A

MINI PROJECT REPORT

On

COLLEGE ENQUIRY CHATBOT

Submitted in partial fulfillment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering

Submitted
By

T. MEENAKSHI (22UP5A0515)
A. SHRUTHIKA (21UP1A05D1)
T.AJITHA (22UP5A0515)

Under the Guidance

of

Mrs. G.Anitha
Assistant professor

Professor/Associate Professor/Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN
Accredited to NBA (CSE & ECE) and NAAC A+
(Affiliated to Jawaharlal Nehru Technological University Hyderabad)
Kondapur (Village), Ghatkesar (Mandal), Medchal (Dist.)
Telangana-501301
(2021-2025)

VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN

(An Autonomous Institution)

(Sponsored by Lavu Educational Society)

[Affiliated to JNTUH, Hyderabad & Approved by AICTE, New Delhi]

Kondapur (V), Ghatkesar (M), Medchal –Malkajgiri (D)- 501301.Phone:9652910002/3

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled **"College Enquiry Chatbot"** submitted by **T. Meenakshi (22UP5A0515) A. Shruthika (21UP1A05D1) T. Ajitha (22UP5A0516)** in the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering**, **Vignan's Institute ofManagement and Technology for Women** is a record of bonafide work carried by them under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or institute for the award of any degree.

PROJECT GUIDE

Mrs. G. Anitha

THE HEAD OF DEPARTMENT

Mrs. M. Parimala
(Associate Professor)

(External Examiner)

VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN
(An Autonomous Institution)
(Sponsored by Lavu Educational Society)
[Affiliated to JNTUH, Hyderabad & Approved by AICTE, New Delhi]
Kondapur (V), Ghatkesar (M), Medchal –Malkajgiri (D)- 501301.Phone:9652910002/3

DECLARATION

We hereby declare that the results embodied in the project entitled **"COLLEGE ENQUIRY CHATBOT"** is carried out by us during the year 2024-2025 in partial fulfillment of the award of **Bachelor of Technology** in **Computer Science and Engineering** from **Vignan's Institute of Management and Technology for Women** is an authentic record of our work under the guidance of Guide Name. We have not submitted the same to any other institute or university for the award of any other Degree.

T. Meenakshi (22UP5A0516)
A. Shruthika (21UP1A05D1)
T. Ajitha (22UP5A0516)

ACKOWLEDGEMENT

We would like to express sincere gratitude to **Dr G. APPARAO NAIDU**, **Principal, Vignan's Institute of Management and Technology for Women** for his timely suggestions which helped us to complete the project in time.

We would also like to thank our madam **Mrs. M. Parimala, Head of the Department and Associate Professor, Computer Science and Engineering** for providing us with constant encouragement and resources which helped us to complete the project in time.

We would also like to thank our Project guide **Mrs. G. Anitha Associate Professor, Computer Science and Engineering**, for providing us with constant encouragement and resources which helped us to complete the project in time with his/her valuable suggestions throughout the project. We are indebted to him/her for the opportunity given to work under his/her guidance.

Our sincere thanks to all the teaching and non-teaching staff of Department of Computer Science and Engineering for their support throughout our project work.

T. Meenakshi (22UP5A0515)
A. Shruthika (21UP1A05D1)
T. Ajitha (22UP5A0516)

INDEX

LIST OF FIGURES

Abstract

The College Enquiry Chatbot is an intelligent system designed to streamline the process of answering common queries from students, parents, and visitors regarding college-related information. This project aims to provide instant, accurate, and automated responses to frequently asked questions about admissions, courses, fees, campus facilities, faculty, and events, without the need for human intervention. The chatbot will leverage natural language processing (NLP) to understand and interpret user queries in a conversational manner. It will be trained on a dataset comprising typical questions and corresponding answers relevant to the institution. The system will be accessible through multiple platforms, including a web interface and mobile application, ensuring ease of access. CampusPal is a conversational AI chatbot designed to provide automated support for college-related enquiries. The chatbot utilizes natural language processing (NLP) and machine learning algorithms to understand and respond to student queries, providing instant answers and guidance.

CHAPTER 1

1. INTRODUCTION

A chatbot is software that simulates human-like conversations with users via text messages on chat. Its key task is to help users by providing answers to their questions. This could be a text based (typed) conversation, a spoken conversation or even a non-verbal conversation. Chat bot is typically perceived as engaging software entity which humans can talk to. It can be interesting, inspiring and intriguing. It appears everywhere, from old ancient HTML pages to modern advanced social networking.

College Enquiry Chatbot uses machine learning concepts to have conversations with humans. The purpose of developing this project is based on an intellectual chat-bot system which will deal with the academic activities like admission enquiry, fees structure, scholarship details, time-table of every department, details of the documents required to attach etc. With this chat-bot system it will be easy for the student to directly clear their queries in lesser time. Chat bots typically provide a text-based user interface, allowing the user to type commands and receive text in order to resolve the query. The Chatbot has information stored in its dataset to identify the sentences and making a decision itself as response to answer a given question. The program analyzes the user's query then the bot responds to the query.

1.1 objective

of the College Enquiry Chatbot System Project is to create an advanced, AI-driven solution that transforms the way colleges interact with prospective students, parents, and other stakeholders by providing instant, precise, and customized information. This chatbot aims to simplify the dissemination of essential details, such as courses offered, fee structures, admission procedures, eligibility criteria, faculty credentials, placement records, campus facilities, and other critical aspects of the institution. By utilizing cutting-edge technologies like Natural Language Processing (NLP), machine learning, and conversational AI, the system ensures real-time interaction with users, delivering a seamless and engaging experience.

A key focus of the project is to ensure 24/7 availability, overcoming the limitations of traditional communication methods like office hours, email delays, or in-person visits. The system not only enhances accessibility but also reduces the workload on administrative staff by automating repetitive tasks, such as addressing frequently asked questions, allowing them to focus on strategic and high-value activities. Its multilingual support further broadens its usability, ensuring that individuals from diverse linguistic and cultural backgrounds can access the information effortlessly.

Additionally, the chatbot is designed to integrate with multiple platforms, such as college websites, mobile apps, and social media, to reach a wider audience and provide a consistent user experience across channels. The integration of voice and text-based interactions caters to varied user preferences, while its ability to handle multiple queries simultaneously makes it scalable for peak admission seasons.

Furthermore, the chatbot employs data analytics to gather insights on user behaviour, frequently asked questions, and trends, enabling continuous optimization of its functionality and addressing user needs more effectively. By providing a cost-effective, efficient, and technology-driven solution, the College Enquiry Chatbot System aims to enhance the institution's reputation, improve user satisfaction, and position the college as a leader in adopting modern educational technologies. Ultimately, the project aspires to create a bridge between prospective students and the institution, fostering transparency, engagement, and convenience in the admission process and beyond.

1.2 Existing System for College Enquiry Chatbot System

The existing system for handling college-related inquiries is typically manual and may involve the following methods:

1. **Traditional Manual Assistance**:
    - Students and parents visit the college in person to gather information about courses, fee structures, placements, and other details.
    - Administrative staff or counsellors provide answers to these queries manually, leading to delays and inefficiencies.
2. **Email Communication**:
    - Colleges often rely on email as a mode of communication where prospective students send inquiries.
    - Responses can take hours or days, depending on the staff's availability, causing delays in decision-making for students.
3. **Phone-Based Support**:
    - A dedicated helpline is provided for handling inquiries.
    - This method is limited by working hours, and the staff's ability to handle multiple calls simultaneously, leading to long wait times.

4. **Static Information on Websites**:
    - Colleges provide information on their official websites in the form of FAQs, downloadable brochures, or static pages.
    - This method lacks interactivity, making it difficult for users to get personalized responses to their specific questions.
5. **Third-Party Forums or Social Media Queries**:
    - Prospective students may rely on forums, social media platforms, or third-party review websites to gather information.
    - This information is often unverified or incomplete, which can lead to confusion or misinformation.

1.2.1 Limitations of the Existing System:

- **Time-Consuming**: Manual handling of inquiries is slow and inefficient.
- **Limited Availability**: Most methods are only accessible during working hours or require in-person visits.
- **High Workload for Staff**: Repetitive queries increase the burden on administrative staff.
- **Lack of Personalization**: Static information on websites or brochures fails to address user-specific queries.
- **Limited Scalability**: Handling an increasing number of inquiries becomes challenging with the existing methods.

1.3 Proposed System of College Enquiry Chatbot Project

The proposed **College Enquiry Chatbot System** is an AI-powered solution designed to automate and streamline the process of providing information about the college. The system leverages Natural Language Processing (NLP) and machine learning to interact with users in real-time, providing instant and accurate responses to their queries.

Features of the Proposed System:

1. **Interactive Query Handling**: The chatbot provides answers to frequently asked questions (e.g., courses offered, fee structure, placements, faculty details, admission process, etc.).
2. **24/7 Availability**: Users can access the chatbot anytime, ensuring uninterrupted service.
3. **Personalized Responses**: The chatbot tailors responses based on the user's specific input, such as course preferences or eligibility criteria.

4. **Multichannel Support**: The chatbot can be deployed on multiple platforms, including the college website, mobile app, or messaging platforms like WhatsApp and Telegram.
5. **Voice and Text Interface**: Supports both text-based and voice-based interactions for user convenience.
6. **Multilingual Support**: Provides support for multiple languages to cater to a diverse audience.
7. **Data Analytics**: Collects and analyzes data on user queries to identify trends and improve the system over time.
8. **Real-Time Notifications**: Sends automated updates on admission deadlines, events, or announcements.

1.3.1 Advantages of the Proposed System:

1. **Improved Efficiency**:
    - Reduces response time by instantly answering queries, eliminating delays common in traditional systems.
2. **Cost-Effective**:
    - Minimizes administrative overhead by automating repetitive tasks, reducing the need for human involvement.
3. **Enhanced User Experience**:
    - Provides users with an intuitive and interactive platform, making it easy to access the desired information.
4. **24/7 Availability**:
    - Unlike traditional methods, the chatbot is always available, ensuring uninterrupted assistance even outside office hours.
5. **Scalability**:
    - Capable of handling multiple users simultaneously, making it suitable for high volumes of queries during peak admission periods.
6. **Personalization**:
    - Delivers customized responses, enhancing the user experience by addressing specific needs and preferences.
7. **Multilingual Support**:
    - Caters to a diverse audience by providing information in multiple languages, broadening its accessibility.

8. **Reduced Workload for Staff**:
    - Allows administrative staff to focus on complex or high-priority tasks by automating routine inquiries.
9. **Data Insights**:
    - Provides valuable insights into common questions, user behaviour, and areas where information can be improved.
10. **Accessibility Across Platforms**:
    - Makes college information available on various platforms, increasing its reach and convenience for users.

CHAPTER 2

2. Literature Survey for College Enquiry Chatbot System

S.No	Title	Authors	Year	Summary	Relevance to project
1.	Chatbots in Education: Scoping Review	M. Winkler, J. Soanes	2020	Explores the increasing use of chatbots in education to improve user engagement and provide personalized learning experiences	Demonstrates how chatbots can automate routine queries in educational settings.
2.	AI Chatbots: Applications,Challenges and Opportunities	P. Shukla, A. Maheshwari	2021	Discusses the application of AI and NLP in chatbots, focusing on how they process user queries and deliver intelligent responses.	Provides foundational knowledge for developing an AI-powered college enquiry chatbot
3.	Implementing Chatbot for College Admission:	S. Rajan, T. Prakash	2019	Highlights the implementation of a chatbot in Indian colleges, showing a 40% reduction in administrative workload and 85% faster response times	Demonstrates the feasibility and effectiveness of chatbots in college admission systems.
4.	Challenges in designing Chatbots for Higher Education	E. Brown, L.White	2020	Identifies challenges like handling complex queries, ensuring privacy, and maintaining trust. Provides strategies to overcome these challenges.	Offers insights into challenges and solutions for building a reliable chatbot system.

5.	Designing Centric Chatbots for Education	K. Sharma M. Gupta	2022	Focuses on the importance of user-friendly interfaces and conversational flow to improve user engagement and satisfaction.	Guides the design of a user-centric chatbot for college enquiries.
6.	Chatbots and Data Analytics Improving User-Experience Education	J. Khan S. Mehta	2021	Discusses integrating chatbots with analytics to track user behavior, identify trends, and improve system performance over time.	Shows how data analytics can optimize chatbot functionality for better performance.

CHAPTER 3

3. System Analysis

3.1 Purpose

The purpose of system analysis for the College Enquiry Chatbot System Project is to carefully examine and understand the requirements of all users, including prospective students, parents, and college staff, to design an effective solution. This stage involves analyzing the current enquiry-handling process, identifying inefficiencies, and establishing how the chatbot can streamline responses to commonly asked questions about courses, admissions, fee structures, placement records, and campus facilities. By leveraging technologies like Natural Language Processing (NLP) and machine learning, the analysis ensures the chatbot can provide real-time, accurate, and personalized responses to diverse user queries.

