AI Based Student Enquiry System

O.L.S.S. Kumari
2020

AI Based Student Enquiry System

A dissertation submitted for the Degree of Master of Information Technology

O.L.S.S. Kumari
University of Colombo School of Computing

2020

ABSTRACT

A chatbot is a computer program based on Artificial Intelligence, which emulates the conversation between human and the system in natural language. It can communicate in text or audio format. Chatbot can identify the user queries and decide itself the relevant response to answer the queries. Chatbots are used in messaging applications, embedded in websites and mobile applications.

This project mainly focusses on using this emerging technology to ease the life of students and staff in the field of higher education.

Every year many students visit the college websites, visit administration offices, send hundreds of emails and contact helpdesk to inquire regarding the admission process, course details, scholarships, and many more areas related to university life. Answering these queries by helpdesk systems normally takes much time, requires manpower, and cause some errors in communication. Since most of these questions are repetitive, where chatbots can be used to handle this time-consuming task of replying to all the queries of these students, personally and automatically.

This will save the time of the students and release the burden of administration offices and helpdesks of universities. Moreover, the students will not need to visit administration offices or will not need to wait in telephone lines with helpdesk to gather the information that they require.

This project acts as a real time chat application with an effective Graphical User Interface, embedded in university website and can be accessed by everyone. This system analyses the queries of students and providing them with relevant answers with the help of artificial intelligence. This application can provide answers in both text and audio formats and it is operating as a 24/7 functioning student support system.

DECLARATION

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: O.L.S.S Kumari

Registration Number: 2017/MIT/039

Index Number: 17550392

____

Signature: Date: 11/14/2020

This is to certify that this thesis is based on the work of

Mr./Ms.

under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name:

____

Signature: Date:

ACKNOWLEDGEMENT

I take this opportunity to thank all the people who have given their tremendous support for me to complete this project. Without them it will be a hard journey for me.

I express my gratitude towards Mr. K. P. M. K. Silva, supervisor of the project, for the support he has given me to complete this project. Without his supervision, guidance, and help, successful completion of this project would be unimaginable. His ideas and feedback helped me greatly to reach the targets set for the project tasks.

And, I would like to express gratitude towards Mr. Rifa Salam, advisor of the project, for the guidance provided for me to complete this project successfully. The encouragement you provided me and the trust you had on my skills and capabilities was a great strength for me.

I would also like to thank my parents and all the friends who have given their support and for being a strength throughout this journey.

Finally, I thank every person who gave me their support and assistance to complete this project successfully.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI – Artificial Intelligence

ALICE - Artificial Linguistic Internet Computer Entity

API – Application Programming Interface

CNN – Cable News Network

CSS – Cascading Style Sheets

HTML – Hypertext Markup Language

JWT - JSON Web Token

MIT - Masters in Information Technology

NLP – Natural Language Processing

NoSQL - Not Only SQL

SQL – Structured Query Language

UCSC - University of Colombo School of Computing

XML - Extensible Markup Language

Chapter 01

1. INTRODUCTION

1.1 Motivation

For any university/college or an institute, effective communication is one of the major factors to attract more students and to convert prospective applicants into enrolled/registered students. At present, majority of these prospective applicants are technically sound individuals who lives in a digital world with multiple mobile devices at their hands. Since they are used to access information within few seconds at their fingertips, they expect the same instant responses for their queries from universities.

This project is for implementing an AI based student enquiry chatbot for university students to ask queries and get them responded easily from the university website. With this implementation, students can have a live chat with the chatbot and get responses for their answers within few seconds. They will not need to wait until they get the answers through university support systems. Furthermore, this system will be an ideal solution to release the load from traditional university helpdesk systems. In addition to this, this system consists of a forum, where the registered students can use to see FAQs and any public discussions.

1.2 Problems Identified

During the admissions season, prospective students visit universities with applications or with enquiries about the courses that they wish to follow, course fees and details about scholarships. The college administration office or helpdesk staff proceeds to address these queries, with the help of both online and offline channels. This process shows some drawbacks as below.

- Help desks or the administration of universities does not operate for 24/7 hours. Therefore, the students must wait until these offices resume for work. Specially on public holidays, weekends and at nighttime students won't be able to contact the helpdesk for any queries.
- This process takes a considerable amount of time and requires a manpower. With the limitation of staff and resources, students will have to stay in the telephone line on-hold for some time or like waiting for a reply to their emails.
- This process consumes time and money of the students since they need to visit colleges if they didn't receive enough information from the helpdesk.
- This also can leave a room for human errors and misunderstandings in communication and sometimes it has the possibility of tarnishing the image of the university as well.

"AI Based Student Enquiry Chatbot System" can be used to eliminate these issues. It is programmed with automated answers for the repetitive questions, and any number of students can chat at the same time with them, without any cost and delay at any time of the day.

1.3 Aims and Objectives

The main objective of this project is implementing an AI based student enquiry chatbot system for universities to provide answers for students' queries.

- This system is a web application embedded in University website, which can provide answers to students' queries in both text and text-to-speech format. It replies in natural language using an effective Graphical user interface implying like a real person is chatting.

- It helps in saving the time of students rather than going to the universities and contacting help desk. Since the mobile phones are always in their hands, it will be very easy to get the information via this system.
- Helpdesk staff can process two or three operations in the same time, but this chatbot can process without a specific limit and can scaleup the operations.
- This will take off the workload of helpdesk staff and it will improve the response rate when compared to traditional help desk systems.
- Furthermore, this can automate the repetitive tasks. Like when the university receives the same queries frequently, chatbot is trained to answer them.
- The main advantage of this chatbot is that it's availability. This is up and running for 24/7. No matter what time it is, students can use this to get their queries answered.
- The university management can monitor student queries received via these chats and analyze to identify areas where they need improvement. This helps the universities to get updated with the changing demands and preferences of students.

1.4 Scope

This project mainly focuses on answering students' queries related to the admission processes of university through a web-based chatting system, using artificial algorithms.

In this system user can query about the college activities which is given as the input to the chatbot interface. It analyses the inputs and uses the artificial intelligence algorithms to fetch the suitable replies for the students' queries. All college activities and details are updated in the web application frequently by the admin. In this application user can get all the details like admission details, course information, scholarship details, and the general questions about the university. The chatbot uses Artificial Intelligence Modelling Language (AIML) as knowledge source for filtering the responses and it automates the college manual process.

Any prospective student can ask their queries. Furthermore, to assist the differently abled students like those who have visual impairments, text to speech facility is implemented supporting phone calls. Moreover, students can give the feedback for the service that they were provided by the Chatbot as a rating.

All the student queries will be saved in the database and later university administrator can analyses those and identify the patterns and trends of the student community. Then they can take suitable decisions to uplift their processes and services to cater the needs of this new generation.

1.5 Structure of the dissertation

An introduction of the project "AI based student enquiry system" is given in this chapter along with the motivation to implement this project and the potential problems addressed by this, aims/objectives and the scope of the project. In chapter 02, requirement analysis with diagrams, literature review of similar systems is described. Furthermore, a comparison between alternative design strategies is discussed. Chapter 03 describes software development methodology used, and a detailed design of the project with the technologies used. Project evaluation and testing strategies is included in 04th Chapter. In chapter 05 the conclusion of the project is given. References are included at the end of the dissertation.

Chapter 2

2. BACKGROUND

2.1 Requirement Analysis

When student community is considered, they can have many numbers of queries regarding different aspects and different functionalities of the university. Considering all those university functionalities and implementing this system to cater all those is a bit complex process, since all those data should be feed into this chatbot system, for the chatbot to analyze them and provide answers.

Therefore, some limited and basic requirements of the prospective students are considered in this implementation. Below are the question types that this system is capable to answer.

- General queries about the university. (Example: Opening hours, university address, contact details)
- Helping prospective students to find a degree program that is suitable for their future career.
- Queries related to the admission process.
- Course details that are offered by the university.
- Scholarships that are offered.
- A Forum for public discussions, available for registered students.

Students can provide their feedbacks on the service offered by the chatbot.

The administrator of the system can log in using his credentials. His main responsibility is to maintain the system by adding college admissions related details to the system and by updating the current information whenever it's needed. Furthermore, he can view the user feedbacks, rankings and queries asked by the users. He can generate reports regarding the open enquiries, pending ones and the closed enquiries.

