

ARTIFICIAL INTELLIGENCE BASED CHATBOT FOR STUDENT QUERIES

PMCA698J – Dissertation - 1

Submitted in partial fulfillment of the requirements for the degree of

Master of Computer Applications
in
Department of Computer Applications

by

NAVEEN D

23MCA0285

Under the guidance of

Dr. MYTHILI N

School of Computer Science Engineering and Information Systems

VIT, Vellore.



November, 2024

ARTIFICIAL INTELLIGENCE BASED CHATBOT FOR STUDENT QUERIES

PMCA698J – Dissertation - 1

Submitted in partial fulfillment of the requirements for the degree of

Master of Computer Applications
in
Department of Computer Applications

by

NAVEEN D

23MCA0285

Under the guidance of

Dr. MYTHILI N

School of Computer Science Engineering and Information Systems

VIT, Vellore.



November, 2024

DECLARATION

I hereby declare that the **PMCA698J - Dissertation-1** entitled "**ARTIFICIAL INTELLIGENCE BASED CHATBOT FOR STUDENT QUERIES**" submitted by me, for the award of the degree of Master of Computer Applications in Department of Computer Applications, School of Computer Science Engineering and Information Systems to VIT is a record of bonafide work carried out by me under the supervision of Dr. Mythili N, Associate Professor Grade 2.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date:

Signature of the Candidate

CERTIFICATE

This is to certify that the **PMCA698J - Dissertation-1** entitled “**ARTIFICIAL INTELLIGENCE BASED CHATBOT FOR STUDENT QUERIES**” submitted by **Naveen D (23MCA0285)**, SCORE, VIT, for the award of the degree of **Master of Computer Applications in Department of Computer Applications**, is a record of bonafide work carried out by him / her under my supervision during the period, 15. 07. 2024 to 30.11.2024, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place: Vellore

Date:

Signature of the VIT-SCORE - Guide

Internal Examiner

External Examiner

Head of the Department
Department of Computer Applications

ACKNOWLEDGEMENTS

It is my pleasure to express with a deep sense of gratitude to my **PMCA698J - Dissertation-1** guide **Dr. Mythili N, Associate Professor Grade 2**, School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, for his/her constant guidance, continual encouragement, and understanding; more than all, he/she taught me patience in my endeavor. My association with him/her is not confined to academics only, but it is a great opportunity on my part to work with an intellectual and expert in the field of Natural Language Processing and Web Technology

"I would like to express my heartfelt gratitude to **Dr. G Viswanathan**, Chancellor; **Mr. Sankar Viswanathan**, Vice President; **Dr. Sekar Viswanathan**, Vice President; **Dr. G V Selvam**, Vice President; **Dr. V. S. Kanchana Bhaaskaran**, Vice Chancellor; **Dr. Partha Sharathi Mallick**, Pro-Vice Chancellor; **Dr. Jayabarathi T**, Registrar and **Dr. Sumathy S**, Dean of School of Computer Science Engineering and Information Systems, for providing me with an enriching environment to work in and for their inspirational guidance throughout the tenure of the course.

"In a jubilant mood, I express ingeniously my whole-hearted thanks to **Dr E Vijayan** Professor Grade-1 & Head, Department of Computer Applications, **Dr. Karthikeyan P**, MCA Project Coordinator, **Dr. Srinivasa Koppu**, School Project Coordinator, all teaching staff and members working as limbs of our university for their not-self-centred enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my parents and friends who persuaded and encouraged me to take up and complete this task. Last, but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Vellore

Date:

NAVEEN D

Name of the student

Executive Summary

The aim of the College Enquiry Chatbot is to help students by giving them prompt answers to questions about different facets of college life. This contains details on the college and more. The chatbot analyzing user inquiries using sophisticated machine learning techniques. Users may simply use this system because it is integrated into a web application and is implemented in Python. Through an easy-to-use interface, students may communicate with the chatbot and ask any queries they may have about college-related activities without physically visiting the university. Based on the data it has been educated on, the chatbot can understand a variety of questions and provide precise and pertinent responses. The purpose of the graphical user interface (GUI) is to improve the user experience by simulating a real-world discussion. By using this technology, students can get prompt and effective answers to their questions, making it easier for them to obtain crucial information regarding their academic goals and college experience. In addition to saving time, this enhances accessibility to crucial information that students require in order to make well-informed educational decisions.

Keywords: College Inquiry System, AI Chatbot, Natural Language Processing, Machine Learning, and Automated Communication.

	Page No.
CONTENTS	
Acknowledgement	v
Executive Summary	vi
Table of Contents	vii
List of Figures	ix
List of Tables	xi
Abbreviations	xii
Symbols and Notations	xii
1 INTRODUCTION	1
1.1 Objective	2
1.2 Motivation	3
1.3 Background	4
2 PROJECT DESCRIPTION AND GOALS	5
2.1 Project Description	5
2.2 Goals	7
2.3 Research Gap	8
2.4 Project Statement	8
2.5 Literature Survey	9
3 TECHNICAL SPECIFICATION	13
3.1 Existing System	13
3.2 Proposed System	14
3.3 Module Description	14
3.4 Software Requirements	16

4	DESIGN APPROACH AND DETAILS	19
4.1	System Architecture	19
4.2	Data Flow Diagram	20
4.3	Codes and Standards	21
4.4	Constraints, Alternatives and Tradeoffs	22
5	SCHEDULE, TASKS AND MILESTONES	26
6	PROJECT DEMONSTRATION	28
6.1	Project Configuration	28
6.2	Install the Necessary libraries	28
6.3	Install the MySQL database	29
6.4	Set up Google reCAPTCHA.	30
6.5	In chatbot.py, define the chatbot logic	30
6.6	Launch the Flask application	31
6.7	Testing User Registration and the Chatbot	31
6.8	Tips for Troubleshooting	32
7	RESULT & DISCUSSION	34
7.1	Results	34
7.2	Conclusion	38
7.3	Future Scope	39
8	SUMMARY	40
9	References	41
	Appendix	42

List of Figures

Figure No.	Title	Page No.
3.4.1	Figure 1 VISUAL STUDIO CODE	16
3.4.2	Figure 2 PYTHON	17
3.4.3	Figure 3 DATABASE BROWSER	18
4.1	Figure 4 SYSTEM ARCHITECTURE	19
4.2	Figure 5 DATA FLOW DIAGRAM	20
6.1	Figure 6 TERMINAL	28
6.2	Figure 7 LIBRARIES	29
6.3	Figure 8 DATABASE	29
6.3	Figure 9 UPDATE DB	30
6.4	Figure 10 RECAPTCHA	30
6.6	Figure 11 FLASK APP RUNINNG	31
6.7	Figure 12 USER REGISTER	32
6.7	Figure 13 LOGIN VALIDATION	32
6.7	Figure 14 DB & LOGOUT	33
7.1	Figure 15 REGISTER	34
7.1	Figure 16 LOGIN	34
7.1	Figure 17 FORGOT PASSWORD	35

7.1	Figure 18	HOMEPAGE	35
7.1	Figure 19	ABOUT US	36
7.1	Figure 20	CHATTING WITH BOT	36
7.1	Figure 21	QUERY CHAT	37
7.1	Figure 22	USER REQUEST	37

List of Tables

Table No.	Title	Page No.
5.1	SCHEDULE	26

List of Abbreviations

AI	Artificial Intelligence
NLP	Natural Language Processing
VTOP	VIT Technical Online Portal
UI	User Interface
IDE	Integrated Development Environment
API	Application Programming Interface
SQL	Structured Query Language
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
WCAG	Web Content Accessibility Guidelines
OWASP	Open Web Application Security Project
UX	User Experience
GDPR	General Data Protection Regulation
RECAPTCHA	A CAPTCHA system that distinguishes humans from bots
BERT	Bidirectional Encoder Representations from Transformers
AJAX	Asynchronous JavaScript and XML

Symbols and Notations

- Implies or directs**
- > Arrow notation used in functions**
- % Modulus operator for division remainders**

1. INTRODUCTION

This college inquiry chatbot's main goal is to expedite the admissions process by automating answers to frequently asked questions. By reducing their workload, admissions staff members may concentrate on more intricate and individualized interactions. The chatbot can guarantee that students receive timely assistance by efficiently managing common inquiries, such as those about program specifics, application deadlines, and campus amenities. This will enhance the students' overall experience and happiness. This project also places a strong emphasis on accessibility and inclusivity. Given that potential students may have different backgrounds and differing degrees of technological expertise, the chatbot will be made to be as user-friendly as possible. A global audience will be served by features like multilingual support and an easy-to-use interface, guaranteeing that information is available to all users, including those from other countries.

In the quickly changing world of higher education, it can be quite difficult for potential students to find timely and reliable information on colleges and universities. With so many alternatives, students must navigate websites, pamphlets, and various communication channels, which can cause them to get frustrated and confused. Conventional communication channels, such email and phone calls, usually produce inconsistent information and delayed responses. In addition to impeding the hiring process, this divergence lowers the overall experience for students who want direction and clarity when making decisions. The creation of an intelligent college inquiry chatbot is the suggested solution to these problems in this capstone project. Using natural language processing (NLP) and artificial intelligence (AI) technology, this creative solution gives potential students prompt, thorough, and correct answers to their inquiries. By acting as a virtual assistant and having the ability to have natural, conversational conversations, the chatbot will make it easier for consumers to obtain information. Colleges can improve their communication tactics by putting such a system into place, making them more effective and user-friendly.

They study will be done to determine the most frequent queries and worries of potential students in order to construct this chatbot. To build a knowledge base that represents the institution's programs and beliefs, this research will examine input from prospective applicants, admissions personnel, and current students. Additionally, an engaging and user-friendly interface will be created using user experience (UX) design principles, guaranteeing that users can quickly explore the chatbot and locate the information they need.

A major improvement in the way educational institutions communicate with potential students has been made with the development of an AI-powered college inquiry chatbot. Colleges that embrace innovation and technology can increase communication effectiveness while also creating a more encouraging and educational atmosphere for students.

1.1. OBJECTIVE

The goal of the VTOP Bot project is to develop a powerful chatbot driven by AI that improves the VIT community's ability to acquire and use information. By giving customers access to a responsive, on-demand virtual assistant, this project seeks to expedite the process of responding to common questions about academic programs, administrative services, and campus events. The VTOP Bot is made to understand and react to user inquiries precisely by using machine learning and natural language processing, guaranteeing that pertinent information is always accessible.

By providing a user-friendly platform that enables visitors, parents, teachers, and students to readily access important information at any time and from any location, one of the main goals is to enhance the user experience. In addition to meeting the requirements of the community, this immediate access lightens the administrative staff's workload, freeing them up to work on more difficult or specialized projects. In order to preserve the integrity of the chatbot's environment, the project also aims to encourage safe, user-verified interactions by integrating Google reCAPTCHA.

Also, by using an integrated suggestion box to gather user feedback, the VTOP Bot seeks to continuously adapt and develop. Because of this characteristic, the bot can improve its responses over time to better suit shifting demands and expectations. By creating a smooth communication platform that satisfies the many needs of its users, the VTOP Bot project ultimately aims to provide a dependable, effective, and safe chatbot solution that serves VIT's administrative and educational objectives.

1.2 MOTIVATION

The essential demand for quick, accurate, and easily accessible knowledge in educational institutions is what drove the creation of the VTOP Bot. Students, instructors, and other stakeholders in a dynamic college setting often need prompt answers to questions about academic timetables, exam procedures, administrative services, and campus events. However, due to excessive demand, traditional support channels—such as help desks and office inquiries—frequently endure delays and provide a less acceptable user experience. By offering a virtual assistant that can handle routine tasks on its own, VTOP Bot aims to close this gap and free up administrative staff members to concentrate on more intricate and individualized support.

The increasing demand for digital interaction and real-time solutions from users who are becoming more and more used to utilizing AI-based technologies in their daily lives is another factor propelling this endeavor. Using an AI-driven chatbot is in line with the changing demands and tastes of a tech-savvy audience as technology continues to permeate education. This contemporary change is reflected in the VTOP Bot initiative, which seeks to encourage a self-service culture and enable users to find solutions on their own. The chatbot promotes productivity and convenience by giving users a tool that is easily available at all times, allowing them to access necessary resources without being constrained by office hours.

A powerful motivator is the VTOP Bot's continuous learning feature, which allows the chatbot to adapt based on user feedback. This feedback loop encourages the bot's flexibility and is consistent with the organization's dedication to continuous innovation and development. The creation of VTOP Bot is an example of a forward-thinking strategy that can establish a new benchmark for how educational institutions engage and support their communities, in addition to showcasing the revolutionary potential of AI in educational settings. The ultimate goal of the VTOP Bot is to serve as an example of how to employ technology in an educational setting to improve user experience, increase engagement, and expedite processes.

1.3 BACKGROUND

The possibility of chatbots to revolutionize the way educational institutions serve their communities is what motivated the VTOP Bot initiative. Educational chatbots have advanced in sophistication in recent years, utilizing machine learning and natural language processing (NLP) to evaluate user input and provide precise answers to a variety of queries. These developments in artificial intelligence (AI) technology enable chatbots to deliver timely, pertinent information to users, simulating human contact and providing a round-the-clock level of service. Notably, chatbots like Chatterbot, Google Dialogflow, and IBM's Watson have established benchmarks for AI-powered support, and numerous colleges are implementing comparable programs to improve employee and student engagement.

These advancements, the VTOP Bot incorporates an NLP-based chatbot into a Flask web framework and makes it available via an intuitive web interface. The VTOP Bot was created with Python's Chatterbot framework and leverages an organized database-driven methodology to efficiently handle and retrieve data. Google reCAPTCHA adds even more functionality to the system by guaranteeing safe user interactions and guarding against unwanted access to the platform. With the use of a suggestion box feature to collect user input, the design enables continuous learning and improvement, enabling the bot to develop and better serve customer demands over time.

The enhance communication, accessibility, and operational efficiency, educational institutions are increasingly implementing AI solutions, as seen by the project's background. By creating a chatbot specifically for the VIT community, this project hopes to further the institution's information-sharing objectives while simultaneously advancing a larger vision of incorporating cutting-edge AI technologies into academia. With its thoughtful AI integration and ongoing user feedback learning, the VTOP Bot is positioned as a useful tool that fosters a more engaging and dynamic learning environment.

Due to the quick development of artificial intelligence, chatbots have become useful resources in a number of industries, such as customer service, healthcare, and education. It gets harder to handle a large number of questions and deliver timely information as educational institutions expand and diversify. Conventional support techniques, such as phone support, email correspondence, and in-person help desks, are frequently resource-intensive and might not adequately meet the demands of a population that is more tech-savvy and demands immediate access to information.

2. PROJECT DESCRIPTION & GOALS

2.1 PROJECT DESCRIPTION

The VTOP Bot project is to develop an AI-powered chatbot that can be customized to meet the various informational requirements of the VIT community. This community consists of visitors, parents, teachers, and students who frequently require easy access to trustworthy information on a range of administrative and academic subjects. The VTOP Bot is a centralized, easily available resource that offers consumers real-time replies as an effective substitute for more conventional approaches like emails, phone calls, or in-person questions. In addition to meeting the institution's information demands by automating these interactions, VTOP Bot improves user experience by providing a seamless, on-demand service.

Users may easily access information at any time and from any location with this chatbot's user-friendly web interface, which is compatible with a variety of devices. Because of its intuitive design, even inexperienced users may easily use the interface. Based on the Chatterbot library and incorporated into the Flask framework, VTOP Bot interprets user inputs and produces precise, contextually relevant responses by utilizing the advantages of Python-based natural language processing (NLP) techniques. With the help of this functionality, users can obtain solutions to their inquiries without requiring direct human support, turning the bot into a potent self-service tool.

The database-driven architecture that powers the chatbot is essential to its rapid and precise response times. The bot can obtain information very instantly since frequently asked questions and their accompanying answers are kept in a structured database. This methodical approach to data storage guarantees that the chatbot can manage a large number of inquiries without sacrificing the caliber of its responses. In addition to being effective, the database-driven backend makes it simple to update when new data becomes available, enabling the chatbot to stay up to speed with institutional changes.

The VTOP Bot's design places a high priority on security. The bot uses Google reCAPTCHA, a well-known and reliable method for user authentication and preventing automated bot misuse, to safeguard user data and stop unwanted access. The platform's security is improved by this connection, resulting in a dependable and secure environment where users can freely engage with the bot without worrying about data privacy. The reCAPTCHA tool prevents abuse by ensuring that the chatbot is only available to legitimate users by confirming their lawful access.

The VTOP Bot is made to constantly develop and grow. Every user contact gives the bot a chance to grow and adapt, making sure it stays applicable and helpful over time. In an academic setting where information is continuously changing—from course updates and exam schedules to administrative processes and event announcements—this flexibility is essential. Even when institutional information changes, the bot's capacity to improve itself in response to criticism guarantees that it will continue to be a useful tool.

The creation of VTOP Bot is an important step in the VIT community's digital revolution. The bot reduces administrative staff workload by automating information retrieval and question resolution, freeing them up to concentrate on more intricate, customized tasks. Also, the VTOP Bot project provides as an example of how AI and machine learning can be used in educational environments. The bot's ability to accurately and efficiently evaluate and comprehend user inquiries through the use of natural language processing demonstrates the revolutionary potential of these technologies in educational settings. This project demonstrates VIT's dedication to implementing state-of-the-art technologies.

Because VTOP Bot is scalable, it may be readily extended to cover other subjects and resources as the institution's information requirements change. Because of its modular nature, the chatbot may be easily updated to accommodate any future additions or modifications. As user expectations rise and the range of information needed increases, this scalability guarantees that the bot can continue to provide the VIT community with efficient services.

The aim of the VTOP Bot project is to make the VIT community more involved, informed, and connected. The bot encourages a self-service culture by providing users with a dependable and easily available information-gathering tool, enabling people to solve problems independently. VTOP Bot is a priceless addition to VIT's digital environment since it fosters a sense of independence and motivates users to utilize all of the tools at their disposal.

2.2 GOALS

Reliable Information Source: The chatbot is a trustworthy source of information for kids, teachers, parents, and guests since it is built to deliver precise and prompt answers to frequently asked questions. The VTOP Bot guarantees that users can obtain trustworthy information rapidly by centralizing frequently requested inquiries.

Improved User Engagement: The goal of VTOP Bot is to develop an interactive, user-friendly platform that motivates users to interact and communicate with VIT's resources more successfully. A more individualized experience that satisfies the demands of a tech-savvy audience is provided by the AI-driven interface, which adjusts to user needs.

Decreased Administrative Workload: VTOP Bot assists VIT's administrative personnel in reducing their workload by automating answers to common inquiries, freeing them up to concentrate on more intricate and individualized assistance requirements. A more effective distribution of resources throughout the organization is made possible by this feature.

Safe Interactions: One of the project's main objectives is security. The VTOP Bot creates a safe and secure environment for information exchange by integrating Google reCAPTCHA to guarantee that all user interactions are shielded from unwanted access.

Constant Improvement: VTOP Bot has a feedback feature that lets users report problems with the chatbot's responses or make suggestions for enhancements. This feedback loop facilitates ongoing development by enabling the bot to adjust in response to actual interactions and gradually increase its accuracy.

Promotion of Technological Innovation in Education: VTOP Bot demonstrates the possibilities of digital innovation in education through the use of AI and machine learning tools. By improving operational effectiveness and user experience in the academic setting, the initiative establishes VIT as a progressive organization in the use of AI.

2.3 RESEARCH GAP

Improving natural language understanding (NLU) and user personalization represent a major research gap for the VTOP Bot project. At the moment, the system divides users into four major groups: kids, parents, teachers, and guests. Within each group, it provides answers specific to common questions. However, using information on past searches, individual preferences, or particular academic requirements could result in a more tailored user experience. A more immediate and customized engagement could be made possible by predictive capabilities, in which the bot proactively recommends pertinent resources depending on user behavior.

The bot's capacity to react to difficult or multi-layered inquiries is limited by the existing implementation, which depends on logic adapters with predefined answers. Advanced natural language processing (NLP) methods, including transformer-based language models, could greatly enhance the bot's comprehension of complex inquiries. This development would lessen the need for rule-based limitations by improving its capacity to interpret complicated requests, context, and a variety of phrase patterns.

Furthermore, by incorporating sentiment analysis, the bot might be able to determine human moods and modify its responses appropriately. For example, the bot might react empathetically and offer more resources for support if a user indicates irritation. This would enhance the bot's emotional intelligence and make the interaction more interesting and intuitive. Filling up these research gaps could significantly improve the VTOP Bot's usefulness, user experience, and general efficacy in responding to a range of university-related questions.

2.4 PROJECT STATEMENT

The VTOP Bot Project intends to create an automated web-based chatbot system that allows VIT stakeholders, such as students, staff, parents, and visitors, to obtain college-related information in an efficient and user-friendly manner. The system will make it easier to access information on admissions, academic resources, student services, and administrative processes by providing quick responses via a conversational interface. The platform features secure user authentication, registration, and login capability, as well as Google reCAPTCHA for added security. The project will use a rule-based chatbot to offer accurate, classified responses to frequent requests, eliminating the need for direct human support and increasing the accessibility of critical information. This project also lays the groundwork for future additions, such as dynamic learning capabilities, that will better suit the changing needs of VIT's users.

2.5 LITERATURE SURVEY

[1]. Singh (2022) talks about utilizing Python and Natural Language Processing (NLP) approaches to create a healthcare chatbot system. The literature review emphasizes how crucial chatbots are becoming to the healthcare industry in order to give patients accurate and timely information. Chatbots have the ability to increase patient involvement, lessen the workload for medical staff, and increase the accessibility of healthcare services, according to earlier research. The article also discusses how different NLP approaches, like sentiment analysis and machine learning algorithms, might be used to increase chatbots' comprehension and effectiveness in comprehending and reacting to user inquiries. Furthermore, the research highlights the difficulties in guaranteeing the confidentiality and security of patient data as well as the necessity of ongoing training and updates to enhance chatbot performance.

[2]. Ebsen (2024) investigates the creation of a chatbot for customer support that makes use of Python, machine learning, and artificial intelligence (AI) technology. The literature study highlights the growing use of AI-powered chatbots in customer care, emphasizing how well they can handle common questions, offer immediate assistance, and raise customer happiness. Prior studies highlight how effective chatbots are at speeding up responses, cutting down on operating expenses, and improving user experience. The review also covers how machine learning algorithms let chatbots learn from contacts, improve their responses, and more precisely anticipate client demands. Among the difficulties noted in the literature are the requirement for advanced natural language processing (NLP) methods to manage a variety of intricate and varied queries, guaranteeing the dependability and consistency of chatbot responses, and resolving privacy issues pertaining to client information. The review emphasizes how artificial intelligence is transforming customer service across businesses.

[3]. The development of "Aisha," a proprietary AI library chatbot made with the ChatGPT API, is the main focus of Lappalainen and Narayanan (2023). The emerging trend of using AI chatbots in library settings to enhance user engagement, expedite information retrieval, and provide individualized support is highlighted in the research study. Prior research covered in the paper highlights how well AI-powered chatbots can respond to user inquiries, suggest resources, and improve the user experience in digital libraries as a whole. The evaluation also discusses the benefits of using the ChatGPT API, pointing out that it can comprehend and produce responses that

are similar to those of a human, which makes interactions more intuitive and natural. providing a format for a literature review. Managing the correctness and applicability of responses, maintaining data security and privacy, and the requirement for continual training to adjust to the changing demands of library patrons are among the difficulties noted. The research demonstrates how AI chatbots have the potential to transform library services by providing effective and scalable assistance.

[4]. The creation of an AI-based medical chatbot intended to forecast infectious diseases is examined by Chakraborty et al. (2022). The growing use of AI in healthcare, especially in the early diagnosis and treatment of infectious diseases, is highlighted in the literature review. Prior studies covered in the paper highlight how AI-powered chatbots can help with quick initial diagnosis, medical guidance, and alleviating the burden on healthcare systems. The research also examines a number of AI and machine learning methods, including deep learning and natural language processing (NLP), that improve the chatbot's capacity to accurately assess symptoms and forecast possible illnesses. It also highlights issues with data quality, the difficulty of predicting diseases, and the requirement for thorough validation of AI models to guarantee their dependability.

[5]. Advanced Natural Language Processing (NLP) models for technical university information chatbots are developed and compared by Attigeri, Agrawal, and Kolekar (2024). The literature study emphasizes the increasing importance of chatbots in learning environments, especially when it comes to giving staff, instructors, and students quick access to reliable information. NLP plays a key role in helping chatbots comprehend and reply to intricate questions about administrative procedures, course information, and university services, according to earlier research. Transformer-based models like BERT and GPT, which have been demonstrated to improve the chatbot's capacity to process and produce contextually relevant responses, are among the NLP techniques included in the paper. Managing domain-specific language, making sure the chatbot can adjust to a variety of user requests, and preserving the quality and dependability of information are some of the other issues it tackles.

[6]. The development of the "CSM" chatbot, intended to handle student concerns regarding payments and university enrollment, is examined by Parrales-Bravo et al. (2024). The growing usage of AI-powered chatbots in higher education to boost administrative effectiveness and increase student support services is highlighted in the literature study. Prior studies demonstrate how well chatbots perform to automate answers to often requested inquiries, which lessens the strain for university employees and gives students timely support. The evaluation covers a number of artificial intelligence (AI) and natural language processing (NLP) strategies that are utilized to guarantee the chatbot's accuracy when responding to questions about financial transactions and enrollment processes. Among the difficulties noted include maintaining data security and privacy, particularly when managing private student data, and the requirement for frequent updates to keep the chatbot up to current with evolving university regulations.

[7]. A thorough analysis of chatbots is provided by Benaddi et al. (2024), who look at their classification, evolution, and impacts on the travel and tourist sector. The assessment follows chatbots' development from simple rule-based systems to complex AI-powered platforms. It talks about the technology enabling the development of chatbots, such as machine learning and natural language processing (NLP), and divides them into sorts including informational, transactional, and conversational. The evaluation emphasizes the important role chatbots play in the travel industry, pointing out how they enhance customer experience by offering real-time responses, tailored suggestions, and booking assistance. There is also discussion of difficulties like merging with current tourism systems and preserving accurate and contextually relevant interactions. The revolutionary potential of chatbots to improve customer interactions' effectiveness and raise user satisfaction levels in the travel sector.

[8]. The construction of a deep learning-based chatbot for GRIET (Gokaraju Rangaraju Institute of Engineering and Technology) is examined by Deepa, Thumati, and Reyya (2022), with an emphasis on how sophisticated deep learning approaches can enhance chatbot performance. Their analysis emphasizes the application of models such as transformers and neural networks, which improve the chatbot's of responses that resemble those of a human. Deep learning provides more accurate, context-aware interactions and superior processing of complex queries as compared to conventional rule-based systems the paper also discusses difficulties such the requirement for large amounts of training data, substantial computer power, and guaranteeing flexibility in response to different user inputs. All things considered, the study emphasizes how deep learning can greatly increase chatbots' efficacy and efficiency in educational contexts.

[9]. The development of an autonomous AI chatbot therapy to help patients with sleeplessness is examined by Shaikh and Mhetre (2022). Their assessment of the literature emphasizes how AI and chatbots are being used more and more in mental health treatment, especially to treat sleep disorders. According to the review, chatbots can provide individualized guidance, therapeutic interventions, and patient progress tracking. It looks at a number of AI methods that assist chatbots in comprehending and successfully addressing patient problems, such as machine learning algorithms and natural language processing (NLP). Additionally covered are issues like guaranteeing clinically accurate answers, offering advice based on evidence, and preserving user interest. The review's overall findings highlight how AI chatbots can help patients with insomnia in a way that is affordable, scalable, and easily accessible, increasing treatment results overall and supplementing conventional therapies.

[10]. In their 2023 study, Assayed, Alkhatib, and Shaalan investigate the creation of an AI-powered chatbot intended to advance equity in high school guidance. The expanding usage of AI chatbots in education to increase accessibility and equity is highlighted in the literature study. It highlights how these chatbots ensure that students from a variety of backgrounds have fair access to academic resources by providing reliable, objective information and assistance. The use of artificial intelligence (AI) and natural language processing (NLP) to deliver unbiased, individualized advice is covered in the paper. Making sure responses are relevant and sensitive to cultural differences is one of the challenges. The review highlights the potential of AI chatbots to improve assistance and equity in educational advising overall.

3. TECHNICAL SPECIFICATION

3.1 EXISTING SYSTEM

A web-based tool called the VTOP Bot system was created to make information access easier for all VIT users, including visitors, parents, teachers, and students. With Google reCAPTCHA integrated to prevent unwanted access, the Flask-built system offers a secure environment for user registration and login. A MySQL database is used to manage user data, guaranteeing dependable and well-organized storage. The ChatterBot-powered chatbot can answer often requested questions about academic resources, college procedures, and campus information because it is set up with predefined responses. A clear, responsive interface made using HTML and Bootstrap allows users to engage with the bot, making it easier to navigate and compatible with a variety of devices. By classifying responses according to user type—for example, faculty or students—this system effectively meets the demands of certain users, offering a targeted, beneficial experience. The chatbot could be improved with more sophisticated natural language processing skills to better comprehend and reply to a variety of user inquiries in the future, even though it is now restricted to pre-set responses.

The existing College Enquiry Chatbot system is a feature-rich web tool designed to make it easier for visitors, parents, teachers, and students to obtain information. It was created using Flask and Python and makes use of the chatterbot library to respond to commonly requested inquiries on college-related topics like admissions, academic programs, exam procedures, and campus resources. With responses customized for each user group, users can browse through a variety of topics, including teacher support, extracurricular activities, student curriculum, and placement services. In order to handle user registration, login, and password recovery, the system integrates a MySQL database, guaranteeing safe access via Google ReCaptcha. A responsive interface that is both aesthetically pleasing and accessible is achieved through the usage of HTML and Bootstrap. The chatbot's front-end interface is made contemporary and easy to use with HTML, CSS, and Bootstrap. The chatbot is simple to use, and users can register, log in, and recover forgotten passwords with ease. With the help of AJAX, the chatbot can instantly respond to user inquiries without requiring page reloads, improving user experience by fostering a smoother, more dynamic dialogue flow.

3.2 PROPOSED SYSTEM

The proposed College Enquiry Chatbot system offers improvements to increase response, security, and usability. More customer satisfaction will result from the chatbot's improved comprehension of intricate, multi-part queries and its ability to respond with more accurate answers by utilizing sophisticated natural language processing models like BERT or GPT. The bot will save user interaction history in order to provide a more customized experience, providing customized answers for frequent users. Data encryption and two-factor authentication will also be used to bolster security measures and safeguard private user data.

Administrators will be able to effortlessly update the bot's knowledge base thanks to the system's dynamic content management functionality, which will guarantee information accuracy as college schedules and policies change. Reliability will also be improved by the integration of a human escalation mechanism that allows users to contact staff with unanswered questions. By monitoring user interactions and detecting response gaps, analytics and feedback gathering will enable ongoing improvement. High traffic levels will be handled by the chatbot with ease thanks to load balancing and scalable infrastructure. Last but not least, multilingual support will promote inclusivity by making the bot available to a larger audience, including users from other countries. With these enhancements, a very effective, safe, and intuitive chatbot for college inquiries will be produced.

3.3 MODULE DESCRIPTION

3.3.1 User Authentication and Registration:

This module is in charge of safely handling user accounts, including login, logout, and registration. For further protection, it integrates Google reCAPTCHA, which guards against illegal access and guarantees authentic interactions. Users must submit necessary information during the registration process, and it is safely saved in a database. This module makes sure that the bot's full functionality is only accessible to users who have been verified.

3.3.2 Chatbot Interaction:

This module serves as the primary interface for users to engage with the bot. It employs the ChatterBot library, allowing the VTOP Bot to answer to predetermined inquiries.

The chatbot is built with logic adapters for speedy response selection, and it provides organized recommendations based on common user queries about university information like admissions, academic resources, and services. The bot provides users with options according to their role, increasing query specificity.

3.3.3 Query Categorization:

This module categorizes user inquiries into predetermined sections (students, faculty, parents, and guests), allowing the bot to respond with information tailored to the user's role and requirements. Each category contains sub-options (such as curricular information, extracurricular activities, administrative support, and so on), allowing visitors to navigate to relevant information more easily. This module guarantees that users get appropriate and role-specific responses quickly.

3.3.4 Database Management:

The database module stores and retrieves user data, session information, and query logs. It maintains data consistency and provides easy access to frequently sought information. The module is essential for controlling user authentication, storing user accounts, and keeping track of user interactions with the bot. This data allows the bot to give a consistent experience and supports future improvements such as customisation.

3.3.5 Sentiment Analysis and Response Enhancement:

This is planned for future development, would provide sentiment analysis capabilities, allowing the bot to identify the user's mood and alter its responses accordingly. This could improve the user experience, particularly in instances where consumers may be frustrated or want further support. This module tries to improve the bot's sympathetic and contextually aware answers by incorporating advanced NLP.

3.4 SOFTWARE REQUIREMENTS

3.4.1 VISUAL STUDIO

Microsoft created Visual Studio as an Integrated Development Environment (IDE) for developing desktop programs, GUIs (Graphical User Interfaces), consoles, web applications, mobile applications, cloud and online services, and other projects. This IDE allows you to build both managed and native code. It makes use of a variety of Microsoft software development platforms, including Windows Store, Microsoft Silverlight, and Windows API. It is not a language-specific IDE; you may use it to write code in C#, C++, VB (Visual Basic), Python, JavaScript, and many other languages. It supports 36 distinct programming languages. It is accessible for both Windows and macOS.

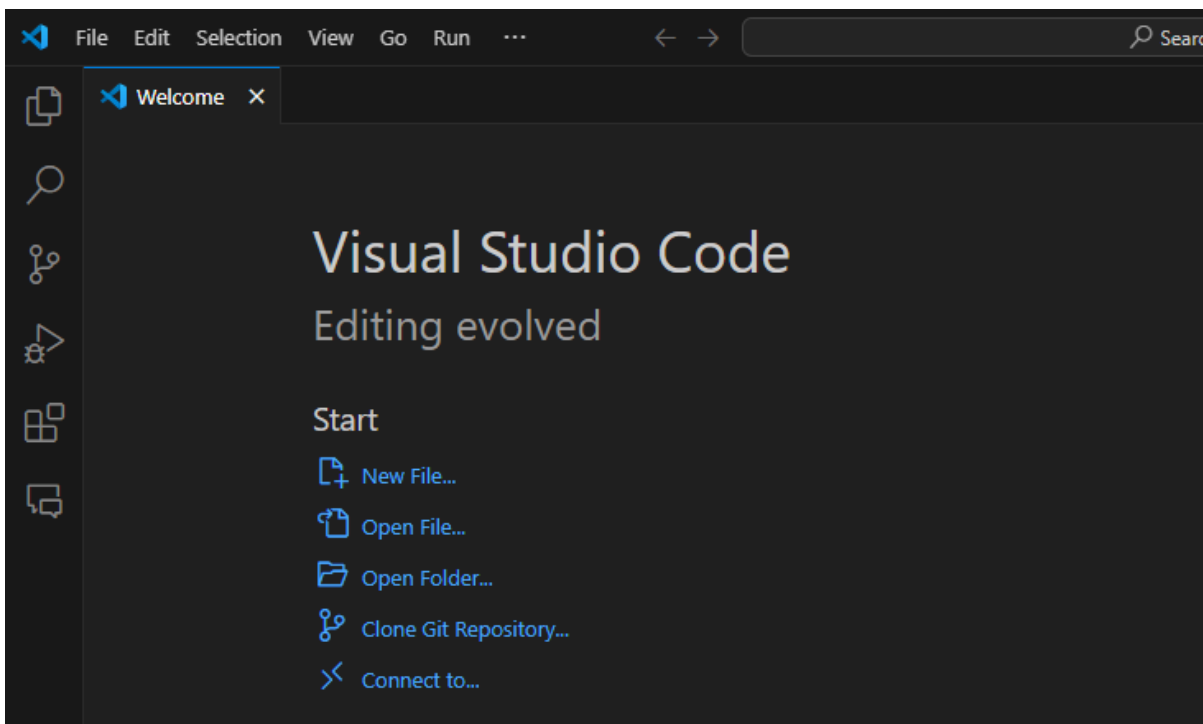


Figure 1. VISUAL STUDIO CODE

Visual Studio IDE is a full-featured programming environment that works with several operating systems, the web, and the cloud. Users can easily navigate the interface, allowing them to write code quickly and precisely.

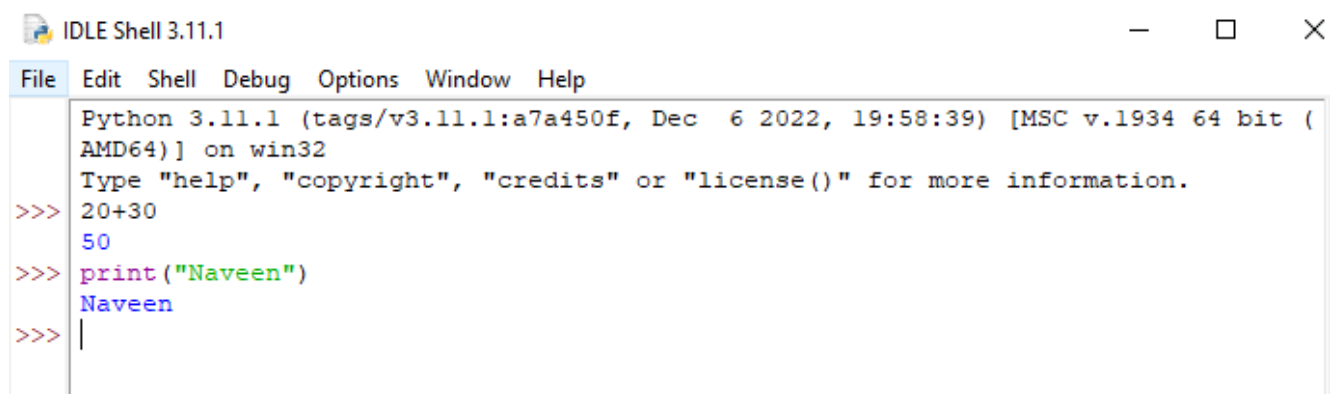
Visual Studio has a sophisticated debugger that helps developers quickly spot code errors. Developers can host their application on the server with confidence now that any potential performance issues have been resolved.

Visual Studio users can get live coding assistance regardless of the programming language they use. The Platform has an autocomplete option to help you develop faster. The built-in intelligent system offers API descriptions and recommendations. The Visual Studio IDE, you may easily collaborate with your colleagues on the same project. This IDE allows developers to share, push, and pull their code with their coworkers.

Visual Studio allows all users to personalize it. They can choose to add features based on their needs. They can, for example, download and install add-ons and extensions in their integrated development environment. Even programmers are allowed to contribute their own extensions.

3.4.2 PYTHON

Python, created by Guido van Rossum and first released in 1991, is a high-level, versatile programming language known for its readability, simplicity, and wide range of applications. Its design emphasizes code readability with its clear syntax and use of indentation, making it a popular choice for beginners and experienced developers alike. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.



```
IDLE Shell 3.11.1
File Edit Shell Debug Options Window Help
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec 6 2022, 19:58:39) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 20+30
50
>>> print("Naveen")
Naveen
>>> |
```

Figure 2. PYTHON

3.4.3 DATABASE BROWSER

A database is a structured collection of data that has been organized and stored in a way that makes it easy to access, manage, and update. Databases are fundamental to modern technology, acting as the backbone for everything from small applications to big enterprise systems. They assist in organizing information such as customer records, inventory data, and transaction logs, allowing for efficient retrieval, updating, and analysis as needed.

A database is a software program that allows you to manage and interact with database contents through an intuitive, graphical interface. These tools are intended to simplify difficult database activities by allowing users to view, update, and query data without requiring substantial SQL knowledge. A database browser allows users to easily view and change tables, create and delete records, and manage the general layout of the database. Common capabilities include query editors for retrieving specific data, schema visualization for understanding table structures and linkages, and data export/import tools for moving data between systems. Many database browsers provide features for managing user permissions, which ensures that data access is secure and regulated. Popular database browsers such as DB Browser for SQLite, phpMyAdmin, and pgAdmin make database management easier and more efficient, benefiting both developers and database administrators.

DB Browser for SQLite - C:\Users\Lenovo\OneDrive\Desktop\capstone-projects\CRCE Bot\db.sqlite

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Undo Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: statement Filter in any column

	id	text	search_text	conversation	created_at
	Filter	Filter	Filter	Filter	Filter
1	1	Hi	hi	training	2024-11-09 12:19:1
2	2	Helloo!	helloo !	training	2024-11-09 12:19:1
3	3	Hey	hey	training	2024-11-09 12:19:1
4	4	How are you?	ADV:be VERB:-pron-	training	2024-11-09 12:19:1
5	5	I'm good.</br> Go ahead and ...	VERB:ahead ADV:write VERB:number ...	training	2024-11-09 12:19:1
6	6	Great	great	training	2024-11-09 12:19:1
7	7	Go ahead and write the number of an...	ADV:write VERB:number NOUN:query ...	training	2024-11-09 12:19:1
8	8	good	good	training	2024-11-09 12:19:1
9	9	Go ahead and write the number of an...	ADV:write VERB:number NOUN:query ...	training	2024-11-09 12:19:1
10	10	fine	fine	training	2024-11-09 12:19:1
11	11	Go ahead and write the number of an...	ADV:write VERB:number NOUN:query ...	training	2024-11-09 12:19:1
12	12	Thank You	VERB:-pron-	training	2024-11-09 12:19:1
13	13	Your Welcome 😊	DET:welcome	training	2024-11-09 12:19:1
14	14	Thanks	thank	training	2024-11-09 12:19:1
15	15	Your Welcome 😊	DET:welcome	training	2024-11-09 12:19:1
16	16	Bye	bye	training	2024-11-09 12:19:1
17	17	Thank You for visiting!..	VERB:visit	training	2024-11-09 12:19:1
18	18	What do you do?	PRON:do VERB:-pron- PRON:do	training	2024-11-09 12:19:1
19	19	I am made to give Information about...	NOUN:vtop PROPN:vellore	training	2024-11-09 12:19:1

Figure 3. DATABASE BROWSER

4. DESIGN APPROACH AND DETAILS

4.1 SYSTEM ARCHITECTURE

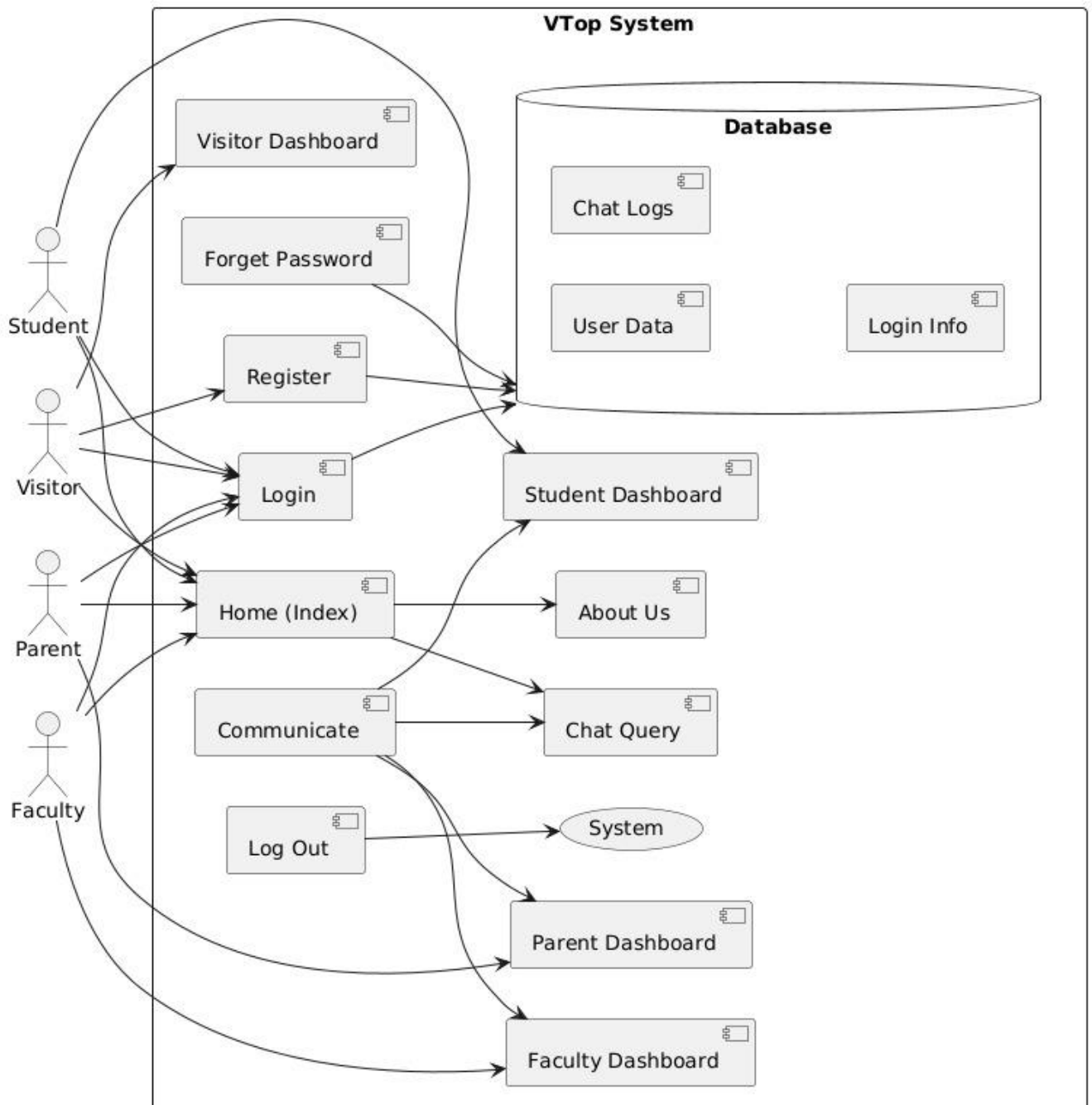


Figure 4. SYSTEM ARCHITECTURE

4.2 DATA FLOW DIAGRAM

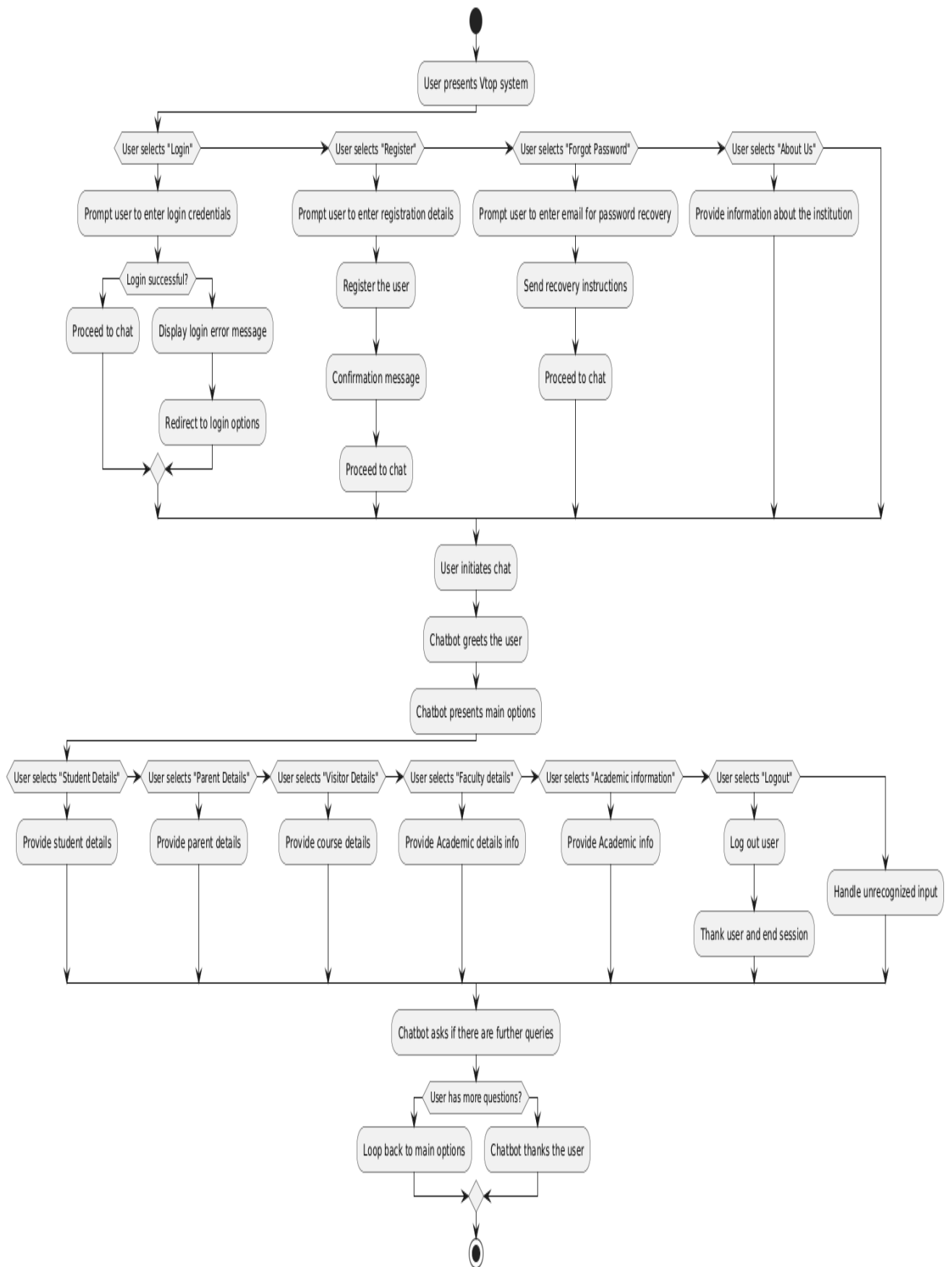


Figure 5. DATA FLOW DIAGRAM

4.3 CODES AND STANDARDS

4.3.1 HTML, CSS, and JavaScript Standard

HTML5: Using semantic HTML tags enhances both accessibility and SEO. Standards also call for correct indentation, comments, and valid syntax.

CSS: CSS should use conventions such as BEM (Block Element Modifier) for organization and prevent inline styling. Use a consistent color palette and responsive design (e.g., viewport units, Flexbox).

JavaScript: JavaScript standards include variable name rules (camelCase), code organization into functions, the use of let and const instead of var, and the minimization of inline JavaScript.

4.3.2 Security Standards:

Open Web Application Security Project: OWASP recommendations guard against threats such as SQL injection, XSS, and CSRF.

ReCAPTCHA: Used to prevent spam and to meet authentication security criteria. Keep RECAPTCHA_SECRET_KEY secret in production contexts.

4.3.3 Python Coding Standard.

PEP 8: The Python style guide promotes indentation and code organization into functions and modules.

Docstrings and comments: Create docstrings for functions and classes, and use inline comments to explain complex reasoning.

Environmental Management: Encrypt keys and configurations with environment variables.

4.3.4 SQL and Database Standards.

SQL Injection Protection: To prevent SQL injection attacks, use parameterized queries rather than Python string formatting for SQL operations.

Data integrity: Use constraints such as NOT NULL and UNIQUE in the database schema to assure data accuracy.

Database Connection Management: Close connections after each operation to avoid memory leaks.

4.3.5 Accessibility and Usability Standards:

Web Content Accessibility Guidelines (WCAG): Ensure that the material is perceptible, operable, comprehensible, and robust.

Responsive Design: Bootstrap is useful, but testing on various devices guarantees that the chatbot interface is accessible on all screen sizes.

4.4 CONSTRAINTS, ALTERNATIVES AND TRADEOFFS

4.4.1 CONSTRAINTS:

Constraints specify the bounds within which the development process must work. Technical limitations, economic restraints, time, resources, and security concerns are all common constraints. For example, using a lightweight framework like Flask may limit sophisticated functionalities that might otherwise be available in more complete frameworks but are limited by project scope or expertise availability. Budget limits may limit the ability to incorporate premium tools, servers, or advanced security measures, whilst stringent time constraints may hinder rigorous testing or execution of all planned features. Resource constraints, such as team expertise, may limit the focus to simpler features or readily available technology. Furthermore, legislative compliance requirements, such as GDPR or CCPA, set stringent standards for data processing and user privacy, limiting the project's development possibilities.

Technical constraints:

Limitations imposed by the project's programming languages, frameworks, or libraries.

For example, Flask's capabilities in comparison to more feature-rich frameworks may limit scalability possibilities.

Budget constraints:

Financial resources for software development, hosting, security, and maintenance. Budget constraints may limit access to premium APIs, third-party services, and high-performance servers.

Time constraints:

Deadlines have been established for development, testing, and deployment. Time constraints may have an impact on the thoroughness with which all intended features are tested and implemented.

Human Resource constraints:

The availability and expertise of team members.

A small team may restrict the complexity and number of features that can be implemented.

Security constraints:

Data protection security protocols, such as GDPR or CCPA compliance when dealing with sensitive information.

Certain functionality and data-sharing capabilities may be limited due to security concerns.

4.4.2 ALTERNATIVES:

Exploring several options aids in balancing project requirements and constraints. For example, where technical constraints exist, alternate frameworks, libraries, or tools can provide identical functionality while remaining within project limits. Different chatbot frameworks (such as ChatterBot, Rasa, and Dialogflow) offer varying degrees of flexibility and scalability, allowing the team to choose the best suitable option. Alternative database selection options, such as SQL versus NoSQL, enable the team to balance data handling needs with scalability. Hosting alternatives also provide varying levels of control and scalability, with the decision between on-premises, cloud, or hybrid solutions having a substantial impact on both performance and cost. Exploring several user authentication approaches, such as single sign-on or multi-factor authentication, reveals how alternatives can help preserve project security and user experience.

Alternative frameworks or technologies:

Consider utilizing a different framework or technology, such as Django rather than Flask for a more solid structure.

Choosing between SQL and NoSQL databases based on the data architectural requirements.

Different Chatbot Libraries:

Comparing ChatterBot to various chatbot frameworks such as Rasa and Dialogflow. Choosing between open-source and proprietary chatbot solutions based on flexibility against simplicity of implementation.

Hosting Options:

On-premise hosting vs cloud hosting (AWS, Google Cloud, etc.).

Tradeoff between on-premise control and cloud scalability.

User Authentication Techniques:

Comparing different authentication mechanisms, such as Auth, regular login, and single sign-on. Weighing security measures against user experience (for example, demanding two-factor authentication).

4.4.3 TRADEOFFS

Tradeoffs are frequently necessary to strike a reasonable balance between opposing project requirements. A key tradeoff may be between cost and quality, with a more cost-effective approach sacrificing some of the final product's performance or resilience. Similarly, time restrictions may drive the project to prioritize core functionality above feature completeness, resulting in a shorter development cycle that sacrifices extra, non-essential features. In terms of user experience, strict security mechanisms such as multi-factor authentication or CAPTCHA can protect user data but also introducing friction, reducing ease of use. Performance and scalability also have drawbacks, as a highly performant solution may not scale effectively, whereas a scalable strategy may necessitate more resources.

Cost Versus Quality:

High-quality solutions (for example, adopting extensive security measures) may raise expenses, whereas low-budget alternatives may jeopardize security or performance.

Speed of development Versus Feature completeness:

Rapid development may necessitate sacrificing certain capabilities or selecting simpler, less configurable components.

Prioritizing important features for speedier deployment over a full feature set that takes longer to build.

User Experience Versus Security:

Implementing rigorous security methods may require additional steps for users (e.g., CAPTCHA, multi-factor authentication), thereby reducing convenience of use.

Balancing user access with necessary authentication protocols.

Performance Versus Scalability:

Using lightweight solutions may enhance short-term performance, but as user numbers increase, scalability difficulties may arise.

Choosing whether to prioritize immediate response or invest in a scalable system for growth.

5. SCHEDULE, TASK AND MILESTONES

5.1. SCHEDULE

Date	Phase	Description
15th July 2024	Topic Selection and Verification	Chose the project topic and received guide's approval to proceed with the research.
18th July – 25th August 2024	Data Collection and Research	Gathered datasets, studied research papers, journals, and articles relevant to the topic.
28th August – 12th September 2024	Literature Review and Project Foundation	Completed the literature review, defined the problem, set objectives, and scoped the project.
18th September – 15th October 2024	Algorithm Exploration and Module Development	Researched algorithms, selected the best ones, and implemented 80% of the project's core modules.
1st November – 15th November 2024	Documentation and Final Implementation	Completed the final implementation and documented all aspects of the project for review.
15th November – 27th November 2024	Conclusion and Final Report Submission	Drafted conclusions, finalized the report, and prepared for the final presentation

Table 1. SCHEDULE

5.2 MILESTONES

5.2.1 Topic Selection and Verification by Guide: July 15, 2024

The first process consisted of deciding on a meaningful and practical project topic. Once the topic had been chosen, the project proposal was presented to the guide for assessment and approval. This stage confirmed that the project was consistent with academic goals and expectations.

5.2.2 Study took place between: July 18th to August 25th, 2024

Sources included datasets, research papers, journals, and publications. The purpose was to lay a solid foundation of information and resources to support the project's development and better understand the issues and existing solutions in the sector.

5.2.3 Literature study and Project foundation: August 28th to September 12th, 2024

The literature study and project foundation were accomplished. This includes assessing past work in the chosen field, defining the challenge, and determining the project's scope and objectives. The review helps to clearly define the study direction and frame the problem for solution development.

5.2.4 Algorithm Exploration and Module Development: September 18th to October 15th, 2024

The Algorithm Exploration and Module Development phase explored and selected the finest algorithms for the project. The algorithms were carefully chosen based on their prospective effectiveness and suitability for the project's needs. In parallel, 80% of the project's fundamental modules, such as basic model creation, data processing, and algorithm implementation, were finished.

5.2.5 Documentation and Final Implementation: November 1st to November 15th, 2024

During this period, the project neared completion, with a thorough assessment of the work documents and final implementation. This involved drafting the pre-final version of the paper, which detailed the methodology, outcomes, and issues encountered throughout the implementation phase. The final execution of the project was also polished and fully integrated to guarantee that all components worked properly.

5.2.6 Conclusion and Report Submission: November 15th to November 27th, 2024

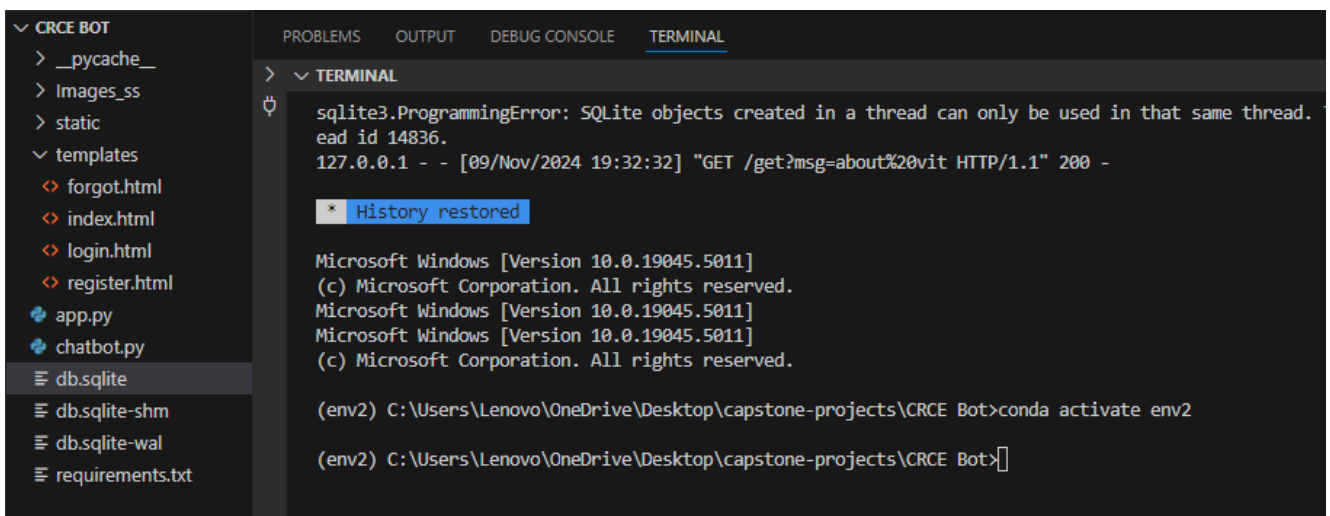
The last step consists of summarizing the entire project, analyzing the outcomes, and drawing conclusions. The final report will be created, with all sections comprehensive and coherent. In addition, preparations will be made for the final presentation, which will emphasize the project's important components, outcomes, and potential influence in the field.