The system analysis also focuses on defining the architecture of the chatbot, including how it will interact with existing college databases and systems to retrieve up-to-date information. It aims to outline the key features of the chatbot, such as multilingual support, user-friendly interfaces, and seamless integration across different platforms, such as websites, mobile apps, and social media.

Moreover, the analysis identifies critical non-functional requirements, including system security, scalability, and performance, ensuring that the chatbot can handle a high volume of users, especially during peak admission seasons, and that it maintains user privacy and data protection. Ultimately, the purpose of the system analysis is to provide a comprehensive blueprint that guides the development of a scalable, efficient, and effective chatbot system that enhances user satisfaction, reduces administrative workload, and improves overall college engagement.

3.2 Scope

The scope of the system analysis for the College Enquiry Chatbot System Project encompasses identifying the key functional and non-functional requirements necessary for the chatbot to efficiently manage and respond to user inquiries. The primary focus is on defining the chatbot's capabilities to handle a wide range of queries related to college admissions, courses, fee structures, placement details, campus facilities, and more. The scope includes ensuring that the chatbot is designed to operate across multiple platforms such as the college website, mobile applications, and popular messaging platforms (e.g., WhatsApp, Telegram).

The system analysis will explore the integration of Natural Language Processing (NLP) and machine learning technologies to enable the chatbot to understand and process user queries in a natural, conversational manner. The scope also covers the development of a user-friendly interface that supports both text and voice-based interactions, making the system accessible to a diverse audience.

In terms of non-functional requirements, the scope includes addressing data security and privacy concerns by ensuring that the system complies with relevant regulations (e.g., GDPR). It also covers the scalability of the chatbot to handle high volumes of user queries, especially during peak admission seasons. Multilingual support is also within the scope to ensure the chatbot can cater to users from various linguistic backgrounds.

Furthermore, the analysis will outline the chatbot's ability to collect data on user interactions, providing valuable insights into common queries and improving system performance over time through data analytics. The scope also involves identifying potential challenges, such as handling complex or ambiguous queries, and suggesting solutions to ensure the system remains efficient and accurate.

Ultimately, the scope of this analysis ensures that the chatbot system is comprehensive, efficient, scalable, secure, and capable of providing an enhanced, user-centric experience to all stakeholders involved.

3.3 Feasibility Study

The feasibility study for the College Enquiry Chatbot System Project is conducted to evaluate the technical, operational, economic, and legal viability of implementing the chatbot system for managing college-related inquiries. This study assesses the practicality of the project in terms of its development, integration, and long-term sustainability, ensuring it aligns with the institution’s goals and resources.

3.2.1. Technical Feasibility

The technical feasibility of the project involves assessing the technological infrastructure required to develop and deploy the chatbot. This includes evaluating the use of Natural Language Processing (NLP) and machine learning algorithms to understand and respond to user queries accurately. The system will also require integration with existing college databases and platforms to provide real-time, accurate responses. Given the availability of advanced tools and frameworks, such as TensorFlow, Google Dialogflow, and Microsoft Azure, the project is technically feasible. Additionally, considerations for scalability, multi-platform deployment (website, mobile apps, social media), and multilingual support will be addressed to ensure the system can handle increasing user demands and diverse user needs.

3.2.2. Operational Feasibility

Operational feasibility refers to the chatbot's alignment with the operational processes and goals of the college. The system is designed to reduce the workload on administrative staff by automating frequently asked questions and other routine tasks, improving efficiency. The chatbot will be available 24/7, ensuring users can get information outside traditional office hours. The operational workflow, including the integration with existing systems (student databases, admission portals, etc.), will ensure that the chatbot provides accurate, up-to-date information. The college's staff will need training to handle exceptions where the chatbot cannot provide a satisfactory response. Additionally, the support required for monitoring and maintaining the system will be minimal but must be planned to ensure smooth functioning.

3.2.3. Economic Feasibility

Economic feasibility evaluates the financial viability of the project. The initial investment required for developing and deploying the chatbot will include costs for software development, integrating existing databases, setting up servers, and implementing necessary security measures. However, the project is expected to yield long-term savings by reducing the administrative burden and improving operational efficiency. With the ability to handle an increased number of inquiries during peak times (e.g., admission season), the system will prevent the need for additional human resources. The cost savings on manpower and enhanced user engagement also make the project a financially viable option. The return on investment (ROI) is expected to materialize through increased student satisfaction, more streamlined operations, and better resource management.

3.2.4. Legal Feasibility

Legal feasibility examines the legal aspects of implementing the chatbot system. This includes ensuring that the system complies with data protection laws, such as GDPR (General Data Protection Regulation), to ensure user privacy and data security. Since the chatbot will handle sensitive user data, including personal information and academic details, it will need to follow strict security protocols and encryption methods. The system will also need to adhere to the college's internal policies regarding data storage, access controls, and user consent. Ensuring that the chatbot operates within the boundaries of these legal frameworks is essential for its successful deployment and operation.

3.2.5. Social Feasibility

Social feasibility assesses the potential social impact of the chatbot system on users, including students, parents, and staff. The introduction of the chatbot will enhance user experience by offering faster responses, making college-related information more accessible, and reducing the effort required to gather information. However, there may be initial resistance from staff who are accustomed to manual inquiry handling. To address this, the project will involve educating and training stakeholders to embrace the new system. The chatbot's ability to offer multilingual support ensures inclusivity and caters to a diverse demographic, making it socially beneficial for students from various linguistic backgrounds.

3.4 Requirement Analysis

The requirement analysis phase is crucial in defining the functional and non-functional requirements of the College Enquiry Chatbot System. This phase ensures that the system will meet the needs of stakeholders, including prospective students, parents, and college staff, and will function as expected. The requirements outlined here are categorized into functional, non-functional, and system requirements.

3.4.1. Functional Requirements

Functional requirements define the core features and functionalities the chatbot must perform to meet the project's objectives.

- **User Interaction and Query Handling**:
    - The chatbot must be capable of handling a wide range of user queries related to courses, admissions, fee structures, placement records, faculty details, and campus facilities.
    - It should support natural language processing (NLP) to interpret and respond to questions in a conversational format.
    - The system must support text-based interactions and, optionally, voice-based interactions for better accessibility.
    - The chatbot should be able to understand and respond to both simple and complex queries accurately.
- **Multilingual Support**:
    - The chatbot must support multiple languages to cater to diverse users. The system should be capable of automatically detecting and responding in the language the user prefers (e.g., English, Hindi, regional languages).

- **Integration with College Systems**:
    - The chatbot must be integrated with the college's existing databases and management systems, such as student information systems, admission portals, and course databases, to provide real-time and accurate information.
    - The system should allow automatic updates, ensuring the chatbot provides up-to-date data about admissions, fees, and other time-sensitive information.
- **User Registration and Personalization**:
    - Users should be able to register with the system for personalized services, such as receiving tailored course recommendations based on their preferences and academic background.
    - The system should store user preferences and history to offer customized responses during future interactions.
- **24/7 Availability**:
    - The chatbot must be available 24/7 to ensure that users can inquire about the college at any time, even outside office hours.
- **Escalation to Human Support**:
    - In case the chatbot is unable to handle a query or when the user requests more detailed information, the system should escalate the query to a human representative, such as a counselor or administrative staff.
- **Data Analytics and Reporting**:
    - The system should collect user interaction data and analyze it to identify common queries, user preferences, and areas for improvement.
    - The chatbot must generate reports on user activity, frequently asked questions, and trends to help improve the system and college services.

3.4.2. Non-Functional Requirements

Non-functional requirements describe the overall characteristics that the system should possess to ensure efficiency, security, and user satisfaction.

- **Performance and Scalability**:
    - The chatbot must handle multiple users simultaneously, especially during peak periods such as the admission season.

    - The system should respond to user queries with minimal latency to ensure a seamless experience.

- **Security and Privacy**:
    - The chatbot must comply with data protection laws, such as GDPR, ensuring the privacy of user information and secure handling of personal data.
    - Sensitive data, such as personal and financial information, should be encrypted and stored securely.
- **User Experience (UX)**:
    - The chatbot should provide an intuitive and easy-to-use interface, ensuring a positive user experience for both tech-savvy users and those with limited technical knowledge.
    - It must support conversational flows that mimic natural human interactions and provide helpful, friendly responses.
- **Availability and Reliability**:
    - The chatbot should be available 99.9% of the time, with minimal downtime. In case of system failures, it should have failover mechanisms to restore services quickly.
- **Cross-Platform Support**:
    - The chatbot must work across multiple platforms, including the college website, mobile applications (iOS and Android), and popular messaging platforms (e.g., WhatsApp, Facebook Messenger).
- **Compliance with Standards**:
    - The system must comply with accessibility standards to ensure it is usable by people with disabilities, including compatibility with screen readers and other assistive technologies.

3.5 Requirement Specification

This requirement specification provides a detailed overview of the hardware, software, and language requirements for the College Enquiry Chatbot System Project. These specifications are essential to ensure the system is developed efficiently, functions optimally, and meets all performance, security, and user experience standards.

3.5.1. Hardware Specifications

The hardware specifications outline the system's physical and cloud infrastructure needed to run the College Enquiry Chatbot efficiently, ensuring it can handle high traffic volumes and provide a seamless user experience.

Server Requirements

- **Cloud Hosting**: The system should be hosted on a scalable cloud platform such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud.
- **Virtual Machines/Instances**:
    - Minimum of 2 virtual machines to handle load balancing and ensure high availability.
    - Each virtual machine should have at least 4 vCPUs and 16 GB of RAM.
- **Storage**:
    - **Database Storage**: At least 100 GB of storage for relational and NoSQL databases.
    - **Backup Storage**: Separate storage for regular backups, with a capacity of 50 GB to store daily backups.
- **Bandwidth**:
    - The system should support high bandwidth (minimum of 1 Gbps) to handle multiple simultaneous user queries and ensure minimal latency.
- **Redundancy & Failover**:
    - Cloud infrastructure should be configured with failover and load balancing for uninterrupted service.
    - Automated backups and disaster recovery protocols to ensure data integrity and service uptime.

End-User Device Requirements

- The system must be accessible on multiple platforms, including:
    - **Desktop Computers**: Compatible with modern browsers (Google Chrome, Mozilla Firefox, Safari).
    - **Mobile Devices**: Accessible via both iOS and Android devices using mobile browsers or native applications.

- **Smart Speakers**: Support for voice-based interactions via smart assistants like Google Assistant, Amazon Alexa (optional for voice-based queries).

3.5.2. Software Specifications

The software specifications define the tools, frameworks, libraries, and platforms required for the development, deployment, and operation of the College Enquiry Chatbot System.

Backend Development

- **Programming Languages**:
  - **Python**: The primary language for backend development and AI model implementation (NLP, machine learning).
  - **JavaScript/Node.js**: For implementing real-time communication and handling server requests.
- **Web Frameworks**:
  - **Flask/Django (Python)**: Lightweight web frameworks for handling HTTP requests and server-side logic.
  - **Express (Node.js)**: For building a RESTful API for chatbot communication.

Frontend Development

- **Languages**:
  - **HTML5/CSS3**: For designing the web interface, ensuring responsiveness and accessibility.
  - **JavaScript**: For dynamic interaction on the front end, handling chat interactions, and managing session states.
  - **React.js/Angular**: Optional for developing single-page web applications with a smooth user experience.

Natural Language Processing (NLP) and AI

- **NLP Libraries/Tools**:
  - **Google Dialogflow**: For building conversational interfaces and integrating NLP capabilities to interpret user input.
  - **Rasa**: An open-source machine learning framework for building custom NLP models, particularly for more specialized queries.
  - **spaCy**: A Python library for advanced NLP, useful for entity recognition and intent classification.

- **Machine Learning**:
    - **TensorFlow/PyTorch**: Used for training and implementing deep learning models that handle natural language understanding.

Database Management

- **Relational Database**:
    - **MySQL/PostgreSQL**: For storing structured data like user information, course details, and admission records.
- **NoSQL Database**:
    - **MongoDB**: For storing unstructured data, such as conversation logs, chatbot responses, and user queries.
- **Cache System**:
    - **Redis**: For caching frequently accessed data (e.g., FAQs, course details) to improve performance.

Analytics and Reporting

- **Analytics Tools**:
    - **Google Analytics**: For tracking user interactions and engagement with the chatbot.
    - **Power BI/Tableau**: For generating data visualizations and reports based on user behavior and chatbot performance.

Security

- **Encryption**:
    - **SSL/TLS**: For secure communication between the client and server, ensuring data integrity and privacy.
- **Authentication**:
    - **OAuth 2.0 / JWT (JSON Web Tokens)**: For secure user authentication and session management.
- **Database Encryption**:
    - Sensitive user data stored in databases must be encrypted both at rest and in transit.

3.5.3. Language Specifications

Language specifications outline the programming languages, frameworks, and libraries to be used for the development of the chatbot, ensuring the system is scalable, maintainable, and capable of handling natural language processing effectively.

Programming Languages

- **Python**:
    - Primary language for developing the backend, NLP models, and integrating machine learning algorithms.
    - Popular libraries like TensorFlow, Keras, and spaCy will be used for building machine learning models to handle user input.
- **JavaScript (Node.js)**:
    - Used for building real-time communication features and APIs for interaction between the user and the server.
    - Ensures a dynamic user interface and smooth communication between the client-side and backend.
- **HTML5/CSS3**:
    - For designing and styling the user interface of the chatbot on web platforms, ensuring that it is responsive and user-friendly.

Frameworks

- **Dialogflow**: A platform for building conversational interfaces, handling NLP and intent recognition. Ideal for handling most user queries out-of-the-box.
- **Rasa**: An open-source framework for building more customizable chatbots with advanced NLP capabilities, useful for more complex queries.
- **Express (Node.js)**: For building a RESTful API to facilitate communication between the frontend and backend.

Natural Language Processing Libraries

- **spaCy**: A library for advanced NLP tasks like named entity recognition (NER), text classification, and language parsing.
- **NLTK**: For processing and analyzing human language data, particularly useful in educational and FAQ contexts.

Chatbot Integration Platforms

- **WhatsApp API**: To allow the chatbot to function within popular messaging apps, extending reach and accessibility.
- **Facebook Messenger API**: To enable chatbot integration with Facebook, reaching a large user base on social media platforms.

CHAPTER 4

4. System Design

4.1 System Architecture

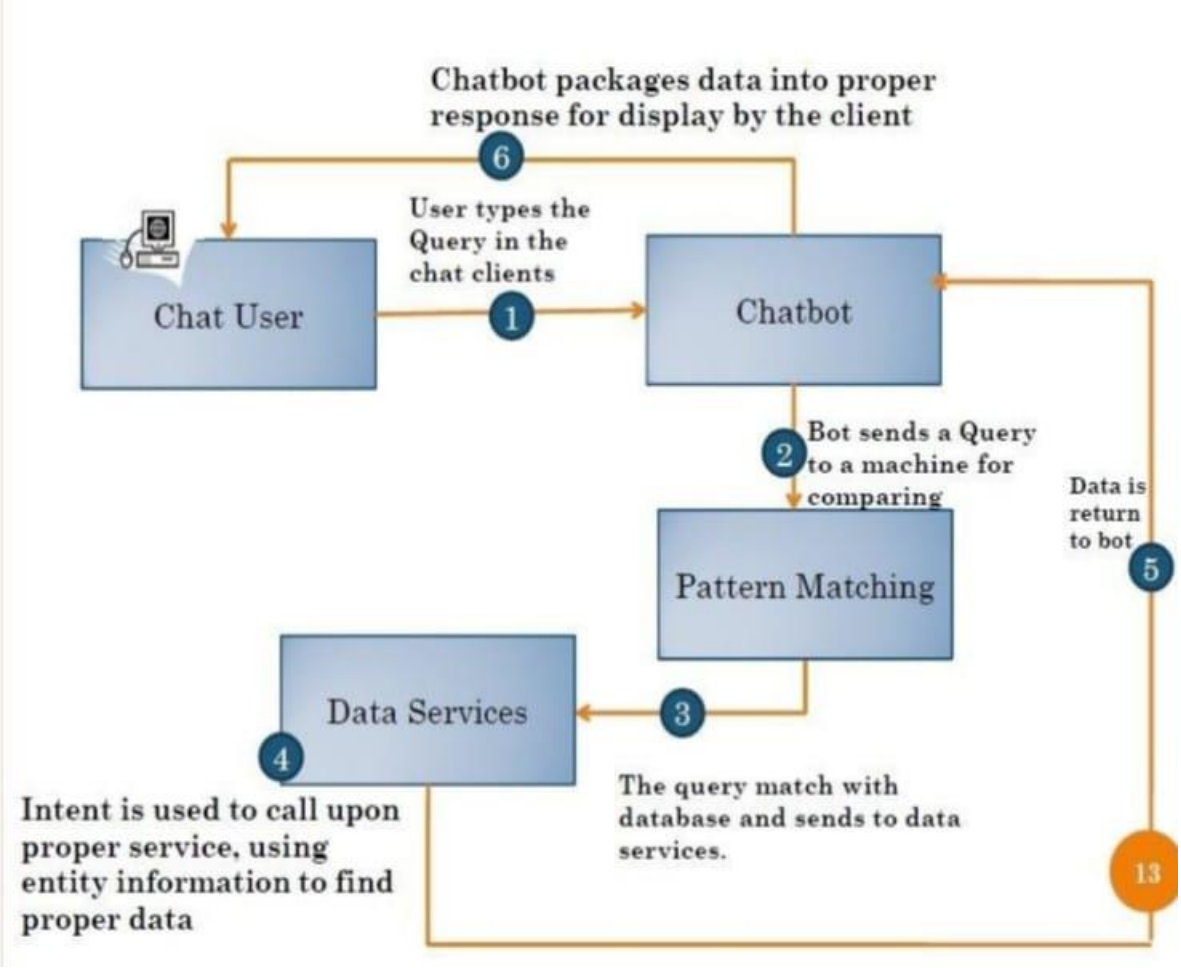

Figure 4.1 System architecture

Modules Client-Server (chat user): The proposed system has a client server architecture. All the information will be kept in an optimized database on the central server. This information can be accessed by the users through the android application installed on their smart phones (client machines). Each client machine will have an improved user interface.

Chatbot: A chatbot is a technology that allows users to have natural conversations to access content and services. Chatbots typically take the form of a chat client, leveraging natural language processing to conduct a conversation with the user. Chatbots control conversation flow based on the context of the users requests and respond with natural language phrases to

provide direct answers, request additional information or recommend actions that can be taken.

Pattern matching: Bot send a query to a machine for comparing. The query match with database sends to data services.

Data Services: Intent is used to call upon proper service.using entity information to find proper data. Hence all the modules are described above are completed in polynomial time sec t, so this problem is P.

4.2 Description

A block diagram is a diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. It may also show how the system operates, what are its inputs and outputs at various stages, and how the information, and/or materials flow through it. The block diagram for" Online chatting system for college enquiry knowledgeable Database" The proposed system has a client server architecture. All the information will be kept in an optimized database on the central server. This information can be accessed by the users through the android application installed on their smartphones (client machines). Each client machine will have an improved user interface. A chatbot is a technology that allows users to have natural conversations to access content and services. Chatbots typically take the form of a chat client, leveraging

4.3 UML Diagrams

4.3.1 Use case Diagram

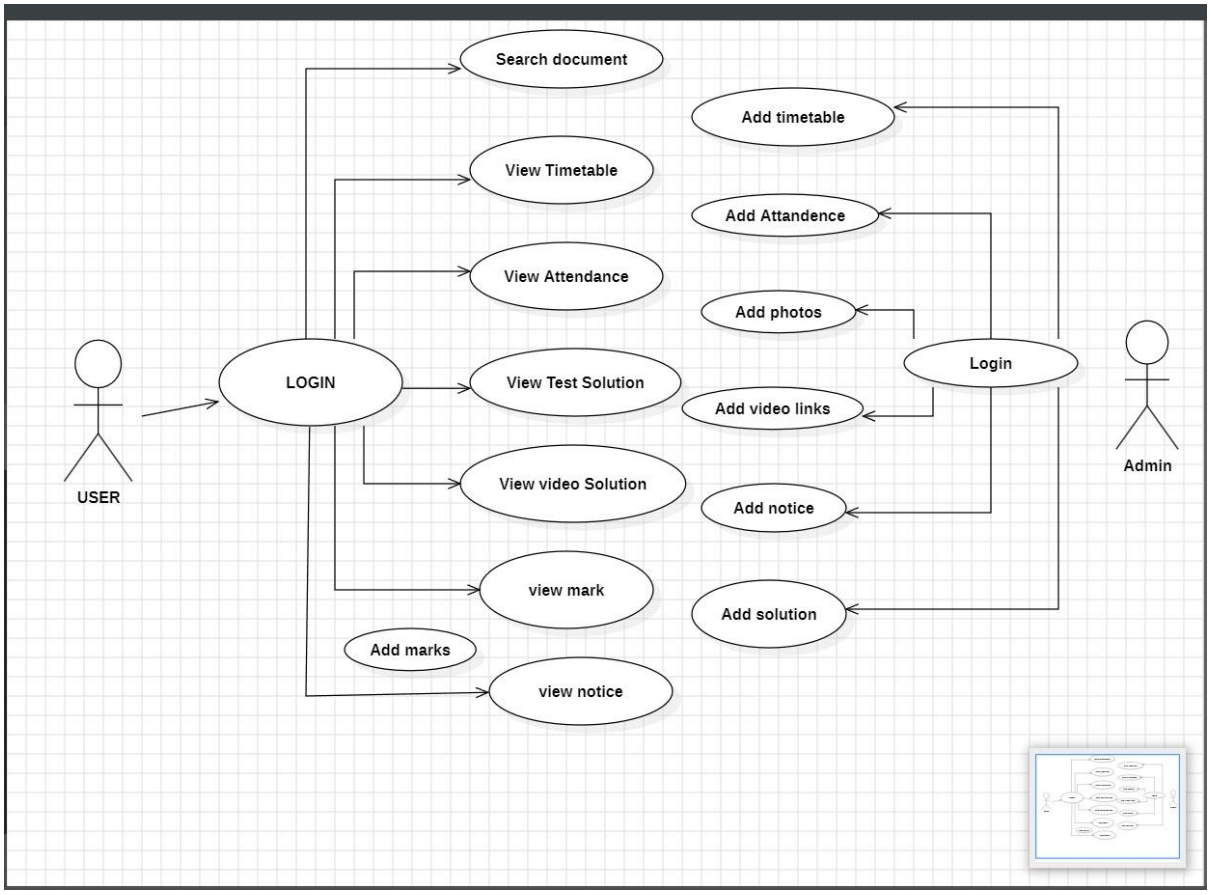

Figure 4.2 Use Case Diagram

Use case diagram is a graphical representation of the interactions between actors (users) and the system. In the context of the chatbot system for college enquiry using a knowledgeable database, a use case diagram can be used to identify the various use cases or scenarios in which the system is used. The use case diagram for the chatbot system can include the actors (users) such as prospective students, parents, and other stakeholders who are interested in obtaining information about the college. The various use cases can include querying information about courses, admission requirements, campus facilities, and other related information.

