decided to move it next week
- Testing part is over

Meeting #10

Date & Time: 6/5/2025 – Tuesday, 10 -11 am
Venue: Online
Attendees: Leena Althekair, Rihab Asif, Rahaf Alshabrawi, Fatima Alamoudi
Agenda:

- Update on project progress, presentation, and report
- Demonstration of the product (mobile app)

Discussion:

- The project presentation (PPT) for the Capstone 2 final is currently 50% complete.
- The remaining work on the presentation will be completed after the team participates in the hackathon, tentatively scheduled for Wednesday and Thursday.
- Product testing and evaluation were conducted by Rahaf and Leena using external users (non-team members). The users provided comprehensive feedback, and Rahaf implemented the necessary changes.
- Rahaf and Rihab will demonstrate the app and the copilot version of the chatbot to Dr. Imed in the next meeting. Screen sharing issues during this meeting prevented the demonstration.
- The team requested to move the project presentation from 8 May to 26 May due to their participation in the hackathon.
- However, Ms. Karima announced during the meeting that the official Capstone presentation will be held on 22 May, and all team members agreed to this date.
- According to Leena, the project report is in progress. The team will continue working on it post-hackathon and submit both the report and the presentation to Dr. Imed for review.
- Submission is due one week before the final presentation date (i.e., by 15 May).

Plan of Action:

- Complete the project report and presentation by next week to allow Dr. Imed to review and provide feedback.
- Present the demo of the app and the copilot chatbot version in person at the next meeting.

Meeting #11

Date: May 13, 2025
Time: 10:15 – 10:45 AM
Location: Room 280
Attendees: Dr. Imed, Leena, Rahaf, Rihab, Fatma
Agenda:

- - Demo presentation to Dr. Imed
- - Presentation progress and feedback
- - Report requirements and rubric
- - Potential conference submission (AITK’25)

Discussion:

1. Demo Approval

- The demo was presented to Dr. Imed, and he approved it.

2. Presentation
   - The presentation is progressing well; the hackathon version is used as a baseline.
   - Recommendations: use fewer slides, aim for 1–2 minutes per slide, keep animations minimal, and maintain a formal tone.
   - Review the presentation rubric. Time limit is likely 20 minutes including the demo.
   - Submit the final version to Dr. Imed by May 20th for review.
3. Capstone Report
   - The report must be completed thoroughly and according to the rubric.
   - Include detailed comparisons, tables, and figures.
   - Literature review should integrate content from the hackathon presentation and Dr. Imed's feedback email.
   - Submit report my May 22 max for Dr. Imed's review.
4. AITK'25 Conference
   - Discussed the upcoming AITK'25 conference in Hyderabad, India (October 2025).
   - The team may submit a paper based on the project.
   - Rahaf will be the primary author, with Dr. Imed as the last author and advisor.

Appendix C – Code and Unit Testing Results

The source code for the proof-of-concept program:

```
import re
from nltk.tokenize import RegexpTokenizer
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from openai import OpenAI

# Initialize OpenAI API client
client = OpenAI(api_key="sk-
aMrflV4Tv_76JR7iSbfC412BX6xUeIzWk3wfsoWwOvT3BlbkFJm9wCXTaAWqO_9d014jL4OBc8VgSQa3Q
QxktXNy0gcA")

# Pre-processing function
def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()

    # Remove special characters
    text = re.sub(r'[^a-zA-Z\s]', '', text)

    # Tokenize
    tokenizer = RegexpTokenizer(r'\w+')
    tokens = tokenizer.tokenize(text)

    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    tokens = [word for word in tokens if word not in stop_words]

    # Lemmatize
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(word) for word in tokens]

    return ' '.join(tokens)


def chat_with_gpt(message):
    response = client.completions.create(
```

```
        model="gpt-3.5-turbo-instruct",
        prompt=message,
        max_tokens=150
    )

    return response.choices[0].text.strip()  # To erase whitespaces


while True:
    user_input = input("You: ")

    if user_input.lower() == 'quit':
        break

    cleaned_input = preprocess_text(user_input)

    # cleaned input to the OpenAI model
    response = chat_with_gpt(cleaned_input)

    # Output the chatbot's response
    print("Chatbot:", response)
```

Unit Testing Results

Task	Criteria	The amount of time to accomplish task	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	***>0 sec***	1		
Task 2:	Correct and comprehensive answers	***4 sec***	1		
Task 3:	view calendar	***3 sec***		1	
Task 4:	view settings	***3 sec***	1		
Task 5:	overall experience	***3 min***	1		
Task 6:	Dark mode	***>0 sec***	1		
Task 7:	Save Chats	***2 sec***		1	
Comments	Reminds of due dates without asking (personalized notifications)				
Major	SLHS				

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers		1	
Task 3:	view calendar		1	
Task 4:	view settings	1		
Task 5:	overall experience		1	
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Add semester schedule/courses, hallucinating answers. Excellent concept.			
Major	Law			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers		1	
Task 3:	view calendar	1		
Task 4:	view settings	1		
Task 5:	overall experience	1		
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Everything very nice, answers are long, could be in points.			
Major	Computer Science			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers	1		
Task 3:	view calendar	1		
Task 4:	view settings	1		

Task 5:	overall experience	1		
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Add floor plan to be able to ask for direction			
Major	Cybersecurity			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers		1	
Task 3:	view calendar	1		
Task 4:	view settings		1	
Task 5:	overall experience		1	
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Add augmented reality map, specify faculty positions.			
Major	Cybersecurity			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers	1		
Task 3:	view calendar	1		
Task 4:	view settings	1		
Task 5:	overall experience	1		
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	NA			
Major	Interior Design			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		

Task 2:	Correct and comprehensive answers	1		
Task 3:	view calendar	1		
Task 4:	view settings	1		
Task 5:	overall experience	1		
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Add Arabic language interface			
Major	Interior Design			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers	1		
Task 3:	view calendar	1		
Task 4:	view settings	1		
Task 5:	overall experience	1		
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		
Comments	Add more faculty.			
Major	Interior Design			

Task	Criteria	Excellent	Acceptable	Unacceptable
Task 1:	Can message chatbot	1		
Task 2:	Correct and comprehensive answers		1	
Task 3:	view calendar		1	
Task 4:	view settings	1		
Task 5:	overall experience		1	
Task 6:	Dark mode	1		
Task 7:	Save Chats	1		

Comments	Very useful, likes the link feature. Recommend university map guider, reader.
Major	Visual Communication

.....

(وَآخِرُ دَعْوَاهُمْ أَنِ الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ)