National University of Computer and Emerging Sciences, Lahore

Automated Registration System

Ahmad Javaid 21L-1885 BS(CS)

Menal Naeem 21L-1872 BS(CS)

Abdul Raheem Jalil 21L-5287 BS(CS)

Supervisor: Saif-ul-Islam

Final Year Project

April 13, 2025

Anti-Plagiarism Declaration

This is to declare that the above publication was produced under the:

Title: Automated Registration System

is the sole contribution of the author(s), and no part hereof has been reproduced as it is the basis (cut and paste) that can be considered Plagiarism. All referenced parts have been used to argue the idea and cited properly. I/We will be responsible and liable for any consequence if a violation of this declaration is determined.

Date: .....

Name: .....

Signature: .....

Name: .....

Signature: .....

Name: .....

Signature: .....

Author's Declaration

This states Authors' declaration that the work presented in the report is their own, and has not been submitted/presented previously to any other institution or organization.

Abstract

This project aims to develop an extensive chatbot system designed to make the course registration process at the university more efficient and student-friendly while offering AI-based answers to common queries. The primary goal is to simplify students' interactions with the university's management, minimize administrative workload, and optimize the overall course registration experience. Leveraging cutting-edge technologies such as Artificial Intelligence (AI) and Natural Language Processing (NLP), the chatbot will engage in human-like conversations, allowing students to easily ask questions and receive immediate responses. The system will provide customized support, tailoring responses based on individual student profiles, previous interactions, and registration history. Additionally, the chatbot will be trained on a large dataset that analyzes frequently asked questions and recurring issues, thus enhancing its ability to provide relevant, real-time information on course availability, prerequisites, deadlines, and university policies. To ensure a seamless user experience, the extensive system registration database will enable students to browse available courses, add or drop classes, and receive notifications about registration deadlines and other vital events. This automation will not only streamline the registration process but also minimize human errors and improve overall efficiency. Furthermore, the chatbot will be accessible 24/7, offering students a reliable and user-friendly platform for resolving queries related to registration, fee payment, course material access, and academic requirements, making university administration more approachable and less time-consuming for both students and faculty.

Executive Summary

Most universities in Pakistan make use of the traditional practices of registering a student for the semester with a manual process which leads to complexities, errors , and frustration for the students involved. The university administration is also burdened which often results in mismanagement. This project focuses on solving the inefficiency of the current system in most universities by introducing an AI powered chatbot website that enhances the experience for both the students and the administration.

This project aims to develop a seamless platform where the overall user experience is greatly elevated as compared to the existing one.The robust and interactive website will allow students to register courses, view their academic schedule, transcripts and access several other features.The main goal of the project is to create an AI powered chatbot that will offer assistance to the students, thereby reducing the administrative burden.The chatbot will be able to provide proper guidance to the students including the course availability, pre-requisites, course selection and even generate the preferred timetable for the user.Additionally, the chatbot will cater to any queries students might have regarding their academic year.College administrators will have a seperate login window where they can access and track student registrations, monitor section allocations amongst other things. Python, Dialogflow and BERT are used to develop the chatbot while JavaScript with React and NodeJS will power the website.

Our website nearly automates the registration process with students finding it easier to register their courses and access technical support via the extensively trained chatbot we have developed. The project is a user-friendly solution for the increased workload that administrators face. The AI powered chatbot developed with python along with an aesthetically pleasing website created with React and NodeJS ensures that the project is scalable and can adapt to future changes and updates as well.

Table of Contents

List of Figures **ix**

1 Introduction **1**

1.1 Purpose of this Document 1

1.2 Intended Audience 2

1.3 Definitions, Acronyms, and Abbreviations 2

1.4 Conclusion 2

2 Project Vision **3**

2.1 Problem Domain Overview 3

2.2 Problem Statement 3

2.3 Problem Elaboration 4

2.4 Goals and Objectives 4

2.5 Project Scope 5

2.6 Sustainable Development Goal (SDG) 5

2.7 Constraints 6

2.8 Business Opportunity 6

2.9 Stakeholders Description/ User Characteristics 6

2.9.1 Stakeholders Summary 7

2.9.2 Key High-Level Goals and Problems of Stakeholders 7

3 Literature Review / Related Work **8**

3.1 Definitions, Acronyms, and Abbreviations 8

3.2 Detailed Literature Review 9

3.2.1 The Development of Chatbot Provided Registration Information Services for Students 9

3.2.2 Intelligent Conversational Agent for Enhancement of Online Communication in Selected Universities Using Pattern Matching Algorithm 10

3.2.3 AI-Chatbot Improve Efficiency in Handling Student Queries 12

3.2.4 iNOUN: Architecture and Usability of a Chatbot for Academic Enquiries 13
3.2.5 **Interactive Advising with Bots: Improving Academic Excellence in Educational Establishments** 14
3.2.6 **Chatbots in the Education System** 15
3.3 Literature Review Summary Table 16
3.4 Conclusion 16

4 Software Requirement Specifications **18**
4.1 List of Features 18
4.2 Functional Requirements 19
4.3 Quality Attributes 20
4.4 Non-Functional Requirements 20
4.5 Assumptions 21
4.6 Use Cases 21
4.6.1 Use Case 1: View Available Courses 22
4.6.2 Use Case 2: Course Recommendation Based on Profile 23
4.6.3 Use Case 3: Access and View Student Profile 24
4.6.4 Use Case 4: Wait-list Management 25
4.6.5 Use Case 5: Manage Add/Drop Requests 26
4.6.6 Use Case 6: Generate Reports 27
4.6.7 Use Case 7: Course Management 28
4.6.8 Use Case 8: Deadline Configuration 29
4.6.9 Use Case 9: Teacher Management 30
4.6.10 Use Case 10: Student Management 31
4.6.11 Use Case 11: Grader Management 32
4.6.12 Use Case 12: View Grades 33
4.6.13 Use Case 13: Login 34
4.6.14 Use Case 14: Chatbot invalid Queries 35
4.7 Hardware and Software Requirements 35
4.7.1 Hardware Requirements 35
4.7.2 Software Requirements 36
4.8 Graphical User Interface 37
4.8.1 ER Diagram 39
4.8.2 Data Dictionary 39
4.9 Risk Analysis 42

4.9.1 1.Technical Risks . . . 42
4.9.2 2.Risks to Business . . . 42
4.9.3 3.Operational Risk . . . 43
4.10 Conclusion . . . 43

5 High-Level and Low-Level Design 45
5.1 System Overview . . . 45
5.2 Design Considerations . . . 45
5.2.1 Assumptions and Dependencies . . . 45
5.2.2 General Constraints . . . 45
5.2.3 Goals and Guidelines . . . 46
5.2.4 Development Methods . . . 46
5.3 System Architecture . . . 46
5.3.1 Subsystem Architecture . . . 47
5.4 Architectural Strategies . . . 48
5.4.1 Programming Language . . . 48
5.4.2 User-interface Paradigm . . . 48
5.4.3 Concurrency and Synchronization . . . 49
5.4.4 Memory management . . . 49
5.5 Domain Model/Class Diagram . . . 49
5.6 Policies and Tactics . . . 50
5.6.1 Plans Of Testing . . . 50
5.6.2 Hierarchical Organization . . . 50
5.6.3 Maintenance . . . 50
5.6.4 End User Interfaces . . . 50

6 Implementation and Test Cases 51
6.1 Implementation . . . 51
6.1.1 Implementation of Data Collection and Preprocessing . . . 51
6.1.2 Back-end Development . . . 52
6.1.3 Frontend Development . . . 52
6.1.4 Chatbot Development Using BotPress . . . 53
6.1.5 User Interface Design . . . 53
6.1.6 Integration and Scalability . . . 53
6.1.7 Conclusion . . . 53

6.2 Test case Design and description 54
6.3 Test Metrics 62

7 User Manual **63**
7.1 Introduction 63
7.2 System Requirements 63
7.3 How to Access the System 63
7.4 Student User Manual 63
7.4.1 Key Features for Students 63
7.5 Teacher (Professor) User Manual 64
7.5.1 Key Features for Teachers 64
7.6 Admin User Manual 65
7.6.1 Key Features for Administrators 65
7.7 Troubleshooting Guide 66
7.8 Support Contact 66

8 Experimental Results and Discussion **67**
8.1 Introduction 67
8.2 Analysis and Discussion 67
8.2.1 Chatbot Response Accuracy 67
8.2.2 System Processing Speed 67
8.2.3 User Satisfaction Survey 67
8.2.4 System Reliability 68
8.3 Conclusion 68

9 Conclusion and Future Work **69**
9.0.1 Summary and Final Thought: 69
9.0.2 Key Findings and Accomplishments: 69
9.0.3 Scope Coverage and Challenges: 70
9.0.4 Goal Fulfillment: 70
9.1 Future Work and Recommendations: 70

List of Figures

2.1 Quality Education 6

4.1 Illustration of Conversation window 37
4.2 Illustration of login page for Admin 38
4.3 ChatBot-ER diagram 39

5.1 System Architecture Layout 47
5.2 Class diagram for University registration system with AI chatbot 49

List of Tables

3.1 Related Work Summary Table 17

4.1 Use Case 1: View Available Courses 22
4.2 Use Case 2: Course Recommendation Based on Profile 23
4.3 Use Case 3: Access and View Student Profile 24
4.4 Use Case 4: Wait-list Management 25
4.5 Use Case 5: Manage Add/Drop Requests 26
4.6 Use Case 6: Generate Reports 27
4.7 Use Case 7: Course Management 28
4.8 Use Case 8: Deadline Configuration 29
4.9 Use Case 9: Teacher Management 30
4.10 Use Case 10: Student Management 31
4.11 Use Case 10: Grader Management 32
4.12 Use Case 12: View Grades 33
4.13 Use Case 13: Login 34
4.14 Use Case 14: Chatbot invalid Queries 35

6.1 Student Panel - Available Courses 54
6.2 AI Course Recommendation System 55
6.3 Student Profile Management 55
6.4 Course Waitlist System 56
6.5 Add/Drop Request Management 56
6.6 Admin Course Management 57
6.7 Deadline Configuration 57
6.8 Report Generation 58
6.9 Admin Add Professor 58
6.10 Admin Add Student 59
6.11 Professor - Upload Grades 59
6.12 Student Dashboard - View Grades 60

6.13 Student Login Module 60
6.14 Teacher Login Module 61
6.15 Chatbot Error Handling 61
6.16 Test case Matric 62

7.1 Common Problems and Solutions 66

Chapter 1 Introduction

Whether you are a university student or an admin, one of the most complex processes you will encounter in your academic year is that of the semester registration. Many educational institutions including FAST NUCES Lahore lack an efficient and user-friendly registration system, leaving both the students and administration frustrated.

University Registration is an integral part of any higher education system. This project aims to address the growing complications regarding the enrollment of students in an academic year to ease the process for both the students and faculty. Our project focuses on developing an AI powered chatbot that streamlines the semester enrollment process.

The chatbot enabled by Natural Language Processing(NLP) and trained on a vast dataset will be able to handle any query related to the registration process. It will provide real-time information regarding course selection, sections availability, and other vital FAQs while also managing to generate preferred timetable of a student's choice. Additionally, the chatbot will be able to assist students by troubleshooting any issues they might be facing in their enrollment process.

The primary objective of this project is to reduce administrative overhead and enhance the experience for the students. Our goal is to develop an extensive chatbot system that automates assistance in such a way that it almost erases the need for physical guidance and delivers a tailored experience for the students.

1.1 Purpose of this Document

This document gives a detailed insight into our designing and implementation of our Final Year Project. The goal of our project is to significantly improve the student registration process by means of Artificial Intelligence (AI). The question relevant to this project is: Does an AI-powered chatbot have the potential to provide robust and accurate information and assistance to students?

This report will analyze the design and methodologies of the chatbot and dive deep into the implementation of the project. It will thoroughly cover the Natural Language processing (NLP) and machine learning (ML) used to develop the chatbot. The document will also showcase the outcome of our project and the potential challenges we encountered. The main goal of this research is to provide a personalised seamless registration process for the students.

1.2 Intended Audience

The primary stakeholders for this project are the university administrators and students. They are the ones whose interest is aligned the most with our project as they have to face the registration problem every other year. This report will focus on mainly addressing the students, a key audience, explaining the functionality of the chatbot and how it serves the purpose of the students in terms of course registration, choosing sections, tackling issues and keeping a track of important dates and deadlines. It will also be intended for the administration and the IT support staff, as to how the project will reduce their workload.

1.3 Definitions, Acronyms, and Abbreviations

List all important definitions, acronyms, and abbreviations used in this document. For example: **AI**: Artificial Intelligence

FYP: Final Year Project

NLP: Natural Language Processing

UI: User Interface

UX: User Experience

ML: Machine Learning

1.4 Conclusion

This document outlines a detailed and comprehensive overview of the project. It starts off by setting a tone and purpose of the project in chapter one. It goes on to define the problem statement and the key goals and objectives alongside project scope and potential business opportunities in chapter two. The report moves on to literature review in chapter three. It then delves into the software requirement specifications in chapter four leading to a detailed analysis of high and low level design in chapter five.

Chapter 2 Project Vision

Chapter two aims to explore the ultimate challenges that our chatbot will tackle while clearly defining the goals and objectives of our project. By going through this chapter, readers will be able to fully comprehend the scope of our project and the sustainable development goals our Final Year Project (FYP) targets.