4.3.2 Class Diagram

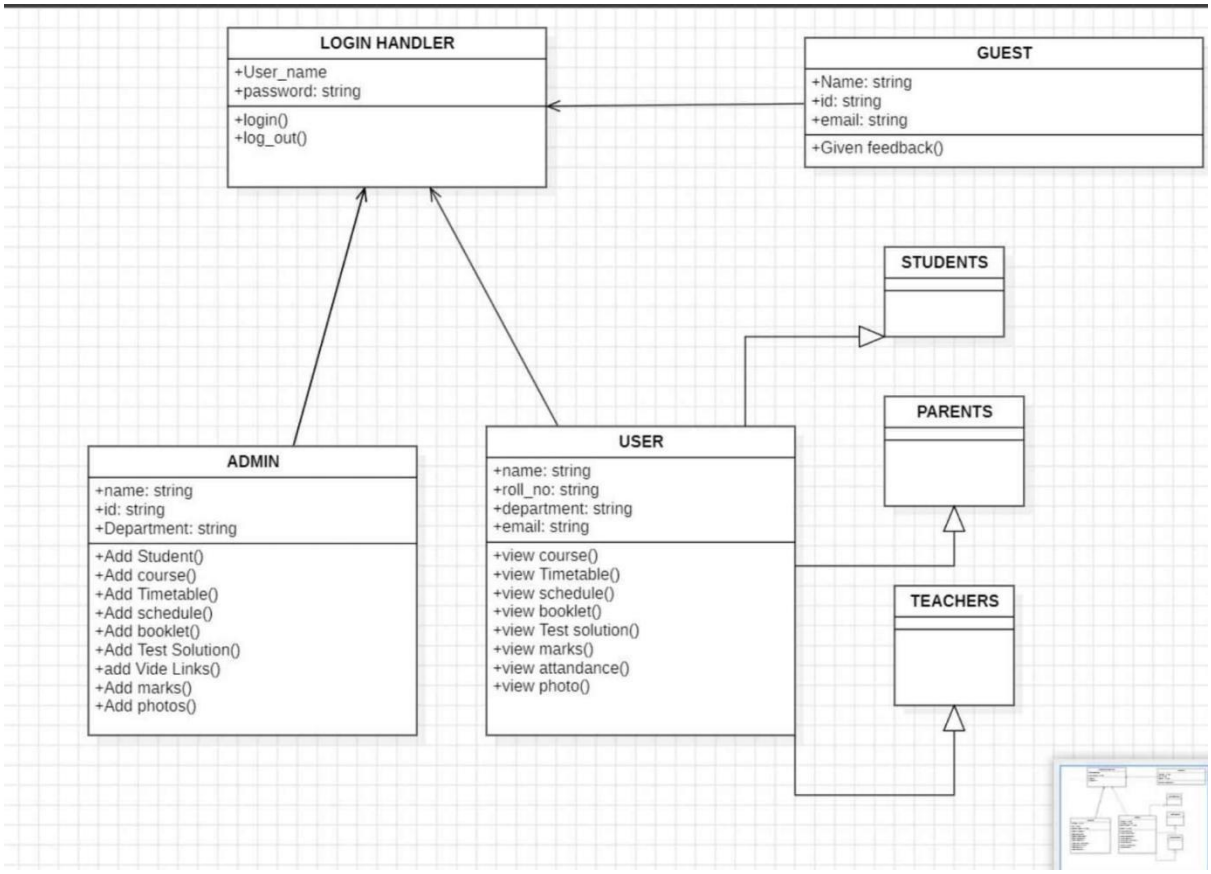

Figure 4.3.2 Class diagram

A class diagram is a type of UML (Unified Modeling Language) diagram that represents the classes and their relationships in a system. In the context of the chatbot system for college enquiry using a knowledgeable database, a class diagram can be used to represent the various classes in the system and their relationships. The class diagram for the chatbot system can include classes such as User, Query, Response, Natural Language Processing Engine, Knowledgeable Database, Retrieval-based Algorithm, Rule-based Algorithm, Machine Learning Algorithm, Hybrid Approaches, and Feedback Mechanism. Each class can have attributes and methods that define its behavior and properties.

4.3.3 State Chart Diagram

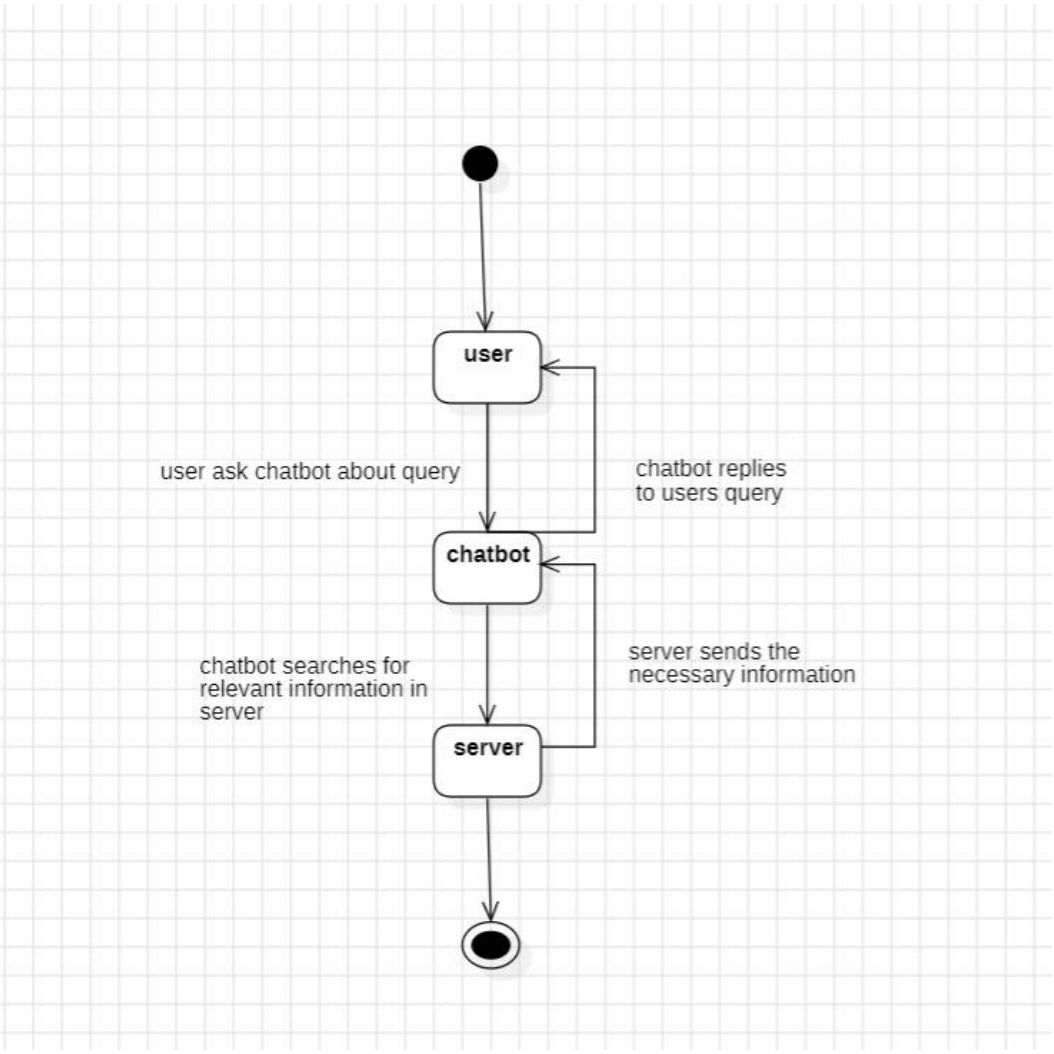

Figure 4.3.3 State Chart Diagram

1. User: Initiates the interaction by asking the chatbot a query.

2. Chatbot: Receives the query from the user. Searches for relevant information by communicating with the server.

3. Server: Provides the necessary information back to the chatbot.

4. Chatbot: Sends the response back to the user.

This diagram depicts the basic flow of a query-response system, where the chatbot acts as an intermediary between the user and the server. The arrows reflect the direction of information flow, clearly indicating the transitions between these states.

4.3.4 Activity Diagram

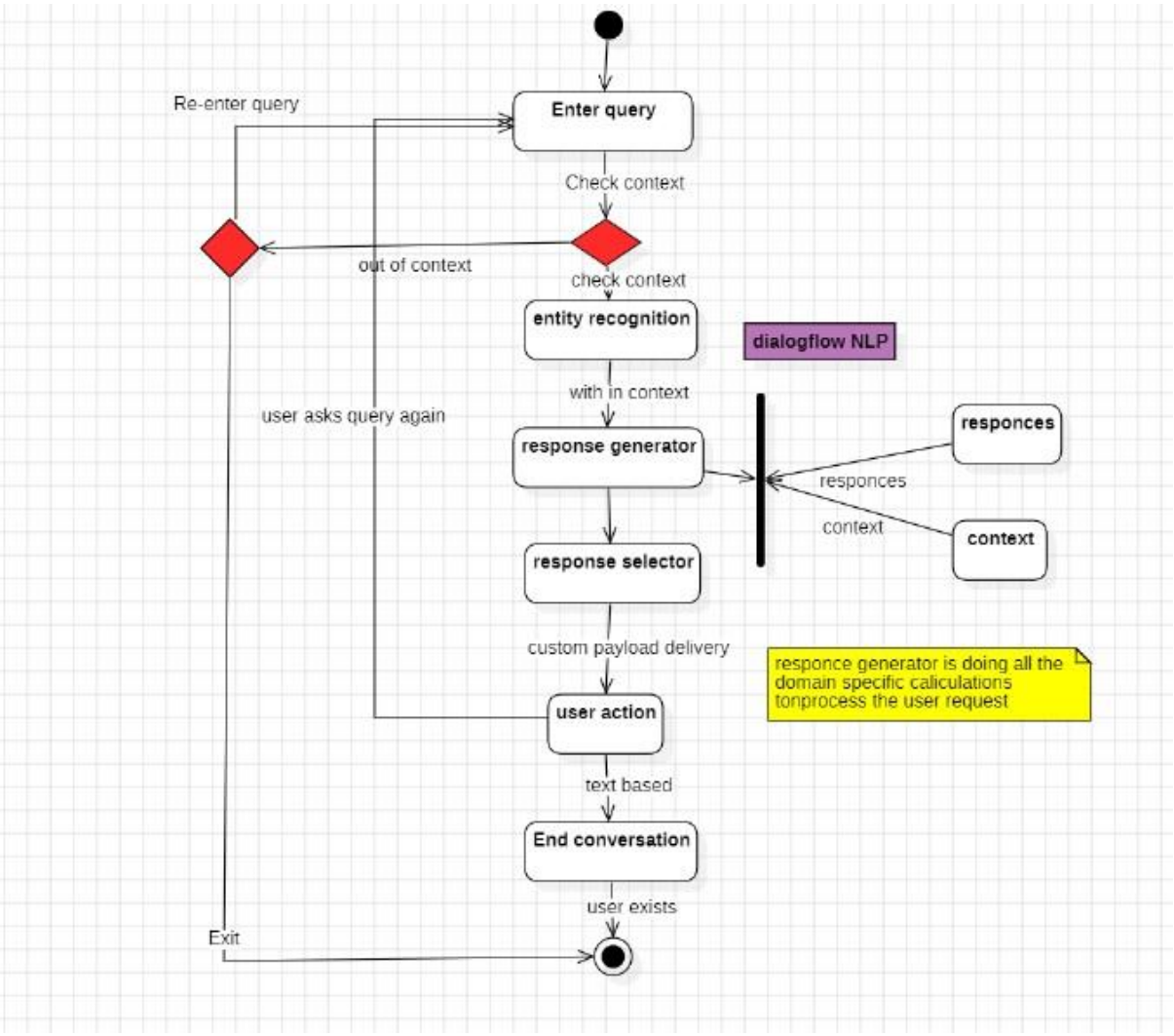

Figure 4.3.4 Activity Diagram

This activity diagram represents the workflow of a college enquiry chatbot project and highlights the flow of interactions between the user and the system. Here is the detailed description of the diagram:

1. User Initiation (Start Point)

- Enter Query: The process begins when a user enters a query into the chatbot system.

2. Query Context Check

- The chatbot verifies whether the query is within context or out of context:

3. Entity Recognition

- At this step, the system checks the query for specific entities and determines what the user is asking about.
- It uses Dialogflow NLP (Natural Language Processing) to analyze and process the query context.

4. Response Generation

- If the query is recognized, the Response Generator performs domain-specific calculations and processes the user request.

5. Response Selection

- The Response Selector selects the most appropriate response based on the query context and user input.

6. User Action

- The selected response is delivered to the user as part of a custom payload.
- This interaction may involve follow-up actions depending on the nature of the query.

7. End Conversation

- If the response satisfies the query and no further input is required, the chatbot proceeds to End Conversation.
- If the user exits, the process is complete.

8. Iterative Process

- If the user asks another query, the process loops back to the Enter Query step.

CHAPTER 5

5. Implementation and Result

5.1 Methods/algorithms

1. Flask app backend

Routes

1. Homepage (/):

- Method: GET
- Purpose: Displays the main homepage of the app. Likely used

2. Admin Chat (/admin_chat):

- Methods: GET and POST
- Purpose: Handles interactions with the admin interface, specifically for managing app data (e.g., faculty or timetable).
- GET: Renders the admin chat interface.
- POST: Processes commands issued by the admin, such as:
    - Adding new faculty records.
    - Viewing the list of faculty.
    - Adding/viewing timetable details.

3. Chatbot (/chatbot):

- Methods: GET and POST
- Purpose: Handles user queries and interactions with the chatbot system.
- POST: Routes user inputs to the process_user_query function for response generation.

Core Functions

1. execute_query:

    - Executes database write operations (e.g., adding new records for faculty or timetables).

2. fetch_query:

    - Fetches data from the database for read operations (e.g., viewing faculty details or timetable information).

3. process_user_query:

    - Processes user-specific chatbot requests, determining the best response for their query.
    - Likely integrates with natural language processing (NLP) or command parsing for intelligent interaction.

Command Processing

a) Admin Commands:
   - Add Faculty: Inserts a new faculty member into the database.
   - Add Timetable: Adds or updates timetable details.
   - View Faculty: Retrieves and displays a list of all faculty members.
   - View Timetable: Retrieves and displays timetable information.

b) Error Handling:
   - Validates admin/user input for proper formats or completeness.
   - Ensures commands are appropriately handled, with fallback responses or errors for invalid inputs.
   - This backend setup ensures the app can handle both admin operations (for data management) and user queries (via the chatbot) while maintaining robust database interaction.

