

## The Hashemite University, Zarqa, Jordan

**Faculty of Prince Al-Hussein Bin Abdallah II For Information Technology Computer Science and Applications Department**

HUBOT

## A project submitted

**in partial fulfillment of the requirements for the B.Sc. Degree in Computer Science and Applications**

## By

***Mohammad Haitham Fathi Salah (2035083) Mohammad Abdulnnasir Abdallah Khattab (2040117) Tariq Mohammad Adel Alkurdi (1437177)***

## Supervised by

Dr. Mohammad.M.A.Bsoul

## Committee Member Names

Dr.Sahar Idwan

Dr.Alaa eddien Attar

## Month Year 1/2024

## CERTIFICATE

It is hereby certified that the project titled <***HUBOT>***, submitted by undersigned, in partial fulfillment of the award of the degree of “Bachelor in Computer Science and Applications” embodies original work done by them under my supervision.

All the analysis, design and system development have been accomplished by the undersigned. Moreover, this project has not been submitted to any other college or university.

Mohammad Haitham Fathi Salah (2035083) Mohammad Abdulnnasir Abdallah Khattab (2040117) Tariq Mohammad Adel Alkurdi (1437177)

## ABSTRACT

The HUBOT mobile app is a comprehensive solution designed to enhance the campus experience for students by providing various features and functionalities. This project aimed to address the challenges faced by students in managing their academic schedules, accessing campus resources, and obtaining relevant information. The investigation involved the development of the app using Flutter for the frontend and Java for the backend, with MongoDB as the real-time database and REST APIs for communication.

HUBOT will have key features such as a CHATBOT, CAMPUS MAP, GPA calculator, FAQ section, and a student day manager. These features were selected to be in the first version of HUBOT during to they are essential features. these features designed to streamline academic tasks, improve navigation within the campus, facilitate communication with a responsive chatbot, and offer a convenient platform for managing daily schedules.

HUBOT future additions and enhancement are countless, and we are not trying just to solve student problems and issues, we are trying to build a whole environment that will add many new things to our campus and open up to new and other doors.

Its benefit will not stop in student it will raise the university services and in future HUBOT will contain many users such as employees, instructors or just and foreign visitor.

In conclusion we are not just trying to build a bot, we are establishing for a new era of university life, HUBOT future addition are countless…

## ACKNOWLEDGEMENTS

This page is optional. It is where you may put your personal word of thanks to anyone who has helped you in your work.

## TABLE OF CONTENTS

[CERTIFICATE II](#_bookmark0)

[ABSTRACT III](#_bookmark1)

[ACKNOWLEDGEMENTS IV](#_bookmark2)

[TABLE OF CONTENTS V](#_bookmark3)

[ABBREVIATIONS VII](#_bookmark4)

[LIST OF FIGURES VIII](#_bookmark5)

[LIST OF Tables X](#_bookmark6)

[Chapter 1: Introduction 1](#_TOC_250008)

* 1. [Overview 1](#_TOC_250007)
  2. [Project Motivation 1](#_TOC_250006)
  3. [Problem Statement 1](#_TOC_250005)
  4. [Project Aim and Objectives 1](#_TOC_250004)
  5. Project Scope 2
  6. Project Software and Hardware Requirements 2
  7. Project Limitations 2
  8. Project Expected Output 2
  9. Project Schedule 2
  10. Project, product, and schedule risks 2
  11. Report Organization 2

[Chapter 2: Literature Review 5](#_bookmark7)

* 1. [Introduction 4](#_TOC_250003)
  2. [Existing Systems 4](#_TOC_250002)
  3. [Overall Problems of Existing Systems 4](#_TOC_250001)
  4. [Overall Solution Approach 4](#_TOC_250000)

Chapter 3: Requirement Engineering and Analysis 5

* 1. [Stakeholders 10](#_bookmark8)
  2. [Use Case Diagram 12](#_bookmark10)
  3. Non-Functional User Requirements 5

3.3 Constraints 6

[Chapter 4: Architecture and Design 18](#_bookmark12)

* 1. Overview 7
  2. [Software Architecture 18](#_bookmark13)
  3. [Software design 20](#_bookmark14)
  4. [User interface design (prototype) 25](#_bookmark15)

[Chapter 5: Implementation Plan 34](#_bookmark16)

* 1. [Description of Implementation 34](#_bookmark17)
  2. [Programming language and technology 34](#_bookmark18)
  3. [part of implementation if possible 38](#_bookmark19)

[Chapter 6: Testing Plan 42](#_bookmark20)

* 1. [Black-box 42](#_bookmark21)
  2. [White-box 42](#_bookmark22)
  3. [Testing automation 43](#_bookmark23)

[Chapter 7: Conclusion and Results 44](#_bookmark24)

* 1. Summary of accomplished project 12
  2. Future Work 12

References 13

## ABBREVIATIONS

 I.e. is the abbreviation for id Est and means “in other words.”

## LIST OF FIGURES

|  |
| --- |
| [Figure 1: Caption for Figure 1 via *References-> example of chatbot UI*](#_bookmark9) |
| Figure 2: Caption for Figure 2 via *References-> stakeHolders* |
| Figure 3: Caption for Figure 3 via *References->user use case diagram* |
| Figure 4: Caption for Figure 4 via *References-> logical view diagram* |
| Figure 5: Caption for Figure 5 via *References-> User uml sequence diagram* |
| Figure 6: Caption for Figure 6 via *References-> class diagram* |
| Figure 7: Caption for Figure 7 via *References-> ER-diagram* |
| Figure 8: Caption for Figure 8 via References->log in page |
| Figure 9: Caption for Figure 9 via References-> chatbot page |
| Figure 10: Caption for Figure 10 via References-> talking with Hubot |
| Figure 11: Caption for Figure 11 via References-> Slide Menu |
| Figure 12: Caption for Figure 12 via References-> GPA Calc Adding Subject |
| Figure 13: Caption for Figure 13 via References-> Filling Subject Grades |
| Figure 14: Caption for Figure 14 via References-> GPA Result |
| Figure 15: Caption for Figure 15 via References-> Adding a subject into std day manager |
| Figure 16: Caption for Figure 16 via References-> Subject added to std day manager |
| Figure 17:Caption for Figure 17 via *References-> Campus map* |
| Figure 18:Caption for Figure 18 via *References-> Campus map internally* |
| Figure 19: Caption for Figure 19 via References-> part of login page |
| Figure 20: Caption for Figure 20 via References-> part of sigu up page code |
| Figure 21: Caption for Figure 21 via References-> part of student registration page code |
| Figure 22: Caption for Figure 22 via References-> part of Chat Page code |
| Figure 23: Caption for Figure 23 via References-> part of Slide Menu code |
| Figure 24: Caption for Figure 24 via References-> part of GPA calculator code |
| Figure 25: Caption for Figure 25 via References-> part of STD Day Manager code |
| Figure 26: Caption for Figure 26 via References-> part of AdminController showing Spring annotation and REST CRUD operations code |
| Figure 27: Caption for Figure 27 via References-> part of USER Service describing business logic code |
| Figure 28: Caption for Figure 28 via References-> part of Instructor Data Link Class Communicating with Mongo DB |
| Figure 29: Caption for Figure 29 via References-> MongoDB GUI |
| Figure 30: Caption for Figure 30 via References-> ENTERING WRONG USERNAME OR PASSWORD |
| Figure 31: Caption for Figure 31 via References-> ENTERING EXISTING USERNAME |
| Figure 32: Caption for Figure 32 via References->ENTERING INVALID EMAIL |
| Figure 33: Caption for Figure 33 via References-> ENTERING DEFFERENT PASSWORDS |
| Figure 34: Caption for Figure 34 via References-> INVALID REQUEST |
| Figure 35: Caption for Figure 35 via References-> BAD REQUEST (METHOD NOT ALLOWED) |
| Figure 36: Caption for Figure 36 via References-> REQUESTING WHILE SERVER IS OFF |
|  |

## LIST OF TABLES

[Table 1: Caption for Table 1 via *References-> comparision table*](#_bookmark11)

## CHAPTER 1:INTRODUCTION

## OVERVIEW

this chapter introduces the hubot project, which is a chatbot application aimed at providing assistance to students in various aspects of their university days. it provides a general overview of the project and its main features.

## PROJECT MOTIVATION

the motivation behind developing the hubot project was to provide students with a convenient and easy-to-use tool to help them in efficent way, guied them and managing their academic tasks. the project aims to address the challenges that students face in managing their coursework and academic schedules. by providing a user-friendly chatbot interface.

the project is important because it has the potential to enhance students' academic experiences by providing them with an accessible and personalized tool for managing their academic work and provide personalized guidness or help in varrious aspects.

additionally, the project aims to increase student engagement and retention by offering a platform that allows for more efficient and effective academic support.

the new idea proposed by this project is the development of an intelligent chatbot that can provide personalized academic assistance to student

## PROBLEM STATEMENT

the hubot project addresses the issue of students struggling on finding the information they need and to manage their academic workload and schedules and many more. many students face challenges in keeping up with coursework, assignments, and deadlines, leading to poor academic performance and increased stress levels. the project aims to provide a solution to these problems by offering a chatbot that can provide relevant and personalized support, help,information and good academic experiance.

## PROJECT AIM AND OBJECTIVES

the overall aim of the hubot project is to develop an intelligent chatbot that can assist students in managing their academic workload, schedules and every thing related to their university life. the specific objectives of the project include:

* developing a chatbot that can understand and respond to student queries and requests. providing personalized academic assistance to students, including reminders for assignments and deadlines, study tips, and recommendations.
* offering a user-friendly and accessible interface for students to manage their academic tasks.
* iimproving student engagement and retention by offering a tool that enhances their academic experience.
* create a map for our university campus that can navigate even inside buildings..
* answer student frequent quiestion..

## PROJECT LIMITATIONS

the hubot project has several limitations and parameters that need to be considered. firstly, the chatbot is designed to provide assistance only to students and may not be suitable for other user groups (till this point). additionally, the project is limited to providing university support and does not address other areas of student life out side the hashemite university campus..

## PROJECT EXPECTED OUTPUT

the desired output of the hubot project is an intelligent chatbot that can provide personalized academic support to students. the chatbot should be user-friendly, accessible, and capable of understanding and responding to student queries and requests.

## PROJECT SCHEDULE

the project schedule includes several milestones, activities, and deliverables, with intended start and finish dates. the project is expected to be completed within six months, with the following major milestones:

* + research and planning phase (MONTH 1)
  + development and testing phase (MONTHS 2-4)
  + integration and deployment phase (MONTHS 5-6)

## REPORT ORGANIZATION

the remainder of this report is organized as follows. chapter 2 provides an overview of the related work in chatbot development and academic support tools. chapter 3 describes the methodology used in developing the hubot project. chapter 4 presents the design and architecture components of the chatbot. chapter 5 discusses the implementation and testing of the chatbot. chapter 6 presents the future work and concludes the report

## CHAPTER 2: LITERATURE REVIEW

## Introduction

This literature review will examine the available literature in the field of chatbots and their applications, with a focus on the development of Hubot. The review aims to provide a comprehensive understanding of the current state of the art in this field and how Hubot fits into it.

But to be considered that the idea of making AI bots for universities doesn’t exist in Jordan and not widely common in the world, and the idea of making such intelligent bot to be the student friend and assistant it’s nearly doesn’t exist.

## Existing Systems

We have done a lot of researches trying to find a tool that is nearly similar to our tool but we didn’t find, in Jordan none of the current universities have any Chabot app, in the world it’s not widely exist in universities some of the big universities have done chatbots as web pages and a very shy number of universities have chatbots as mobile application and they are only accessed from their university country.

Another issue that all the chatbots are connected with their university database and systems so they provide a range of services we are not aiming to provide.

Even if maybe there is some common features but we are varying in the way and the content of the features we have, as an example let’s take campus map feature as an example, many have this feature but nearly none of them do track inside buildings and so on to the other features we have.

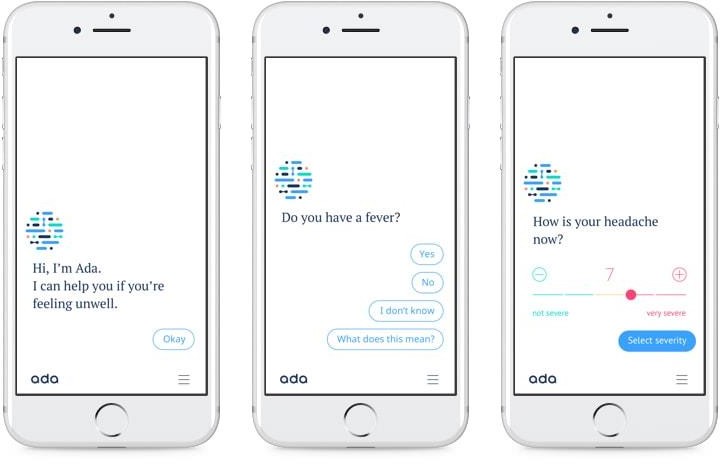
## Tools discussed:

* + 1. **CSE Peer Helper**

The CSE Peer Helper chatbot developed by the University of Minnesota is built using a number of programming languages and technologies. The primary programming language used to develop the chatbot is Java, along with several related libraries and frameworks, including the Microsoft Bot Framework and Apache OpenNLP for natural language processing

## CRIMSON CONCRIEGE

The Crimson Concierge chatbot, developed by the University of Alabama, was built using the Amazon Lex service and implemented in the Node.js programming language. Amazon Lex is a service provided by Amazon Web Services (AWS) that enables developers to build conversational interfaces for applications using natural language understanding and automatic speech recognition.