2.1.1 Functional Requirements

- Chatting
  This is the most basic level requirement of the chatbot. The chatbot must respond to the user when the user enters a query. This gives the impression to the users like they are chatting with a real human being. System informs the user if an answer is not available or if the question is invalid.
- Text to speech
  Text to speech feature will be implemented which, allows user to ask the question from the chatbot via a phone call and it will answer the question in audio format.
- Querying
  Prospective students can query about the general questions of the university, can make use of chatbot to find a course that they like to follow and get to know about the admission process.
- Logs
  System should maintain logs of the queries asked by students from the chatbot.
- Student Forum

This will contain questions like FAQs and public discussions. Registered students can get information via the Forum.

- Feedbacks
  User can rate the service he received from the chatbot by 1 to 5 rating and can provide their feedbacks.
- Administrator portal
  The administrator is responsible for inserting, updating and deleting details, and keywords.
  The administrator also can view the logs and user feedbacks. Administrator can sign up and sign into the system.
  Furthermore, he can generate reports regarding the open enquiries, pending ones and the closed ones.

2.1.2 Non-Functional Requirements

- User Interface
  The most important usability requirement of this system is a user friendly chatbot and the website. The chatbot is user friendly since it has a simple chat interface. Administrator portal also has an easy to use interface. And, this system should be compatible with all the commonly used browsers and operating systems, such as Firefox, Google chrome, Safari, Windows and Mac OS.
- Web Accessibility
  To assist the differently abled students like those who have visual impairments, text to speech feature is supported via phone calls.
- Performance/Response time
  The replies from chatbot should reach the user in a timely manner to keep the flow of the conversation without interruptions. Since time is valuable for users, if the response is not quick enough the users will get pissed off with the application
- Security
  The administrator portal, database and passwords should be protected from unauthorized access.
- Portability
  The system should run on major operating systems and hardware when there is an internet connection. Should be mobile friendly.
- Maintainability
  A clear separation should be there with the frontend and backend codes for the system to be easily maintained. The chatbots must be updated with up to date information. It makes easier for the chatbot to learn information over time.
- Exception handling
  User errors should be displayed to users effectively.
- Session handling
  User sessions should be handled properly, so that the system doesn’t mix-up the users when multiple users access the system same time.
- Reliability
  Consistency of the chatbot’s responses is a must. It should provide answers in a user acceptable manner when communicating.

2.1.3 Use case diagram for the developed system

The following figure 2.1 is a high-level use case diagram of the “AI based student enquiry system”.

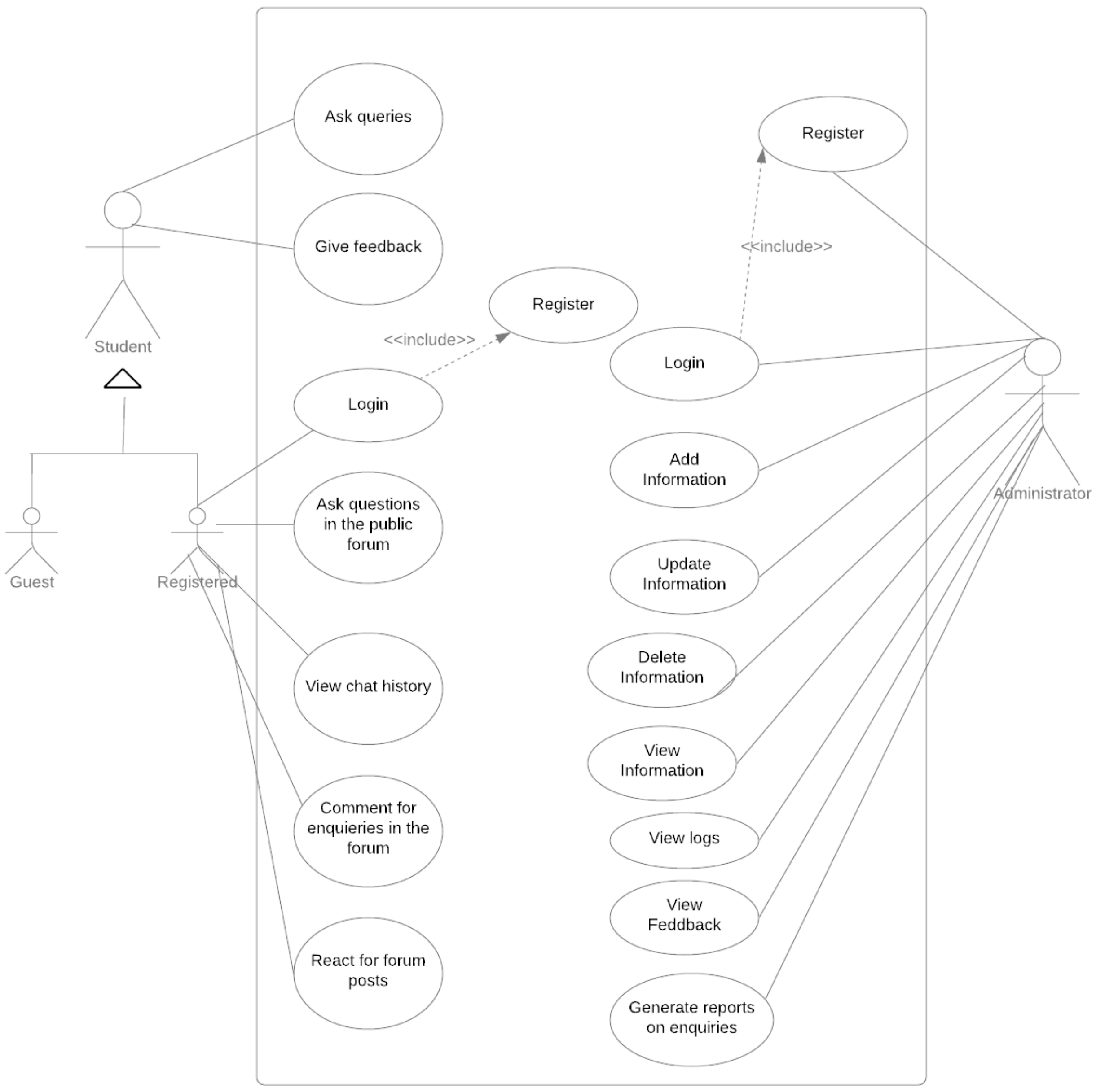


Figure 2.1: Overall Use case diagram of the chatbot system

2.1.4 Use case description

The users in the system are the students and administrator. In the above use case chatbot has been added as an actor to showcase its functionalities as well.

The following use cases are the high-level functional requirements of each actor.

Students

- Chat with the system in text/audio format
- Give feedbacks
- Public forum
- Raise queries
- Login
- Registration
- User chat history for registered users
- Email notifications regarding their enquiries
- Registered students can comment and react for forum posts.

Administrator

- View user enquiries
- View user feedbacks
- Add information to the system
- Login to the system
- Register with the system
- Generate report on enquires(solved/unsolved)
- Training the AI chatbot

Chatbot

- Analyze user query
- Search the answer in knowledge base
- Respond with the answer (using feed-processing)
- Searching details through 3rd party APIs

2.2 Review of similar systems

Different categories of chatbots can be identified in today's world.

- Command based chatbots - These chatbots rely on a dataset of replies and heuristics. User must ask very specific questions for the chatbot to answer. Hence, these bots are implemented to answer only for limited number of questions, and they cannot perform functions outside the code [1].
- Chatbots based on AI or machine learning algorithms - These chatbots can answer ambiguous questions where the users do not need to be specific while asking questions. Thus, they create replies using Natural Language Processing (NLP) [1].

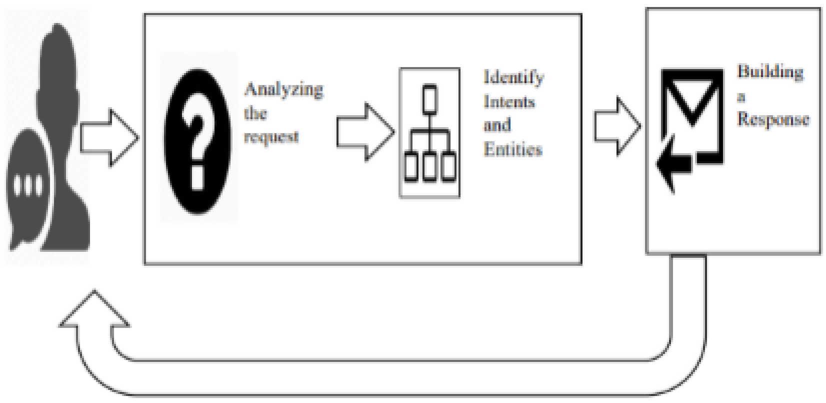


Figure 2.2: How does a chatbot works

Figure 2.2 simply shows generally how a chatbot works. When a user asks any query from the chatbot, it will first analyze the request, and then it identifies intents and entities of the request, finally it builds a response and sends it back to the user. Intention of the query is known as Intents and details of that query is known as Entity.

For example, if a student wants to know the open hours of Post-graduate division, then the intent will be the open hours and entity will be name of the relevant department.

Chatbots are used in many organizations replacing manpower whenever it is feasible and possible to use. Majority of the chatbots are based on the way ELIZA or ALICE chatbot communication methods.

2.2.1 ELIZA

ELIZA is the first natural language processing computer program developed in 1966. Eliza was implemented by simulating the conversations using a "pattern matching" and substitution methodology [2]. Using this pattern matching technique, it can read the user input and search for specific keywords, if those keywords are found then the answer gets retrieved. If it could not find any match for keywords, ELIZA would re-try based on the given rules and will try to get more information from the user and will keep the flow of the conversation.

Following example shows how ELIZA works.

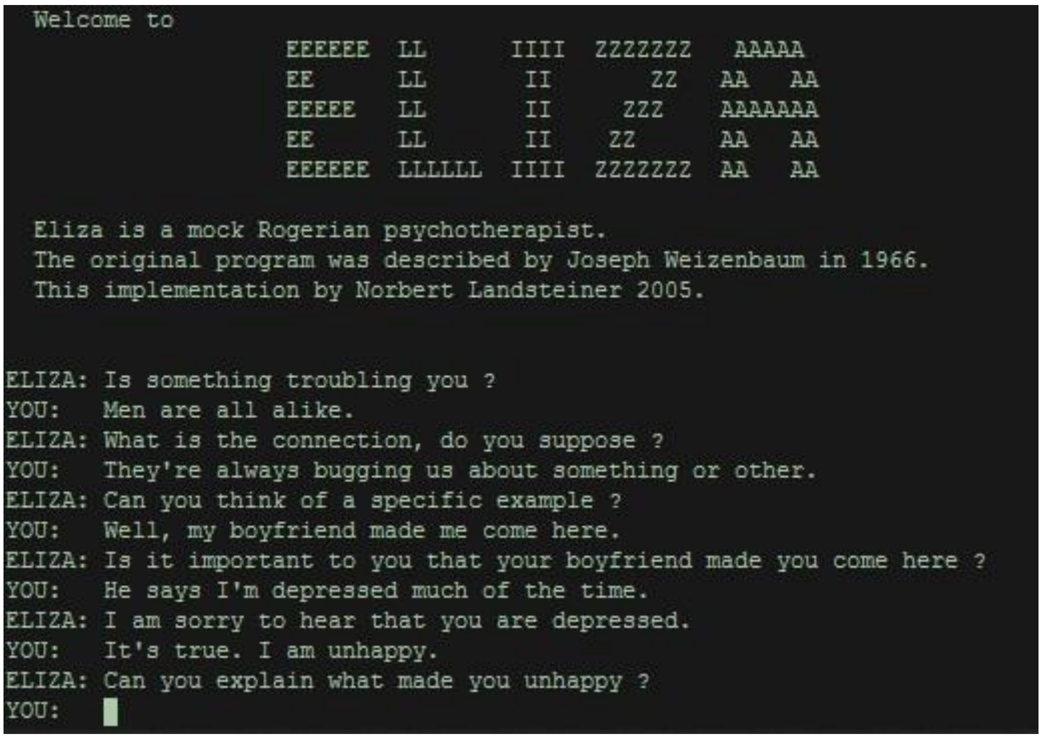
User: Can I know the exam dates?

Here, ELIZA identifies the keyword “exam dates” and would find a connection in “exam dates” and “degree program”. Then it responds to the user with the below question.

ELIZA: Tell me more about the degree program that you follow

ELIZA cannot understand what the user is saying. It gives results based only on the rules. In the above scenario, there was a rule connecting “exam dates” and “degree program”. Then the bot had to respond the user with another question, in order to find the most matching keyword based on the user response. However, some standard replies are there, that ELIZA would use if it was not able to match a keyword. One example is like “Very interesting, please go on” or “Can you think of a special example?” [2]

The following Figure 2.3 is an example of chatting with ELIZA [2].



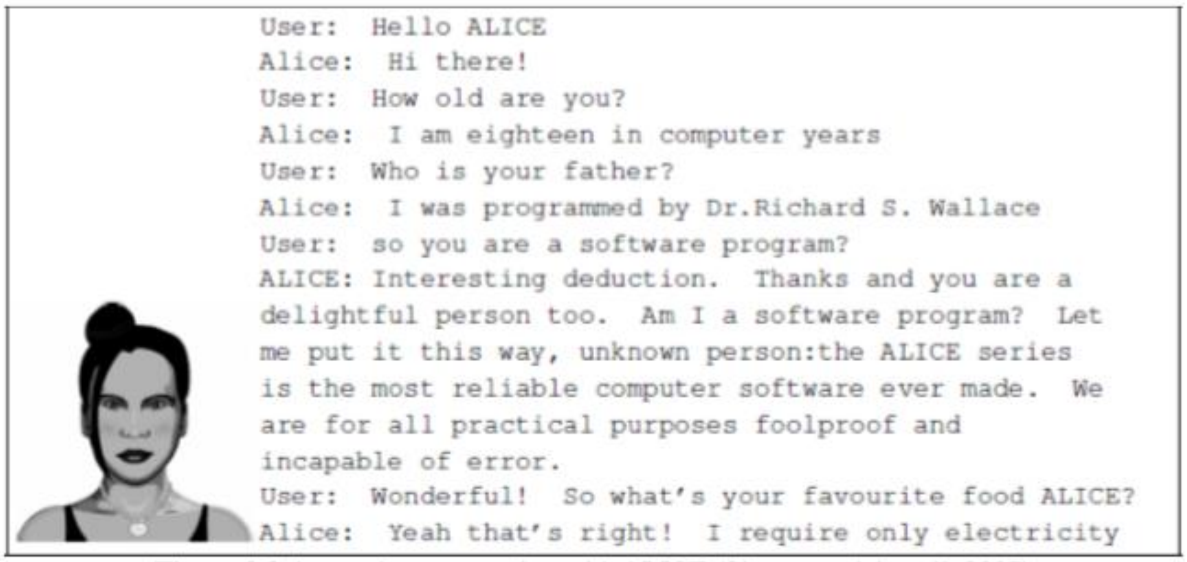
```
 Welcome to
                  EEEEEE  LL      IIII  ZZZZZZZ   AAAAA
                  EE      LL       II       ZZ   AA   AA
                  EEEEE   LL       II     ZZZ    AAAAAAA
                  EE      LL       II    ZZ      AA   AA
                  EEEEEE  LLLLLL  IIII  ZZZZZZZ  AA   AA

 Eliza is a mock Rogerian psychotherapist.
 The original program was described by Joseph Weizenbaum in 1966.
 This implementation by Norbert Landsteiner 2005.


ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:   Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU:   He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:   It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:
```

Figure 2.3: Sample chat with ELIZA

2.2.2 A.L.I.C.E

A.L.I.C.E. (Artificial Linguistic Internet Computer Entity), also known as Alicebot, is another natural language processing chatbot. This engages in conversations by applying some heuristical pattern matching rules to the user's input [3]. It is also inspired by ELIZA. But ALICE is more significant and complex. Since it generates responses to queries by using some pattern matching inputs like <pattern> (input) <template> (output) pairs stored in documents in a knowledge base. These documents are written in Artificial Intelligence Markup Language (AIML). AIML is an extension of XML.