6. PROJECT DEMONSTRATION

6.1 Project Configuration:

Get the project or clone it: Make sure you have all the relevant files, such as the Python scripts (app.py and chatbot.py), HTML templates (for the front end), and any additional assets that may be needed.

Create an Environment Virtually: Dependencies should be kept separate, therefore establish and turn on a virtual environment.



```
CRCE BOT
├── __pycache__
├── Images_ss
├── static
├── templates
│   ├── forgot.html
│   ├── index.html
│   ├── login.html
│   └── register.html
├── app.py
├── chatbot.py
├── db.sqlite
├── db.sqlite-shm
├── db.sqlite-wal
└── requirements.txt

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
> ▼ TERMINAL
sqlite3.ProgrammingError: SQLite objects created in a thread can only be used in that same thread.
ead id 14836.
127.0.0.1 - - [09/Nov/2024 19:32:32] "GET /get?msg=about%20vit HTTP/1.1" 200 -

* History restored

Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.
Microsoft Windows [Version 10.0.19045.5011]
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.

(env2) C:\Users\Lenovo\OneDrive\Desktop\capstone-projects\CRCE Bot>conda activate env2

(env2) C:\Users\Lenovo\OneDrive\Desktop\capstone-projects\CRCE Bot>
```

Figure 6. TERMINAL

6.2 Install the necessary libraries:

Install Dependencies: After turning on the virtual environment, install all necessary libraries, such as:

Flask: An application development microweb framework.

Flask-reCAPTCHA: For Google reCAPTCHA form security.

ChatterBot: Offers chatbot functionality.

MySQL Connector: Used to link a MySQL database to an application.

Execute the subsequent command:

```

requirements.txt
1  pip install flask
2  pip install chatterbot
3  pip install chatterbot-corpus
4  pip install spacy
5  pip install flask-recaptcha
6  pip install mysql-connector-python
7  pip install mysql-connector-python-rf
8

```

Figure 7. LIBRARIES

6.3 Install the MySQL database

Install MySQL: Get it from the official MySQL website and install it.

Make a fresh database:

Get MySQL Workbench or the MySQL command line open.

Create the database and a table for user data storage by running the SQL statements shown below:

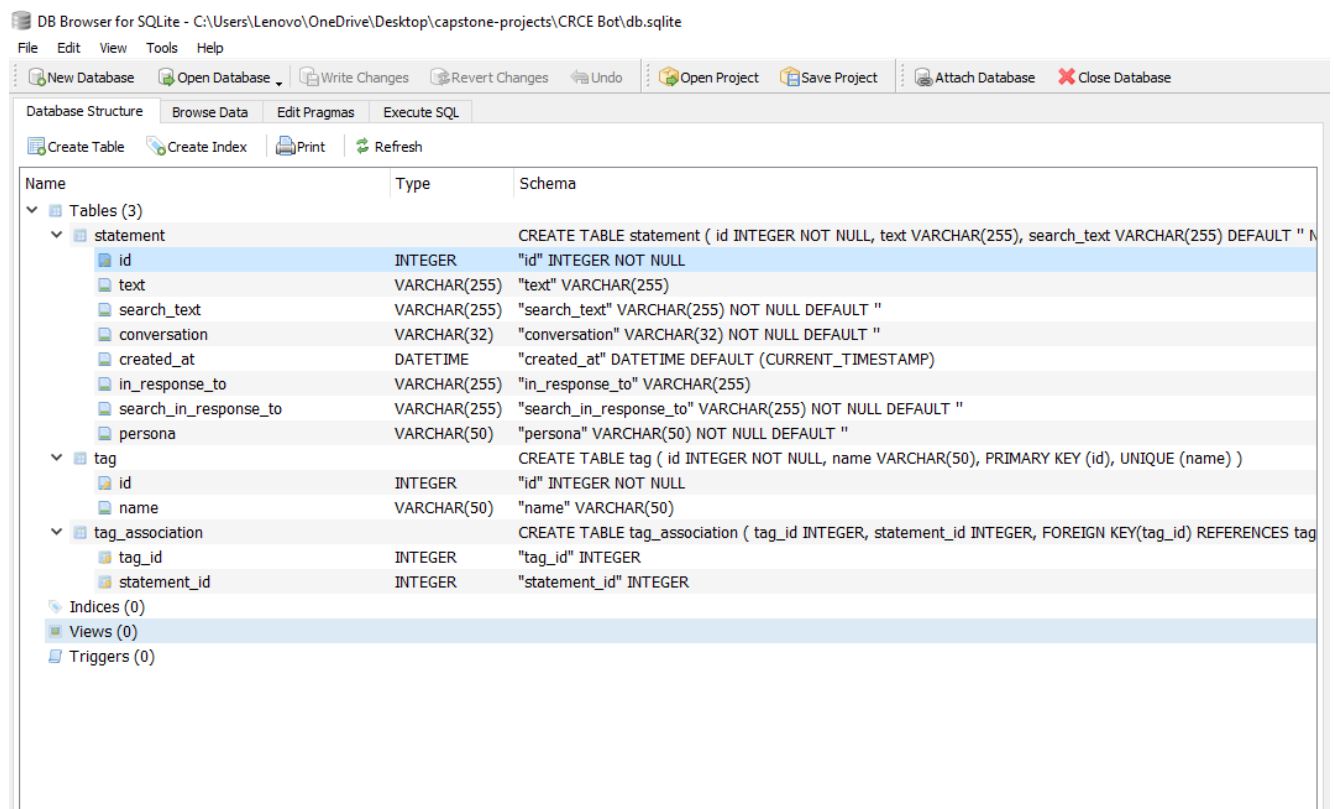


Figure 8. DATABASE

Update the app.py's database connection: Verify that the MySQL connection details correspond to your configuration:

```
#database connectivity
conn=mysql.connector.connect(host='localhost',port='3306',user='root',password='12345',database='register')
cur=conn.cursor()
```

Figure9. UPDATE DB

6.4 Set up Google reCAPTCHA.

Generate reCAPTCHA keys: The RECAPTCHA_SITE_KEY and RECAPTCHA_SECRET_KEY can be obtained by visiting Google reCAPTCHA and creating a new site.

The Enter reCAPTCHA keys[app.py]: In the Flask app settings, enter your keys:

```
app.config.update(dict(
    RECAPTCHA_ENABLED = True,
    RECAPTCHA_SITE_KEY = "6LdbAx0aAAAAAANl04WHtDbraFMufACHccHbn09L",
    RECAPTCHA_SECRET_KEY = "6LdbAx0aAAAAAMmkgBKJ2Z9xsQjMD5YutoXC6Wee"
))

recaptcha=ReCaptcha()
recaptcha.init_app(app)
```

Figure 10. RECAPTCHA

Use: The reCAPTCHA will assist protect login and user registration forms from automated bot attacks.

6.5 In chatbot.py, define the chatbot logic.

Personalize Chatbot Reactions: To answer user questions, the chatbot.py file makes use of the ChatterBot framework. Adapt the answers to your needs by adding more questions and answers to the chat list.

Get the chatbot to learn: According to chatbot.py, the chatbot can be educated with distinct answers for every kind of query. For instance, "1.1" requests for "Curricular" data.

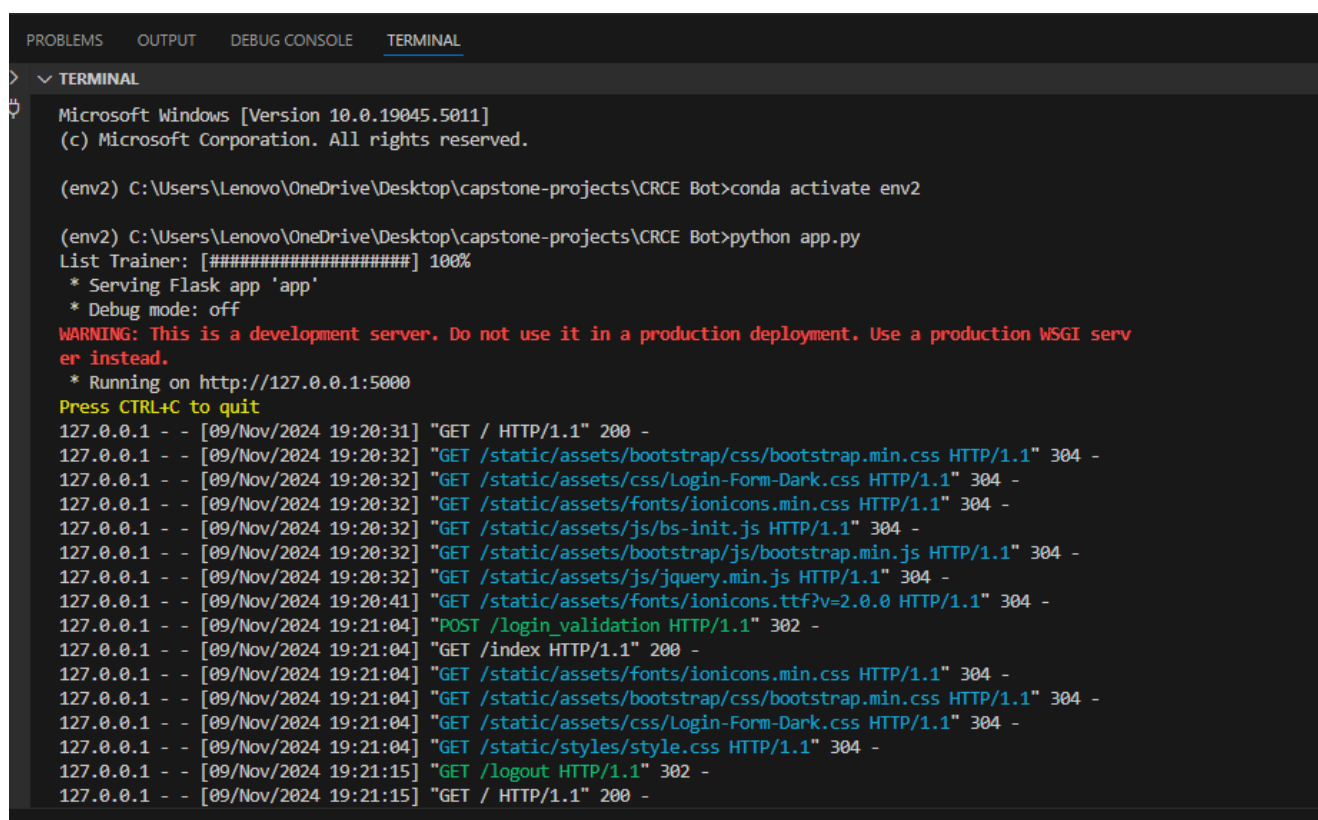
The conversation list can be expanded to include more FAQs or categories.

6.6 Launch the Flask application.

Launch the application: Verify that you are still in the virtual world before running:

Python app.py

Open the application on a web browser: To use the application, launch a web browser and navigate to <http://127.0.0.1:5000>. You ought to be able to access the main chatbot interface as well as pages like register and login.



```
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.

(env2) C:\Users\Lenovo\OneDrive\Desktop\capstone-projects\CRCE Bot>conda activate env2

(env2) C:\Users\Lenovo\OneDrive\Desktop\capstone-projects\CRCE Bot>python app.py
List Trainer: [#####] 100%
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [09/Nov/2024 19:20:31] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/bootstrap/css/bootstrap.min.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/css/Login-Form-Dark.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/fonts/ionicons.min.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/js/bs-init.js HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/bootstrap/js/bootstrap.min.js HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:32] "GET /static/assets/js/jquery.min.js HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:20:41] "GET /static/assets/fonts/ionicons.ttf?v=2.0.0 HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "POST /login_validation HTTP/1.1" 302 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "GET /index HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "GET /static/assets/fonts/ionicons.min.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "GET /static/assets/bootstrap/css/bootstrap.min.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "GET /static/assets/css/Login-Form-Dark.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:21:04] "GET /static/styles/style.css HTTP/1.1" 304 -
127.0.0.1 - - [09/Nov/2024 19:21:15] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [09/Nov/2024 19:21:15] "GET / HTTP/1.1" 200 -
```

Figure 11. FLASK APP RUNINNG

6.7 Testing User Registration and the Chatbot

Create a New User Account: To establish an account, visit the `/register` page and provide your name, email address, and password.

To access the chatbot, log in: Once you've registered, enter your login information and navigate to the `index.html` page, which is where the chatbot is located.

Communicate with the chatbot: To check if the chatbot provides the right information, test it by entering questions like "student" for the Student Section or "faculty" for the Faculty Section. Based on the chatbot.py discussion structure, the bot is configured to offer predetermined data and links for every category.

```
@app.route("/index")
✓ def home():
    ✓ if 'id' in session:
        return render_template('index.html')
    ✓ else:
        return redirect('/')
    ✓ @app.route('/')
    ✓ def login():
        return render_template("login.html")

    ✓ @app.route('/register')
    ✓ def about():
        return render_template('register.html')

    ✓ @app.route('/forgot')
    ✓ def forgot():
        return render_template('forgot.html')
```

Figure 12. USER REGISTER

```
@app.route('/login_validation',methods=['POST'])
def login_validation():
    email=request.form.get('email')
    password=request.form.get('password')

    cur.execute("""SELECT * FROM `users` WHERE `email` LIKE '{}' AND `password` LIKE '{}'""".format(email,password))
    users = cur.fetchall()
    if len(users)>0:
        session['id']=users[0][0]
        flash('You were successfully logged in')
        return redirect('/index')
    else:
        flash('Invalid credentials !!!')
        return redirect('/')
    # return "The Email is {} and the Password is {}".format(email,password)
    # return render_template('register.html')
```

Figure 13. LOGIN VALIDATION

```

@app.route('/add_user',methods=['POST'])
def add_user():
    name=request.form.get('name')
    email=request.form.get('uemail')
    password=request.form.get('upassword')
    #cur.execute("UPDATE users SET password='{}'WHERE name = {}".format(password, name))
    cur.execute("INSERT INTO users(name,email,password) VALUES('{}','{}','{}')".format(name,email,password))
    conn.commit()
    cur.execute("SELECT * FROM `users` WHERE `email` LIKE '{}'"format(email))
    myuser=cur.fetchall()
    flash('You have successfully registered!')
    session['id']=myuser[0][0]
    return redirect('/index')
@app.route('/add_user',methods=['POST'])
def register():
    if recaptcha.verify():
        flash('New User Added Successfully')
        return redirect('/register')
    else:
        flash('Error Recaptcha')
        return redirect('/register')
@app.route('/logout')
def logout():
    session.pop('id')
    return redirect('/')
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    return str(chatbot.get_response(userText))

```

Figure 14. DB & LOGOUT

6.8 Tips for Troubleshooting

Database Connection Problems: Verify that MySQL is operating and that the connection information in app.py is accurate if there are problems connecting to the database.