[Figure 1: Caption for Figure 1 via *References-> example of chatbot UI*](#_bookmark9)

* + 1. **POUNCE**

# Georgia State University's chatbot, Pounce, was built using the Microsoft BotFramework and implemented in the C# programming language. The Bot Framework provides a platform for building, testing, and deploying chatbots,and C# is a commonly used language within the framework. Additionally, Pounce utilizes natural language processing (NLP) to understand user input.

* + 1. **ASK BLUE**

# Ask Blue, the chatbot developed by the University of Michigan, was built using the Oracle Digital Assistant platform, which allows developers to buildconversational interfaces using natural language processing and machine learning technologies. The programming languages used to create the chatbot are not publicly available, as the Oracle Digital Assistant platform abstracts the underlying implementation details.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tool\ Features | Chatbot | Assignment Help | GPA Calculator | Student day  manager | Financial Aid | map |
| CSE  Helper | T | T | X | X | X | X |
| Crimson | X | X | X | X | X | X |
| Pounce | X | X | X | X | T | X |
| Ask Blue | X | X | X | X | X | X |
| HUBOT | T | X | T | T | X | T |

[Table 1: Caption for Table 1 via *References-> comparision table*](#_bookmark11)

* + Green T denoted to existing feature.
  + Red X denoted to excluded feature.

## Overall Problems of Existing Systems

Despite the popularity of chatbots, there are still several challenges that exist in their development and deployment. One of the main challenges is natural language processing, which is critical for understanding user input and generating appropriate responses.

While existing systems have made significant contributions to the field, they are not without their limitations and challenges. Through our literature review, we have identified several common issues encountered in these systems. These include limited functionality, lack of personalized recommendations and complex user interfaces.

## Overall Solution Approach

Based on our analysis of the existing systems and the identified problems, we have developed an overall solution approach for HUBOT. Our approach involves the utilization of Flutter as the frontend framework and Java as the backend programming language. To overcome the limitations of dependency on university servers and databases, we have opted to integrate MONGODB as a real-time database and REST API provider.

By leveraging MONGODB, we can ensure the availability of real-time data updates, seamless integration with the app's frontend, and the ability to store user-specific information securely. This approach enables us to create a mobile app that functions independently of university infrastructure while still providing valuable assistance to students.

## CHAPTER 3: REQUIREMENT ANALYSIS

## Stakeholders

In the case of HUBOT, we have identified the following primary stakeholders:

Students: The primary users of the HUBOT system who will benefit from its features and functionalities in accessing information, receiving assistance, and navigating the campus effectively.

Administration: Administrative personnel who oversee the overall management.

A diagram of a chatbot project

Description automatically generated

Figure 2: Caption for Figure 2 via *References-> stakeHolders*

The diagram depicts the relationships within the HUBOT project, showcasing how different individuals and components interact. Here's a brief explanation:

## Stakeholders:

**Students:** They interact with the system by engaging with the Chatbot component, seeking assistance and accessing information.

## Components:

**HUBOT Project:** This encompasses the overall project, including the Chatbot and Knowledge Base components.

**Chatbot:** It serves as the interactive interface through which students communicate with the system, asking questions and receiving responses.

**Knowledge Base:** It acts as a repository of knowledge, which the Chatbot accesses to provide students with answers and assistance.

The diagram visually represents the interaction between students and the Chatbot component, which, in turn, relies on the Knowledge Base for relevant information.

As the admin, I oversee the management and maintenance of the Knowledge Base to ensure its accuracy and relevance.

Overall, the diagram effectively represents the relationships between stakeholders and components within the HUBOT project, emphasizing the flow of information and interactions between them.

## Use Case Diagram

* + 1. User

A diagram of a person with a person in the middle

Description automatically generated

Figure 3: Caption for Figure 3 via *References->user use case diagram*

The use case diagram for the "Hubot" system depicts a user-centric interaction model with a comprehensive set of functionalities.

Users, represented by the "User" actor, can seamlessly engage with the system through distinct use cases.

The "Login" use case allows users to enter credentials and authenticate, facilitating secure access to the platform.

For account creation, the "Signup" use case lets users provide necessary information to establish their accounts.

A communication-centric feature, "Make a Chat," enables users to initiate and engage in conversations by sending messages. Additionally,

the system offers utility in navigating the campus environment through the "Access Campus Map" use case, allowing users to view the map and search for specific locations.

Managing academic responsibilities is made convenient with the "Access Student Day Manager" use case, offering tools to view schedules and handle tasks efficiently.

Lastly, the "FAQ" use case empowers users to explore frequently asked questions and obtain relevant information.

## Together, these use cases encapsulate the diverse functionalities of the "Hubot" system, enhancing the user.

## non-Functional User Requirements

When designing a chatbot like Hubot, it's important to consider not just the functional requirements (i.e. what the chatbot should be able to do), but also the non-functional requirements that will ensure the chatbot is safe, secure, usable, and easy to maintain over time. Here are the non-functional user requirements for the Hubot project:

## Execution Qualities

* Safety: Hubot should not be able to cause harm to users or their data. It should be designed with appropriate safeguards and access controls to prevent any malicious actions.
* Security: Hubot should be designed with strong security measures to protect users' privacy and data. It should use encryption and secure communication protocols to protect sensitive information.
* Usability: Hubot should be easy to use and understand for users of all technical levels. It should have a clear and simple interface, with intuitive commands and responses.

## Evolution Qualities

* Testability: Hubot should be designed with testability in mind, so that it can be easily tested and debugged during development and maintenance. This includes having clear error messages and logging features.
* Maintainability: Hubot should be designed with maintainability in mind, so that it can be easily updated and modified as needed over time. This includes using modular, reusable code and following best practices for code organization and documentation.
* Extensibility: Hubot should be designed with extensibility in mind, so that it can be easily extended with new functionality as needed. This includes having a clear plugin architecture and API for developers to build upon.
* Scalability: Hubot should be designed with scalability in mind, so that it can handle a growing number of users and requests over time.

By considering these non-functional user requirements, we can ensure that Hubot is not just a functional chatbot, but a reliable and sustainable tool for users to interact with over time.

## CHAPTER 4: ARCHITECTURE AND DESIGN

* 1. **Software (System) Architecture**
     1. Logical view

A screenshot of a chatbot

Description automatically generated Figure 4: Caption for Figure 4 via *References-> logical view diagram*

## Details of each component in a separate section.

* + - 1. **Chatbot:**

The Chatbot component serves as the core component of the system. It includes the following interfaces:

**User Interaction:** This interface enables the Chatbot to interact with users, specifically students. It handles user inputs, processes requests, and provides responses.

**Data Retrieval:** This interface allows the Chatbot to retrieve relevant data from the database. It can fetch information such as student records, course details, assignments, and deadlines.

**Data Storage:** This interface enables the Chatbot to store data into the database. It can save user preferences, progress, or any other relevant information for future reference.

**External Services:** This interface facilitates communication with external services or APIs. It allows the Chatbot to retrieve additional data or perform actions beyond the system's scope.

## Database:

The Database component is responsible for storing and managing data related to the system. It stores information such as student records, course details, assignments, and other relevant data. The HUBOT component interacts with the Database component through the Data Retrieval and Data Storage interfaces to fetch and store data as needed.

## External Services:

The External Services component represents external services or APIs that the system may utilize. It can include services for accessing additional educational resources, retrieving real-time information, or integrating with other systems. The Chatbot component interacts with the External Services component through the Data Retrieval and External Service Interaction interfaces to retrieve relevant data or perform specific actions.

Each component plays a crucial role in the system's overall functionality, enabling the Chatbot to provide user interaction, retrieve and store data, and interact with external services when necessary.

## Software design

* + 1. User UML sequence/communication diagram

A screenshot of a computer

Description automatically generatedFigure 5: Caption for Figure 5 via *References-> User uml sequence diagram*

1. **Access All Features (sync): The user registers and logs in synchronously and then accesses all features synchronously through Hubot.**
2. **Request Campus Map (sync): The user requests the campus map synchronously, and Hubot interacts with the Map API synchronously.**
3. **Request GPA Calculator (sync): The user requests GPA calculation synchronously, and Hubot processes the calculation synchronously.**
4. **Request Comprehensive Chat (async): The user initiates a comprehensive chat asynchronously, and Hubot interacts with the Chatbot API asynchronously. This reflects the potential delay in responses as the Chatbot API processes messages in the background.**
5. **Exit (sync): The user exits the Hubot system synchronously.**
   * 1. Class diagram­­­­

A diagram of a computer

Description automatically generated with medium confidenceFigure 6: Caption for Figure 6 via *References-> class diagram*

* + 1. ER-Diagram

A diagram of a company

Description automatically generatedFigure 7: Caption for Figure 7 via *References-> ER-diagram*