Below Figure 2.4 shows a conversation done with ALICE.



```
User:  Hello ALICE
Alice:  Hi there!
User:  How old are you?
Alice:  I am eighteen in computer years
User:  Who is your father?
Alice:  I was programmed by Dr.Richard S. Wallace
User:  so you are a software program?
ALICE: Interesting deduction.  Thanks and you are a
delightful person too.  Am I a software program?  Let
me put it this way, unknown person:the ALICE series
is the most reliable computer software ever made.  We
are for all practical purposes foolproof and
incapable of error.
User:  Wonderful!  So what's your favourite food ALICE?
Alice:  Yeah that's right!  I require only electricity
```

Figure 2.4: Sample chat with ALICE

2.3.3 Modern Chatbots

Amazon shopping website is also using this concept of chatbot to address one of the major issues that their customers are facing, that is when a customer needs to return any item that he/she has already bought, he/she has to undergo an hectic process. Because customers don’t have information on how to return items. To get those, they have to send emails, call help desk and wait for a ling time until a customer representative connects with the call. This process is difficult and sometimes can be very irritating for customers. In order to address this issue, Amazon has implemented a chatbot to answer simple queries of customers. It is Amazon Shopping App. One drawback observed in this chatbot is the lack of personality and poor conversational flow.

CNN has also developed a chatbot. But a major drawback of it is that lack of sympathy and effortlessness. Chatbot should be able to relate and associate with the users. For example, a in CNN chatbot, if a user says anything irrelevant to news or some other thing, it answers with news items which contains those words and at the end it says "Not sure I understand what you're looking for. Try again”. This implies that CNN bot needs compassion [4].

Sentiment analysis must be incorporated in websites that has embedded chatbots to handle user queries. Understanding user’s feelings is known as sentiment analysis. Whether the user expresses his/her feelings in positive, negative or in a neutral manner. The bot is trained to provide his answers with an empathy to the users based on that. If a user says, “I am sad”, then the bot should reply with some empathy like “I’m sorry to hear that, how can I help you?”. Not just reply with the default message like “Sorry did not understand your question.” [5] .

Another issue of chatbots are they are designed in a way to follow a specific code and answer specific set of questions. So, they fail to answer anything outside the code. This means that if the question asked by the user is not properly handled by the code, bot will fail to understand the question and it will give a bad experience for the users. A concept called active learning can be introduced to the system to overcome such issues[6].

Using Active learning, it builds a conversation with user to obtain the desired output. When a user asks something outside code, the chatbot will ask several more questions from the user and based on the user’s input, the bot returns the most suitable answer to the asked query[6].

2.3 Comparison of alternative design strategies

In this chapter a general comparison of the alternative design strategies is given. Here it discusses the approaches, frameworks and third-party APIs used to implement this project, along with a clear justification to use them and a comparison with other alternative technologies available.

- **Using Keyword Recognition Based Chatbots approach over Contextual Chatbots approach**
  There are two main chatbot approaches to be considered when implementing a chatbot application. One is Contextual and the other one is Keywords based.
  Contextual Chatbots – Contextual are more advanced than the Keyword Recognition Based Chatbots. They are based on Machine Learning and Artificial Intelligence. They remember the conversations with users, and they learn from it and grow over time [7]. They are smart in recognizing and learning based on what user is asking and how user is asking it.

Example: A contextual chatbot which facilitates users to order clothes in a shopping website and it will store the data from all the conversations happening with users and learn what the users like to order. When a user chats with this chatbot, it will eventually remember their size, favorite colors, delivery address and will ask if they would like to repeat this order in future when the user tries to order something again.
Keyword Recognition Based Chatbots – Keyword Recognition Based chatbots can listen to what users type and respond accordingly. These chatbots are implemented based on customizable keywords and Artificial Intelligence to give the most appropriate response to the user [7].
Example: If a user asks 'When will the university opens?', the bot will use the keywords 'when', 'university', and 'opens', to best determine the best answer to that he can use to respond.
Since this project is implemented to respond the queries of prospective students in a university, the chatbot should reply to students with up to date and correct answers. Since contextual chatbots evolve with the time and by learning user inputs, they cannot specifically answer to the updating questions specially when related to dates and time. In this project Administrator is feeding the chatbot with up to date answers and chatbot learns them through AI and returns the most appropriate answers by matching the keywords. In order to implement a project in this domain, Keyword recognition chatbots is the best approach to use.

- **Using Flask Framework over Django Framework**
  A framework is a code storage. It helps developers' work easier, scalable, efficient and maintainable. It provides reusable code or extensions for common operations [8]. Both Django and Flask are web frameworks for Python. Django provides a full-featured Model-View-Controller (MVC) Framework while Flask is a micro-framework.
  Django – Django helps with simplifying website creation. It is mainly focused reusability of components, less code, low coupling, rapid and fast-paced development. Python is the primary language used by Django [9]. Django is better to integrate with databases like Oracle, SQLite, PostgreSQL and MySQL.
  Flask - Flask is a lightweight web application micro-framework. It has been designed to help you get started quickly and easily with web development. Best to integrate with NoSQL databases. Flask has a number of extensions as well such as Flask-RESTful, Flask-Classful, Flask-REST Plus for Views, Flask-Marshmallow for Serialization, Flask-JWT, Flask-JWT-Extended for Authentication [10].
  In this project Flask is used as the python framework since it is a lightweight micro framework. It has less complications and easy to use. It is easy to start with Flask and can scale up to complex applications. Since Flask is smaller and having few layers, it performs well than Django. Flask-JWT is used for authentication purposes in this application.

- **Using Google Cloud Platform over Amazon Alexa**
  Both Google Cloud Platform and Amazon Alexa are smart voice assistants. In this project a voice assistant is required to implement the text-to-speech and speech-to-text features. Both these services have released third-party APIs to consume.

Creating an Amazon account and a Google cloud platform account is needed. Google Cloud Platform services can be used with 1USD and whereas for Amazon Alexa a payment is required. When considering the cost Google Cloud Platform APIs are used as third-party APIs to implement voice assistant features in the chatbot.

- **Using Python over Java**
  Both Java and Python are object-oriented programming languages that can be used for web-based application development process. In this project aim is to develop a chatbot system based on artificial intelligent. Python is a powerful machine learning language when compared to java and python has inbuilt libraries that can be reused in Artificial Intelligence related projects. Therefore, Python is the best choice for a programming language here.

- **Using NoSQL Database over SQL database**
  MongoDB, a NoSQL database is used as the database of this developed application. NoSQL database is used over a SQL database is that NoSQL databases can store large amount of data and it is scalable. Also, high-performance data processing is a great feature of NoSQL databases than SQL.

Chapter 3

3. METHODOLOGY

3.1 Software development methodology

Incremental development methodology along with agile concepts is used as the development methodology of this system. In incremental process, software is built and delivered in pieces. Each piece represents a complete functionality [11]. In this project the functionalities of the project were broken down into high-level user stories as below. At the end of each user story a working piece of application is delivered.

- Finalizing an approach to implement the system and the technologies.
- Designing the system.
- Chatbot algorithm is implemented with some dummy values to be run in the command prompt. A chat application that runs in the command prompt was delivered.
- UI was created for the chat application.
- More data were inserted to brain files and tested whether the chat application is working as expected.
- Admin portal was created and added data to brain files through the admin portal.
- Database integration is done.
- User queries of the chat application were saved in the database and displayed through the administrator portal
- A static webpage for a university was developed and embedded the chat application in it.
- A student forum was developed.
- Text-to-speech feature was developed for phone calls.

3.2 System Analysis

The basic requirement of this chatbot system is to provide the most suitable answers for the students' enquiries related to the university processes. Queries can be answered in both text and text-to-speech format. A web interface of a chatbot is implemented for students to interact with the system and an administration interface for admin related activities. This chat application is embedded inside the general website of the university, so that everyone has the access to it.

The chatbot developed for this project is a contextual chatbot using Natural Language Processing. These are so far the most advanced chatbots. It is a combination of rule-based and keyword chatbots. These chatbots use NLP (Natural Language Processing) to understand the context and intent in users' requests and acts accordingly. These chatbots can handle multiple requests at the same time.

Chatbot responding with text output

The below Figure 3.1 depicts a high-level workflow diagram of the chatbot system when user input queries in the text format.

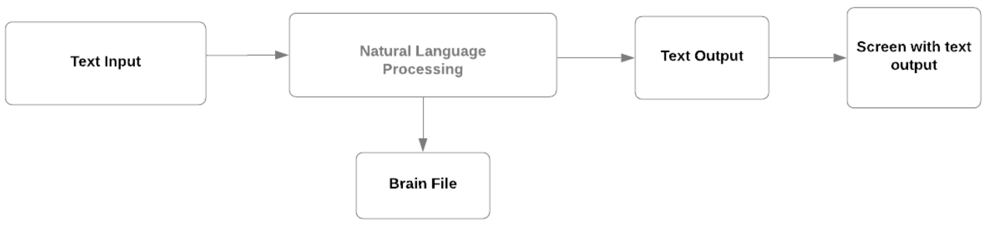


Figure 3.1: Data flow diagram of text chat

Chatbot responding with audio output

The below Figure 3.2 depicts a high-level workflow diagram of the system, when a user asks their question in audio format.