2.1 Problem Domain Overview

The system is designed to give students a rich and enhanced platform where they can register semester courses in a smooth flow. It follows all the standard security protocols, allowing students and administrators likewise to securely login and access their account. The key aspect of the project is the AI powered chatbot. It allows students to easily gather any information or answer any query in a matter of seconds. The chatbot is extensively trained so as to assist students in any technical query they might have such as course selection, eligibility, registration procedures and much more. It eliminates the need for manual assistance which removes the burden of administrators. The system is built using JavaScript with react for an interactive and aesthetic frontend while NodeJS and MongoDB handle the backend of the website. The chatbot is trained with a large dataset on Python, while also considering Dialogflow and BERT.

2.2 Problem Statement

The universities in Pakistan lack a modernized registration system. The outdated and traditional system has become a headache and a challenge for the administrators as they encounter an extremely low efficiency rate every year. Inaccurate query responses, delayed course registration and long queues to resolve minor inconveniences discourage and frustrate both the students and faculty. The peak registration season attracts a large amount of traffic and often forces the website to crash for several hours particularly in FAST NUCES. The project aims to resolve all of these issues by streamlining the registration process with a smooth user interface, and nearly automating the system with the AI powered chatbot that handles all of the technical support and queries of the students while prioritising accurate and fast responses.

2.3 Problem Elaboration

The primary objective of this project is to maximise efficiency of the system while reducing the need for manual support by a significant percentage. However, we face a lot challenges that hinder our goal to build an advanced system:

1. **Outdated error prone process**: Most of the traditional registration systems either include manual registrations ie by filling out forms,or, slow and not so user-friendly websites which are hard to navigate. These outdated processes usually tend to raise more problems rather than solve the existing ones. Incorrect course selections and system glitches of ignoring a student's pre-requisite warning are then to be manually resolved by an admin, increasing their workload.
2. **Administrative burden**: The peak registration season sees a massive workload piled on the university administration. This is due to the high influx of students queries and issues which often lead to delays in deadlines and complexities.
3. **Lack of Real time support**: To check course requirements, available sections, or access class timings before registration requires a student to physically visit the administrative office which ends up taking a lot of time. In many cases, students are not guided properly and they end up with a bunch of issues and queries which are mostly not addressed. Section change and incomplete registrations are the most common problems faced and in traditional systems often only resolved by the admin.

2.4 Goals and Objectives

The objectives of our final year project are as follows:

- To employ a chatbot system for the university's course registration process.
- To advance the chatbot's use of AI with NLP by training it rigorously on a relatively large dataset so as to respond to extensive and complex queries.
- To greatly minimize the amount of administrative workload involved in student inquiries and course registration
- To enhance the relevance and accuracy of the details that students are given during the registration process.
- To facilitate the students with a smooth process for semester enrollment by providing real time information on course availability and other vital aspects.

2.5 Project Scope

Our project aims to streamline the university registration process from both the perspective of students and administration. The project will allow the users to access an AI-powered chatbot with high efficiency and accuracy in order to deal with any query related to the enrollment process. Our system will allow users to:

- Authenticate login
- Access their full portfolio including transcripts
- offer personalized recommendations via chatbot
- ask chatbot for the available sections and courses
- ask chatbot for the best possible timetable excluding any clashes
- prompt chatbot to resolve or troubleshoot any issue

The front-end of this project will make use of JavaScript with React to develop an interactive and highly responsive UI. The back-end of the project will rely upon Node.js to ensure robust performance. Additionally, Python, DialogFlow, and BERT will be utilized to develop and train chatbot. Lastly, MongoDB is preferred for a flexible and efficient database. The project deliverables will include:

- A fully functional website which university administrators and students both can access
- A detailed documentation outlining the design, implementation, and testing process

The project scope does not include the following:

- Providing hosting or server infrastructure for the application
- Developing mobile applications (iOS or Android)
- Integrating the application with third-party services or APIs

2.6 Sustainable Development Goal (SDG)

Our project perfectly aligns with the Sustainable Development Goal of Quality Education. We aim to make the lives of both students and faculty easier by developing an advanced registration system. This objective clearly reflects our ambition to serve the world with quality education and to promote out-of-the-box thinking among our readers by fostering broad problem solving skills.

Figure 2.1: Quality Education

2.7 Constraints

Our project's main focus is the chatbot system for university registration. However, developing a chatbot involves several constraints. Data limitations refer to lack of a reliable and large dataset to fully train the chatbot. Another technical constraint might be user's ambiguous inputs. It will be quite challenging for the chatbot to interpret and respond to the user's queries if they are in a language it can't comprehend. Lastly, high latency rate can result in a degraded website performance leading to an unsatisfactory user experience.

2.8 Business Opportunity

Our project "University Registration System with AI chatbot" has the potential to attract a lot of Business opportunities. Globally, over the past few years, university registrations have skyrocketed with a large number of educational institutions still facing the challenge of smooth and timely enrollment process. Our system reduces the administration overhead and with the assistance of AI chatbot almost automates the registration process. This project tackles the challenges faced by numerous colleges worldwide and offers a highly scalable solution. Moreover, diving into today's trend of AI in streamlining everything exponentially increases our chances to market our product successfully.

2.9 Stakeholders Description/ User Characteristics

The key stakeholders of our system are as follows:

- Students: Our chatbot enabled registration system will be widely used by the students of that particular organization.

- University administration: Administrators can easily track and maintain courses and student records with our easy to use website.
- Educational institutions: Other educational based institutions might be inclined to use our chatbot system by tailoring it to their needs.

2.9.1 Stakeholders Summary

The University Registration system has a diverse audience from students and high level administration to IT support staff, everyone has their own interests in it. The key stakeholders are the students who are greatly dependant on the system for queries and information such as course offerings, availability of sections, registering online complaints etc. However, the faculty has more or less the same shared stake in it. Colleges worldwide have a large expenditure on their registration related administrative teams. Still, they fall short of the expectations and under-perform mostly as they are unable to enroll the students in a timely and smooth manner. Our system eases the task for these educational institutions, with our AI-powered chatbot paving the way for a seamless enrollment process.

2.9.2 Key High-Level Goals and Problems of Stakeholders

Stakeholders in our project have set targets to achieve. This will be possible only after we eliminate the challenges they encounter. Our project plays a critical role in a student's enrollment process. With the current system in place, students are unable to register in the courses of their choice in the sections they prefer with any clashes or issues only being resolved via physically going to the registration office. Our system aims to completely remove the notion of physical meetings and instead shift to a hundred percent online and automated system. This point also covers the demand set by the faculty as they want to reduce their burden and increase productivity in other matters. Their goal of an effective real time registration system is being answered by our AI powered chatbot.

Chapter 3 Literature Review / Related Work

This section contains a thorough analysis of prior researches done on; incorporating AI-Driven chatbots in educational environments, developing conversationalist bots for student support and streamlining the registration process via advavnced NLP and AI-Algorithims.

3.1 Definitions, Acronyms, and Abbreviations

Some of the key abbreviations , acronyms and definitions of concepts are given below:

- **Chatbot:** A software application designed to simulate human-like conversations with users, typically using natural language processing (NLP) techniques.
- **Natural Language Processing (NLP):** A branch of artificial intelligence that focuses on the interaction between computers and humans through natural language. It enables chatbots to understand, interpret, and generate human language.
- **Artificial Intelligence (AI):** The simulation of human intelligence in machines that are programmed to think, learn, and solve problems autonomously.
- **Decision Support System (DSS):** A computer-based system that supports decision-making activities by providing relevant data and analyses. In the context of chatbots, it can assist in advising students based on their academic records.
- **User Interface (UI):** The space where interactions between humans and machines occur. In the chatbot project, the UI refers to the chat page or conversation interface where students communicate with the bot.
- **Knowledge Base:** A collection of information that the chatbot draws upon to answer queries. This includes predefined responses, FAQs, and data relevant to the university's academic processes.
- **Machine Learning (ML)**: A subset of AI that allows machines to learn from data, improving their performance on specific tasks over time without being explicitly programmed for each task.
- **API**: Application Programming Interface. A set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other services.
- **AI**: Artificial Intelligence. The broader concept of machines being able to carry out tasks in a way that we would consider "smart."
- **UI**: User Interface. The part of the system with which users interact.

- **ER**: Entity-Relationship (Diagram). Used for database modeling and structure representation.
- **DSS:** Decision Support System. A tool that helps in making informed decisions based on data.
- **SRS:** Software Requirements Specification. A document that outlines the functional and non-functional requirements of the system being developed.
- **LTSM:** Long Short-Term Memory. A type of artificial recurrent neural network architecture used in machine learning to handle sequence prediction problems like language modeling.
- **RNN**: Recurrent Neural Network. A class of neural networks that is effective for sequential tasks such as language modeling and chatbots.
- **SUS:** System Usability Scale. A scale used to measure the usability and user satisfaction of a system.
- **AIML:**Artificial Intelligence Markup Language.

3.2 Detailed Literature Review

A thorough literature review of existing researches related to the use of AI-driven generative chatbots in universities for student support and enrollment is given below:

3.2.1 The Development of Chatbot Provided Registration Information Services for Students

:

Summary of the Research Item:

The research paper[1] focuses on;designing chatbots given the registration related information, the structure of communication between the bot and students and the evaluation and analysis of the accuracy of chatbots responses. This study used a Research and development approach involving 16 staff members, to obtain registration related information, and 255 undergraduate students that used a complete prototype of the chatbot. Results showed majority of the students were highly satisfied with the responses and accuracy of the chatbot. This paper's main objective is to utilize registration data to better understand student needs and guide the development of AI chatbots for student services, with a focus on distance learning. With the evolution of artificial intelligence in recent years, chatbots have become more and more popular due to their ability to handle the myriad of questions that come up during the registration process. The study highlights how important it is that students get timely and accurate information about rules, notifications, and enrollment processes.Furthermore, the study emphasizes how availability

and working hours frequently restrict staff response times, which may hamper timely transmission of information and makes the process difficult. Chatbots appear to be a plausible solution in this situation. They have proven to be affordable and scalable solutions for responding to student inquiries because they provide prompt, accurate responses around-the-clock (Bii, 2013; Rahman et al., 2017; Fryer et al., 2019; Khanthavit & Khanthavit, 2023). Additionally, the authors offer a useful classification of chatbots according to their intelligence, user experience (UX), and intended uses. The goal of the chatbot's development and design phase is to make the interface easy to use and improve user interactions by utilizing natural language processing (NLP).

Critical Analysis of Research Item:

This research provides useful information about how AI chatbots can be used to handle registration queries efficiently. Its dual focus on users (students) and staff (who maintain the chatbot) is one of its main advantages. The system will continue to be easy for students to use and manageable for the staff who will be using it due to this comprehensive approach. The study's conclusions are further strengthened by the statistical data used to highlight the importance of timely and accurate information.

However, the research has certain limitations . Its dependence on Dialogflow may limit customization and flexibility for more complicated scenarios. Further study and development on this topic should take this into account.

3.2.1.1 Relationship to the proposed research work:

This case study, which also intends to streamline university registration and improve student support through chatbots, is closely aligned with the propsed project. The proposed project aims to manage a large number of student inquiries and expedite the registration process by using AI-driven chatbots, which aligns with the research findings. Notably, the fact that Dialogflow is used in both projects demonstrates how well-suited the platform is for handling these kinds of jobs, streamlining design and facilitating multi-modal interactions. All things considered, this study offers a strong framework for designing the chatbot and assessing its performance using metrics like accuracy, user satisfaction, and overall system performance.

3.2.2 Intelligent Conversational Agent for Enhancement of Online Communication in Selected Universities Using Pattern Matching Algorithm

Summary of the Research Item:

This study [2]looks into the development and implementation of chatbots powered by artificial intelligence, specifically in educational environments. The goal is to develop a chatbot that can be easily

integrated with the university's website to replace the need for current human assistance in online communication. Through the use of Pandorabots and Artificial Intelligence Markup Language (AIML), the system offers 24/7 support that can respond to questions from students. The chatbot's knowledge base can be customized, enabling it to be tailored for various types of institutions. This study utilized a qualitative research approach and further classified the development of the chatbot into 4 phases; design ,implementation, testing and measurement. Every stage provides valuable insights, especially when it comes to creating the chatbot's knowledge base, importing information, and evaluating the system's overall effectiveness . The evaluation of the study revealed that chatbot was able to accurately respond to the student queries in 95 percent of the cases. This study illuminates the development of AI technologies, highlighting the significance of machine learning (ML) and natural language processing (NLP) in expanding chatbot interaction attributes.

Critical Analysis of Research Item:

With a clear focus on the technical architecture and customization aspects, the research offers useful information about the process of developing chatbots. In an academic setting in particular, it effectively illustrates how conversational agents can be created through the use of AIML and Pandorabots. The study's strength is its thorough approach to creating an adaptable chatbot system that can be used in different kinds of educational settings. The study does, however, also point out important limitations. For example, the chatbot's adaptability is reduced by limiting input to proper English and depending on a predefined knowledge base. This could hinder its ability to handle more dynamic conversations or serve a diverse, multilingual student body. Furthermore, even though the paper explores chatbot architecture, it should have looked into advanced AI models that provide more flexibility and superior handling of complex, unstructured queries.

Relationship to the proposed research work:

This study is extremely relevant to our project since it also entails building a chatbot system to manage registration procedures at universities. The goal of both projects is to improve student services by using conversational agents to automate important tasks. The study's incorporation of AIML and the chatbot's adaptable knowledge base is similar to our strategy of employing AI to accelerate student interactions—particularly with regard to registration-related inquiries.

Furthermore, the research's focus on the stages of development—from design to evaluation—aligns with the development process our chatbot system will employ. The information obtained from the study's limitations, such as the significance of natural language processing and the need to extend chatbot functionality beyond text input, will be especially helpful in improving our system's adaptability and user experience.

3.2.3 AI-Chatbot Improve Efficiency in Handling Student Queries

Summary of the Research Item:

The study[3] highlights the growing necessity for academic institutions to implement AI-powered tools for better communication, particularly in light of the COVID-19 pandemic's forced transition to hybrid learning. The flood of new students made it impossible for conventional means of communication to keep up with the increasing number of questions from students. To address this, the researchers developed a chatbot using AI, NLP, and the LTSM model to manage student inquiries efficiently. Through the use of a quantitative methodology, the chatbot's accuracy, usability, and speed of information retrieval were evaluated. The bot did, however, have certain limitations, such as its reliance on the English language and its inability to respond to questions outside of its knowledge base. Notwithstanding these drawbacks, the chatbot effectively handled topics like scheduling, exams, admissions, and registration while offering round-the-clock accessibility. According to Adam, Wessel, and Benlian (2021), AI-based applications have grown more popular due to their affordable pricing and up to 80% response time reduction. This study built a chatbot using RNN to assess its accuracy and efficacy in order to investigate how AI-driven systems can be used to handle student inquiries. A hybrid strategy may be the most successful, according to the research, which divided chatbots into two categories: retrieval models (with pre-established patterns) and generative models (self-learning). The system's components, which included an NLP engine and a user interface (UI), allowed it to efficiently manage dialogues and classify intent.

The waterfall methodology used in the development process offered an organized method for developing systems. Efficiency-wise, the chatbot received an average rating of 4.10/5, but the user interface (UI) received a lower rating (3.5/5), especially from non-technical users.

Critical Analysis of the Research:

The research provides insight into the advantages and practical difficulties of deploying AI chatbots in academic settings. Positively, by responding to students' repetitive questions quickly and accurately, the chatbot helped ease the workload of university staff. The study also identifies two important limitations of the chatbot, including its exclusive reliance on the English language and its incapacity to handle queries that are outside of its scope. Future developments must take these flaws into account as they restrict the chatbot's ability to effectively serve a diverse student body. It was appropriate to employ a quantitative testing strategy, which offered concise information about the chatbot's functionality. For a better understanding of the bot's limitations, more qualitative feedback regarding students' and staff's experiences using it would be helpful. Though the study offers useful technical information on chatbot architecture, it lacks of a thorough discussion on enhancing user experience, particularly for

non-technical users, which would have increased the applicability of the findings to actual use cases.

Relationship to the Proposed Research:

This research is crucial to our project because we are also developing an AI chatbot to handle university-related inquiries, such as registration. The insights into using AI and NLP for query handling, particularly the LTSM model for intent recognition, are important for the technical development of our system. Furthermore, the research findings on the limitations of the chatbot's knowledge base and language capabilities will assist us in anticipating and addressing similar issues in our own chatbot.

In addition, user feedback on the interface's usability, particularly from non-IT students, will help us ensure that our chatbot is accessible and simple to use for a wider audience. The findings support the effectiveness of AI chatbots in reducing administrative burdens and providing timely responses, reinforcing our project's goals of streamlining student support and improving the overall registration process and student support.

3.2.4 iNOUN: Architecture and Usability of a Chatbot for Academic Enquiries

Summary of the Research Item:

This study[4] focuses on the creation and implementation of iNOUN, a chatbot designed to handle academic queries at the National Open University of Nigeria. As universities adapt to new communication technologies, a chatbot was introduced to help students with frequent inquiries about course registration, exam schedules, admission requirements, and study guidelines. The chatbot, which runs on Google's Dialogflow NLP engine, was designed to provide real-time assistance via text and voice interaction. To assess the chatbot's effectiveness, the researchers conducted a usability test using the System Usability Scale (SUS), which yielded a 77% satisfaction rate, exceeding the industry average of 67%. By answering repetitive questions and making information available throughout , the system significantly reduced the workload of university staff. This improved students' access to immediate responses and reduced the wait time often associated with traditional human-based systems. However, the chatbot was still limited by its predefined knowledge base, which prevented it from answering more complicated questions.

Critical Analysis of the Research Item:

The study provides important insights into the successful use of chatbots in higher education. One of the study's key strengths is its emphasis on user experience and satisfaction, which were assessed using the System Usability Scale (SUS). The SUS results, with a score of 77%, show that the chatbot provides a highly efficient and user-friendly interface, allowing students to obtain answers quickly and conveniently. Furthermore, the use of Dialogflow enabled the developers to incorporate advanced features such as natural language understanding and real-time interaction. However, the chatbot was not without

its flaws. Its overall scope of application is limited because it relied mainly on a predefined knowledge base, making it incapable of handling more complex or infrequently asked questions. Even though this problem is widespread with AI chatbots, it highlights the necessity of regular updates and the inclusion of backup plans, like referral to human support in the event that the chatbot is unable to offer sufficient responses. Additionally, even though the chatbot's voice and text features worked well, adding more personalized interaction handling capabilities could enhance the overall learning experience for students. In spite of these drawbacks, the system worked well for handling academic inquiries, demonstrating the chatbot's capacity to minimize administrative workloads.

Relationship with Proposed work:

Our project and this study both investigate how AI-driven chatbots might enhance university services, so they are highly relevant to each other. The iNOUN chatbot's use of Dialogflow, especially its natural language processing features, offers a solid foundation for the development and deployment of our chatbot system. Automating student interactions that speed up academic services like course registration and admissions is the common goal of both projects.

Furthermore, the difficulties this study found—like the chatbot's dependence on a static knowledge base—reflect the issues we may run into when developing ourselves. This study highlights how crucial it is to keep the knowledge base up to date and include fallback options for queries that the system is unable to respond to. In our project, this could mean including support channels for human assistance when needed, as well as enhancing the chatbot's capacity to handle increasingly complicated queries. Furthermore, the System Usability Scale employed in this study provides a precise framework for assessing our chatbot's efficacy after it is put into use, guaranteeing that it satisfies the user satisfaction benchmarks identified in this study.

3.2.5 Interactive Advising with Bots: Improving Academic Excellence in Educational Establishments

Summary of the Research item:

This research paper[5] investigates the development of AdvisorBot, a virtual advising system designed for Nigerian tertiary institutions. It draws attention to the increasing burden that growing student populations puts on academic advisors, which results in ineffective advising procedures. The paper offers a chatbot as a remedy for this, helping students with registration, course selection, and academic inquiries. AdvisorBot provides real-time, personalized advice by utilizing student data and academic performance. The goal of this automation is to make the advising process more efficient overall and lessen the workload for academic staff. To provide accurate and timely advice, the chatbot makes use

of decision support systems (DSS) and an agent-based framework. The system's architecture, its natural language processing capabilities, and its potential for broad adoption by other academic institutions are all described in this paper. In addition to offering regular advice, AdvisorBot supports academic success by assisting students in making viable decisions based on prior performance.

Critical Analysis of the Research Item:

The overwhelming demand for academic advising in institutions with limited staff resources has a workable solution because of this study. AdvisorBot enhances the advising process by freeing up human advisors to work on more complicated cases by automating repetitive queries. One of the system's best features is its capacity to offer customized advice in real time. Additionally, the DSS integration gives the advising process a clever layer that guarantees students receive accurate information catered to their individual needs.

On the downside, the chatbot is still dependent on a predefined knowledge base, which limits its ability to handle complicated student inquiries. Furthermore, human intervention is still necessary in complex student cases that call for a deeper comprehension of personal or academic circumstances. It would have been beneficial for the study to look into ways to make the chatbot more flexible and able to handle more complicated problems.

Relationship to the Proposed Work:

Our project, which also seeks to deploy a chatbot for accelerating university services like registration and student support, is directly related to this research. Our chatbot, like AdvisorBot, will help students and lessen administrative burden by utilizing real-time data. The incorporation of DSS into this study offers a useful framework that we can apply to enhance the decision-making capabilities of our system. This study also highlights the significance of keeping an updated knowledge base, which is essential to guaranteeing the continued effectiveness and dependability of our chatbot.

3.2.6 Chatbots in the Education System

Summary of the Research Item:

The growing use of AI-powered chatbots in educational settings is covered in this article[6]. The report demonstrates how chatbots can be used to automate repetitive tasks like responding to commonly asked questions (FAQs), reminding people of deadlines, and helping with administrative work. Educational institutions can reduce staff workloads and free up time for more important tasks by streamlining these repetitive processes. The paper also explores how chatbots can be used as virtual tutors, offering interactive learning experiences through quizzes, assessments, and personalized learning paths.

The growing importance of AI chatbots in education is highlighted by our increasing reliance on voice assistants like Siri, Alexa, and Google Assistant. In addition to offering round-the-clock assistance, these chatbots enhance the effectiveness of administrative and instructional procedures in educational establishments.

Critical Analysis of the Research Item:

The study offers a comprehensive examination of the advantages AI-powered chatbots have for the education industry. Chatbots greatly lessen administrative burdens by managing a variety of repetitive tasks, increasing institutional efficiency. Furthermore, chatbots improve the quality of the learning process by giving students immediate, individualized support—something that is frequently not feasible in conventional learning settings.

Even though chatbots have a lot of potential for use in education, the study also highlights some of the difficulties. For example, chatbot development is still in its early stages, and their ability for conversation may at times be restricted. When a poorly constructed chatbot is unable to comprehend or appropriately answer a user's question, it can irritate the user.

Relationship to the Proposed Research:

This paper directly addresses the goals of our project by offering insights into the use of chatbots for administrative and student registration tasks. The study emphasizes how critical well-thought-out chatbot interfaces are to guaranteeing seamless user interactions within our system. The paper's emphasis on round-the-clock support is also especially pertinent to our project because our chatbot is made to help students with registration-related questions whenever they need it. The study also highlights potential drawbacks, like limited conversational capabilities, which we must continually address by improving the knowledge base and design of the chatbot.

3.3 Literature Review Summary Table

The summary of all the literature reviews is given in the following table:

3.4 Conclusion

In conclusion, the reviewed literature emphasizes how much AI-driven chatbots can improve student services in educational settings. Research such as those conducted on iNOUN and AdvisorBot show that these chatbots reduce staff workloads and deliver timely, accurate information, which in turn boosts student satisfaction and engagement. Still, there are issues, mainly with regard to chatbots' inability to

Table 3.1: Related Work Summary Table

Application	Features	Relevance	Limitations
AI-Driven Chatbots for Student Services.	Uses registration data for guidance, 24/7 support.	Assists with timely information delivery; reduces administrative burdens.	Relies heavily on Dialogflow, limiting customization for complex queries
iNOUN	Handles academic queries, 24/7 availability, NLP for text/voice	Streamlines student inquiries, reduces staff workload, supports round-the-clock assistance	Struggles with complex conversations, restricted by predefined knowledge base and English-only input
Chatbots in Education	Automates FAQs, quizzes, and administrative tasks	Enhances learning experience, improves administrative efficiency, 24/7 support	Begining development stage, potential user frustration due to limited conversational ability.
Interactive Advising with Bots.	Offers real-time responses, handles academic queries.	Addresses increasing demand for academic advising; enhances student experience.	The chatbot's effectiveness dependens upon the quality of the knowledge base.
Conversational AI-Driven Chatbots.	Integration with university websites, tailorble knowledge base	Helps in enhancing online communication and support; scalable for multiple institutions.	Limited to predefined knowledge base; text-only input restricts interaction types.
AdvisorBot	Decision support, real-time academic advice, course registration .	Minimizes advisor workload, customizes advice based on student data.	Limited to predefined knowledge base, requires human intervention for complicatd queries.

handle complicated questions and exchanges that go beyond preset answers. To maximize their effectiveness, updates and continuous improvement are essential (Adam et al., 2021; Rahman et al., 2017).As we look to the future, these insights will be critical in directing the development of our chatbot system. Through the resolution of constraints like language barriers and adaptability, we can develop a user-friendly solution that fulfills the changing requirements of both educational institutions and students.

Chapter 4 Software Requirement Specifications

This chapter explores the key features of our System. it contains list of features,use-cases, functional an non-functional requirements, quality attributes and hardware and software requirements for our system. A comprehensive detail of each section is given below:

4.1 List of Features

The list of all important features of our system is given below:

1. Student Profile Integration

The system will acquire student's academic information such as Course History, Program Details, and Grades (if required) to validate the course eligibility requirements and for personalized recommendations.

2. Real-time Course Availability

The system will retrieve and provide real-time information regarding list of available courses, schedules, and availability of seats.

3. Registration Workflow

The system will present the list of core and elective courses. furthermore, it will identify any conflicts/-clashes and will alert the students and will also suggest the alternatives available.

4. Personalized Course Recommendation

The chatbot will provide AI-based course recommendations,based on grades history and areas of interests of the student, for elective courses.

5. Wait-list Management

If the available seats for the course are filled, the system will automatically add the students to a wait list. In addition, it will notify the student if a seat becomes available during the registration process.

6. Add/Drop Utility

The system will automatically monitor the student's registration process and notify the students about important deadlines(registration, course add/drop,payment deadlines). Moreover, it will allow students to Add/Drop courses before deadline.

7. Automated Timetable Generation

The system will be able to create a personalized timetable for students after registration. Also, it will

notify and confirm the students about timetable details via email.

8. Reporting and Analytics for Admin

The system will provide admin with a comprehension about course popularity and student preferences. Thus, it will help the administration with effective and enhanced semester planning for future.