## User interface design (prototype)

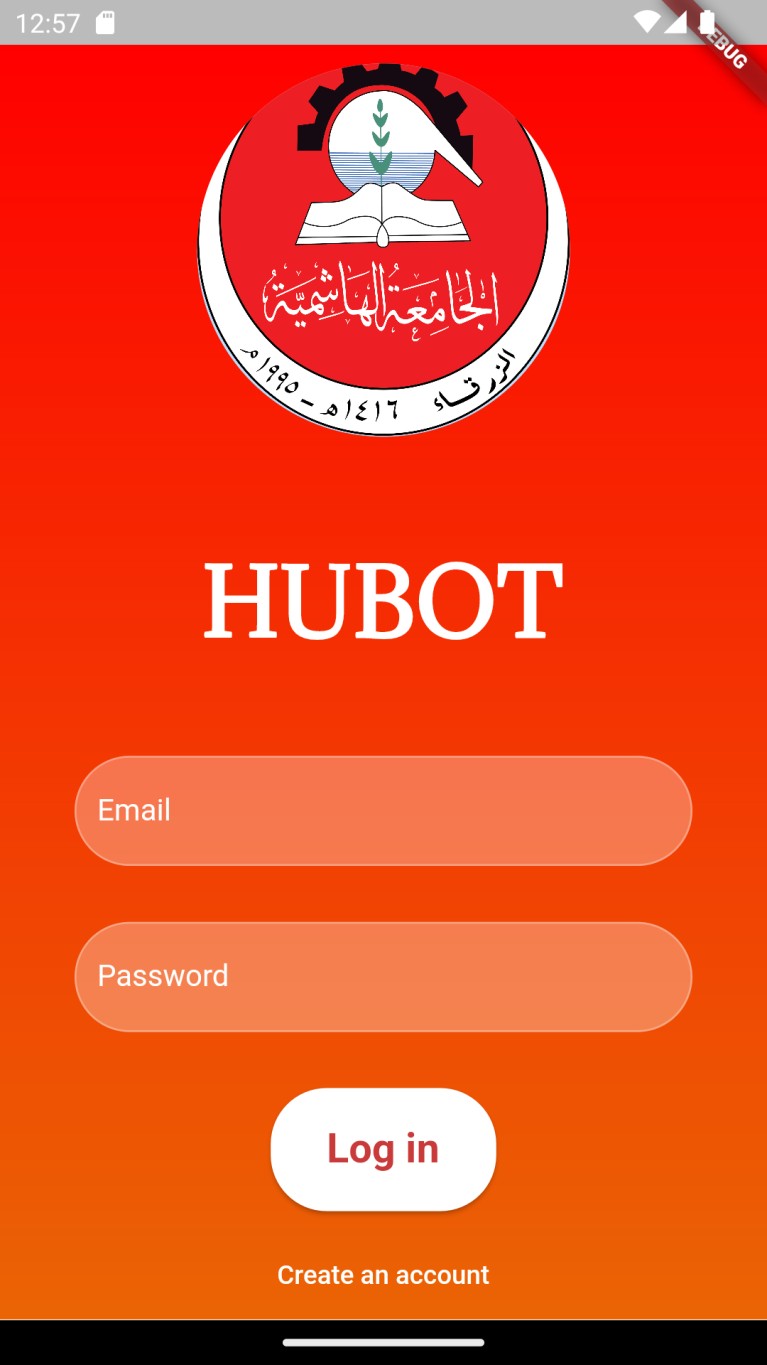


Figure 8: Caption for Figure 8 via *References->log in page*

A screenshot of a phone

Description automatically generated

Figure 9: Caption for Figure 9 via *References-> chatbot page*

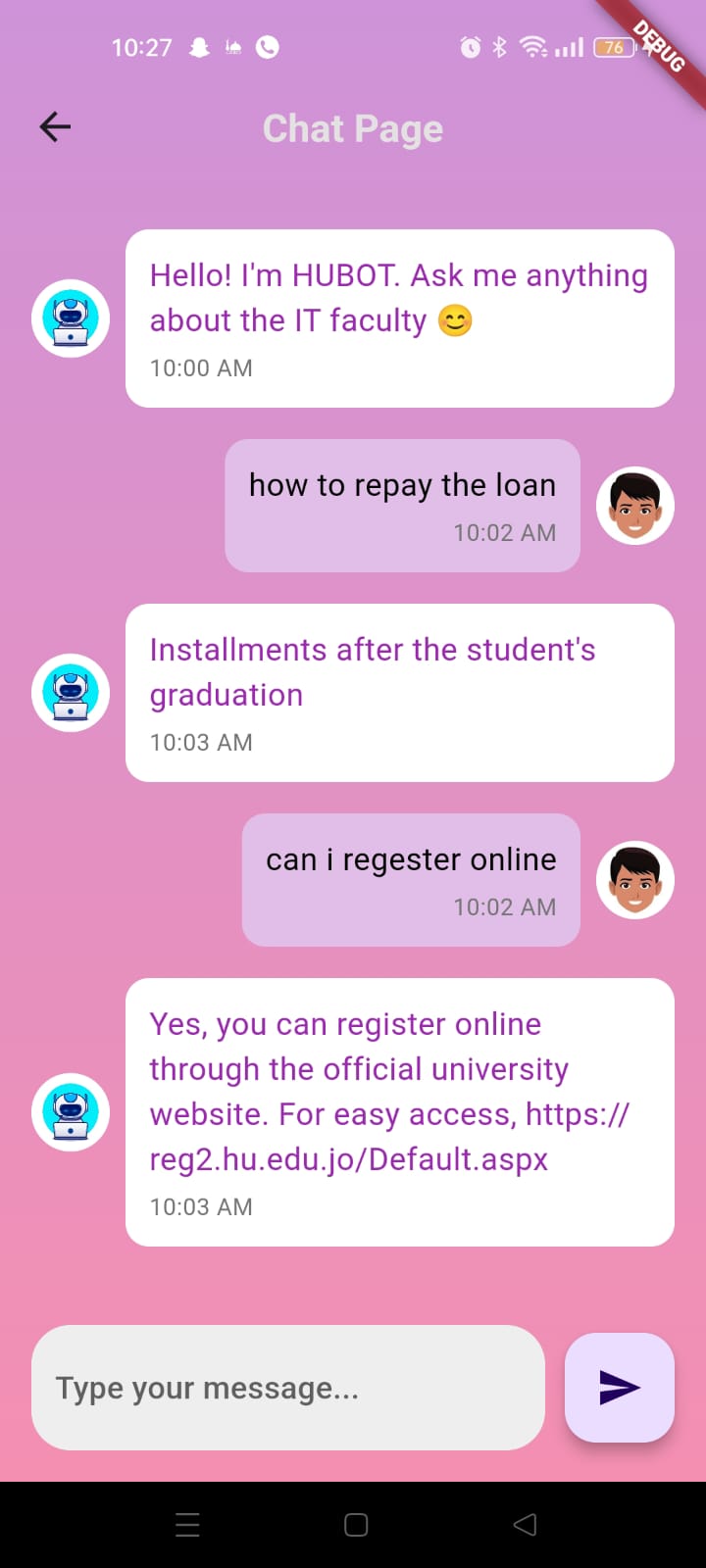


Figure 10: Caption for Figure 10 via *References-> talking with Hubot*

A screenshot of a cell phone

Description automatically generated

Figure 11: Caption for Figure 11 via *References-> Slide Menu*

A screenshot of a computer

Description automatically generated

Figure 12: Caption for Figure 12 via *References-> GPA Calc Adding Subject*

A screenshot of a computer

Description automatically generated

Figure 13: Caption for Figure 13 via *References-> Filling Subject Grades*

A screenshot of a cell phone

Description automatically generated

Figure 14: Caption for Figure 14 via *References-> GPA Result*

A screenshot of a calendar

Description automatically generated

Figure 15: Caption for Figure 15 via *References-> Adding a subject into std day manager*

A screenshot of a phone

Description automatically generated

Figure 16:Caption for Figure 16 via *References-> Subject added to std day manager*