Problems with reCAPTCHA Validation: Check your site key and secret key if reCAPTCHA doesn't work. To make form submissions easier for testing, you can temporarily disable reCAPTCHA.

Inaccurate Chatbot Reactions: Check the chatbot.py conversation list to make sure the questions and answers match the format you intended if the bot gives unexpected answers.

7. RESULT & DISCUSSION

7.1 RESULTS

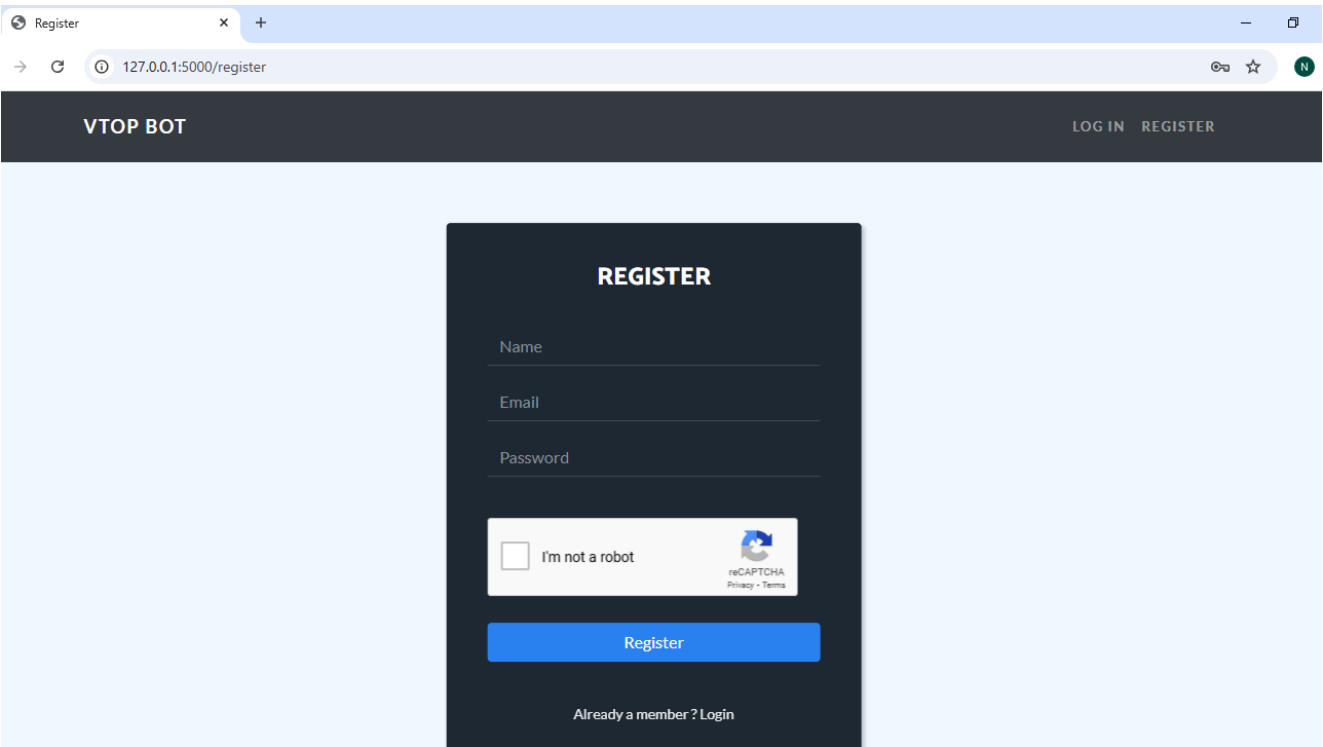


Figure 15. REGISTER

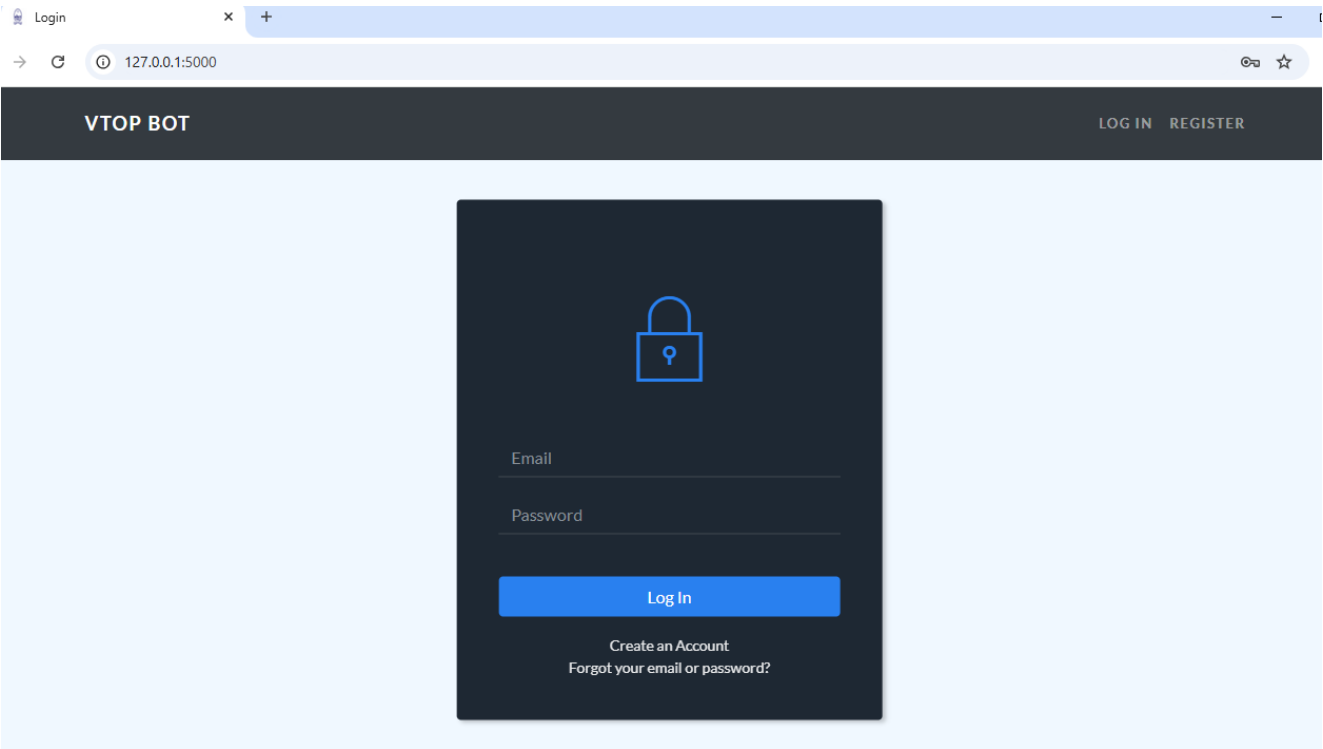


Figure 16. LOGIN

VTOP BOT

LOG IN REGISTER

FORGOT PASSWORD

Name

Email

Password

☐ I'm not a robot

reCAPTCHA

Submit

Figure 17. FORGOT PASSWORD

VTOP BOT

Home About Us Log Out

VTOP BOT

Hi there, Welcome to VTOP! 🤖 If you need any assistance, I'm always here.

VTOP BOT

Which of the following user groups do you belong to?

Student's Section Enquiry.

Faculty Section Enquiry.

Parent's Section Enquiry.

Visitor's Section Enquiry.

Enter your enquiry number...