9. Automated Alerts and Notifications

The system will keep the students updated about important deadlines and will send reminders about them. furthermore, it will also inform students about wait-listed courses and registration confirmation details

4.2 Functional Requirements

The system will be able to do the following tasks:

1. **Student Profile Integration:** To fetch and display the student's academic profile, the system will connect with the university's database.
2. **Course Availability and Requirements Check:** The system will verify course availability and prerequisites by fetching real-time information (number of seats, schedules, prerequisites).
3. **Personalized Course Recommendation:** The system will employ AI-based algorithms to suggest courses based on the student's academic performance.
4. **Registration Workflow:** The system will enroll students in their core courses, suggest electives, notify them about registration deadlines, and alert them if there are any conflicts in the schedule.
5. **Wait-list Management:** The system will automatically place students on a waitlist if the available seats in an elective are filled and will notify students if a seat becomes available during the registration period.
6. **Timetable Generation:** After registration, the system will generate a personalized timetable for the new semester and notify students to confirm their registration.
7. **Course Add/Drop:** The system will allow students to add or drop courses during registration and will automatically check prerequisite requirements before adding a course.
8. **Automated Notifications/Reminders:** The system will alert students about various academic deadlines (add/drop deadline, registration period, payment deadline, etc.).
9. **Analytics and Reporting:** The system will provide the admin with detailed reports by analyzing trending courses, student preferences, and more.

4.3 Quality Attributes

Quality attributes of our system are listed below:

- The chatbot will be available for the users 24/7 with negligible downtime.
- The system will be thoroughly accurate in understanding and responding to user queries ensuring user satisfaction.
- The system will be highly adaptable, learning from user queries and adapting to new categories of questions with time.
- The structure of the system will be extensible, permitting us to include new features over time.
- The system will respond to the user queries efficiently and quickly.
- The system will have a secure authentication process in order to protect confidential information.
- The system will be accurate and precise in providing all course information, suggestions, and wait-list updates.

4.4 Non-Functional Requirements

The non-functional requirements of the system are described as follow:

1. Scalability

The system should be able to assist various users simultaneously, particularly during important procedures such as, registration. System should be able to cater evolving number courses, students and data accessibility without any performance hindrance.

2. Reliability

The system should have an operation time of 99.9% , specifically during high-traffic session such as registration or other important deadlines.

3. Performance

The response time of the chatbot should be between 3-5 seconds to ensure effective and smooth communication.

4. Maintainability

The maintenance of system should be easy and should not require a downtime while update and improvement.

5. Usability

The web interface of the system should be user-friendly, should have easy to find navigation and help options.

6. Security

The system should follow communication and authentication protocols to ensure a worry-less experience for the users.

4.5 Assumptions

The key assumptions made in designing phase are as follow:

1. The academic profile and course databases of the University can deliver real-time information and are up-to-date
2. Students will use the system frequently to see the deadlines, timetable,registration status, and ask questions
3. The system will use already existing data to train the AI algorithms for accurate responses and conflict management
4. Both system and users are expected to have a reliable internet connection during high-traffic registration periods.

4.6 Use Cases

This section lists use cases used for our system.

4.6.1 Use Case 1: View Available Courses

Table 4.1: Use Case 1: View Available Courses

Name	View available courses
Actors	Student, admin
Summary	Enables the student and admin to see the list of available courses, course requirements, and prerequisites.
Pre-Conditions	Student's profile is accurately integrated and course catalog is up-to-date.
Post-Conditions	User is able to check course availability, seat count, and prerequisites.
Special Requirements	Real-time data is integrated from the university's database.
Basic Flow	**Actor Action** 1. The user navigates to the "List of Available Courses" section. 2. The student clicks on a course to check details about prerequisites, available seats, and schedule. **System Response** 1. The system shows the list of available courses for the upcoming semester. 2. The system verifies the email and password, establishes a session for the user, and redirects the user to the home page.
Alternative Flow	**S**tudent's prerequisites are not met. **System Response:** The system notifies the student about the missing prerequisites and suggests courses to meet prerequisite requirements. **4-A.** The course detail is unavailable. **System Response:** The system notifies the student about the temporary unavailability of details and suggests checking again in a while.

4.6.2 Use Case 2: Course Recommendation Based on Profile

Table 4.2: Use Case 2: Course Recommendation Based on Profile

Name	Course Recommendation based on profile.
Actors	Student, Admin (view only)
Summary	The system recommends elective courses based on the student's academic history.
Pre-Conditions	Student's profile is up-to-date.
Post-Conditions	The system displays personalized course recommendations.
Special Requirements	AI engine must examine the student's profile (academic progress) to make suggestions.
Basic Flow	**Actor Action** 1. The user navigates to "Recommended Courses" section. **System Response** 1. The system displays a list of suggested courses.
Alternative Flow	**S**tudent has already taken the displayed courses. **System Response:** The system notifies and recommends other electives.

4.6.3 Use Case 3: Access and View Student Profile

Table 4.3: Use Case 3: Access and View Student Profile

Name	Access and view student profile.
Actors	Student, admin
Summary	Allows the users to access and view the academic profile (academic progress and history).
Pre-Conditions	The student is registered with the system and the profile is up-to-date.
Post-Conditions	User is able to view the profile.
Special Requirements	System must set the level of permissions for students and admin (student can access only their own profile, admin can access all students' profiles).
Basic Flow	**Actor Action** 1. The user clicks the "View Profile" option. **System Response** 1. The system fetches and displays the student profile.
Alternative Flow	**S**tudent tries to access another profile. **System Response:** The system displays "Access not granted" and denies permission to access another profile.

4.6.4 Use Case 4: Wait-list Management

Table 4.4: Use Case 4: Wait-list Management

Name	Wait-list management
Actors	System, admin (if necessary)
Summary	System will automatically enter the student into the waitlist when the available seats are filled and enroll students accordingly.
Pre-Conditions	The available seats of the course are filled.
Post-Conditions	Students are notified and moved from the waitlist to the course based on available seats and system rules.
Special Requirements	System must incorporate automatic notification and waitlist management functions.
Basic Flow	**Actor Action** 1. Course enrollment limit is reached. **System Response** 1. The system notifies and places the student on the waitlist.
Alternative Flow	**N**o seats are available within the registration period. **System Response:** Students are notified about the unavailability of seats.

4.6.5 Use Case 5: Manage Add/Drop Requests

Table 4.5: Use Case 5: Manage Add/Drop Requests

Name	Manage add/drop requests
Actors	Student, admin (if necessary)
Summary	Students can add or drop courses within the registration period, and admins can supervise and handle the requests.
Pre-Conditions	Add/Drop period is on, and students meet the course's prerequisite requirements.
Post-Conditions	Courses are added or dropped according to the student's action from the schedule.
Special Requirements	System must verify prerequisite requirements and check seat availability before adding a course.
Basic Flow	**Actor Action** 1. Student heads to the "Add/Drop" section and chooses a course to add or drop. **System Response** 1. The system checks seat availability and prerequisites and confirms or denies the action accordingly.
Alternative Flow	**C**ourse seats are filled or prerequisites are not met. **System Response:** The system notifies the student and suggests alternatives.

4.6.6 Use Case 6: Generate Reports

Table 4.6: Use Case 6: Generate Reports

Name	Generate reports
Actors	Admin
Summary	Admins can create reports about registration statistics, trending electives, and student preferences.
Pre-Conditions	The system contains registration data.
Post-Conditions	A detailed report is created for the admin to review.
Special Requirements	The system must have the ability to process a large amount of data and create written reports.
Basic Flow	**Actor Action** 1. The admin logs into the system and navigates to the ”Reports” section. 2. The admin selects the type of report to generate. **System Response** 1. The system processes the data and generates the requested report.
Alternative Flow	**T**he system does not have enough data to generate the report. **System Response:** The system alerts the admin and asks for more information.

4.6.7 Use Case 7: Course Management

Table 4.7: Use Case 7: Course Management

Name	Course management
Actors	Admin
Summary	Admins can create, update, or remove courses in the system.
Pre-Conditions	Admin must be logged into the system.
Post-Conditions	Course status is changed in the system and visible to students and staff.
Special Requirements	The system must verify if all the course details are provided.
Basic Flow	**Actor Action** 1. Admin logs into the system and heads to the ”Course Management” section. 2. Admin selects to create, update, or remove a course. **System Response** 1. The system confirms the changes made by the admin and saves them.
Alternative Flow	**C**ourse details are invalid or incomplete. **System Response:** The system prompts the admin to add valid and complete details.

4.6.8 Use Case 8: Deadline Configuration

Table 4.8: Use Case 8: Deadline Configuration

Name	Deadline configuration
Actors	Admin
Summary	Admin can adjust deadlines for various important procedures like registration and withdrawal.
Pre-Conditions	Admin must have access to deadline configuration functionality.
Post-Conditions	Deadlines are adjusted and students are notified.
Special Requirements	The system must notify the students after deadlines are set.
Basic Flow	**Actor Action** 1. Admin logs into the system and heads to the ”Deadline Configuration” section. 2. Admin sets or updates deadlines for academic procedures. **System Response** 1. The system saves the changes and informs the students via notifications.
Alternative Flow	**C**onflict arises between existing schedules or policies. **System Response:** The system alerts the admin and prompts to resolve the conflict.

4.6.9 Use Case 9: Teacher Management

Table 4.9: Use Case 9: Teacher Management

Name	Teacher management
Actors	Admin
Summary	Admins can create, update, or remove teachers in the system.
Pre-Conditions	Admin must be logged into the system.
Post-Conditions	Teacher is added/removed/updated in the system and the result is visible to students and staff.
Special Requirements	The system must verify if all the teacher details are provided.
Basic Flow	**Actor Action** 1. Admin logs into the system and heads to the ”Teacher Management” section. 2. Admin selects to create, update, or remove a teacher. **System Response** 1. The system confirms the changes made by the admin and saves them.
Alternative Flow	**T**eacher details are invalid or incomplete. **System Response:** The system prompts the admin to add valid and complete details.

4.6.10 Use Case 10: Student Management

Table 4.10: Use Case 10: Student Management

Name	Student management
Actors	Admin
Summary	Admins can create, update, or remove Students in the system.
Pre-Conditions	Admin must be logged into the system.
Post-Conditions	Student is added/removed/updated in the system and the result is visible to students and staff.
Special Requirements	The system must verify if all the Student details are provided.
Basic Flow	**Actor Action** 1. Admin logs into the system and heads to the ”Student Management” section. 2. Admin selects to create, update, or remove a Student. **System Response** 1. The system confirms the changes made by the admin and saves them.
Alternative Flow	**S**tudent details are invalid or incomplete. **System Response:** The system prompts the admin to add valid and complete details.

4.6.11 Use Case 11: Grader Management

Table 4.11: Use Case 10: Grader Management

Name	Grader management
Actors	Teacher
Summary	Teacher can upload Students' grades in the system.
Pre-Conditions	Teacher must be logged into the system.
Post-Conditions	Students' grades are added/updated in the system and the result is visible to students.
Special Requirements	The system must verify if all the marks marks are in the correct range.
Basic Flow	**Actor Action** 1. Professor logs into the system and heads to the "Marks Management" section. 2. Professor selects to create, update a Student's record. **System Response** 1. The system confirms the changes made by the Professor and saves them.
Alternative Flow	**M**arks details are invalid or incomplete. **System Response:** The system prompts the Professor to input valid and complete marks.

4.6.12 Use Case 12: View Grades

Table 4.12: Use Case 12: View Grades

Name	View Grades
Actors	Teacher, Student
Summary	Student can view the uploaded grades in the system.
Pre-Conditions	Student must be logged into the system.
Post-Conditions	Grades are visible to the students grouped by categories.
Special Requirements	The system must verify if all the marks are uploaded.
Basic Flow	**Actor Action** 1. Student logs into the system and heads to the ”Marks” section. 2. Student selects a record to view. **System Response** 1. System displays the selected record.
Alternative Flow	**M**arks details are invalid or incomplete. **System Response:** The system prompts the Student to wait or contact teacher.

4.6.13 Use Case 13: Login

Table 4.13: Use Case 13: Login

Name	Login
Actors	Teacher, Student, Admin
Summary	Students,Teachers, and Admins can login into the system.
Pre-Conditions	The database must be live with credentials stored
Post-Conditions	Students, Teachers and Admins are logged into their particular profile.
Special Requirements	The system must verify if the credentials are correct.
Basic Flow	**Actor Action** 1. Student/Admin/Professor enter their login credentials. 2. Press enter **System Response** 1. System allows them access by validating their login
Alternative Flow	**C**redentials are invalid or incomplete. **System Response:** The system prompts the user to input valid credentials or email HQ.

4.6.14 Use Case 14: Chatbot invalid Queries

Table 4.14: Use Case 14: Chatbot invalid Queries

Name	Chatbot invalid Queries
Actors	Teacher, Student, Admin
Summary	Student,Teacher, or Admin prompt the chatbot with invalid query and it has a fallback message
Pre-Conditions	Students,Teachers, or Admins prompt the chatbot with invalid query.
Post-Conditions	Chatbot responds with a fallback message.
Special Requirements	The system must verify if the chatbot API is up and running
Basic Flow	**Actor Action** 1. Student/Admin/Professor open chatbot. 2. Student/Admin/Professor enter their unknown query **System Response** 1. Chatbot responds with a proper message
Alternative Flow	**T**he Query is Valid **System Response:** Chatbot responds to the query

4.7 Hardware and Software Requirements

List the hardware and software requirements that will be required to develop and deploy the project are given below:

4.7.1 Hardware Requirements

The details of Hardware requirements for our system are as follow:

1. **Development Phase**:

- **Processor:** Minimum Intel Core i5 or Core i7 for optimized performance.
- **RAM:** Minimum 8GB, 16GB for handling huge datasets.
- **GPU:** A mid range GPU will be sufficient.
- **Network Requirements:** High-speed network connection for faster API calls,cloud services.