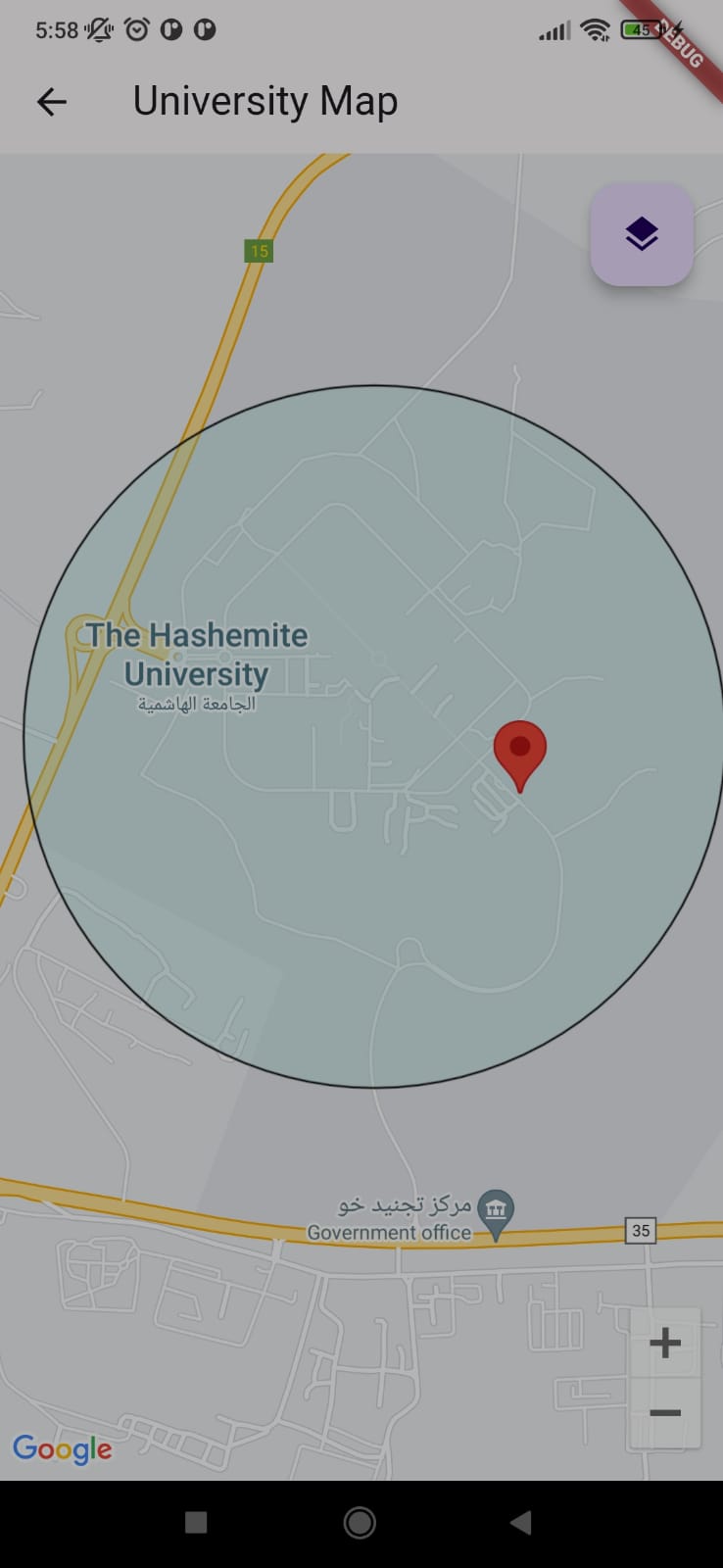


Figure 17:Caption for Figure 17 via *References-> Campus map*

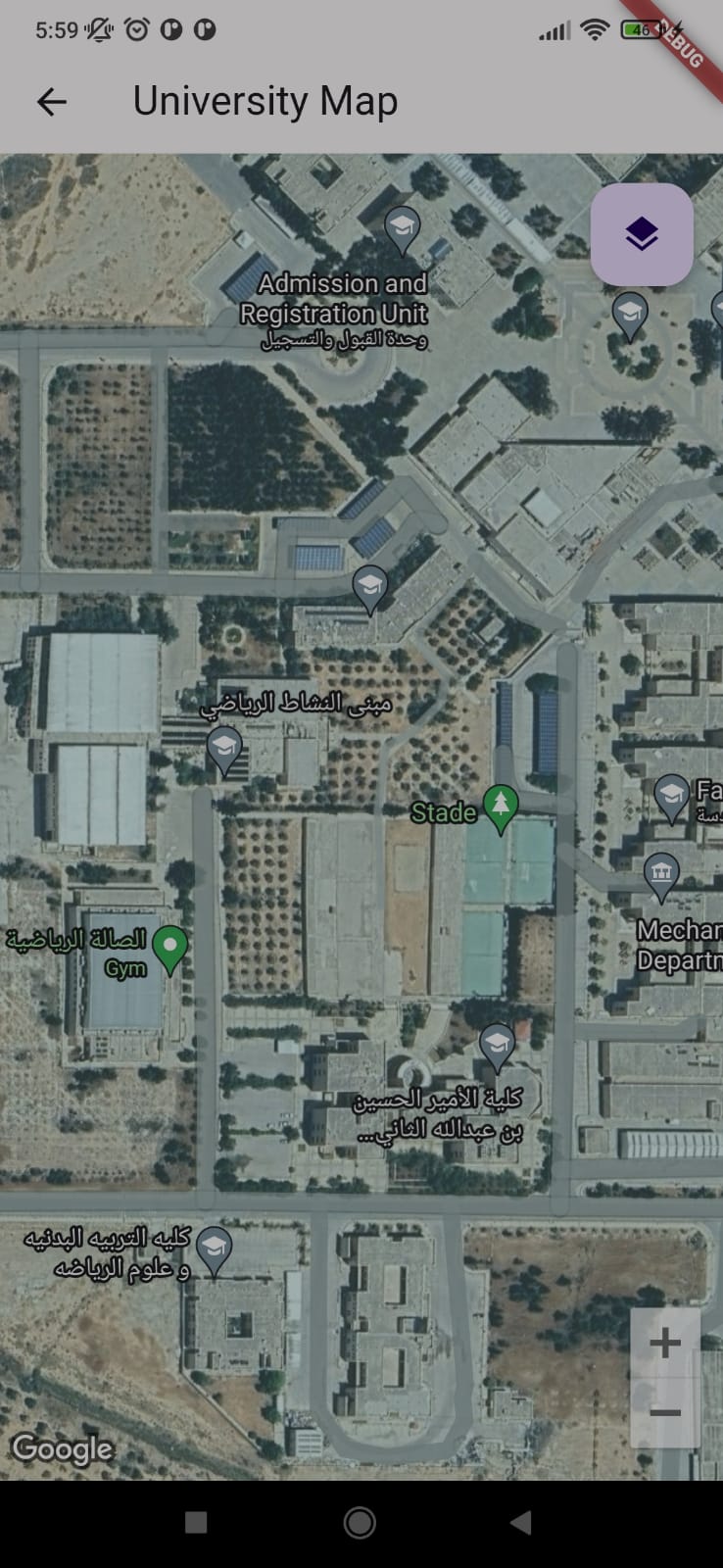


Figure 18:Caption for Figure 18 via *References-> Campus map internally*

## CHAPTER 5: IMPLEMENTATION PLAN

## Description of Implementation

In this section of the Project Implementation Plan, we will provide a detailed description of the implementation process for the HUBOT solution. It outlines how the information system will be deployed, installed, and transitioned into an operational system. This includes an overview of the major tasks and components involved in the implementation, as well as the necessary resources to support the implementation effort.

The implementation of HUBOT will involve several key steps and components. These include:

**Deployment and Installation:** The HUBOT system will be deployed and installed on the necessary hardware infrastructure. This includes setting up servers, network configurations, and software installations. The installation process will ensure that all components of the system are properly installed and configured to function seamlessly.

**Transition to Operational System:** Once the installation is complete, the HUBOT system will undergo a transition phase to become fully operational. This includes conducting testing and quality assurance procedures to ensure the system's functionality, reliability, and performance meet the required standards.

**Resource Requirements:** The implementation effort will require various resources to support its success. These resources include hardware, such as servers and networking equipment, software applications, databases, etc.

## Programming language and technology

The implementation of the HUBOT system will utilize the following programming languages and technologies:

**Flutter:** The HUBOT mobile application will be developed using the Flutter framework. Flutter provides a cross-platform development environment that allows for the creation of native-like mobile apps for both Android and iOS platforms.

**Java:** The server-side components and backend functionalities of the HUBOT system will be implemented using Java. Java is a widely used programming language known for its reliability and scalability.

**Visual Studio Code:** also commonly referred to as VS Code, IDE is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git [10].