Send

Figure 18. HOME PAGE

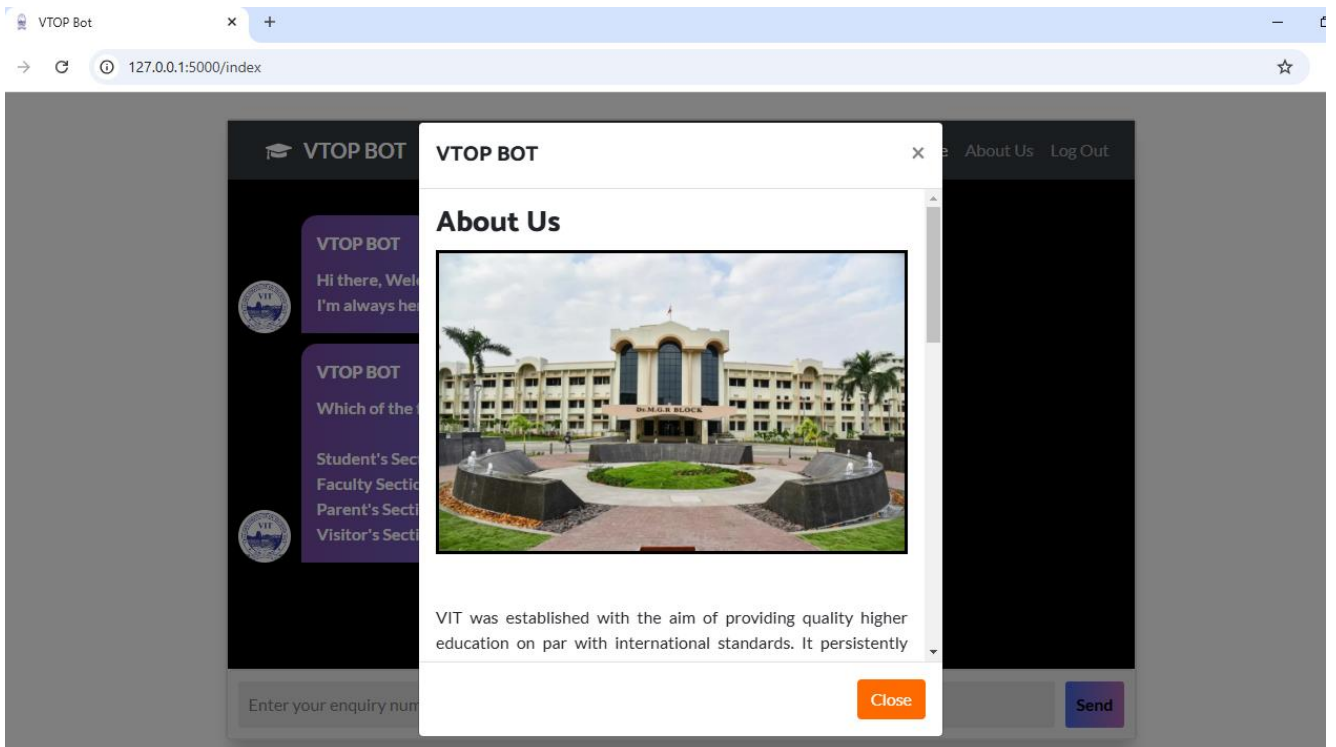


Figure 19. ABOUT US

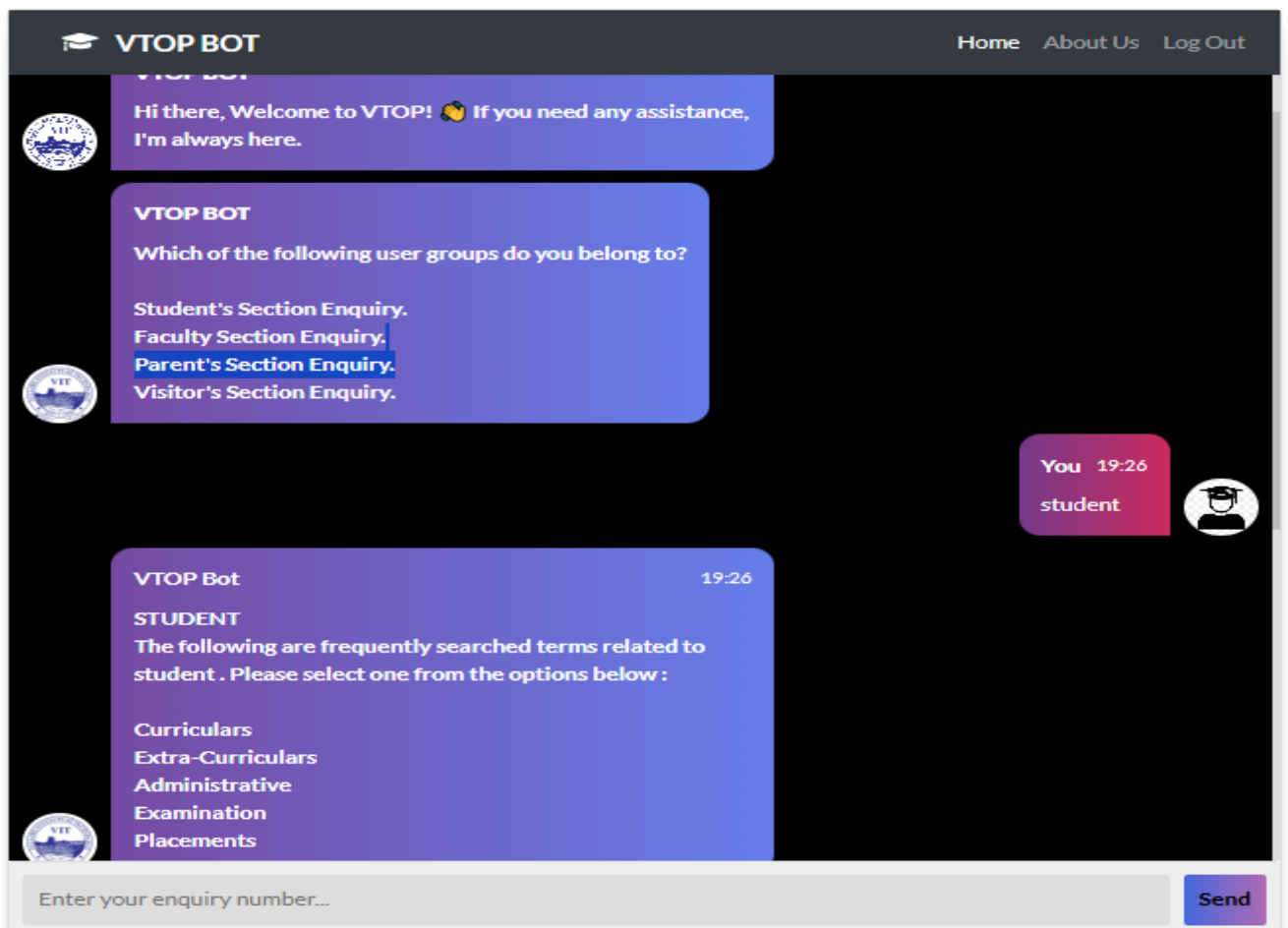


Figure 20. CHATTING WITH BOT

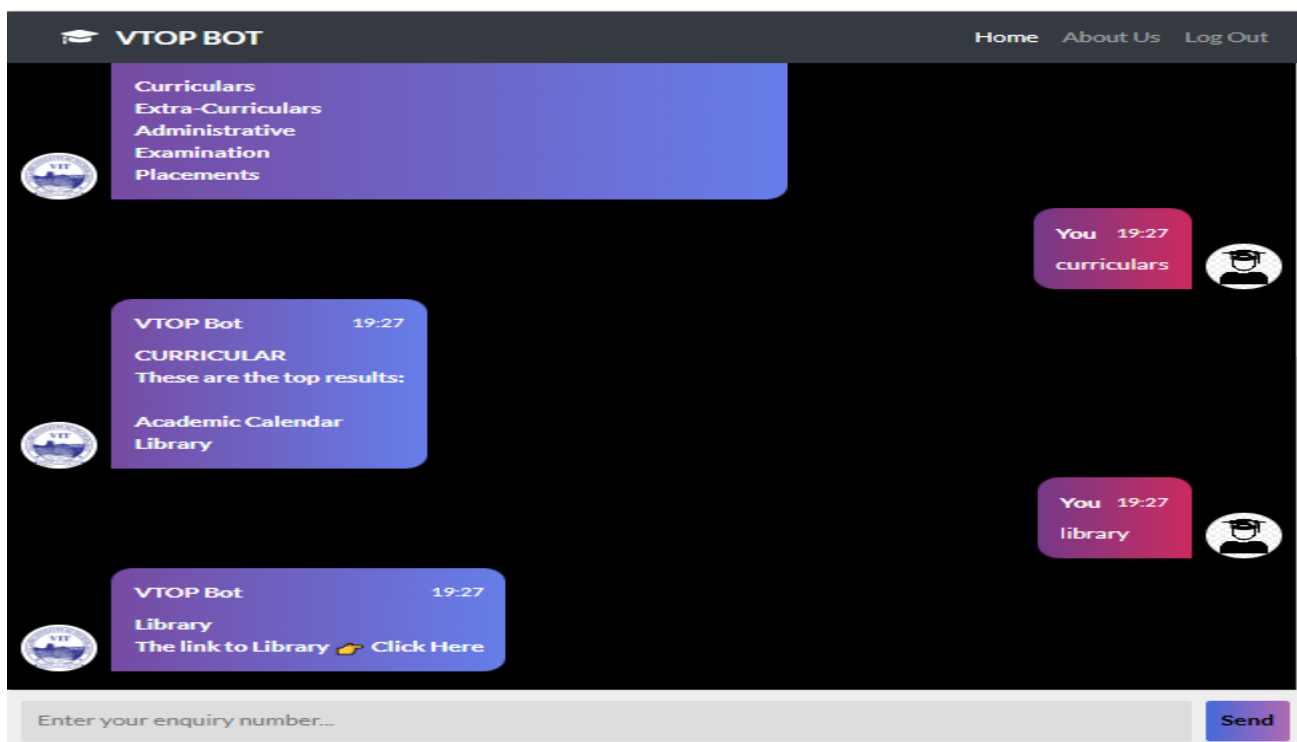


Figure 21. QUERY CHAT

The screenshot shows the VIT Library website. The header includes the VIT logo and navigation links: Home About, Academics, Admissions, CBC, International, Research, Campus Life. The main banner features a photo of the "PERIYAR EVR CENTRAL LIBRARY" building.

PERIYAR EVR CENTRAL LIBRARY

The Central Library is spread over an area of 7770 sq m with six floors (Excluding ground floor). It has specialised collections of books, journals & other resources in mathematics and sciences, engineering and technology, biotechnology, humanities, social sciences and management ranging from printed books, e-books, and back volumes. The Library subscribes to national and international journals in electronic version and print. The Library has a video conferencing facility, other E-learning resources initiated by the Government of India. The Library is fully automated with the web-centric integrated library management system (KOHIA 10.05) with barcode and RFID technology with self-issue and return kiosk, VEDS-CRAC (Online Public Access Catalogue), and online renewal facility.

Link Resources (as on 09/09/2024)

Library Resources	Values-Campus
Number of Books	2,14,036
Number of Back Volumes	19,350
Journals/Magazines	402
SVET	330
national	60
16 databases/Journals (On and Off Campus access facility)	1670
Inform Complete (Pro-Quest), ACM-DL, ACS Publications, ASCE-DB, ASME-DL, ASTM Journals and Standards, Bentham Science, British Standard Euro Codes, CHE Preprint (ProQuest), Economic Outlook and Industry Outlook, EBSCO Business Source Complete, Electrochemical Society (ECS Digital Library), EmeraldSD, Emerald Emerging Market Case Studies (EMERG), IABR Cases, IET, IEEE & IET, Indian Standards, Institute, IOP Journals, MathSciNet, Springer Nature Journals, Nature Research Journals and Scientific American, Royal Society of Chemistry (RSC) Journals, Proquest e-library, VIT Technical Papers (VIT TAP), VIT Online Journals, ScienceDirect, Jstor, Scholar, Scopus, Institute and Proquest Dissertation & Theses (DT) Part A & B (Science and Engineering, Humanities and Social Science) web of Science and Wiley e-journals.	1670
ok - (On and Off Campus access facility)	164,071

Library

- [View our Library Policy](#)
- [Home](#)
- [Online MultiSignal Dictionary](#)
- [Disabled User Friendly Services \(DAVANGAM\)](#)
- [INDOAT \(Online Union Catalogue of Indian Universities\)](#)
- [RISC \(Indian Research Information Networking System\)](#)
- [Shodhganga](#)
- [Shodhganga \(STO\)](#)
- [VDRMAN \(Expert Database and National Research Network\)](#)

Figure 22. USER REQUEST

7.2 CONCLUSION

Students, teachers, parents, and visitors can now obtain college-related information much more easily thanks to the planned College Enquiry Chatbot system. The chatbot provides an effective, customized, and safe user experience by combining cutting-edge natural language processing (NLP) algorithms, improved security features, and real-time content management. By automating repetitive queries, this technology lessens administrative effort and frees up staff members to work on more difficult assignments. The institution's outreach and engagement are strengthened by the chatbot's scalability and multilingual capabilities, which also make it available to a larger audience. All things considered, the chatbot simplifies college inquiries, making data readily available and promoting a satisfying user experience.

7.3 FUTURE SCOPE

The VTOP Bot project effectively illustrates how chatbots may expedite information access in a learning environment. The system gives VIT stakeholders, such as students, teachers, parents, and visitors, an effective tool to answer frequently asked questions about academics, admissions, and services by leveraging Flask, MySQL and ChatterBot. The bot is particularly useful for managing structured queries because of its responsive UI, which was created with Bootstrap to guarantee a user-friendly experience and lead users through predefined alternatives. In addition to giving users round-the-clock assistance, automation of these operations has significantly decreased the workload for administrative staff, freeing them time to concentrate on more complicated duties. A crucial layer of protection is added by security features like Google reCAPTCHA and session-based authentication, which guarantee user data integrity and stop unwanted access. Presently the bot lacks natural language processing (NLP) to comprehend more complicated or open-ended queries, its functionality is still restricted to predefined responses. Future improvements might involve linking the bot to VTOP's live data for real-time updates and incorporating NLP to increase conversational versatility. The VTOP Bot lays the groundwork for future developments in AI-powered educational support systems and is, all things considered, a useful step toward better automation and user experience.

8. SUMMARY

The creation of a VTOP Bot to help VIT stakeholders with information regarding admissions, academics, student services, and administration is described in the project document. MySQL is used for database administration, HTML is used for the front end, and Flask is used for the back end of the system. Enhanced with Bootstrap design and Google reCAPTCHA for security, the article offers thorough descriptions of HTML templates used for registration, login, and password recovery pages.

Through settings tailored to each user group, the ChatterBot-created chatbot capability is designed to accommodate various user groups, including parents, visitors, faculty, and students. Students can obtain information about tests, placements, administrative services, and curriculum activities, for instance. Along with explaining how to integrate MySQL for user data storage, the document also covers the backend logic in app.py, including session handling, user validation, and bot response management. The goal of this project is to enhance VTOP's user experience and expedite information retrieval.

For VIT stakeholders, the VTOP Bot project has greatly expedited the information retrieval process by offering an intuitive and effective interface for obtaining vital information on academics, student services, and admissions. The system provides a dependable and responsive experience for kids, teachers, parents, and guests by combining Flask and MySQL for safe user management with a ChatterBot-powered chatbot. Users and administrative staff have saved time by relying less on human support thanks to the automation of frequently asked questions and organized access to resources.

Future improvements might involve giving the chatbot dynamic learning capabilities, which would enable it to use natural language processing (NLP) to comprehend and reply to more complicated queries. Users might be kept up to date on the most recent announcements with further integration with real-time VTOP system updates. Other worthwhile future options that could further improve user engagement and system robustness include adding multilingual support, expanding chatbot support for mobile access, and strengthening user data security with sophisticated authentication procedures. With these enhancements, the bot will be better equipped to handle a greater variety of queries and offer a more tailored user experience.

9. REFERENCES

- 1) Singh, J., Deshwal, V., Kumar, S., Khalaria, M., Yadav, M., & Negi, P. (2022). A Healthcare Chatbot System Using Python And NLP. *Journal of Pharmaceutical Negative Results*, 5528-5536.
- 2) EBSSEN, T., SEGALL, R. S., ABOUDJA, H., & BERLEANT, D. (2024). A Customer Service Chatbot Using Python, Machine Learning, and Artificial Intelligence. *Journal of Systemics, Cybernetics and Informatics*, 22(3), 38-46.
- 3) Lappalainen, Y., & Narayanan, N. (2023). Aisha: A custom AI library chatbot using the ChatGPT API. *Journal of Web Librarianship*, 17(3), 37-58.
- 4) Chakraborty, S., Paul, H., Ghatak, S., Pandey, S. K., Kumar, A., Singh, K. U., & Shah, M. A. (2022). An AI-based medical chatbot model for infectious disease prediction. *Ieee Access*, 10, 128469-128483.
- 5) Attigeri, G., Agrawal, A., & Kolekar, S. (2024). Advanced NLP Models for Technical University Information Chatbots: Development and Comparative Analysis. *IEEE Access*.
- 6) Parrales-Bravo, F., Caicedo-Quiroz, R., Barzola-Monteses, J., Guillén-Mirabá, J., & Guzmán-Bedor, O. (2024). CSM: a Chatbot Solution to Manage Student Questions About payments and Enrollment In University. *IEEE Access*.
- 7) Benaddi, L., Ouaddi, C., Jakimi, A., & Ouchao, B. (2024). A Systematic Review of Chatbots: Classification, Development, and their Impact on Tourism. *IEEE Access*.
- 8) Deepa, A., Thumati, S. S., & Reyya, S. (2022, April). An efficient deep learning based chatbot for GRIET. In *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)* (pp. 1-5). IEEE.
- 9) Shaikh, T. A. H., & Mhetre, M. (2022, April). Autonomous AI Chat Bot therapy for patient with insomnia. In *2022 IEEE 7th International conference for Convergence in Technology (I2CT)* (pp. 1-5). IEEE.
- 10) Assayed, S. K., Alkhatib, M., & Shaalan, K. (2023, May). Artificial intelligence based chatbot for promoting equality in high school advising. In *2023 4th International Conference on Intelligent Engineering and Management (ICIEM)* (pp. 1-4). IEEE.

APPENDIX

REGISTER

register.html

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
  <title>Register</title>
  <link rel="icon" href="{{ url_for('static', filename='images/icon.png') }}" type="image"
    sizes="16x16">
  <link rel="stylesheet" href="{{ url_for('static',
    filename='assets/bootstrap/css/bootstrap.min.css') }}">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Catamaran:100,200,300,400,500,600,700,800
    ,900">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Lato:100,100i,300,300i,400,400i,700,700i,90
    0,900i">
  <link rel="stylesheet" href="{{ url_for('static', filename='assets/fonts/ionicons.min.css') }}">
  <link rel="stylesheet"
    href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.min.css">
  <link rel="stylesheet" href="{{ url_for('static', filename='assets/css/Login-Form-Dark.css')
    }}">

</head>
<body>
  <nav class="navbar navbar-dark navbar-expand-lg fixed-top bg-dark navbar-custom">
    <div class="container">
      <a class="navbar-brand" href="#" style="font-size: large;">VTOP Bot</a><button data-
        toggle="collapse" class="navbar-toggler" data-target="#navbarResponsive"><span
```

```

class="navbar-toggler-icon"></span></button>
<div class="collapse navbar-collapse" id="navbarResponsive">
  <ul class="navbar-nav ml-auto">
    <li class="nav-item"><a class="nav-link" href="/">Log In</a></li>
    <li class="nav-item"><a class="nav-link" href="/register">Register</a></li>
  </ul>
</div>
</div>
</nav>
<section class="login-dark">

<form action="/add_user" method="POST">

<div class="form-signup">
  <centre>
    <h4 style="text-align: center;">REGISTER</h4>
  </centre>
  <br>
  <div class="form-group">
    <input class="form-control" type="text" name="name" placeholder="Name" required>
  </div>

  <div class="form-group">
    <input class="form-control" type="email" name="uemail" placeholder="Email"
required>
  </div>

  <div class="form-group">
    <input class="form-control" type="password" name="upassword"
placeholder="Password" pattern=".{8,}" required title="8 characters minimum">
  </div>

  <br>
  {{recaptcha}}

```

```

    </br>
    <div class="form-group">
        <button class="btn btn-block" data-bss-hover-animate="pulse" type="submit"
style="background-color: #2980ef; color: white">Register</button>
    </div>
    <br>

    <p class="forgot" >Already a member ? <a href="/">Login</a></p>
</div>
</form>

<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta2/dist/js/bootstrap.bundle.min.js" integrity="sha384-
b5kHyXgcpbZJO/tY9UI7kGkf1S0CWuKcCD38l8YkeH8z8QjE0GmW1gYU5S9FOnJ0"
crossorigin="anonymous"></script>
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkVYIK3UENzmM7KChRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYI"
crossorigin="anonymous"></script>