2. **Deployment phase:**

- **Processor:** Multi-core CPUs.

- **RAM:** 16-32GB based on number of users and Chatbot's complexity.
- **Storage**: 1TB HDD or SDD for course content and data logs and backups.
- **Network Requirements:** High-speed network connection for smooth interaction between chatbot and users.

4.7.2 Software Requirements

Following are the software requirements for our Chatbot: **1.Operating System:**

- Windows 10/11,
- macOS

2. Programming Languages:

- Python (for Chatbot), NLP libraries(like NLTK, Transformers)
- HTML/CSS to style User-Interface.
- ReactJS and Javascript for front-end development.

3.Database:

- MongoDB or MySQL to handle student data(profiles,course content, registration records)

4.Framework and Libraries:

- Dialogflow to build and train the chatbot.
- Node.js to manage server side operations.
- TensorFlow/PyTorch t manage AI-based functionalities.
- Flask/django for API handling and back-end development

5.Integrated Development Environment:

- Visual Studio Code and PyCharm for coding and debugging.

6.Version Control

- Git with GitHub to handle version management and collaboration.

7.APIs and External services:

- Google cloud NLP for advance NLP understanding.
- SendGrid or Twilio to manage notifications and deadline reminders.

8.Cloud Hosting:

- AWS (Amazon Web Services) or Microsoft Azure to deploy the project.

4.8 Graphical User Interface

The GUI sample of the system is given below:

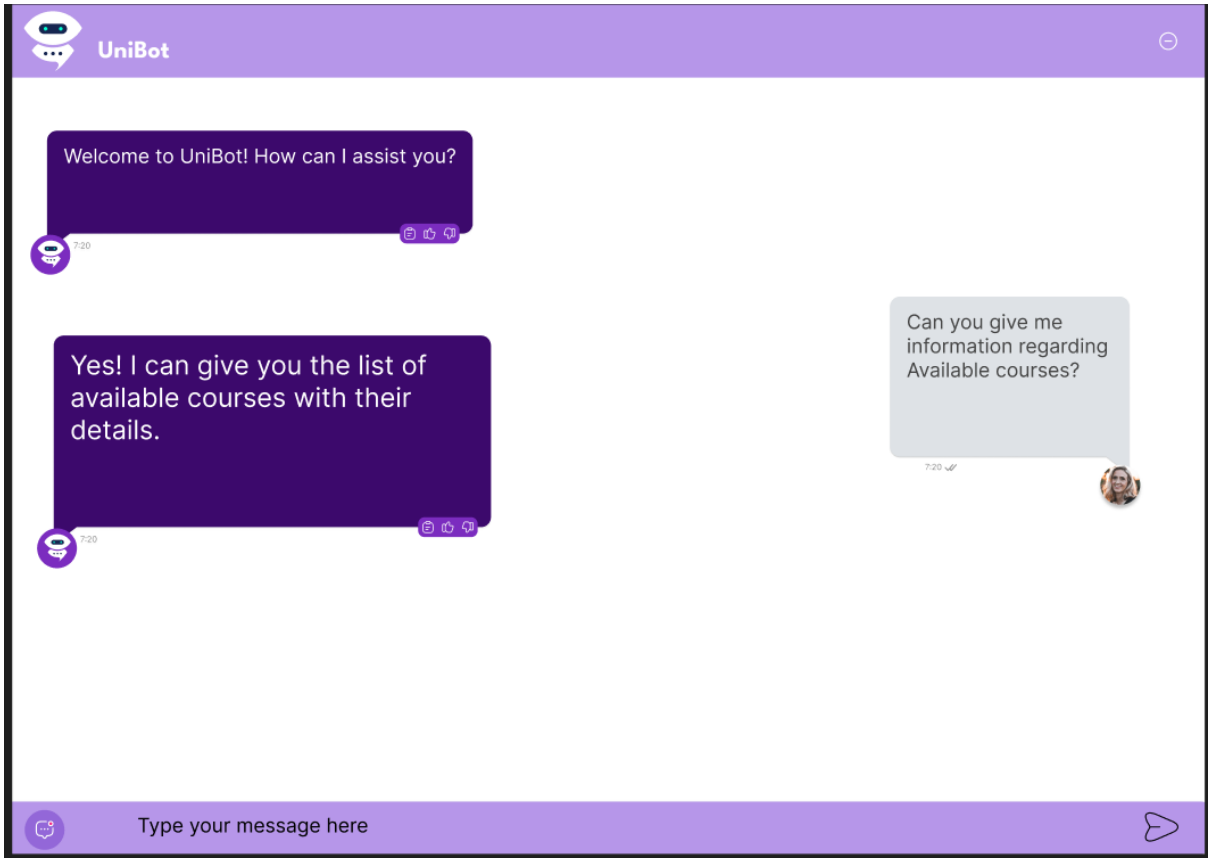

Figure 4.1: Illustration of Conversation window

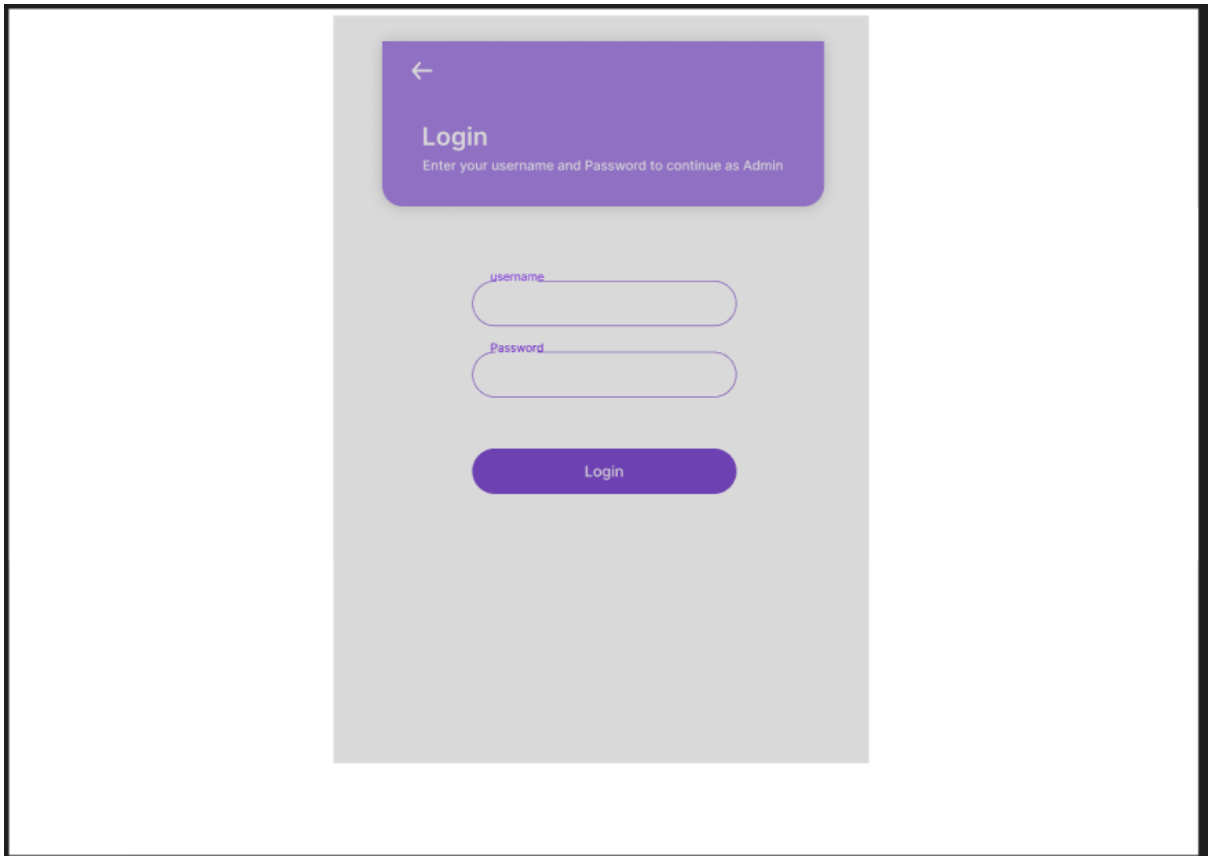

Figure 4.2: Illustration of login page for Admin

4.8.1 ER Diagram

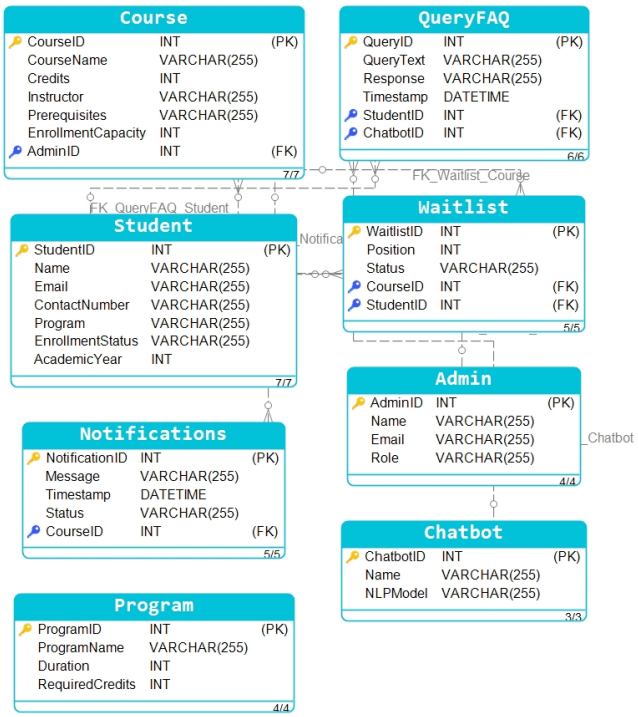

Figure 4.3: ChatBot-ER diagram

4.8.2 Data Dictionary

Entity	Attribute	Datatype	Relation To	Nullable	Description
Student	StudentID	INT	N/A	No	Unique identifier for each student.
	Name	VARCHAR(100)	N/A	No	Full name of the student.
	Email	VARCHAR(100)	N/A	No	Student's email address for communication.
	ContactNumber	VARCHAR(15)	N/A	Yes	Student's phone number.
	Program	VARCHAR(100)	ProgramID	No	The academic program the student is enrolled in.

Continued on next page...

Entity	**Attribute**	**Datatype**	**Relation To**	**Nullable**	**Description**
	EnrollmentStatus	VARCHAR(50)	N/A	No	Current status of enrollment (active, inactive, graduated).
	AcademicYear	INT	N/A	No	The academic year the student is currently in.
Admin	AdminID	INT	N/A	No	Unique identifier for each admin.
	Name	VARCHAR(100)	N/A	No	Full name of the admin user.
	Email	VARCHAR(100)	N/A	No	Admin's email address for communication.
	Role	VARCHAR(50)	N/A	No	Specific role or title of the admin within the university.
Course	CourseID	INT	N/A	No	Unique identifier for each course.
	CourseName	VARCHAR(100)	N/A	No	Title of the course.
	Credits	INT	N/A	No	Number of credits awarded for completing the course.
	Instructor	VARCHAR(100)	N/A	Yes	Name or ID of the instructor teaching the course.
	Prerequisites	VARCHAR(255)	N/A	Yes	List of courses or requirements needed before enrolling.
	Seats	INT	N/A	No	Maximum number of students allowed to enroll in the course.
Chatbot	ChatbotID	INT	N/A	No	Unique identifier for the chatbot instance.
	Name	VARCHAR(100)	N/A	No	Name of the chatbot.
	NLPModel	VARCHAR(100)	N/A	No	Description of the NLP model used for processing queries.

Continued on next page...

Entity	**Attribute**	**Datatype**	**Relation To**	**Nullable**	**Description**
Program	ProgramID	INT	N/A	No	Unique identifier for each academic program.
	ProgramName	VARCHAR(100)	N/A	No	Name of the academic program.
	Duration	INT	N/A	No	Typical duration (in years) required to complete the program.
	RequiredCredits	INT	N/A	No	Total credits needed to graduate from the program.
Query/FAQ	QueryID	INT	N/A	No	Unique identifier for each query or FAQ entry.
	QueryText	VARCHAR(255)	N/A	No	Text of the question or inquiry.
	Response	TEXT	N/A	Yes	Response provided by the chatbot for the query.
	Timestamp	DATETIME	N/A	No	Date and time when the query was made.
Waitlist	WaitlistID	INT	N/A	No	Unique identifier for each waitlist entry.
	Position	INT	N/A	No	The position of the student in the waitlist.
	Status	VARCHAR(50)	N/A	No	Current status of the waitlisted student (active, removed).
	CourseID (FK)	INT	CourseID	No	Reference to the CourseID for which the student is waitlisted.
	StudentID (FK)	INT	StudentID	No	Reference to the StudentID for the student on the waitlist.
Notifications	NotificationID	INT	N/A	No	Unique identifier for each notification.
	Message	TEXT	N/A	No	Content of the notification sent to users.

Continued on next page...

Entity	Attribute	Datatype	Relation To	Nullable	Description
	Timestamp	DATETIME	N/A	No	Date and time when the notification was sent.
	Status	VARCHAR(50)	N/A	No	Read/unread status of the notification.