**Intellij IDE:** Intellij is a popular IDE that offers a comprehensive set of tools and features for software development, including support for Java development. It provides a user-friendly interface, code editing capabilities, debugging tools, and project management features.

**MongoDB:** MongoDB will be used as the backend infrastructure and database for the HUBOT system. MongoDB is a popular open-source NoSQL (non-relational) database management system that is designed to store, query, and manage large amounts of data. It falls under the category of document-oriented databases, which means it stores data in a flexible, JSON-like format called BSON (Binary JSON). BSON stands for Binary JSON and is a binary-encoded serialization of JSON-like documents.

**Rest API:** In the context of the HUBOT system, the RESTful API will provide a set of endpoints that the mobile app can interact with to retrieve and manipulate data. These endpoints will follow the principles of REST, such as using HTTP methods (GET, POST, PUT, DELETE) to perform specific actions on resources and returning responses in a standardized format, typically JSON (JavaScript Object Notation).

**Spring Boot framework:** The implementation of the HUBOT system will leverage the Spring Boot framework to develop the server-side components and RESTful API. Spring Boot is a powerful Java-based framework that simplifies the development of robust and scalable applications.

Using Spring Boot with the Eclipse IDE, we can take advantage of its extensive tooling support and seamless integration. Eclipse provides features like code auto-completion, debugging, and project management, enhancing productivity during the development process.

Additionally, Spring Boot offers integration with various databases, including Firebase, allowing us to store and retrieve data efficiently. We can leverage the Firebase SDK and APIs to interact with the Firebase Real-time Database or Firestore for data persistence.

## By utilizing Spring Boot, we can build a scalable, efficient, and maintainable server- side application for the HUBOT system. It provides a robust foundation for implementing the RESTful API, handling requests, managing data, and ensuring the security .

## part of implementation if possible

## Flutter:

## A screenshot of a computer program Description automatically generated

Figure 19: Caption for Figure 19 via *References-> part of login page*

A screenshot of a computer program

Description automatically generatedFigure 20: Caption for Figure 20 via *References-> part of sigu up page code*

*A screen shot of a computer

Description automatically generated*Figure 21: Caption for Figure 21 via *References-> part of student registration page code*

A screen shot of a computer

Description automatically generatedFigure 22: Caption for Figure 22 via *References-> part of Chat Page code*

*A screen shot of a computer

Description automatically generated*Figure 23: Caption for Figure 23 via *References-> part of Slide Menu code*

A screen shot of a computer screen

Description automatically generatedFigure 24: Caption for Figure 24 via *References-> part of GPA calculator code*

*A screenshot of a computer program

Description automatically generated*Figure 25: Caption for Figure 25 via *References-> part of STD Day Manager code*

## JAVA including SpringBoot & REST API & MONGODB:

## A screenshot of a computer program Description automatically generated

Figure 26: Caption for Figure 26 via *References-> part of AdminController showing Spring annotation and REST CRUD operations code*

*A screenshot of a computer program

Description automatically generated*Figure 27: Caption for Figure 27 via *References-> part of USER Service describing business logic code*

*A screenshot of a computer program

Description automatically generated*Figure 28: Caption for Figure 28 via *References-> part of Instructor Data Link Class Communicating with Mongo DB*

*A screenshot of a computer

Description automatically generated*Figure 29: Caption for Figure 29 via *References-> MongoDB GUI*

integrity of the system.

## CHAPTER 6: TESTING PLAN

Describe the scope, approach, resources and schedule of intended test activities. It identifies amongst others test items, the features to be tested, the testing tasks, test coverage, degree of tester independence, the test environment, the test design techniques and entry and exit criteria to be used, and the rationale for their choice.

## Black-box

In order to ensure the quality and reliability of the HUBOT system, various black-box testing techniques will be employed. These techniques focus on testing the system's functionality without considering its internal structure or implementation details. The following black-box techniques will be utilized:

Equivalence Partitioning: Test cases will be designed to cover different equivalence classes of input data. For example, if the system accepts numerical input, test cases will be created to represent valid and invalid ranges of numbers.

Boundary Value Analysis: Test cases will focus on the boundary values of input ranges to ensure that the system handles them correctly. For instance, if a field accepts values

from 1 to 100, test cases will include values like 1, 100, and values just below and above these boundaries.

Decision Table Testing: This technique will be used to test the system's behavior based on different combinations of input conditions. Decision tables will be created to represent different scenarios and test cases will be designed accordingly.

EXAMPLES ON BLACK BOX TESTING:

A screenshot of a phone

Description automatically generated

Figure 30: Caption for Figure 30 via *References-> ENTERING WRONG USERNAME OR PASSWORD*