2. Admin Chatbot Frontend

This section explains the frontend implementation of the Admin Chatbot Interface using HTML, CSS, and JavaScript.

HTML & CSS

1. Chat Interface:

The main chat interface consists of a clean and user-friendly layout.

Separate styles are applied for user messages (e.g., aligned to the right, different background color) and bot messages (aligned to the left with another style).

2. Modals:

Purpose: Provides a pop-up interface for adding faculty and adding timetable details.

These modals typically include input fields (e.g., text boxes, dropdowns, or date pickers) for entering data.Modals are styled to blend seamlessly with the main app design.

JavaScript

The JavaScript code handles interactions between the user, the chatbot interface, and the backend.

1. sendMessage:

- Purpose: Processes user inputs and displays chatbot responses in real-time.
- Workflow:
  - Captures user input from the chatbox.
  - Sends the input to the backend /admin_chat route using a POST request (via fetch or XHR).

- Receives the chatbot's response from the backend.
- Dynamically updates the chatbox with both user and bot messages

2. submitFacultyForm:

- Purpose: Processes inputs from the Add Faculty modal and sends them as a formatted command to the backend.
- Workflow:
  - Collects data entered in the modal (e.g., faculty name, department).
  - Formats the data into a recognizable command (e.g., "Add Faculty: Name, Department").
  - Sends this command to the backend /admin_chat route for processing.
  - Updates the chat interface with feedback from the backend.

3. submitTimetableForm:

- Purpose: Handles inputs from the Add Timetable modal and sends them as a command to the backend.
- Workflow:
- Gathers details (e.g., date, time, faculty, subject) entered in the modal.
- Formats the input into a timetable command (e.g., "Add Timetable: Subject, Time, Faculty").
- Sends the command to the backend /admin_chat for processing.
- Updates the chat interface dynamically with the result or status of the operation.
- Dynamic Message Display in Chatbox

Real-Time Updates:

Both user and bot messages are appended dynamically to the chatbox.

New messages appear at the bottom, ensuring smooth scrolling.

Styling:

Messages use pre-defined CSS classes for consistent formatting.

Each message type (user or bot) has its alignment, padding, and color scheme.

This frontend design ensures smooth and interactive communication between the admin and the chatbot system, with easy handling of database operations via modals.

3. User chat Frontend

The User Chatbot Frontend is designed to facilitate user interactions with the system, enabling them to query details like faculty information or

timetables. While it shares similarities with the Admin Chatbot Frontend, it has distinct differences in styling and functionality tailored for user interactions.

HTML & CSS

1. Chat Interface: A clean and intuitive design that caters to the end-user experience.

Styling Differences: User messages and bot responses have distinct visual styles to differentiate between the two.

Colors, fonts, and alignments are user-centric, with lighter tones and friendly visuals to make the interface inviting.

2. Responsive Design:Ensures compatibility across devices (desktop, tablet, mobile).

Adapts chatbox dimensions and button layouts for smaller screens.

3. No Modals: Unlike the admin interface, there are no modals for data input since users typically query existing information rather than add new data.

JavaScript

The JavaScript code enables smooth interaction between the user, the chatbot interface, and the backend.

1. send Message: Purpose: Handles user queries and displays chatbot responses dynamically.

- Workflow:
    - Captures the user's input from the text input field.
    - Sends the input to the backend /chatbot route using a POST request. Receives a response (e.g., faculty details, timetable information) from the backend.Dynamically appends both the user's query and the bot's response to the chat interface.

2. Message Formatting: User queries are displayed as right-aligned messages with user-specific styling.Bot responses are displayed as left-aligned messages with friendly, informative text.

3. Error Handling: Ensures proper feedback is given for invalid or unrecognized queries.

Displays error messages (e.g., "Sorry, I didn't understand that.") in the chatbox when necessary.

Dynamic Message Display in Chatbox

Real-Time Updates: The chatbox dynamically updates with each new query and response.Automatically scrolls to show the latest messages.

Styling:

Messages use unique CSS classes for user and bot messages. The user-side interface emphasizes clarity and ease of understanding.

This frontend provides a user-friendly experience with a visually appealing and interactive chat interface, ensuring smooth communication between users and the chatbot system. It is optimized for quick queries and responses, maintaining a focus on simplicity and functionality.

4. Additional Features

I. Database Operations

The chatbot integrates database operations to manage and retrieve faculty and timetable records effectively.Command Separation: There are distinct commands for viewing or adding records.

Examples:

- Admin: "Add Faculty: John Doe, Math, Room 101" adds a new record.
- User: "Show faculty for Math" retrieves relevant records.
- Error Handling: The system ensures accurate operations and guides users when input is incomplete or invalid.

For instance:

- Admin: If a required field (e.g., faculty name) is missing, the bot might respond, "Please provide a valid faculty name."
- User: For unsupported queries, the bot could reply, "I'm sorry, I don't understand that. Try asking about faculty or timetables."
- Feedback System: Successful commands return clear confirmation, like "Faculty added successfully!" or "Here is the timetable for Math."

II. Styling

Styling differentiates and enhances the visual experience for both admin and user interfaces.

Themes for Admin and User Interfaces:

- Admin: Professional and robust design with darker tones. Prioritizes functionality, featuring modals for adding or modifying records.
- User: Friendly and inviting design with lighter colors. Simplified interface focusing on retrieving information.
- Responsive Chatboxes and Modals:

- Responsive Design: Both chatboxes and modals adapt seamlessly across devices, ensuring usability on desktops, tablets, and mobiles.
- Admin Modals: Modals allow admins to input data easily with structured forms for adding faculty or timetable records.

Example: A modal for adding faculty may include fields like name, subject, and room.

- Dynamic Adjustments: Chatboxes adjust to accommodate message lengths and ensure smooth scrolling for longer conversations. Modals scale appropriately, ensuring readability and ease of interaction on smaller screens.

By combining database operations with user-centric styling, the system ensures a seamless experience for both admins and users, supporting efficient data management and intuitive interactions.

5.2 Sample Code

1. index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>VMTW College Chatbot</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<!-- Scattered Background with Logos -->

<div class="logo-container">

<imgsrc="https://www.vmtw.in/images/header/logo.png" alt="Logo 1" class="logo-

1">

<imgsrc="https://www.vmtw.in/images/header/logo.png" alt="Logo 2" class="logo-

2">
```

```
<imgsrc="https://www.vmtw.in/images/header/logo.png" alt="Logo 3" class="logo-
3">
<imgsrc="https://www.vmtw.in/images/header/logo.png" alt="Logo 4" class="logo-
4">
</div>

<!-- Chatbot Container -->
<div class="chatbot-container">
<div class="chatbot-header">
<h2>VMTW College Chatbot</h2>
</div>
<div class="chatbot-messages" id="chatbot-messages"></div>
<div class="chatbot-input">
<input type="text" id="user-input" placeholder="Ask me anything..." />
<button onclick="sendMessage()">Send</button>
</div>
</div>

<!-- FAQ Section -->
<div class="faq-section">
<h3>Frequently Asked Questions</h3>
<ul id="faq-list">
<li onclick="displayAnswer('What is the admission process?')">What is the
admission process?</li>
<li onclick="displayAnswer('What undergraduate courses are offered?')">What
undergraduate courses are offered?</li>
<li onclick="displayAnswer('What are the college contact details?')">What are the
```

```
college contact details?</li>
<li onclick="displayAnswer('What are the college details?')">What are the college
details?</li>
<li onclick="displayAnswer('What are the college timings?')">What are the college
timings?</li>
<li onclick="displayAnswer('Tell me the HOD details')">Tell me the HOD details</li>
</ul>
</div>

<script src="script.js"></script>
</body>
</html>
```

2. Styles.css

```
body {
    font-family: Arial, sans-serif;
    display: flex;
    flex-direction: column;
    align-items: center;
    height: 100vh;
    margin: 0;
    background-color: #ffffff;
    position: relative;
    overflow: hidden;
}

/* Scattered Background */
.logo-container {
    position: absolute;
```

```
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    pointer-events: none;
    z-index: -1;
}

.logo-container img {
    position: absolute;
    width: 120px; /* Increased size of the logos */
    height: auto;
    opacity: 0.1; /* Subtle appearance */
}

/* Randomly placed logos */
.logo-1 {
    top: 10%;
    left: 20%;
}

.logo-2 {
    top: 30%;
    left: 70%;
}

.logo-3 {
    top: 60%;
    left: 40%;
}
```

```
.logo-4 {
    top: 80%;
    left: 10%;
}

/* Chatbot Container */
.chatbot-container {
    width: 350px;
    height: 500px;
    background-color: #ffffff; /* White background for the chatbot */
    border-radius: 10px;
    box-shadow: 0 0 15px rgba(0, 0, 0, 0.1);
    display: flex;
    flex-direction: column;
    overflow: hidden;
    margin-top: 20px;
}

/* Chatbot Header */
.chatbot-header {
    background-color: #4b0082; /* Purple */
    color: white;
    text-align: center;
    padding: 10px;
}

/* Chat Messages Section */
.chatbot-messages {
    flex-grow: 1;
```

```
    padding: 10px;
    overflow-y: auto;
    background-color: #f7f7ff; /* Light purple/white */
    font-size: 1rem;
}

.chatbot-messages div {
    margin: 5px 0;
    padding: 10px;
    border-radius: 10px;
    width: fit-content;
    max-width: 80%;
}

/* User Message Bubble */
.user-message {
    background-color: #4b0082; /* Purple */
    color: white;
    align-self: flex-end;
}

/* Bot Message Bubble */
.bot-message {
    background-color: #1f3d7a; /* Navy blue */
    color: white;
    align-self: flex-start;
}

/* Chat Input Section */
.chatbot-input {
```

```
    display: flex;
    padding: 10px;
    border-top: 1px solid #ddd;
    background-color: #ffffff; /* White input area */
}

#user-input {
    flex-grow: 1;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    margin-right: 10px;
}

button {
    padding: 10px 20px;
    background-color: #1f3d7a; /* Navy blue */
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #4b0082; /* Purple on hover */
}

/* FAQ Section */
.faq-section {
    margin: 20px auto;
```

```
    width: 350px;
    text-align: left;
}

.faq-section h3 {
    text-align: center;
    color: #4b0082; /* Purple */
}

.faq-section ul {
    list-style: none;
    padding: 0;
}

.faq-section ul li {
    margin: 5px 0;
    padding: 5px 10px;
    background: #e6e6fa; /* Light lavender */
    border-radius: 5px;
    cursor: pointer;
}

.faq-section ulli:hover {
    background: #d8bfd8; /* Thistle purple */
}
```

3. Script.js

```
const collegeData = {
course_details: [
        "B.Tech in Computer Science",
        "B.Tech in AI & ML",
```

```
        "B.Tech in Data Science",
        "B.Tech in AI & DS",
        "B.Tech in IT",
        "B.Tech in Electronics & Communication Engineering",
        "Basic Sciences and Humanities"
    ],
hod_details: {
computer_science: {
            name: "Mrs. M. Parimala",
            email: "parimala@vmtw.in",
            phone: "+91 98499 86685"
        },
        "AI & ML": {
            name: "Dr. D. Shanthi",
            email: "drshanthicse@gmail.com",
            phone: "+91 8125174595"
        },
        "Data Science": {
            name: "Dr. M. Vishnu Vardhana Rao",
            email: "vishnucse@vmtw.in",
            phone: "+91 8639936639"
        },
        "AI & DS": {
            name: "Dr. M. Vishnu Vardhana Rao",
            email: "vishnucse@vmtw.in",
            phone: "+91 8639936639"
        },
        "Information and Technology": {
            name: "Dr. S. Ranga Swamy",
            email: "sirisatiranga@vmtw.in",
```

```
            phone: "+91 9989896548"
        },
        "Electronics and Communication Engineering": {
            name: "Mr. P. Harikrishna",
            email: "ece@vmtw.in",
            phone: "+91 7093246048"
        },
        "Basic Sciences and Humanities": {
            name: "Dr. Rapolu Sridhar",
            email: "rapolu31@vmtw.in",
            phone: "+91 9490215079"
        }
    },
college_timings: "The college is open from 8:55 AM to 3:50 PM, Monday to Saturday.",
faqs: [
        {
            question: "What is the admission process?",
            answer: "The admission process is offline. You can check for college contact details
to contact the management."
        },
        {
            question: "What undergraduate courses are offered?",
            answer: "We offer courses like Computer Science, AI & ML, Data Science, AI & DS, IT,
and Electronics & Communication Engineering."
        },
        {
            question: "What are the college contact details?",
```

answer: "VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN,

Kondapur(V), Hyderabad, Medchal Dist-501301. Phone: +91 9652910002/3. Email:

info.vmtw@gmail.com"

},

{

question: "What are the college details?",

answer: `ABOUT VMTW

Vignan's Institute of Management & Technology for Women (VMTW) is a prestigious

educational institution located in Kondapur (V), Ghatkesar (M), Medchal (D), Telangana,

India. VMTW is the brainchild of Dr. L Rathaiah, Chairman, Vignan Group Institutions, and

was established in August 2008.

VMTW provides quality education to empower women. It is affiliated with JNTUH,

approved by AICTE, accredited by NBA, and certified by ISO 14001:2015 & 50001-2018.

The campus spans 5 acres and offers modern facilities, including state-of-the-art

infrastructure, laboratories, libraries, hostels, and a dedicated placement cell.

Beyond academics, VMTW encourages extracurricular activities, cultural events, and

sports for holistic student development.`

},

{

question: "What are the college timings?",

answer: "The college is open from 8:55 AM to 3:50 PM, Monday to Saturday."