4.9 Risk Analysis

This section entails a complete and thorough risk analysis regarding ou system and based on the three categories of risks that are explained below:

4.9.1 1.Technical Risks

Integration Issues: Integrating the chatbot with current university systems, like student information databases, carries a sizable risk. Incompatibilities between the databases and APIs may cause delays and inconsistent data. Therefore, to guarantee a smooth integration, careful planning and testing are essential. **Performance and Scalability:** The chatbot may encounter poor response times or even crashes if it is not appropriately optimized when the user base grows. Designing a system that can scale effectively and handle peak loads without compromising performance is crucial to preventing this. **Security Vulnerabilities:** Since the chatbot will handle private student data, cyberattacks could potentially target it. Therefore, it's essential to put strong security measures in place—like data encryption and secure coding techniques—to safeguard user information and uphold confidence. **Limitations of Natural Language Processing:** The chatbot's ability to process natural language is a major factor in determining how effective it is. Insufficient training or incomplete data may cause the chatbot to be unable to comprehend user inquiries. The user experience may be greatly impacted by this limitation, requiring regular training and updates.

4.9.2 2.Risks to Business

- **Budget Restrictions:** Creating and keeping up a chatbot can be costly. The project's scope or quality may be compromised if costs turn out to be higher than projected. For this reason, careful resource allocation and budget planning are crucial to the project's success.
- **Engagement of Stakeholders:** It is imperative to secure backing from significant stakeholders,

including IT departments and university administration. The project's progress and the chatbot's potential may be hindered if these stakeholders oppose it or don't participate in an effective manner.

- **Market Competition:** If a chatbot fails to live up to user expectations or is lacking in cutting-edge features, it could cause a facility that uses AI-driven solutions to lag behind others. Therefore, maintaining competitiveness requires keeping up with user demands and industry trends.

4.9.3 3.Operational Risk

- **Difficulties with User Adoption:** The chatbot's ability to succeed depends on how well students and employees accept it. Adoption rates could decline if users find it difficult to use or if they favor more conventional approaches. It's critical to make user training investments and make sure the chatbot is intuitive in order to reduce this risk.
- **Maintenance and update:**Updating and maintaining the chatbot on a regular basis is essential to keeping it functional and relevant. If adequate funds aren't set aside for continuous maintenance, the chatbot's usefulness could quickly dwindle.
- **Knowledge Base Management**: The success of the chatbot depends on creating an extensive knowledge base. User frustration will result from the chatbot's inability to respond accurately if the knowledge base is not kept up to date.

4.10 Conclusion

Our chatbot project's Software Requirements Specification (SRS), Chapter 4, provides a clear and comprehensive development plan. The chapter initiates with a long list of features and functional requirements that highlight the distinctive needs of the academic staff, administrative staff, and students. Along with non-functional requirements and functional requirements that indicate the system's overall goals, it also specifies critical quality attributes like security, performance, and usability. The chapter also covers graphical user interface design, with the goal of producing a simple and easy-to-use interface. A data dictionary that explains the different data elements and their relationships is included, along with an entity-relationship diagram that shows how the database will be set up.

Moreover, an exhaustive risk analysis determines possible obstacles, allowing us to establish innovative strategies in place to lessen these risks and strengthen the project's adaptability.Essentially, Chapter 4 provides an essential basis upon which the chatbot system can be developed. We are ready to proceed with implementation by methodically attending to the features, requirements, design considerations,

and risk management techniques. By taking this careful approach, we can be sure that our chatbot will successfully promote student engagement and enable more seamless interactions within the university, which will ultimately lead to the achievement of our project goals.

Chapter 5 High-Level and Low-Level Design

Provide the high- and low-level design of your system in this chapter. Select the design that is appropriate for your project.

5.1 System Overview

This website of ours will provide automated registration for the students. Furthermore the integrated chat bot will help students to Add/drop courses,it will help students to make easy decisions before choosing a timetable as it will list down the possible options automatically. Moreover our system will be trained to answer all kind of queries online hence a complete relief from manual registration.

5.2 Design Considerations

The Design considerations are as follows:

5.2.1 Assumptions and Dependencies

The following are the assumptions we are making regarding to this project:

- User has a good understanding of basic English
- User will have an internet connection to access the website
- Assumption that are server can handle multiple users at one single instance
- Back end is assumed to handle multiple users at on single instance

5.2.2 General Constraints

5.2.2.1 Hardware or Software Environment

- Assumption that the OS being used is at least Windows 7 supported
- Requirement of a stable internet connection is fulfilled
- Device used has at least 4GB RAM

5.2.2.2 End-User Environment

- User can navigate through the browser used to access the website (e.g., Firefox, Safari, etc.)

- Lack of basic internet knowledge in the user

5.2.2.3 Interoperability Requirements

- The information in the database is fetched correctly and is up to date

5.2.2.4 Interface/Protocol Requirements

- A user-friendly and intuitive interface for easy navigation of the system

5.2.2.5 Security Requirements

- Separation of admin/user accounts
- Strict measures to secure data

5.2.2.6 Performance Requirements

- Target minimum uptime is 95 percent
- Support at least 500 users during peak times

5.2.3 Goals and Guidelines

To accomplish the goal this website will follow these guidelines:

- **Simplicity**: This will ensure the interface is easy to use and is admired by the students
- **Response time**: Imposition on an acceptable performance specially in peak registration times.
- **Flexibility**: Easy to add future enhancements

5.2.4 Development Methods

Agile method will be used for development as this system will involve continuous evolving using student's and admins feedback. Furthermore with agile, modular method is also followed because this system will consist of different modules such as chat bot, registration etc.

5.3 System Architecture

Our project, university registration system with a chatbot, has a multi-layered highly functioning architecture to handle the complexities of user interactions, data management, and AI processing. Here's a summary of the system architecture:

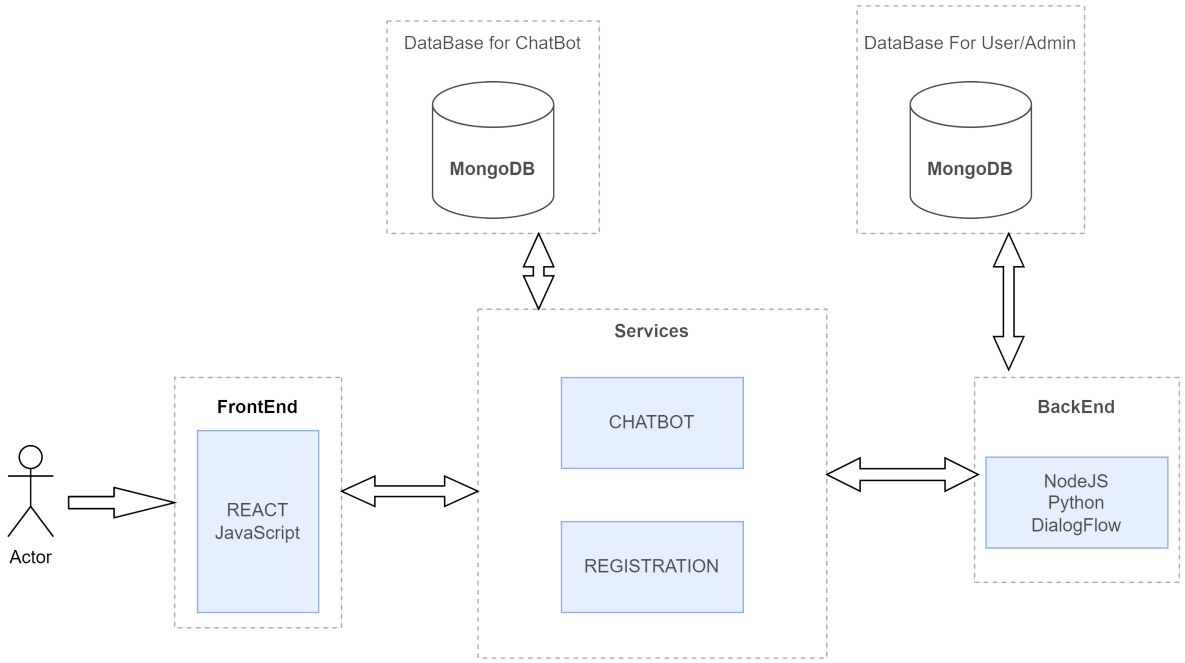

Figure 5.1: System Architecture Layout

5.3.1 Subsystem Architecture

Creating a university registration system with a chatbot involves several subsystems that work together to provide a seamless experience. Here's a breakdown of the subsystems and their architecture:

5.3.1.1 1. User Interface (UI)

- **Website**: For students to interact with the system and chatbot.
- **Chatbot Interface**: A proper interface for users to interact with via messaging.
- **Dashboard for Admins**: To manage student registrations, view analytics, and handle inquiries.

5.3.1.2 2. Chatbot Engine

- **Natural Language Processing (NLP)**: To understand and process user queries. Usage of DialogFlow, Python, and BERT(considered).
- **Intent Recognition**: Classifying user inputs to determine what action to take.
- **Response Generation**: Providing appropriate responses based on user intent.

5.3.1.3 3. Backend Services

- **API Layer**: RESTful or other APIs to handle communication between the frontend and backend.

- **Authentication Service**: To handle user authentication and authorization (e.g JWT).

5.3.1.4 4. Database Management

- **Student Database**: Stores student profiles, course information, registration details, and transcripts.
- **Course Management Database**: Contains information about courses, schedules, faculty, and prerequisites.
- **Chatbot Database**: Contains additional data that was trained to answer a wide set of queries.

5.3.1.5 6. Integration Layer

- **Third-party Services**: Integrate with external services for functionalities like emails.
- **Notification System**: Sends notifications (emails/SMS) to students regarding registration updates, deadlines, etc.

5.3.1.6 7. Admin Management

- **Admin Portal**: Interface for university staff to manage courses, view student registrations, and analyze data.
- **Reporting Module**: Generates reports on registration statistics and student performance.

5.4 Architectural Strategies

Automated Registration system follows the following architectural strategies to ensure its sublime and easy to use:

5.4.1 Programming Language

This project will consist a front-end Developed in JavaScript and react to ensure a responsive user interface. Furthermore back-end will consist of Node.js for its high performance and AI Frameworks like python, Dialog flow And Bert (NLP) will be used to ensure efficiency. In addition the database will work on MongoDB for flexible management.

5.4.2 User-interface Paradigm

We will be incorporating a very familiar user interface which will assure its understood my majority of the students. Furthermore the switch from chat bot to other UI modules will be seamless to ensure

consistent performance.

5.4.3 Concurrency and Synchronization

To achieve maximum concurrency we will build the back-end using a multi threaded approach.Moreover synchronization will be achieved by continuously updating critical data like registration deadlines,seat count etc.

5.4.4 Memory management

A major use of memory cache will surround this project as many user queries are repeated which can be answered directly using the cache data.

5.5 Domain Model/Class Diagram

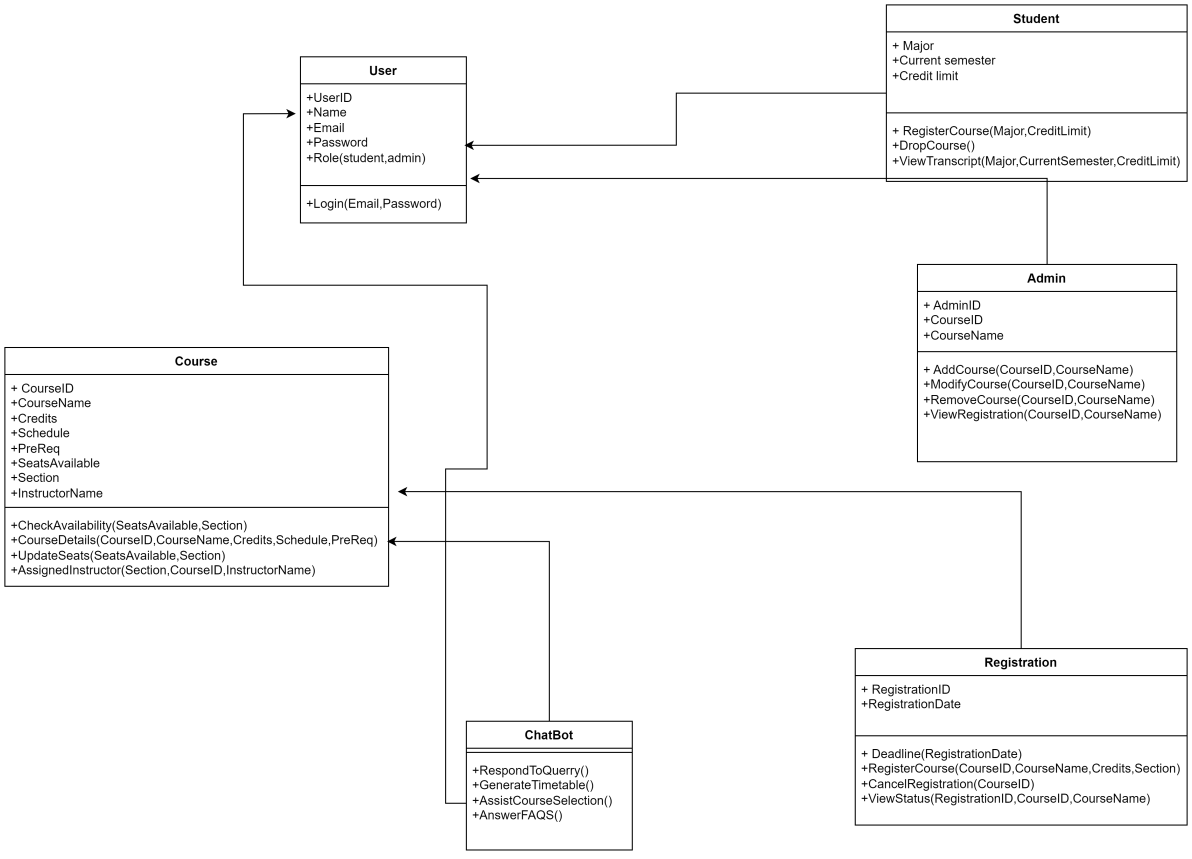

Figure 5.2: Class diagram for University registration system with AI chatbot

5.6 Policies and Tactics

5.6.1 Plans Of Testing

High level of testing will be done to ensure every module is working as it should. In addition to this tools like selenium will be opted to check correct chatbot interaction with the user.