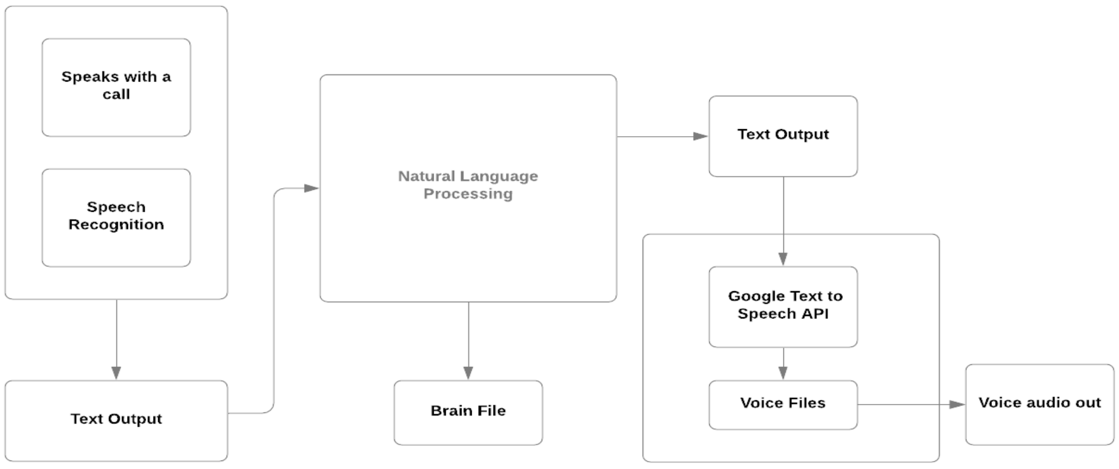


Figure 3.2: Data flow diagram of audio chat

3.3 High level architecture

Chatbot can receive "input" as plain text about the college related queries and "output" is a trained conversational agent which is able to answer mainly the questions related to the admission process.

This project is developed using MVC (Model-View-Controller) architecture.

Here the system administrator feeds some knowledge to the machine so that machine can identify the sentences and taking a decision itself as a response to a question. The chat used is English conversational pattern and the Natural Language Processing is used. Python is the platform in which the bot is created in this project using a Flask framework which integrates with MongoDB. REST APIs are created for effective and efficient data communication between frontend and backend of the system. Pusher channel, instance of a Stun Server is used for Session controlling and to handle messages used to open or close communication. This helps in real time chatting.

ChannelID, a unique id is created for each user which stores in the browser local storage, used to control the session of the user and it helps the user to have a conversational chat with the chatbot, which preserves user's chat history in the browser.

A JWT token is created for user authentication. Therefore, unauthorized users cannot access the administrator portal of the system. The expiration time of the token can be configured.

Below Figure 3.3 shows the structure of the JWT token.

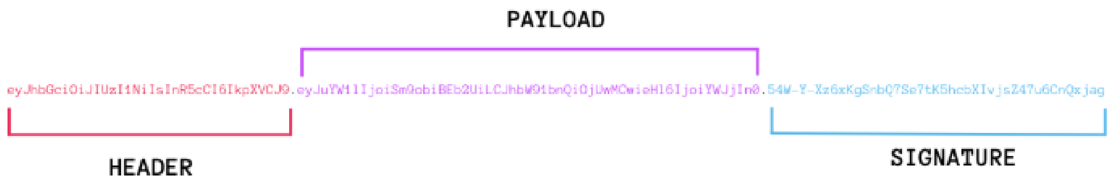

PAYLOAD
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYW1lIjoiSm9obiBEb2UiLCJhbW91bnQiOjUwMCwieHl6IjoiYWJjIn0.54W-Y-Xz6xKgSnbQ7Se7tK5hcbXIvjsZ47u6CnQxjag
HEADER
SIGNATURE

Figure 3.3: JWT Token structure

<u>Header</u>

```
{
"alg": "HS256",
"typ": "JWT"
}
```

Alg – Algorithm used

Typ - Defines the type of the token. Here it is JWT

<u>Payload</u>

```
{
"userId": "dhaodhod79203g2wkdb",
"exp": " 2020-05-17T07:22Z"
}
```

Custom values can be given for the payload. In this project userId and the JWT token expiration time is used when calculating the payload.

Signature

Signature is composed by base64url encoding of header and payload. Then concatenates it with a period as a separator. Finally, it is encrypted with HMAC-SHA256 along with the secret key and the result of the first step.

```
key = 'secretkey';
unsignedToken = encodeBase64Url(header) + '.' +
encodeBase64Url(payload);
signature = HMAC-SHA256(key, unsignedToken) // As mentioned in
header section.
```

Administrator passwords are stored in the database encoded as SHA256. Therefore, user details are safely stored in the database.

Chatbot takes decisions with Python as the main aid, various answers to the questions of the students is stored as the list in Python. The code checks the related answers in the list for a specific query and fetches the information. There are various references stored in the Artificial Intelligence markup language that are used to fetch the answers matching the reference.

All this fetching process is done by using the Python programming using REST APIs. Specifically, it uses the packages Flask to create the APIs. All the data is stored in the MongoDB and admin can create a brain file. Therefore, whenever a user asks a query, it will fetch data from maintained by admin and MongoDB is used to store user session details, admin login details, user feedbacks and user queries.

Figure 3.4 displays the high-level architecture of the system.

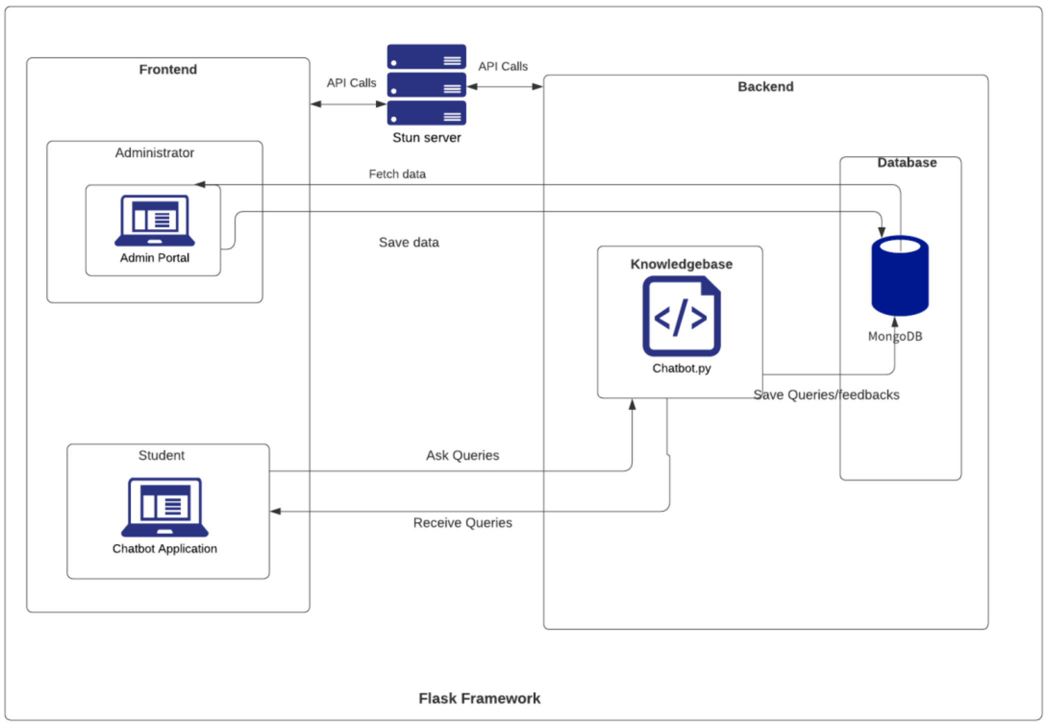
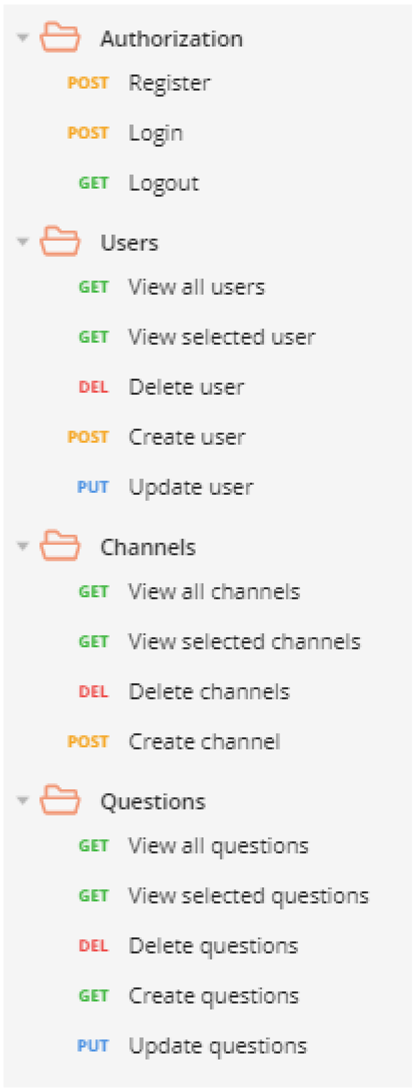