```
        },
        {
            question: "Tell me the HOD details",
            answer: `
<ul>
<li><b>Computer Science</b><br>Name: Mrs. M. Parimala<br>Email:
parimala@vmtw.in<br>Phone: +91 98499 86685</li>
<li><b>AI & ML</b><br>Name: Dr. D. Shanthi<br>Email:
drshanthicse@gmail.com<br>Phone: +91 8125174595</li>
<li><b>Data Science</b><br>Name: Dr. M. Vishnu Vardhana Rao<br>Email:
vishnucse@vmtw.in<br>Phone: +91 8639936639</li>
<li><b>AI & DS</b><br>Name: Dr. M. Vishnu Vardhana Rao<br>Email:
vishnucse@vmtw.in<br>Phone: +91 8639936639</li>
<li><b>Information and Technology</b><br>Name: Dr. S. Ranga
Swamy<br>Email: sirisatiranga@vmtw.in<br>Phone: +91 9989896548</li>
<li><b>Electronics and Communication Engineering</b><br>Name: Mr. P.
Harikrishna<br>Email: ece@vmtw.in<br>Phone: +91 7093246048</li>
<li><b>Basic Sciences and Humanities</b><br>Name: Dr. Rapolu
Sridhar<br>Email: rapolu31@vmtw.in<br>Phone: +91 9490215079</li>
</ul>
            `
        }
    ]
};

// Function to handle user queries
function sendMessage() {
    const userInput = document.getElementById("user-
input").value.trim().toLowerCase();
    const chatMessages = document.getElementById("chatbot-messages");
```

```
    if (!userInput) return;

    // Display user message
addMessage(userInput, "user");

    // Determine the response
    let response = "Sorry, I don't understand that. Please ask about courses, HODs, or
timings.";

    if (userInput === "tell me the hod details") {
        response = collegeData.faqs.find(f =>f.question.toLowerCase() ===
userInput)?.answer || response;
    } else if (userInput === "what are the college timings?") {
        response = collegeData.college_timings;
    } else {
        const faq = collegeData.faqs.find(f =>userInput.includes(f.question.toLowerCase()));
        if (faq) response = faq.answer;
    }

    // Display the bot response
addMessage(response, "bot");

    // Clear the input field
document.getElementById("user-input").value = "";
}

// Function to display answers for FAQ clicks
function displayAnswer(question) {
```

```
    const faq = collegeData.faqs.find(f =>f.question === question);
    const response = faq ?faq.answer : "Sorry, I don't have an answer for that.";
addMessage(question, "user");
addMessage(response, "bot");
}

// Helper function to add messages to the chat
function addMessage(message, sender) {
    const chatMessages = document.getElementById("chatbot-messages");
    const messageDiv = document.createElement("div");
messageDiv.className = sender === "user" ? "user-message" : "bot-message";
messageDiv.innerHTML = message; // Use innerHTML for rich content
chatMessages.appendChild(messageDiv);
chatMessages.scrollTop = chatMessages.scrollHeight;
}
```

4. Instalizer_db

```
import sqlite3
import os

DB_PATH = 'database/chatbot.db'

def create_tables():
    os.makedirs(os.path.dirname(DB_PATH), exist_ok=True)
    conn = sqlite3.connect(DB_PATH)
    cursor = conn.cursor()

    with open('database/schema.sql', 'r') as f:
        cursor.executescript(f.read())
```

```
    conn.commit()
    conn.close()
    print("Database initialized successfully.")

if _name_ == '_main_':
    create_tables()
```

5. app.py

```
from flask import Flask, render_template, request, jsonify
from utils.db_utils import fetch_query, execute_query
from utils.chatbot_utils import process_user_query

app = Flask(_name_, template_folder='templates', static_folder='static')

@app.route('/')
def home():
    return render_template('index.html')



@app.route('/admin_chat', methods=['GET', 'POST'])
def admin_chat():
    if request.method == 'GET':
        return render_template('admin_chat.html')

    if request.method == 'POST':
        user_message = request.json.get('message', '').lower()

        # Add Faculty
```

```
    if 'add faculty' in user_message and ':' in user_message:
        try:
            _, details = user_message.split(':', 1)
            name, department, email, phone, designation = map(str.strip, details.split(','))
            query = "INSERT INTO faculty (name, department, email, phone, designation) VALUES (?, ?, ?, ?, ?)"
            execute_query(query, (name, department, email, phone, designation))
            return jsonify(reply="Faculty added successfully!")
        except Exception as e:
            print(f"Error: {e}")
            return jsonify(reply="Failed to add faculty. Please ensure correct format.")

    if 'add faculty' in user_message:
        return jsonify(reply="Please provide details: name, department, email, phone, designation.")

    # Add Timetable
    if 'add timetable' in user_message and ':' in user_message:
        try:
            _, details = user_message.split(':', 1)
            department, semester, day, time_slot, subject, faculty_name = map(str.strip, details.split(','))
            query = """
            INSERT INTO timetable (department, semester, day, time_slot, subject, faculty_name)
            VALUES (?, ?, ?, ?, ?, ?)
            """
            execute_query(query, (department, semester, day, time_slot, subject, faculty_name))
            return jsonify(reply="Timetable added successfully!")
```

```
        except Exception as e:
            print(f"Error: {e}")
            return jsonify(reply="Failed to add timetable. Please ensure correct format.")

    if 'add timetable' in user_message:
        return jsonify(reply="Please provide details: department, semester, day, time slot, subject, faculty name.")

    # View Faculty
    if 'view faculty' in user_message:
        try:
            result = fetch_query("SELECT name, department, email, phone, designation FROM faculty")
            if result:
                faculty_details = "\n".join(
                    [f"{i+1}. Name: {row[0]}, Dept: {row[1]}, Email: {row[2]}, Phone: {row[3]}, Designation: {row[4]}"
                    for i, row in enumerate(result)]
                )
                return jsonify(reply=f"Faculty Details:\n{faculty_details}")
            else:
                return jsonify(reply="No faculty records found.")
        except Exception as e:
            print(f"Error: {e}")
            return jsonify(reply="Failed to fetch faculty details.")

    # View Timetable
    if 'view timetable' in user_message:
        try:
            result = fetch_query("SELECT department, semester, day, time_slot, subject, faculty_name FROM timetable")
```

```
            if result:
                timetable_details = "\n".join(
                    [f"{i+1}. Dept: {row[0]}, Sem: {row[1]}, Day: {row[2]}, Time: {row[3]}, Subject: {row[4]}, Faculty: {row[5]}"
                    for i, row in enumerate(result)]
                )
                return jsonify(reply=f"Timetable Details:\n{timetable_details}")
            else:
                return jsonify(reply="No timetable records found.")
        except Exception as e:
            print(f'Error: {e}')
            return jsonify(reply="Failed to fetch timetable details.")

    # Command Not Recognized
    return jsonify(reply="Command not recognized. Please try again."

@app.route('/chatbot', methods=['GET', 'POST'])
def chatbot():
    if request.method == 'POST':
        user_message = request.json.get('message', '')
        response = process_user_query(user_message)
        return jsonify({'response': response})
    return render_template('chatbot.html')