5.6.2 Hierarchical Organization

We will follow a conventional structure to organize our code.This is achieved by making the source code into modules. This separation will greatly help in the following policy as well that is maintenance of the system.

5.6.3 Maintenance

We are using git to integrate our modules and managing our team responsibilities. This will also ensure correct maintainability and decent performance.

5.6.4 End User Interfaces

Major focus on chat bot interface so we can assure that it offers a clear and user friendly interface. This will be achieved by keeping our focus on key things that are familiarity,accessibility and readability.

Chapter 6 Implementation and Test Cases

The university chatbot system's implementation involves designing an AI-powered platform that manages student inquiries, automates the registration process, and offers personalized course recommendations. The main backend functions and algorithms employed to create a reliable and efficient user experience for administrators and students alike are described in this section.

6.1 Implementation

Our Chatbot system is designed using advance Narural Languagae Processing (NLP), to improve the student support and automate the regestration process. The carefully chosen set of technologies and framework will allow us to build a scalable and efficient system. The crucial components of the technology stack include:

- **Natural Language Processing**: Google Dialogflow powers the chatbot's ability to understand user intent, manage entities, and handle conversational flows with accuracy and ease.
- **Backend Framework**: The backend, developed using Node.js, manages database integration and processes API requests to ensure real-time data access and updates.
- **Database**: MongoDB is used to store and manage structured data, such as student profiles, course information, and registration details.
- **Frontend Development**: React.js is utilized to create a responsive and user-friendly interface, allowing students to access the chatbot on both web and mobile platforms.
- **APIs**: Custom RESTful APIs facilitate secure communication between the chatbot, backend, and database, ensuring seamless and reliable data exchange.

6.1.1 Implementation of Data Collection and Preprocessing

The first step in building the chatbot includes collecting and preparing relevant datasets. it involves:

- **Course Data** extracting and organizing data from official university websites regarding programs, credit hours, and courses. The data that has been scraped has been processed, verified, and formatted to ensure consistency.
- **Student Profiles** generating a fake dataset with information like names, email addresses, GPAs, academic years, registered courses, and student IDs. In order to replicate real-world situations, the data is created programmatically.

- **Data Cleaning and Standardization** It involves procedures including eliminating duplicates, fixing discrepancies, and making sure that data entries follow specified forms. This guarantees the chatbot will receive high-quality input.

6.1.2 Back-end Development

The main hub for controlling communication between the chatbot and the database is the backend. Here, the primary functions are:

- **Database Integration** Course information, student profiles, and other resources are stored using MongoDB. The schema of the database is made to facilitate fast updates and queries.
- **API Development** The database and chatbot are connected using RESTful APIs. Tasks like retrieving available courses, verifying requirements, and handling course registrations are made more convenient by these APIs.
- **Business Logic Implementation** The backend programming includes algorithms for checking prerequisites, selecting courses based on GPA, and maintaining seat availability.
- **Student Information Storage**: Securely stores student records, enrollment details, and profile data.
- **Faculty Information Storage:** Securely stores information and data regarding faculty.
- **Course Database**: Maintains updated course catalogs, prerequisites, schedules, and registration details.
- **User Query Logs**: Stores past chatbot interactions to enhance future responses using AI-driven learning.

6.1.3 Frontend Development

The user interface was developed using ReactJS to ensure a seamless experience. Key features include:

- **Chatbot UI**: A visually appealing and interactive chatbot interface for student queries.
- **Dashboard for Students**: Provides access to registered courses, chatbot history, and notifications.
- **Admin Panel**: Allows administrators to update course data, monitor chatbot performance, and view analytics.
- **Responsiveness**: Ensures compatibility across different devices, including desktops, tablets, and mobile devices.

6.1.4 Chatbot Development Using BotPress

The conversational interface of the chatbot is built using BotPress, allowing natural language interaction and understanding.

- **Intent Management** Several intents, such as ”Check Available Courses”, ”Register for a Course”, and ”Check Prerequisites”, are set up to analyze user queries and respond appropriately.
- **Entity Configuration** Key entities like course_name, student_name, and academic_year are defined to allow the chatbot to interpret structured input dynamically.
- **Webhook Integration** Dialogflow communicates with the backend APIs via a webhook, ensuring that replies are suited to real-time database data.
- **Dataset Preparation**: Collected and labeled student inquiries to train the model.
- **Continuous Learning**: Integrated feedback loops to refine chatbot accuracy over time.

6.1.5 User Interface Design

To provide a user-friendly experience, the chatbot is integrated within a visually appealing and functional interface.

- **Web Portal** The chatbot will be integrated into the university’s online portal using React.js to ensure responsiveness and ease of use. Students can use the chatbot to access academic information and services.
- **Mobile Support** The UI is tailored for mobile devices, allowing students to utilize the chatbot with ease on smartphones and tablets.

6.1.6 Integration and Scalability

As the chatbot becomes operational, ongoing upgrades are prioritized:

- **Feedback Mechanisms** User feedback is gathered to improve intent identification, response accuracy, and uncover new use cases.
- **Future Scalability** The system is designed to support new features, such as automated timetable production or enhanced analytics, without requiring major architectural changes.

6.1.7 Conclusion

The goal of the chatbot’s implementation is to offer a full framework for streamlining registration procedures by fusing strong backend capabilities, an easy-to-use user interface, and seamless connection

with university systems. The chatbot provides students with a personalized experience, scalability, and efficiency by utilizing contemporary tools such as Dialogflow and MongoDB. This guarantees that the project not only achieves its short-term objectives but also lays the groundwork for upcoming improvements to student services.

6.2 Test case Design and description

All the test cases performed to ensure proper functionality of our system during the testing phase are elaborated below.

Table 6.1: Student Panel - Available Courses

Student Panel - Available Courses			
UC-01			
Test Case ID:	*1*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.1*
Revision History:	*None*		
Objective:	*Verify that students can view a list of available courses.*		
Module:	*Student Panel course registration*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Courses are already added by the admin.*		
Pre-Requisite:	*Student must be logged in.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Navigate to "Courses" section from the dashboard.*	*List of available courses is displayed with course name, code, credits, and instructor details.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.2: AI Course Recommendation System

AI Course Recommendation System			
UC-02			
Test Case ID:	*2*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.2*
Revision History:	*None*		
Objective:	*Ensure that the system recommends courses based on the student's academic history/profile.*		
Module:	*Student Panel Chatbot*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Recommendation logic is implemented.*		
Pre-Requisite:	*Student profile has completed data*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Ask chatbot for course suggestions.*	*A list of personalized course recommendations is displayed.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.3: Student Profile Management

Student Profile Management			
UC-03			
Test Case ID:	*3*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.3*
Revision History:	*None*		
Objective:	*Verify that students can view their personal and academic information.*		
Module:	*Student Panel Profile*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Profile data is stored correctly in the database.*		
Pre-Requisite:	*Student must be logged in.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Click on "Profile" in the sidebar menu.*	*Student's full profile is displayed.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.4: Course Waitlist System

Course Waitlist System			
UC-04			
Test Case ID:	*4*	**QA Test Engineer:**	*Menal Naeem*
Test case Version:	*1*	**Reviewed By:**	*Raheem Jalil*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.4*
Revision History:	*None*		
Objective:	*Ensure that students can join the waitlist if a course is full and get updated automatically when slots free up.*		
Module:	*Student Panel Course Registration*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Waitlist functionality is integrated with course registration.*		
Pre-Requisite:	*Course must be full (no available slots).*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Attempt to register for a full course and join the waitlist.*	*Student is added to the waitlist and notified of their position.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.5: Add/Drop Request Management

Add/Drop Request Management			
UC-05			
Test Case ID:	*5*	**QA Test Engineer:**	*Menal Naeem*
Test case Version:	*1*	**Reviewed By:**	*Raheem Jalil*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.5*
Revision History:	*None*		
Objective:	*Validate that students can submit course add/drop requests and view their status.*		
Module:	*Student Panel Course Registration*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Add/Drop window is open.*		
Pre-Requisite:	*Student must be enrolled in at least one course.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Submit an add/drop request for a course.*	*Request is saved and displayed with 'Approved/Rejected' status.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.6: Admin Course Management

Admin Course Management			
UC-07			
Test Case ID:	*6*	**QA Test Engineer:**	*Raheem Jalil*
Test case Version:	*1*	**Reviewed By:**	*Ahmad Javaid*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.7*
Revision History:	*None*		
Objective:	*Confirm that admins can create, update, and delete courses.*		
Module:	*Admin Panel Course Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Database is connected and permissions are correctly set.*		
Pre-Requisite:	*Admin logged in.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Add a new course with all required fields and save.*	*New course appears in the course catalog.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.7: Deadline Configuration

Deadline Configuration			
UC-08			
Test Case ID:	*7*	**QA Test Engineer:**	*Raheem Jalil*
Test case Version:	*1*	**Reviewed By:**	*Ahmad Javaid*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.8*
Revision History:	*None*		
Objective:	*Verify that admins can set and edit deadlines (e.g., registration deadline, add/drop deadline).*		
Module:	*Admin Panel Deadline Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Deadline fields are editable through admin interface.*		
Pre-Requisite:	*Admin logged in.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Update the deadline fields and save.*	*Deadlines are updated across the system and displayed on the student portal.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.8: Report Generation

Report Generation			
UC-06			
Test Case ID:	*8*	**QA Test Engineer:**	*Raheem Jalil*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.6*
Revision History:	*None*		
Objective:	*Verify that admins can generate student performance, registration, and enrollment reports.*		
Module:	*Admin Panel Deadline Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Report generation functionality is active.*		
Pre-Requisite:	*Admin logged in.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Click "Generate Report" in the Reports section.*	*Report is generated in PDF/Excel and available for download.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.9: Admin Add Professor

Admin Add Professor			
UC-09			
Test Case ID:	*9*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Raheem Jalil*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.9*
Revision History:	*None*		
Objective:	*Ensure the admin can add a new professor to the system.*		
Module:	*Admin Panel Teacher Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Admin is already logged in.*		
Pre-Requisite:	*Valid professor information is available (name, email, department etc)*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Admin fills in professor form fields and submits.*	*Professor record is stored in MongoDB, and confirmation message appears.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.10: Admin Add Student

Admin Add Student			
UC-10			
Test Case ID:	*10*	**QA Test Engineer:**	*Menal Naeem*
Test case Version:	*1*	**Reviewed By:**	*Ahmad Javaid*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.10*
Revision History:	*None*		
Objective:	*Ensure the admin can add a new Student to the system.*		
Module:	*Admin Panel Student Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Admin is already logged in.*		
Pre-Requisite:	*Valid Student information is available (name, email, department etc)*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Admin fills in Student form fields and submits.*	*Student record is stored in MongoDB, and confirmation message appears.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.11: Professor - Upload Grades

Professor - Upload Grades			
UC-11			
Test Case ID:	*11*	**QA Test Engineer:**	*Menal Naeem*
Test case Version:	*1*	**Reviewed By:**	*Ahmad Javaid*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.11*
Revision History:	*None*		
Objective:	*Verify that professors can upload and save student grades.*		
Module:	*Teacher Panel Marks Management*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*The course and student enrollments already exist.*		
Pre-Requisite:	*Professor is logged in and navigates to the Grader page.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Professor selects a student, uploads grades, and saves.*	*Grades are stored correctly in the database, and success alert appears.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.12: Student Dashboard - View Grades

Student Dashboard - View Grades			
UC-12			
Test Case ID:	*12*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Raheem Jalil*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.12*
Revision History:	*None*		
Objective:	*Verify that a student can view their uploaded grades accurately.*		
Module:	*Student Panel Marks*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Grades are already uploaded by professors.*		
Pre-Requisite:	*Student is logged in and enrolled in at least one course.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Navigate to the "Marks" page and verify displayed grades.*	*Grades appear correctly for each enrolled course.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.13: Student Login Module

Student Login Module			
UC-13			
Test Case ID:	*13*	**QA Test Engineer:**	*Raheem Jalil*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.13*
Revision History:	*None*		
Objective:	*Verify that a student can successfully log in with valid credentials.*		
Module:	*Student Panel Login*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Student credentials already exist in the database.*		
Pre-Requisite:	*Student account is created by the admin and is active.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Attempt login using correct email and password combination.*	*Student is successfully redirected to the Student Dashboard.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.14: Teacher Login Module

Teacher Login Module			
UC-13			
Test Case ID:	*14*	**QA Test Engineer:**	*Raheem Jalil*
Test case Version:	*1*	**Reviewed By:**	*Ahmad Javaid*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.13*
Revision History:	*None*		
Objective:	*Verify that a Teacher can successfully log in with valid credentials.*		
Module:	*Teacher Panel Login*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Teacher credentials already exist in the database.*		
Pre-Requisite:	*Teacher account is created by the admin and is active.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*Attempt login using correct email and password combination.*	*Teacher is successfully redirected to the Professor Dashboard.*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

Table 6.15: Chatbot Error Handling

Chatbot Error Handling			
UC-14			
Test Case ID:	*15*	**QA Test Engineer:**	*Ahmad Javaid*
Test case Version:	*1*	**Reviewed By:**	*Menal Naeem*
Test Date:	*13-4-2025*	**Use Case Reference(s):**	*4.6.14*
Revision History:	*None*		
Objective:	*Validate chatbot behavior when an unknown query is asked.*		
Module:	*Teacher Panel Login*		
Environment:	*Web Browser (Desktop, Mobile)*		
Assumptions:	*Chatbot fallback message is properly configured.*		
Pre-Requisite:	*Chatbot API service is live.*		
Step No.	**Execution Description**	**Procedure Result**	
1	*User asks an unknown/invalid query eg "What is 12345656 the weather in Paris?"*	*Chatbot responds with a fallback message like "I'm sorry, I can only assist with university-related queries."*	
Comments: The test case is passed. Our system is working according to our requirements.			
Passed			

6.3 Test Metrics

The summary of the common ground of attributes of test case metrics is given below.