</body>
</html>

```


LOGIN

Login.html

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
  <title>Login</title>
  <link rel="icon" href="{ { url_for('static', filename='images/vit.png') } }" type="image"
    sizes="16x16">
  <link rel="stylesheet" href="{ { url_for('static',
    filename='assets/bootstrap/css/bootstrap.min.css') } }">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Catamaran:100,200,300,400,500,600,700,800
    ,900">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Lato:100,100i,300,300i,400,400i,700,700i,90
    0,900i">
  <link rel="stylesheet" href="{ { url_for('static', filename='assets/fonts/ionicons.min.css') } }">
  <link rel="stylesheet"
    href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.min.css">
  <link rel="stylesheet" href="{ { url_for('static', filename='assets/css/Login-Form-Dark.css')
    } }">

</head>

<body>
  <nav class="navbar navbar-dark navbar-expand-lg fixed-top bg-dark navbar-custom">
    <div class="container">
      <a class="navbar-brand" href="#" style="font-size: large;">VTOP Bot</a><button data-
        toggle="collapse" class="navbar-toggler" data-target="#navbarResponsive"><span
        class="navbar-toggler-icon"></span></button>
```

```

<div class="collapse navbar-collapse" id="navbarResponsive">
  <ul class="navbar-nav ml-auto">
    <li class="nav-item"><a class="nav-link" href="/">Log In</a></li>
    <li class="nav-item"><a class="nav-link" href="/register">Register</a></li>
  </ul>
</div>
</div>
</nav>

<section class="login-dark">
  <form action='/login_validation' method='POST'>
    <div>
      { % with messages = get_flashed_messages() % }
      { % if messages % }
        { % for message in messages % }
          <div class="alert alert-danger alert-dismissible fade show" role="alert">
            <button type="button" class="close" data-dismiss="alert" aria-label="Close">
              <span aria-hidden="true">&times;</span>
            </button>
            <strong>{{ message }}</strong> <br><small> Try again or forgot the
password!</small>
          </div>
        { % endfor % }
      { % endif % }
    { % endwith % }
  </div>
  <div class="illustration"><i class="icon ion-ios-locked-outline"></i></div>
  <div class="form-group">
    <input class="form-control" type="email" name="email" placeholder="Email"
required/>
  </div>
  <div class="form-group">
    <input class="form-control" type="password" name="password"
placeholder="Password" pattern=".{8,}" required title="8 characters minimum">

```

```

</div>
</br>
<div class="form-group">
    <button class="btn btn-block" data-bss-hover-animate="pulse" type="submit"
style="background-color: #2980ef; color:white;">
        Log In
    </button>
</div>

<a class="forgot" href="/register"> Create an Account </a>
<a class="forgot" href="/forgot">Forgot your email or password?</a>
</form>
</section>
<script src="{{ url_for('static', filename='assets/js/jquery.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/bootstrap/js/bootstrap.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/js/bs-init.js') }}"></script>
</body>

</html>

```

FORGOT

forgot.html

```

<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
    <title>Login</title>
    <link rel="icon" href="{{ url_for('static', filename='images/icon.png') }}" type="image"
        sizes="16x16">
    <link rel="stylesheet" href="{{ url_for('static',
        filename='assets/bootstrap/css/bootstrap.min.css') }}">

```

```

<link rel="stylesheet"
  href="https://fonts.googleapis.com/css?family=Catamaran:100,200,300,400,500,600,700,800
,900">
<link rel="stylesheet"
  href="https://fonts.googleapis.com/css?family=Lato:100,100i,300,300i,400,400i,700,700i,90
0,900i">
<link rel="stylesheet" href="{ { url_for('static', filename='assets/fonts/ionicons.min.css') } }">
<link rel="stylesheet"
  href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.min.css">
<link rel="stylesheet" href="{ { url_for('static', filename='assets/css/Login-Form-Dark.css')
} }">

</head>
<body>
<nav class="navbar navbar-dark navbar-expand-lg fixed-top bg-dark navbar-custom">
  <div class="container">
    <a class="navbar-brand" href="#" style="font-size: large;">VTOP Bot</a><button data-
toggle="collapse" class="navbar-toggler" data-target="#navbarResponsive"><span
class="navbar-toggler-icon"></span></button>
    <div class="collapse navbar-collapse" id="navbarResponsive">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item"><a class="nav-link" href="/">Log In</a></li>
        <li class="nav-item"><a class="nav-link" href="/register">Register</a></li>
      </ul>
    </div>
  </div>
</nav>
<section class="login-dark">

  <form action='/add_user' method='POST'>

    <centre>
      <h4 style="text-align: center;">FORGOT PASSWORD</h4>
    </centre>

```

```

<br>
<div class="form-signup">

    <div class="form-group">
        <input class="form-control" type="text" name="name" placeholder="Name" required>
    </div>

    <div class="form-group">
        <input class="form-control" type="email" name="uemail" placeholder="Email" required>
    </div>

    <div class="form-group">
        <input class="form-control" type="password" name="upassword"
placeholder="Password" pattern=".{8,}" required title="8 characters minimum">
    </div>

    <br>
    {{recaptcha}}
</br>
<div class="form-group">
    <button class="btn btn-block" data-bss-hover-animate="pulse" type="submit"
style="background-color: #2980ef; color: white">Submit</button>
</div>
</div>
</form>
</section>
<script src="{{ url_for('static', filename='assets/js/jquery.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/bootstrap/js/bootstrap.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/js/bs-init.js') }}"></script>
</body>
</html>

```

INDEX

index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
  <title>Login</title>
  <link rel="icon" href="{ { url_for('static', filename='images/icon.png') } }" type="image"
    sizes="16x16">
  <link rel="stylesheet" href="{ { url_for('static',
    filename='assets/bootstrap/css/bootstrap.min.css') } }">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Catamaran:100,200,300,400,500,600,700,800
    ,900">
  <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Lato:100,100i,300,300i,400,400i,700,700i,90
    0,900i">
  <link rel="stylesheet" href="{ { url_for('static', filename='assets/fonts/ionicons.min.css') } }">
  <link rel="stylesheet"
    href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.min.css">
  <link rel="stylesheet" href="{ { url_for('static', filename='assets/css/Login-Form-Dark.css')
    } }">
</head>
<body>
  <nav class="navbar navbar-dark navbar-expand-lg fixed-top bg-dark navbar-custom">
    <div class="container">
      <a class="navbar-brand" href="#" style="font-size: large;">VTOP Bot</a><button data-
toggle="collapse" class="navbar-toggler" data-target="#navbarResponsive"><span
class="navbar-toggler-icon"></span></button>
      <div class="collapse navbar-collapse" id="navbarResponsive">
        <ul class="navbar-nav ml-auto">
          <li class="nav-item"><a class="nav-link" href="/">Log In</a></li>
          <li class="nav-item"><a class="nav-link" href="/register">Register</a></li>
```

```

        </ul>
</nav>
<section class="login-dark">
  <form action="/add_user" method="POST">
    <centre>
      <h4 style="text-align: center;">FORGOT PASSWORD</h4>
    </centre>
    <br>
    <div class="form-signup">
      <div class="form-group">
        <input class="form-control" type="text" name="name" placeholder="Name" required>
      </div>
      <div class="form-group">
        <input class="form-control" type="email" name="uemail" placeholder="Email" required>
      </div>
      <div class="form-group">
        <input class="form-control" type="password" name="upassword"
placeholder="Password" pattern=".{8,}" required title="8 characters minimum">
      </div>
      <br>
      {{recaptcha}}
    </br>
    <div class="form-group">
      <button class="btn btn-block" data-bss-hover-animate="pulse" type="submit"
style="background-color: #2980ef; color: white">Submit</button>
    </div>
  </div>
</form>
</section>
<script src="{{ url_for('static', filename='assets/js/jquery.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/bootstrap/js/bootstrap.min.js') }}"></script>
<script src="{{ url_for('static', filename='assets/js/bs-init.js') }}"></script>
</body>
</html>

```

CHATBOT

Chatbot.py

```
from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
import spacy
spacy.load('en_core_web_sm')
# from spacy.lang.en import English
from chatterbot.trainers import ChatterBotCorpusTrainer

# Creating ChatBot Instance
chatbot = ChatBot('<b>VTOP BOT</b>')

# nlp = spacy.load("en_core_web_sm")

chatbot = ChatBot(
    'ChatBot for College Enquiry',
    storage_adapter='chatterbot.storage.SQLStorageAdapter',
    logic_adapters=[
        {
            'import_path': 'chatterbot.logic.BestMatch',
            'default_response': "Hi there, Welcome to VTOP! 🙌 If you need any assistance, I'm  
always here.Go ahead and write the number of any query. 😊🌟<b><br><br> Which of the  
following user groups do you belong to? <br><br> Student's Section Enquiry.</br>Faculty  
Section Enquiry. </br>Parent's Section Enquiry.</br>Visitor's Section Enquiry.</br> <br>",
            'maximum_similarity_threshold': 0.90
        }
    ],
    database_uri='sqlite:///db.sqlite'
)
trainer = ListTrainer(chatbot)

# python app.py
# Training with Personal Ques & Ans
```


conversation = [

"Hi",

"Hello!",

"Hey",

"How are you?",

"I'm good.</br>
Go ahead and write the number of any query. 😊 ✨
 Student's Section
Enquiry.</br>Faculty Section Enquiry. </br>Parent's Section Enquiry.</br>Visitor's Section
Enquiry.</br>",

"Great",

"Go ahead and write the number of any query. 😊 ✨
 Student's Section
Enquiry.</br>Faculty Section Enquiry. </br>Parent's Section Enquiry.</br>Visitor's Section
Enquiry.</br>",

"good",

"Go ahead and write the number of any query. 😊 ✨
 Student's Section Enquiry.</br>
Faculty Section Enquiry. </br>Parent's Section Enquiry.</br>Visitor's Section
Enquiry.</br>",

"fine",

"Go ahead and write the number of any query. 😊 ✨
 Student's Section Enquiry.</br>
Faculty Section Enquiry. </br>Parent's Section Enquiry.</br>Visitor's Section
Enquiry.</br>",

"Thank You",

"Your Welcome 😊",

"Thanks",

"Your Welcome 😊",

"Bye",

"Thank You for visiting!..",

"What do you do?",

"I am made to give Information about VTOP Vellore. ",

"What else can you do?",

"I can help you know more about VTOP",

"student",

"STUDENT
The following are frequently searched terms related to student . Please
select one from the options below :

 Curriculars
Extra-Curriculars

Examination
 Placements ",

"curricular",

" CURRICULAR
 These are the top results:

 Academic Calendar

Moodle ",

"academic calendar",

"Academic Calendar
The link to Academic Calender👉 Click Here ",

"moodle",

"Moodle
The link to Moodle 👉 Click Here ",

"extra curriculars",

"EXTRA-CURRICULAR
These are the top results:

Events
Student
Chapters
Student's Council",

"events",

" Events
The link to Events👉 Click
Here",

"student chapters",

" Student Chapters
The link to Student Chapters👉 Click Here
",

"student council",

"Student's Council
The link to Student's Council👉 Click Here ",

"examination",

" EXAMINATION
These are the top results:
 Exam Notices
 Exam Question",

"exam question",

" Exam Question
The link to Notices👉 Click Here ",

"exam notices",

" Exam Notices
The link to Notices👉 Click Here ",

"placement",

" PLACEMENTS These are the top results:
 Overview
 Our Recruiters
 Placement Statistics",

"overview",

" Overview
The link to Placements👉 Click Here ",

"placement statistics",

" Placement Tracker
The link to Placement tracker👉 Click Here ",

"faculty",

" FACULTY
The following are frequently searched terms related to faculty. Please select one from the options below :

 Portals",

"portals",

" PORTALS These are the top results:
 Biometric Attendance
Faculty Moodle

",
 "biometric",
 " Biometric Attendance
The link to Biometric Attendance 🖱️ Click Here ",
 "faculty moodle",
 " Moodle
The link to Moodle 🖱️ Click Here ",
 "parent",
 " PARENTS
The following are frequently searched terms related to Parents. Please select one from the options below :

 About Us
 Notices
 Placements ",
 "about us",
 " ABOUT US
These are the top results:

 About Vit ",
 "about vit",
 "About vit
The link to About Vit 🖱️ Click Here ",
 "notices",
 " NOTICES
These are the top results:

 Annoucement ",
 "annoucement",
 "annoucement
The link to All Notices 🖱️ Click Here ",
 "placement result",
 " PLACEMENTS These are the top results:

 Placement Detail
 Placement Contact
 Placement Tracker ",
 "placement detail",
 " Placements
The link to Placements 🖱️ Click Here ",
 "placement tracker",

" Placement Tracker
The link to Placement Tracker👉 Click Here ",

"visitors",

"<b VISITORS
The following are frequently searched terms related to visitors. Please select one from the options below :

 Visitor detail
 Programs We Offer ",

"vistior detail",

" ABOUT US
These are the top results:

 About Vit ",

"about vit",

"About Vit
The link to About CRCE👉 Click Here ",

"Programs We Offer",

" PROGRAMS WE OFFER
These are the top results:

Application",

"application",

"Application
The link to Application👉 Click Here ",

]

trainer.train(conversation)

APP

app.py

```
from chatbot import chatbot
from flask import Flask, render_template, request, session, logging, url_for, redirect, flash
from flask_recaptcha import ReCaptcha
import mysql.connector
import os

app = Flask(__name__)
recaptcha = ReCaptcha(app=app)
app.secret_key=os.urandom(24)
app.static_folder = 'static'
app.config.update(dict(
    RECAPTCHA_ENABLED = True,
    RECAPTCHA_SITE_KEY = "6LdbAx0aAAAAAANI04WHtDbraFMufACHccHbn09L",
    RECAPTCHA_SECRET_KEY = "6LdbAx0aAAAAAMmkgBKJ2Z9xsQjMD5YutoXC6Wee"
))

recaptcha=ReCaptcha()
recaptcha.init_app(app)

app.config['SECRET_KEY'] = 'cairocoders-ednalan'

#database connectivity
conn=mysql.connector.connect(host='localhost',port='3306',user='root',password='12345',database='register')
cur=conn.cursor()

# Google recaptcha - site key : 6LdbAx0aAAAAAANI04WHtDbraFMufACHccHbn09L
# Google recaptcha - secret key : 6LdbAx0aAAAAAMmkgBKJ2Z9xsQjMD5YutoXC6Wee

@app.route("/index")
def home():
    if 'id' in session:
        return render_template('index.html')
```

```

else:
    return redirect('/')
@app.route('/')
def login():
    return render_template("login.html")

@app.route('/register')
def about():
    return render_template('register.html')

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

@app.route('/login_validation',methods=['POST'])
def login_validation():
    email=request.form.get('email')
    password=request.form.get('password')

    cur.execute("""SELECT * FROM `users` WHERE `email` LIKE '{ }' AND `password` LIKE
        '{ }'""".format(email,password))
    users = cur.fetchall()
    if len(users)>0:
        session['id']=users[0][0]
        flash('You were successfully logged in')
        return redirect('/index')
    else:
        flash('Invalid credentials !!!')
        return redirect('/')

    # return "The Email is { } and the Password is { }".format(email,password)
    # return render_template('register.html')

@app.route('/add_user',methods=['POST'])
def add_user():

```

```

name=request.form.get('name')
email=request.form.get('uemail')
password=request.form.get('upassword')
#cur.execute("UPDATE users SET password='{ }'WHERE name = '{ }'".format(password,
    name))
cur.execute("INSERT INTO users(name,email,password)
    VALUES('{ }','{ }','{ }')".format(name,email,password))
conn.commit()
cur.execute("SELECT * FROM `users` WHERE `email` LIKE '{ }'".format(email))
myuser=cur.fetchall()
flash('You have successfully registered!')
session['id']=myuser[0][0]
return redirect('/index')
@app.route('/add_user',methods=['POST'])
def register():
    if recaptcha.verify():
        flash('New User Added Successfully')
        return redirect('/register')
    else:
        flash('Error Recaptcha')
        return redirect('/register')
@app.route('/logout')
def logout():
    session.pop('id')
    return redirect('/')
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    return str(chatbot.get_response(userText))

if __name__ == "__main__":
    # app.secret_key=""
    app.run()

```