*A screenshot of a login form

Description automatically generated*

Figure 31: Caption for Figure 31 via *References-> ENTERING EXISTING USERNAME*

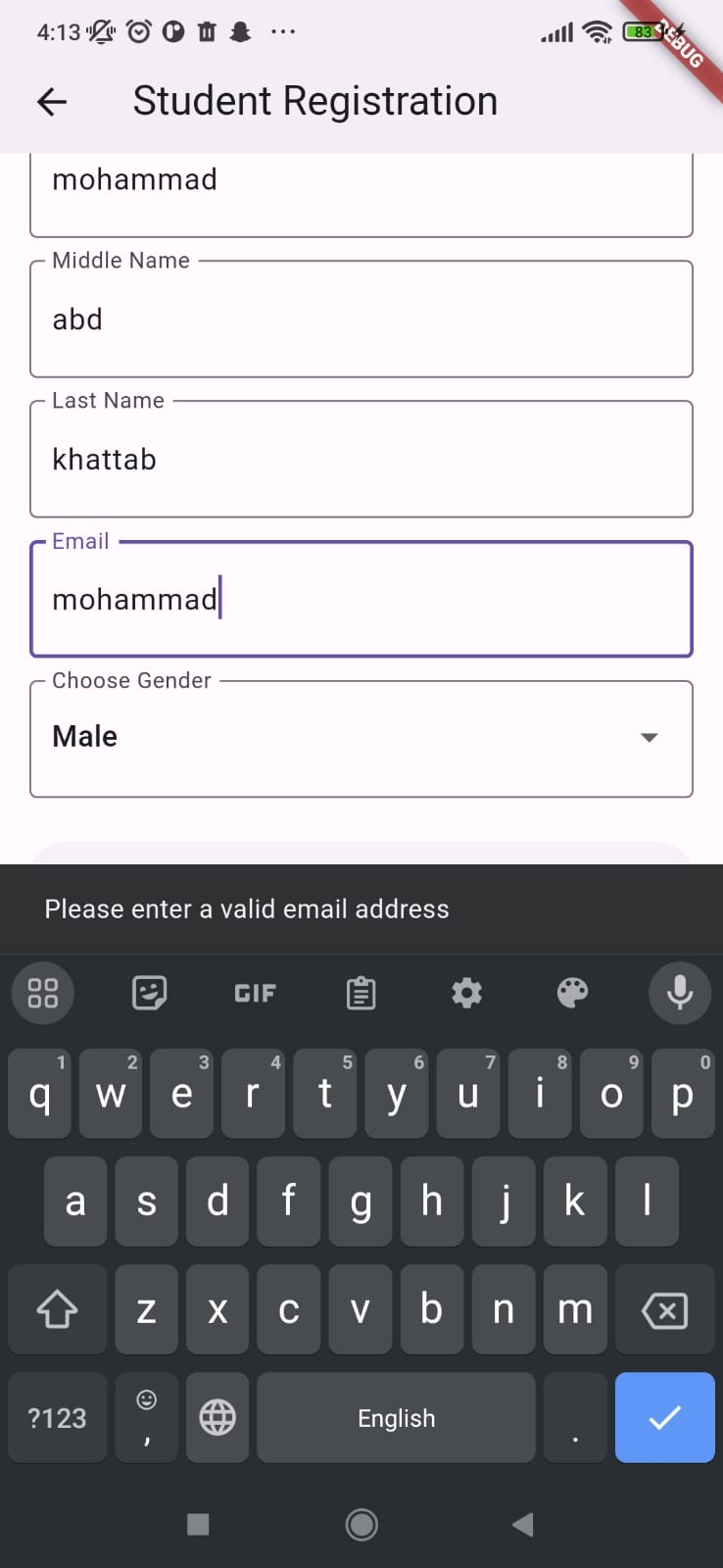
**

Figure 32: Caption for Figure 32 via *References->ENTERING INVALID EMAIL*

*A screenshot of a login screen

Description automatically generated*

Figure 33: Caption for Figure 33 via *References-> ENTERING DEFFERENT PASSWORDS*

## White-box

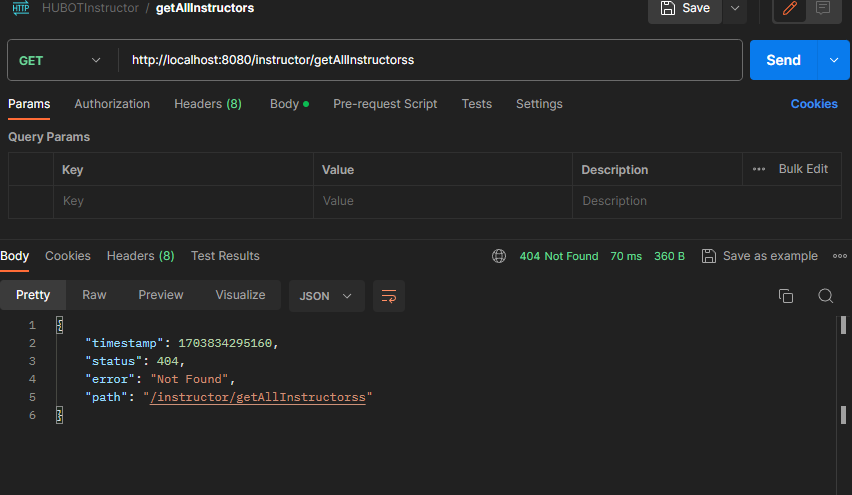
White-box testing techniques will be employed to assess the internal structure and implementation of the HUBOT system. As the code for the project is available, the following white-box techniques will be used:

Statement Coverage: Test cases will be designed to ensure that every line of code is executed at least once. This technique helps identify any dead or unreachable code segments.

Branch Coverage: Test cases will be created to ensure that all possible branches and decision points within the code are exercised. This helps uncover potential logical errors or missing conditions.

Path Coverage: Test cases will be designed to cover all possible paths through the code. This technique ensures that every possible combination of conditions and loops is tested.

EXAMPLES ON BLACK BOX TESTING:

Figure 34: Caption for Figure 34 via *References-> INVALID REQUEST*

A screenshot of a computer

Description automatically generatedFigure 35: Caption for Figure 35 via *References-> BAD REQUEST (METHOD NOT ALLOWED)*

A screenshot of a computer

Description automatically generatedFigure 36: Caption for Figure 36 via *References-> REQUESTING WHILE SERVER IS OFF*

## Testing automation

1. To improve the efficiency and effectiveness of testing, automation tools will be utilized. The following automation tools will be used for controlling test execution and comparing actual outcomes with predicted outcomes:

J Unit: JUnit is a widely used testing framework for Java applications. It provides a set of annotations and assertions that facilitate the creation and execution of automated tests. JUnit will be used for unit testing individual components of the HUBOT system.

1. Deciding which test cases to automate will be based on factors such as the frequency of execution, complexity, and criticality of the test case. Test cases that are repetitive, time-consuming, or require extensive manual effort will be prioritized for automation. Conclusion and Results

The conclusion is a required part that closes the document with a brief summary of the study including the problems found and the proposed solution. Most importantly, it should recommend to the readers the benefits of pursuing the project based on the researcher’s analysis.

## CHAPTER 7:CONCLUSION AND RESULTS

* 1. **Summary of Accomplished Project:**
     + more than 95% of prototype interfaces are ready.
     + we have determined the main functions we want to provide in HUBOT.
     + implemented the Java Spring Boot backend and Flutter framework frontend.
     + Implemented the REST API.
  2. **Future Work:** there is still room for further development and enhancement. Here are some recommendations for future work:

Enhancing Chatbot Intelligence: Continuously improve the Chabot’s natural language processing capabilities to understand and respond to a wider range of user queries accurately.

Expanding Smart Recommendation System: Extend the smart recommendation feature to suggest not only subjects and class schedules but also extracurricular activities, study resources, and campus events based on the user's preferences and academic profile.

Integration with External Systems: Explore the possibility of integrating the app with other university systems and services, such as the library system, student information system, or event management platform, to provide a more comprehensive and seamless user experience.

Collaborative Features: Implement features that enable users to collaborate on group projects, share study materials, and facilitate communication among students within the app.

Personalization and Customization: Allow users to personalize their app experience by customizing the app's theme, layout, and notification preferences according to their preferences.

Overall, the HUBOT mobile app presents a promising solution to improve the efficiency and convenience of campus life. Its current features provide a solid foundation, and future enhancements can further elevate the app's functionality and user experience. By pursuing this project, users can benefit from a comprehensive and intuitive tool that simplifies various aspects of their academic journey.