Figure 3.4: High-level architecture diagram of the system

Application Programming Interfaces(API)

Several APIs were developed to communicate the information among the several components within the system. Below Figure 3.5 is a snapshot taken from postman collection with all the APIs that were implemented for this system.



Authorization
- POST Register
- POST Login
- GET Logout

Users
- GET View all users
- GET View selected user
- DEL Delete user
- POST Create user
- PUT Update user

Channels
- GET View all channels
- GET View selected channels
- DEL Delete channels
- POST Create channel

Questions
- GET View all questions
- GET View selected questions
- DEL Delete questions
- GET Create questions
- PUT Update questions

Figure 3.5: APIs of the system

3.4 Database Design

Since this project is an AI based system, it deals with two databases. One is knowledgebase and the other one is the typical database.

Knowledgebase is necessary to store all the data files that contains the knowledge of the system with possible answers for the student's queries. It contains several data files related to various topics that the students might ask questions. Information entered by administrator in the admin portal will be stored in these data files and those will be treated as the knowledge to answer user queries.

MongoDB is used as the database of this system to store the session details of the users, admin details, queries of the users and feedbacks given by users.

3.4.1 Knowledgebase

NLU (Natural Language Understanding)

Natural Language Processing has 3 main concepts as Entities, Intents and Context.

Entities: This represents a concept in the Chatbot.

Intents: This defines the action a chatbot should perform when a user asks for an enquiry. Example, intent can trigger the same thing if user asks, "What is the university starts date?", "When will the university opens again?" all these user enquiries trigger single command and display the university start date to the user.

Context: When a NLU algorithm analyzes a sentence, it does not have the history of the user's chat conversation. That means, if it receives the answer to a question it has just asked, it will not remember the question. to differentiate the phases during a chat, it's state should be stored. With context, intents can be related easily. Here it is not needed to know what the previous question was.

NLP (Natural Language Processing)

NLP takes some steps to convert the customer's text or speech into structured data that is used to select the related answer. Some of the Natural Language Processing steps are as follows:

- Sentiment Analysis: Here it tries to learn whether the user is having a good experience or whether the chat should be forwarded to a human.
- Tokenization: NLP divides a string of words into pieces or tokens which are linguistically symbolic or are differently useful for the application.
- Named Entity Recognition: Chatbot program looks for categories of words, like the names of the courses, the student's names, or whatever the required data.
- Normalization: Here it processes texts in order to find common spelling mistakes or typographical errors that the user might be doing. With this feature, users will feel like they are chatting with a real human.

- Dependency Parsing: Chatbot looks for the objects and subjects- verbs, nouns and common phrases in the user’s text to find dependent and related phrases that users might be trying to ask *[12]*.

3.4.2 Database

MongoDB is used as the database of this system to store the session details of the users, admin details, queries of the users, expired session tokens, posts, user comments, user reactions, enquiry and feedbacks given by users. MongoDB Database is “Chatbot” and below Figure 3.6 shows the Database Structure.

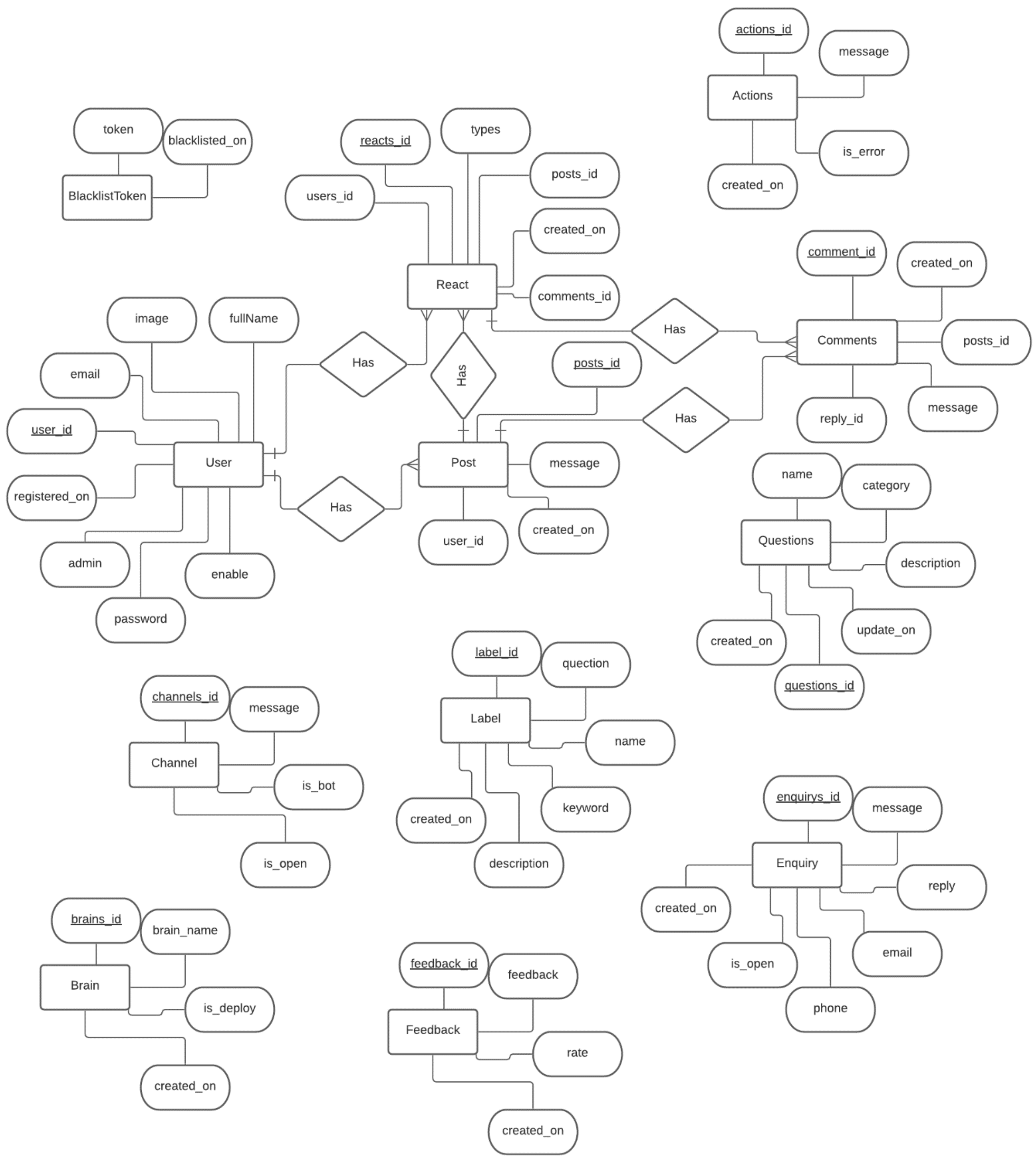
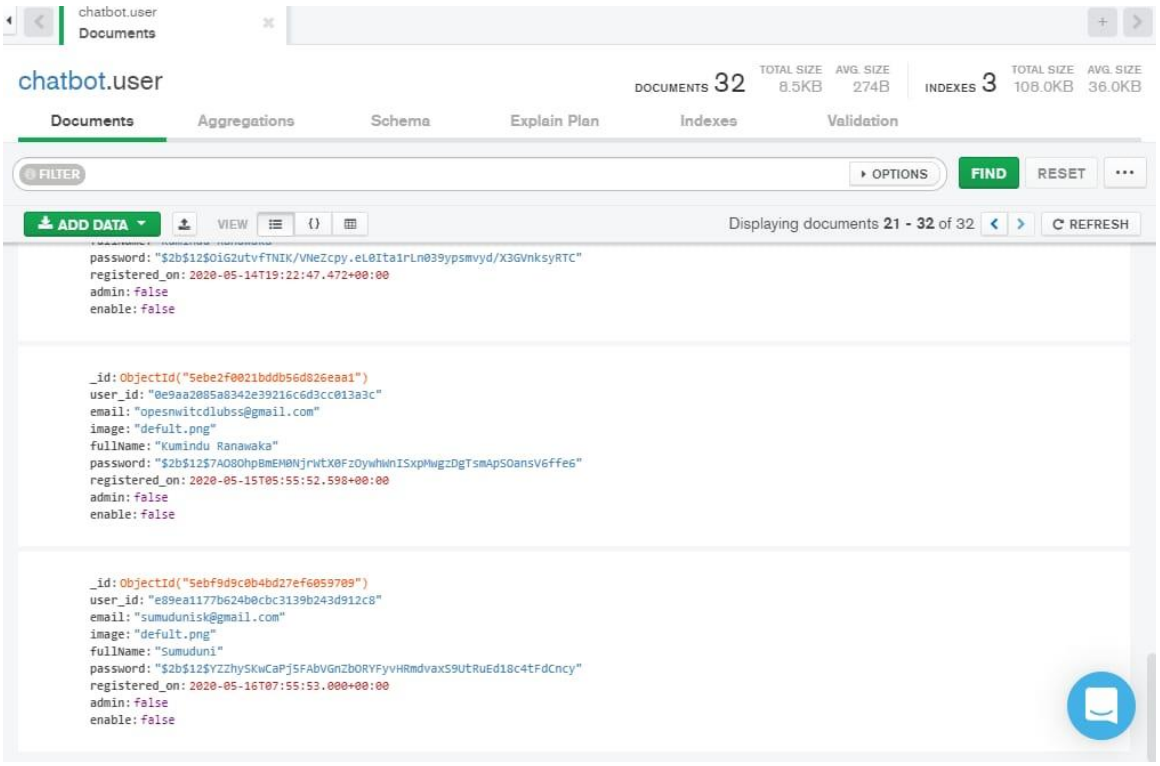