Table 6.16: Test case Matric

Metric	**Value**
Number of Test Cases	14
Number of Test Cases Passed	14
Number of Test Cases Failed	0
Test Case Defect Density	0
Test Case Effectiveness	0
Traceability Matrix	File is enclosed separately.

Chapter 7 User Manual

7.1 Introduction

This User Manual provides a step-by-step guide for three types of users — Students, Teachers, and Administrators — on how to efficiently use the University Registration System with Chatbot. It includes instructions for logging in, navigating key features, and troubleshooting common problems.

7.2 System Requirements

- Internet-enabled device (PC, Laptop, Tablet, Smartphone)
- Updated web browser (Google Chrome, Firefox, Microsoft Edge)
- Registered User Account (Student / Teacher / Admin)

7.3 How to Access the System

1. Open a web browser and visit the University Portal URL.
2. Enter your Email ID and Password.
3. Click Login.

For new account registration or issues, contact the administration.

7.4 Student User Manual

7.4.1 Key Features for Students

7.4.1.1 View Available Courses

Click Courses after login. Browse, search, or filter courses by department or semester.

7.4.1.2 Access and Update Student Profile

Click your profile icon and select Profile. View your contact information, major, etc., and save any updates.

7.4.1.3 Access FAQS and Report Error

Click your profile icon and select Reports. View Frequently Asked Questions and report any errors.

7.4.1.4 Manage Add/Drop Requests

Go to My Courses and click Add Course or Drop Course. Confirm changes when prompted and review updated course enrollments.

7.4.1.5 Wait-list Management

If a course is full, click Join Waitlist. Track your status in My Courses Waitlist section.

7.4.1.6 Chatbot Assistance

Open the chatbot via the bottom-right icon and ask questions like:

- "How to drop a course?"
- "Deadline for registration?"

Receive instant guidance.

7.4.1.7 View Deadlines

Important academic dates are visible on the Dashboard.

7.4.1.8 View Attendance

Click Workflow after login. Access your attendance records.

7.4.1.9 View Marks

Click Marks after login. Access your marks by categories ie mid1, mid2 etc.

7.5 Teacher (Professor) User Manual

7.5.1 Key Features for Teachers

7.5.1.1 Manage Course Listings

Navigate to Manage Courses. View all courses assigned to you. Update course details, schedule, and syllabus.

7.5.1.2 Access FAQS and Report Error

Click your profile icon and select Reports. View Frequently Asked Questions and report any errors.

7.5.1.3 Grade Management

Navigate to Grader. Enter or upload student marks. Use the Generate Grades feature based on predefined criteria. Download detailed grade reports.

7.5.1.4 Track System Usage

Monitor activities like number of enrolled students, marks average etc.

7.5.1.5 Chatbot Assistance

Ask queries related to grading policies, course details, or technical help.

7.5.1.6 Access and Update Teacher Profile

Click your profile icon and select Profile. View your contact information, department, etc., and save any updates.

7.6 Admin User Manual

7.6.1 Key Features for Administrators

7.6.1.1 Student Management

Add, update, or delete student profiles.

7.6.1.2 Teacher Management

Add, update, or remove faculty members. Assign teachers to courses.

7.6.1.3 Course Management

Add new courses, update existing ones, or remove outdated courses. Configure prerequisites and seat limits.

7.6.1.4 Manage Add/Drop Requests

Approve special cases manually when required.

7.6.1.5 Deadline Configuration

Set and modify deadlines for registration, add/drop, and grading.

7.6.1.6 Generate System Reports

Generate reports like:

- Student enrollment statistics
- Faculty course loads
- System usage analytics

Export reports in PDF/Excel formats.

7.7 Troubleshooting Guide

Table 7.1: Common Problems and Solutions

Problem	**Solution**
Forgot Password	Click **Forgot Password** on the login page and follow instructions.
Course Missing	Contact Admin to verify course assignment.
Cannot Drop Course	Ensure deadline has not passed. If issue persists, contact Admin.
System Slow/Unresponsive	Refresh the page or clear browser cache.
Chatbot Error	Refresh the chatbot window or reload the page.

7.8 Support Contact

- Email: l211885@lhr.nu.edu.pk
- Phone: +923108491111
- Office Hours: Monday–Friday — 9:00 AM to 5:00 PM

Chapter 8 Experimental Results and Discussion

8.1 Introduction

This chapter details the experimental results from the University Semester Registration System, which incorporates an AI-powered chatbot. The primary focus is on evaluating the chatbot's performance, user satisfaction, system efficiency, and its overall impact on streamlining the university registration process. Each set of results is accompanied by a discussion to highlight the system's effectiveness and identify potential areas for improvement.

8.2 Analysis and Discussion

8.2.1 Chatbot Response Accuracy

The study aimed to evaluate the chatbot's accuracy in effectively addressing user queries. The results indicated strong performance, with the chatbot achieving a 90% accuracy rate for straightforward questions, such as those related to course availability, prerequisites, and registration deadlines. However, its accuracy dropped to 81% when handling more complex queries, particularly those requiring personalized suggestions. This difference underscores the strength of the chatbot's Natural Language Processing (NLP) algorithms in managing commonly asked questions while also revealing opportunities for improvement. Enhancing the algorithm with more specialized training data could help overcome challenges in delivering personalized responses.

8.2.2 System Processing Speed

The study aimed to evaluate the system's response time, particularly under heavy user load. Testing revealed that the average response time was 4 seconds per query under normal conditions, increasing to 7.5 seconds during high-traffic simulations. While there was a slight delay under peak load, the chatbot's performance remained within acceptable limits, which is especially important during critical periods like registration deadlines. These results highlight the efficiency of the backend infrastructure, built with Node.js and MongoDB.

8.2.3 User Satisfaction Survey

This evaluation aimed to assess user satisfaction with the chatbot's functionality and interface. A survey of 50 students revealed a high satisfaction rating of 4.3 out of 5, with users praising its ease of use and friendly conversational tone. Students particularly valued the real-time assistance, while faculty

appreciated the system's efficiency in handling repetitive queries. Feedback suggested enhancing the interface by incorporating more intuitive visual cues to improve navigation and further elevate the user experience.

8.2.4 System Reliability

The objective of this assessment was to evaluate the system's uptime and error rates during testing. The results were highly encouraging, with a satisfactory uptime performance and only minor disruptions caused by backend maintenance during heavy load. Error rates were equally low, with just 9% of interactions failing due to input misunderstandings. These findings demonstrate the system's strong reliability and robustness. The few issues observed were primarily linked to unusual input formats, indicating that future updates could focus on refining NLP algorithms to handle a broader range of user expressions and further minimize errors.

8.3 Conclusion

The University Semester Registration System, with its integrated AI-powered chatbot, has proven to be a valuable tool in enhancing the registration process. By streamlining course selection, improving access to key information, and easing administrative tasks, the system offers significant benefits to both students and staff. While there are opportunities for further refinement, the results confirm its effectiveness and potential for broader implementation in academic institutions. The insights gained from this study, supported by both quantitative and qualitative data, provide a strong foundation for future improvements, ensuring that the system continues to evolve in line with user needs and technological advancements.

Chapter 9 Conclusion and Future Work

This chapter summarizes the achievements, challenges, and lessons learned from the Course Registration Chatbot project. It also outlines potential future enhancements to improve the system's performance and usability. The project aimed to develop an AI-powered chatbot system to streamline the course registration process, reduce administrative workload, and enhance student experience. Key components such as natural language processing (NLP), customized support, and real-time query resolution were successfully implemented, demonstrating the platform's potential to optimize university management processes.

9.0.1 Summary and Final Thought:

The Course Registration Chatbot project effectively leveraged artificial intelligence to provide a seamless and user-friendly solution for course registration. The chatbot's ability to deliver real-time answers to frequently asked questions, assist in course selection, and notify users about deadlines marked significant progress towards achieving its goals. The integration of an extensive registration database enabled students to efficiently browse, add, and drop courses. While the chatbot successfully simplified student interactions and minimized administrative workload, certain areas, such as handling complex queries and enhancing dataset diversity, still require further refinement. The project has laid a strong foundation for future enhancements in FYP-2.

9.0.2 Key Findings and Accomplishments:

- **Chatbot Development**: Successfully implemented an AI-powered chatbot system capable of human-like conversations for course registration queries.
- **Database Integration**:Integrated an extensive registration database, allowing efficient course browsing, addition, and removal.
- **Customized Responses**:Customized responses based on individual student profiles, previous interactions, and registration history.
- **Real-time Notifications**:Enabled real-time notifications for deadlines and important events, improving user awareness.
- **Areas of improvment:**Identified areas for improvement, including handling advanced NLP cases and expanding dataset diversity.
- **Data Collection and Preprocessing**: Gathered and structured course and student-related data.

9.0.3 Scope Coverage and Challenges:

The project achieved most of its objectives, including efficient query resolution and automation of the registration process. However, challenges were encountered in handling highly complex queries and optimizing NLP algorithms for better contextual understanding. Additionally, expanding the chatbot's dataset to include more diverse queries and issues remains a priority. Features such as advanced query resolution and comprehensive student analytics are planned for FYP-2.

9.0.4 Goal Fulfillment:

The project met its primary goals of streamlining the course registration process, reducing administrative workload, and enhancing the overall student experience. By combining AI and NLP, the chatbot provided accurate and timely responses, creating a scalable and user-friendly platform. These achievements serve as a foundation for further development in FYP-2, where the system's functionality and reliability will be enhanced.

9.1 Future Work and Recommendations:

FYP-2 will focus on refining the chatbot's capabilities, optimizing its performance, and expanding its feature set. The following improvements will be prioritized:

- **Advanced NLP for Complex Queries** Enhancing the chatbot's ability to understand and respond to complex and multi-layered queries, ensuring higher accuracy and user satisfaction.
- **Expanded Dataset for Improved Accuracy** Training the chatbot on a larger and more diverse dataset to improve its ability to handle varied student questions and scenarios.
- **Enhanced User Profile Integration** Deepening the integration of individual student profiles to provide even more personalized responses and recommendations.
- **24/7 Accessibility Optimization** Ensuring reliable and uninterrupted service availability, with additional features like multilingual support for international students.
- **Advanced Analytics for Continuous Improvement** Implementing user activity tracking and analytics to gain insights into student needs, enabling iterative improvements in the chatbot's functionality.
- **Seamless Integration with University Systems** Finalizing integration with all university systems, including fee payment, course material access, and academic requirement tracking, to provide a holistic solution.

- **Multi-Language Support**To make the chatbot more accessible, we plan to add support for multiple languages, enabling students to communicate in their preferred language. This will be implemented using NLP models that can detect languages and translate responses, ensuring a smooth and user-friendly experience for everyone.

With these enhancements, the Course Registration Chatbot will achieve its full potential as a comprehensive, user-centric platform, establishing itself as a valuable tool for both students and university administration. The improvements will ensure a more efficient, reliable, and personalized course registration experience, aligning with the project's overarching vision.

Bibliography

[1] J. Doe and J. Smith, "The use of conversational ai in university registration systems," *ABAC Journal*, vol. 43, no. 2, pp. 80–100, 2023. Available: http://www.assumptionjournal.au.edu/index.php/abacjournal/article/view/7730/3858.

[2] J. Doe and J. Smith, "Designing ai chatbots for university communication systems," *Kenyatta University Repository*, 2023. Available: https://ir-library.ku.ac.ke/server/api/core/bitstreams/6654b1e3-0078-43a0-bda3-7fbd10809feb/content.

[3] W. O. Amollo, "The design and implementation of ai chatbots for educational institutions," *University of Nairobi Repository*, 2023. Available: http://erepository.uonbi.ac.ke/bitstream/handle/11295/161598/Wycliffe%20Onyalo%20Amollo-%20UON.pdf?sequence=1&isAllowed=y.

[4] Sakulwijitsinthu and Mindajao, "Design and evaluation of ai-based chatbots in educational institutions," in *International Conference on AI in Education*, March 2022.

[5] Botnerds and Clark, *AIML and Pandorabots: Building Intelligent Chatbots for Universities*. TechEdu Press, 2020.

[6] Khanthavit and Khanthavit, "Ai chatbots for distance learning: Enhancing communication and support in universities," *Distance Learning Journal*, vol. 12, pp. 67–84, February 2023.