if _name_ == '_main_':
    app.run(debug=True)
```

CHAPTER 6

6. System Testing

Testing has become an integral part of any system or project, especially in the field of information technology. It plays a crucial role in determining if one is ready to proceed further or capable of withstanding the rigors of a particular situation. This importance cannot be understated, making testing before development a critical step.

Before software is delivered to users, it must be thoroughly tested to ensure it fulfills its intended purpose. This testing involves various methods to verify the software's reliability. The program was tested logically, with the execution patterns repeated for a specific set of data. The code was exhaustively checked for all possible valid inputs, and the outcomes were carefully reviewed to ensure accuracy.

Module Testing:

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

Integration Testing:

After module testing, integration testing is performed. During the process of linking modules, there is a chance for errors to occur. These errors are identified and corrected through integration testing. In this system, all modules are connected and tested together. The testing results were accurate, ensuring that the mapping of jobs to resources was performed correctly by the system.

Accepting Testing:

When users find no major issues with its accuracy, the system undergoes a final acceptance test. This test ensures that the system meets the original goals, objectives, and requirements established during the analysis phase. It eliminates unnecessary execution, saving both time and money. With acceptance testing being the responsibility of users and management, the system is deemed acceptable and ready for operation.

CHAPTER 7

7. Screenshots

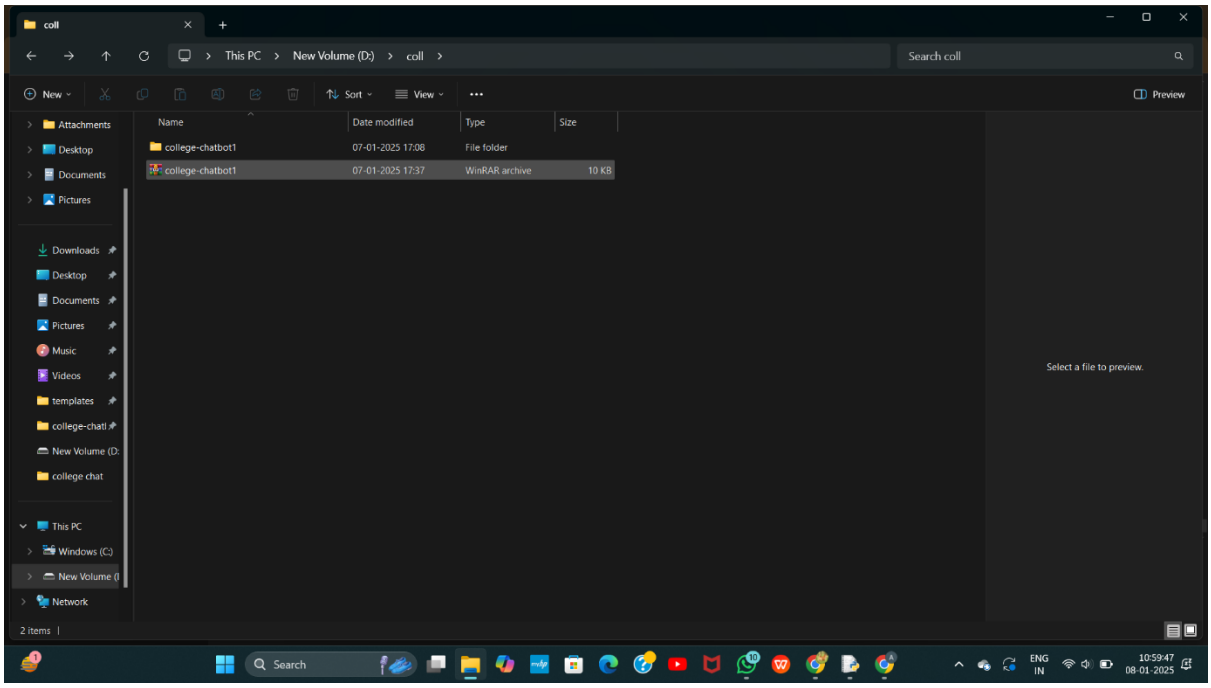

Figure 7.1 Execution Step 1

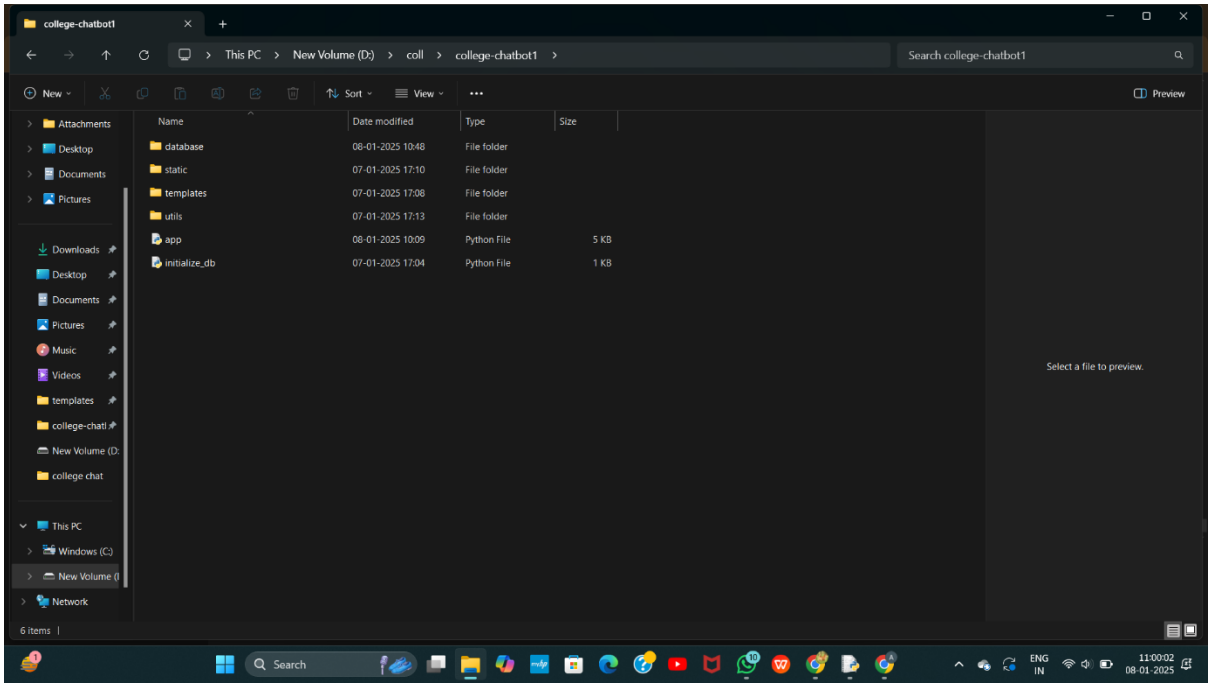

Figure 7.2 Execution Step 2

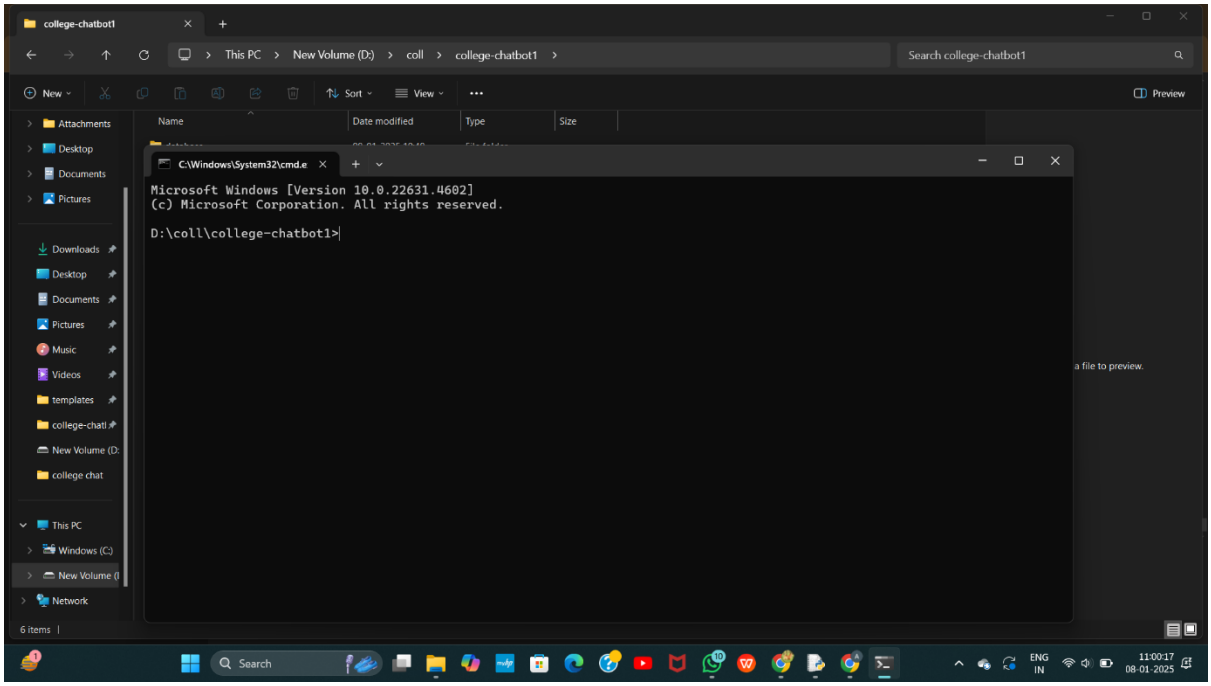

Figure 7.3 Execution Step 3

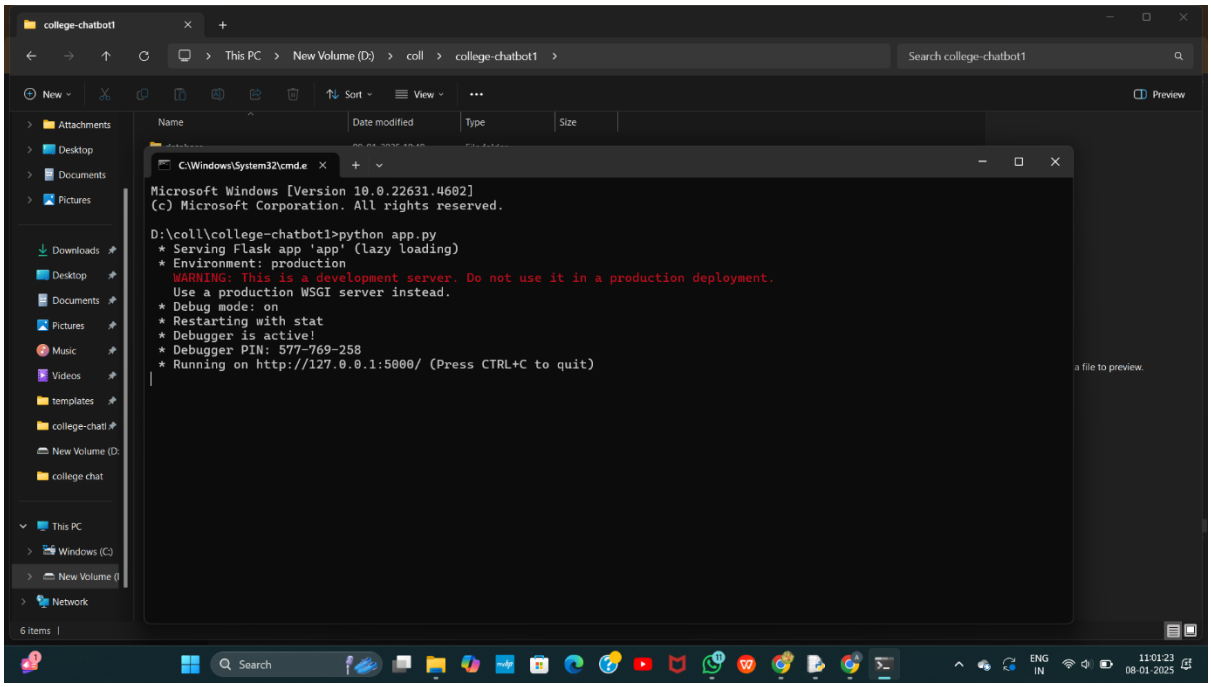

Figure 7.4 Execution Step 4

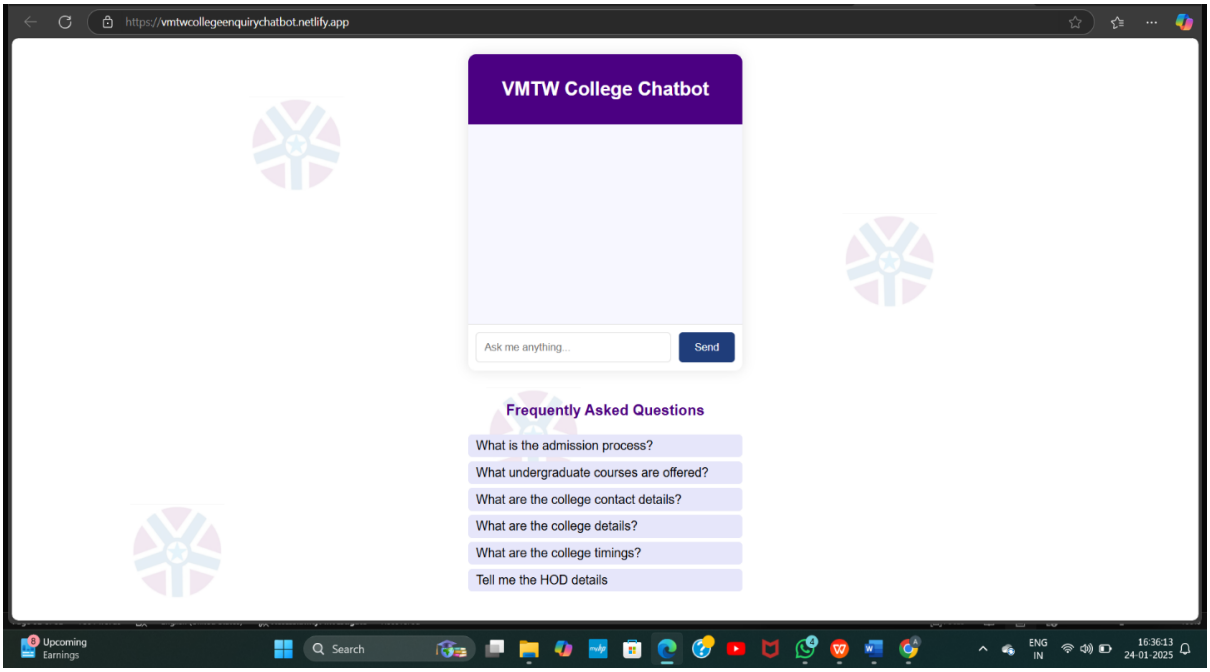

Figure 7.5 Output

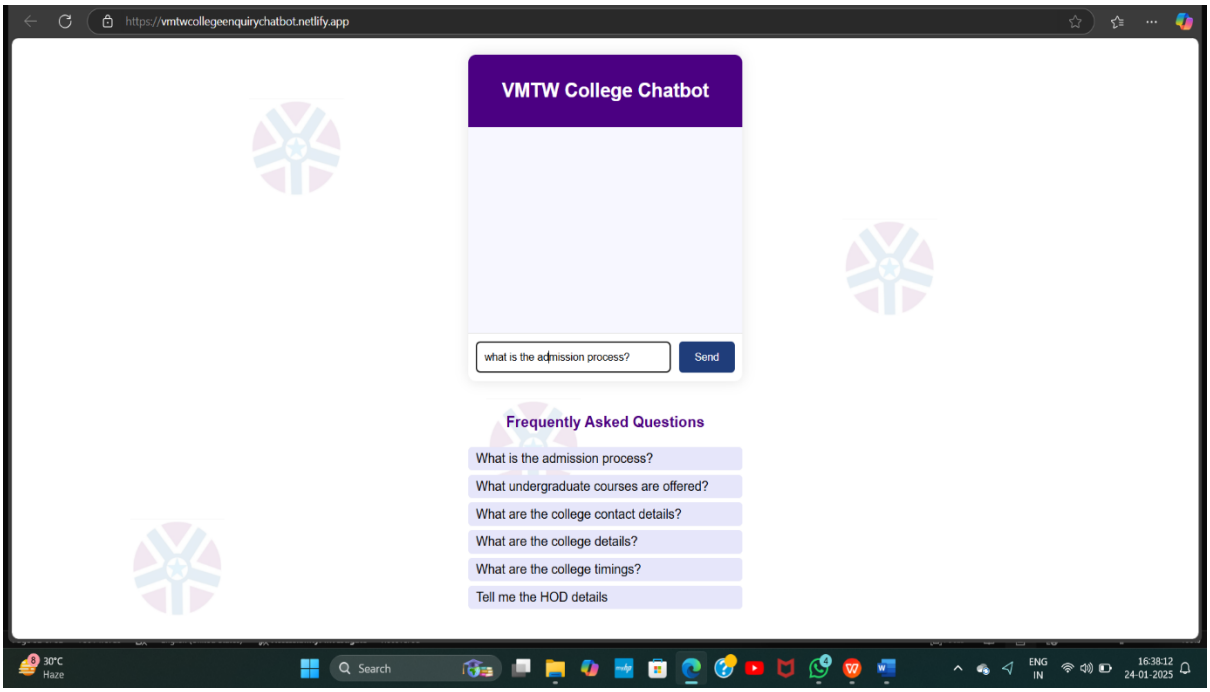

Figure 7.6 Output

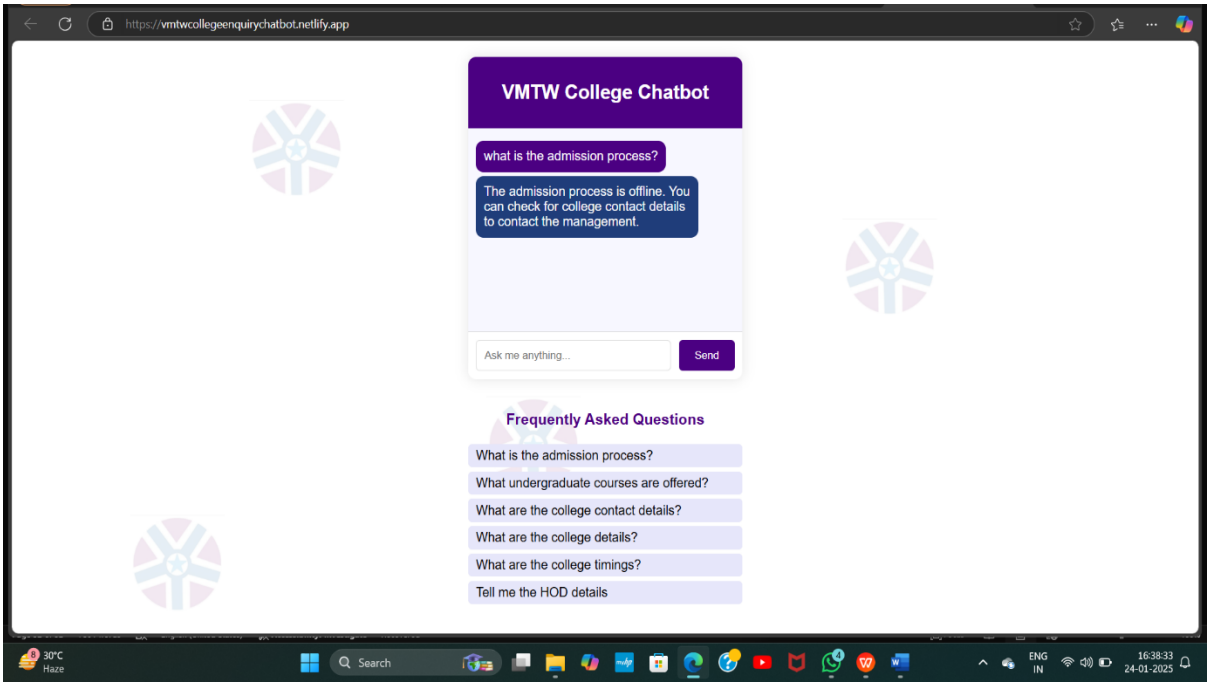

Figure 7.7 Output

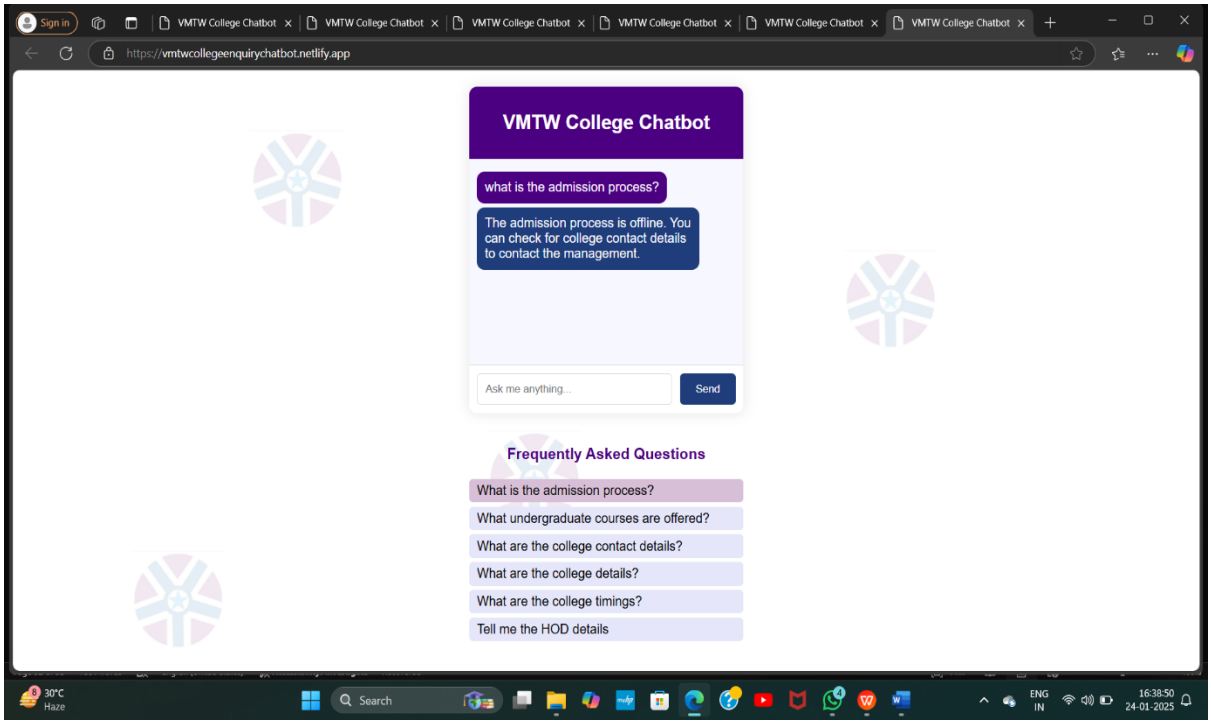

Figure 7.8 Output

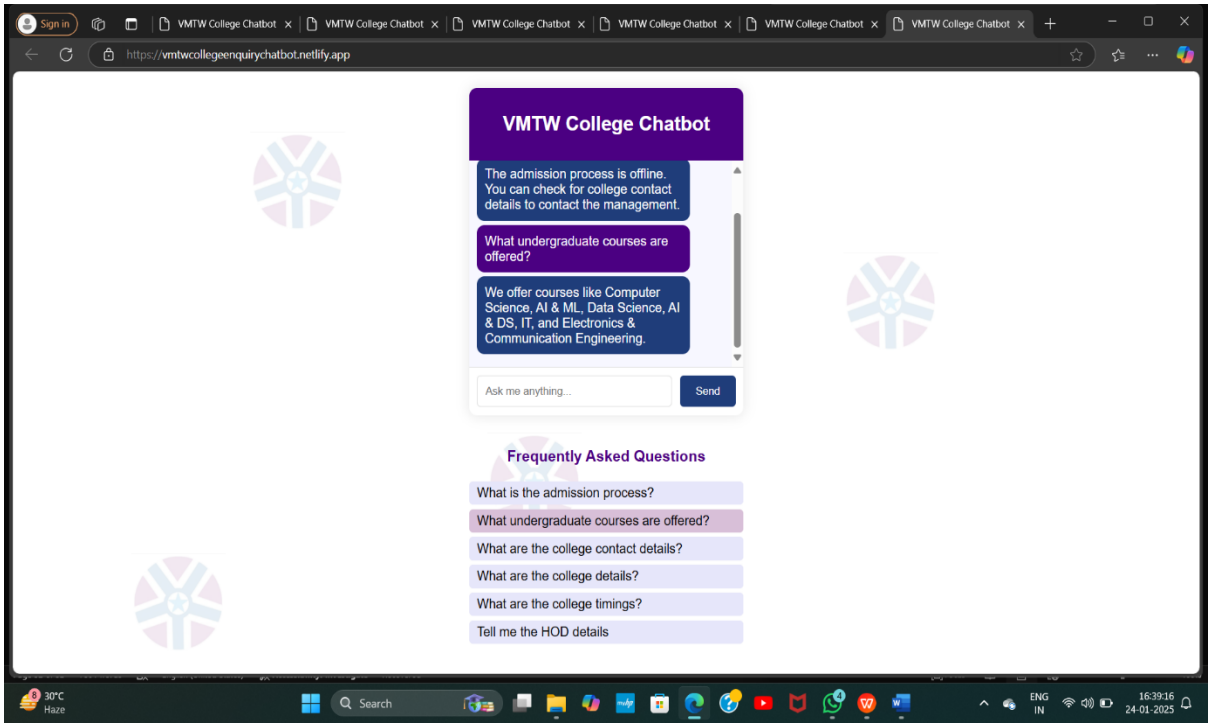

Figure 7.9 Output

CHAPTER 8

8.Conclusion

What we can conclude from the above situation and usage is that greater the database and more the models and use cases, the better is the reaction produced for the client. In any case, the issues are many. For adjusting more uses, the scope changes from the investigation of AI to language examine. We additionally need to recollect that we are taking a shot at a cell phone. The NLP is very broad and maybe utilizing them on a server and isolating this application into customer and server side application can fathom the speed issue as when we do that the speed of the program won't be restricted to the equipment. We live in a time of intelligent technology. Our watches let us know the time, however they likewise remind us to work out. Our telephones prescribe the best places to eat, and our PCs foresee our inclinations, helping us to do our everyday work all the more productively. In conclusion, our implementation of the college enquiry chatbot has demonstrated its potential to streamline the information retrieval process, improve user engagement, and provide efficient support to college-related queries. The successful deployment and positive feedback received from users underscore the value and impact of our chatbot in the educational domain. We are confident that our college enquiry chatbot lays the foundation for future advancements in conversational AI and stands as a testament to the power of technology in enhancing educational services.