Figure 3.6: Database design diagram

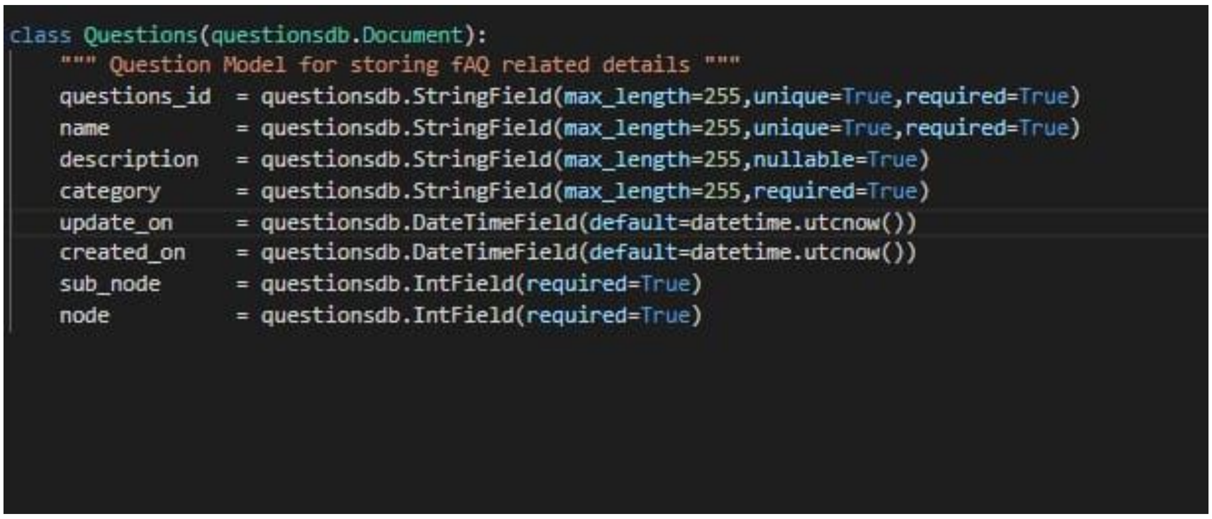
Below Figure 3.7, shows the collection that stores user data. A unique userId is given for each user. Password is encoded with SHA256. A flag is used as "admin" to identifies if the user is an admin or not. And "enable" attribute shows whether the user is active or not.



```
password: "$2b$12$OiG2utvfTNIK/VNeZcpy.eL0Ita1rLn039ypsmvyd/X3GVnksyRTC"
registered_on: 2020-05-14T19:22:47.472+00:00
admin: false
enable: false

_id: ObjectId("5ebe2f0021bddb56d826eaa1")
user_id: "0e9aa2085a8342e39216c6d3cc013a3c"
email: "opesnwitcdlubss@gmail.com"
image: "defult.png"
fullName: "Kumindu Ranawaka"
password: "$2b$12$7AO8OhpBmEM0NjrWtX0FzOywhWnISxpMwgzDgTsmApSOansV6ffe6"
registered_on: 2020-05-15T05:55:52.598+00:00
admin: false
enable: false

_id: ObjectId("5ebf9d9c0b4bd27ef6059709")
user_id: "e89ea1177b624b0cbc3139b243d912c8"
email: "sumudunisk@gmail.com"
image: "defult.png"
fullName: "Sumuduni"
password: "$2b$12$YZZhySKwCaPj5FAbVGnZbORYFyvHRmdvaxS9UtRuEd18c4tFdCncy"
registered_on: 2020-05-16T07:55:53.000+00:00
admin: false
enable: false
```

Figure 3.7: User details collection

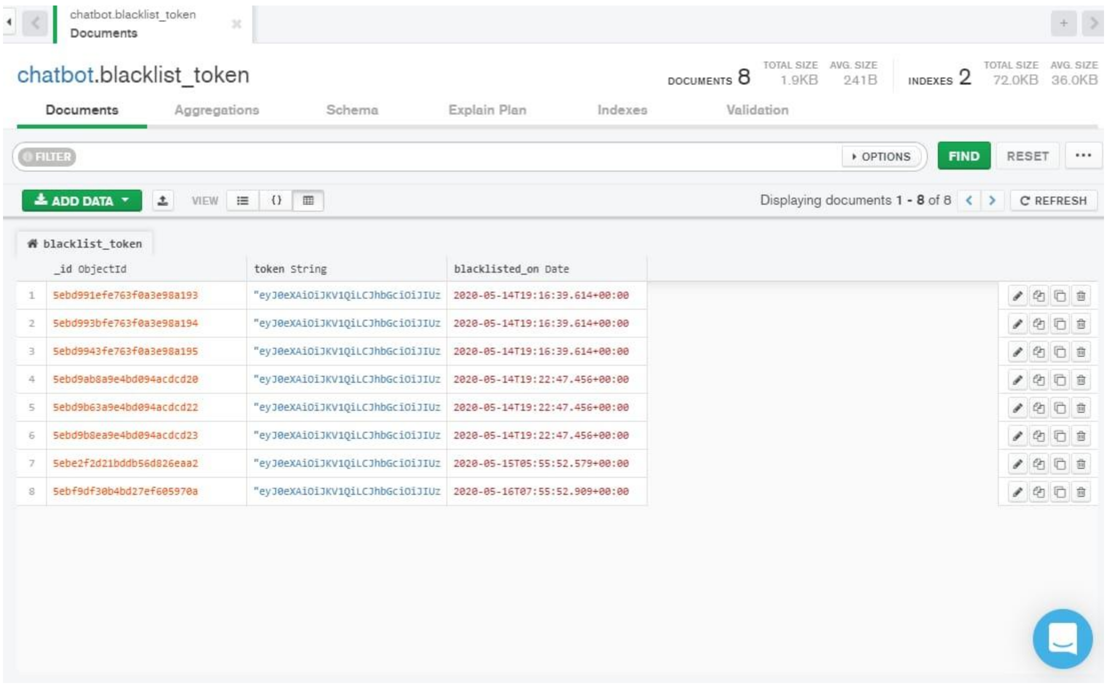
Below Figure 3.8 shows the class written for user details collection.



```python
class Questions(questionsdb.Document):
    """ Question Model for storing fAQ related details """
    questions_id  = questionsdb.StringField(max_length=255,unique=True,required=True)
    name          = questionsdb.StringField(max_length=255,unique=True,required=True)
    description   = questionsdb.StringField(max_length=255,nullable=True)
    category      = questionsdb.StringField(max_length=255,required=True)
    update_on     = questionsdb.DateTimeField(default=datetime.utcnow())
    created_on    = questionsdb.DateTimeField(default=datetime.utcnow())
    sub_node      = questionsdb.IntField(required=True)
    node          = questionsdb.IntField(required=True)
```