CHAPTER 9

9. Future Scope

- Regularly update the knowledge base to reflect any changes or additions to college policies, programs, or resources.
- Enable the chatbot to send proactive notifications or reminders to users regarding important dates, deadlines, or announcements.
- Implement personalized recommendation features that suggest relevant courses, events, or resources based on the user's interests, academic program, or past interactions with the chatbot.
- Explore the integration of voice-based interaction capabilities, allowing users to interact with the chatbot through voice commands.
- Implement mechanisms to gather user feedback and insights on the chatbot's performance and user experience.
- Enhance the chatbot's capabilities by adding support for multiple languages.

CHAPTER 10

Bibliography

10.1 References

1. Ms. Ch. Lavanya Susanna, R. Pratyusha, P. Swathi, P. Rishi Krishna, V. Sai Pradeep, “College Enquiry Chatbot”, International Research Journal of Engineering and Technology(IRJET), e-ISSN: 23950056, p-ISSN: 2395-0072, Volume: 07 Issue: 3 Mar 2020 pp784788.2.

2.Assistant Prof Ram Manoj Sharma, “Chatbot based College Information System”, RESEARCH REVIEW International Journal of Multidisciplinary, ISSN: 2455-3085(Online), Volume-04, Issue03, March-2019, pp 109-112.3.

3.P. Nikhila, G. Jyothi, K. Mounika, Mr. C Kishor Kumar Reddy and Dr. B V Ramana Murthy on , “Chatbots Using Artificial Intelligence”, International Journal of Research and Development, Volume VIII, Issue I, January/2019, ISSN NO:22366124, pp 1-12.4.

4.Payal Jain, “College Enquiry Chatbot Using Iterative Model”, International Journal of Scientific Engineering and Research (IJSER),ISSN (Online): 2347-3878, Volume 7 Issue1, January 2019, pp 80-83.5.

5.Sagar Pawar, Omkar Rane, Ojas Wankhade, Pradnya Mehta, “A Web Based College Enquiry Chatbot with Results”, International Journal of Innovative Research in Science, Engineering and Technology, ISSN(Online): 2319-8753, ISSN (Print): 2347-6710, Vol. 7,Issue 4, April 2018, pp 3874-3880.6.

6.Harsh Pawar , Pranav Prabhu, Ajay Yadav, Vincent Mendonca , Joyce Lemos, “College Enquiry Chatbot Using Knowledge in Database”, International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN: 2321-9653; IC Value:45.98, SJ Impact Factor: 6.887, Volume 6, Issue IV, April 2018, pp 2494- 2496.7.

7.Prof. K .Bala, Mukesh Kumar ,Sayali Hula wale, Sahil Pandita, “Chatbot For College Management System Using A.I”, International Research Journal of Engineering and Technology (IRJET) e ISSN: 2395-0056, p-ISSN: 2395-0072, Volume: 04 Issue: 11 | Nov-2017, pp 2030-2033.8.

10.2 Websites

1.http://vmtwcollegeenquirychatbot.netlify.app/

2.https://www.ibm.com/cloud/learn/natural-language-processing

3.http://iq.opengenus.org/re l u-activation/

4.http://docs.python.org.io/

5.https://keros.io/

6.https://www.nltk.org/

7.https://numpy.org/

8.https://stackoverflow.com/