Figure 3.8: Class of User queries collection

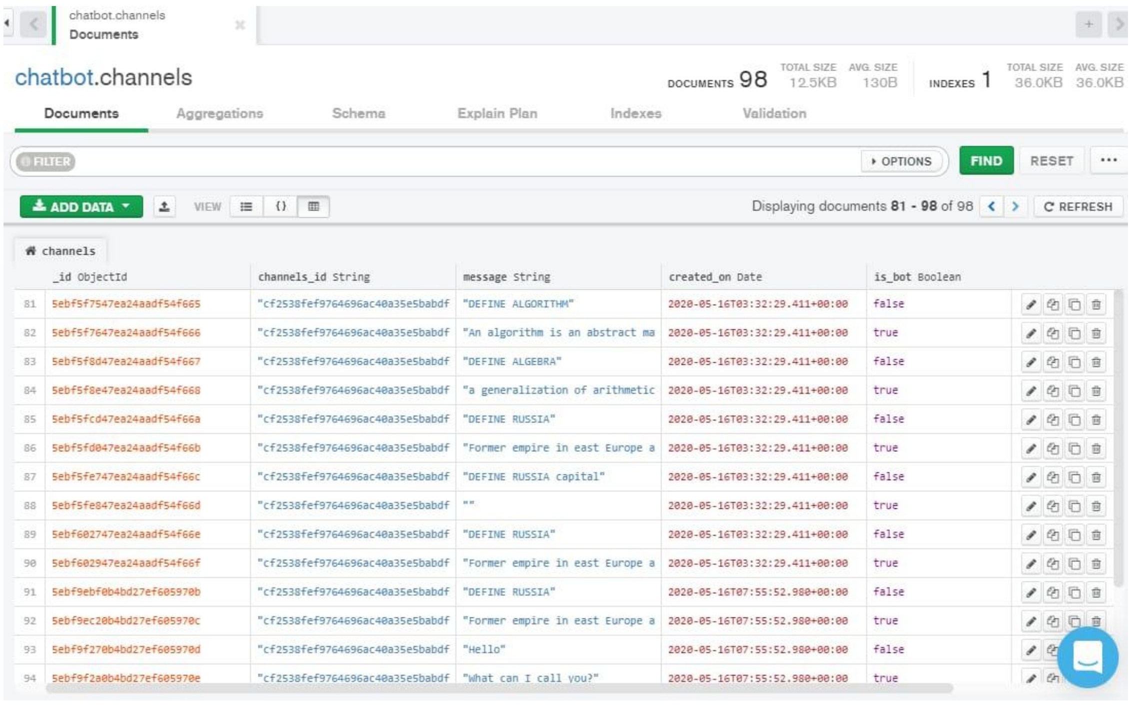
User session expiration time can be configured in the property file. Default expiration time is given as 2 days. Below Figure 3.9 shows the JWT tokens and their expiration time.



	_id ObjectId	token String	blacklisted_on Date
1	5ebd991efe763f0a3e98a193	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:16:39.614+00:00
2	5ebd993bfe763f0a3e98a194	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:16:39.614+00:00
3	5ebd9943fe763f0a3e98a195	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:16:39.614+00:00
4	5ebd9ab8a9e4bd094acdcd20	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:22:47.456+00:00
5	5ebd9b63a9e4bd094acdcd22	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:22:47.456+00:00
6	5ebd9b8ea9e4bd094acdcd23	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-14T19:22:47.456+00:00
7	5ebe2f2d21bddb56d826eaa2	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-15T05:55:52.579+00:00
8	5ebf9df30b4bd27ef605970a	"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz	2020-05-16T07:55:52.909+00:00

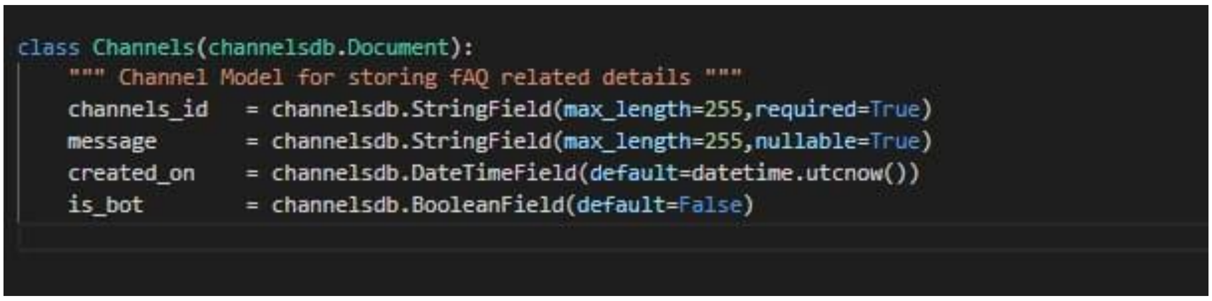
Figure 3.9: User sessions collection

A unique channelID is issued for each user to uniquely identify their chat conversations and to preserve data. The user queries and the answers given by the bot as stored in the database as in the below Figure 3.10. "Is_bot" attributes define whether it's a question asked by the user or whether it's the answer given by the chatbot.



	_id ObjectId	channels_id String	message String	created_on Date	is_bot Boolean
81	5ebf5f7547ea24aadf54f665	"cf2538fef9764696ac40a35e5babdf	"DEFINE ALGORITHM"	2020-05-16T03:32:29.411+00:00	false
82	5ebf5f7647ea24aadf54f666	"cf2538fef9764696ac40a35e5babdf	"An algorithm is an abstract ma	2020-05-16T03:32:29.411+00:00	true
83	5ebf5f8d47ea24aadf54f667	"cf2538fef9764696ac40a35e5babdf	"DEFINE ALGEBRA"	2020-05-16T03:32:29.411+00:00	false
84	5ebf5f8e47ea24aadf54f668	"cf2538fef9764696ac40a35e5babdf	"a generalization of arithmetic	2020-05-16T03:32:29.411+00:00	true
85	5ebf5fcd47ea24aadf54f66a	"cf2538fef9764696ac40a35e5babdf	"DEFINE RUSSIA"	2020-05-16T03:32:29.411+00:00	false
86	5ebf5fd047ea24aadf54f66b	"cf2538fef9764696ac40a35e5babdf	"Former empire in east Europe a	2020-05-16T03:32:29.411+00:00	true
87	5ebf5fe747ea24aadf54f66c	"cf2538fef9764696ac40a35e5babdf	"DEFINE RUSSIA capital"	2020-05-16T03:32:29.411+00:00	false
88	5ebf5fe847ea24aadf54f66d	"cf2538fef9764696ac40a35e5babdf	""	2020-05-16T03:32:29.411+00:00	true
89	5ebf602747ea24aadf54f66e	"cf2538fef9764696ac40a35e5babdf	"DEFINE RUSSIA"	2020-05-16T03:32:29.411+00:00	false
90	5ebf602947ea24aadf54f66f	"cf2538fef9764696ac40a35e5babdf	"Former empire in east Europe a	2020-05-16T03:32:29.411+00:00	true
91	5ebf9ebf0b4bd27ef605970b	"cf2538fef9764696ac40a35e5babdf	"DEFINE RUSSIA"	2020-05-16T07:55:52.980+00:00	false
92	5ebf9ec20b4bd27ef605970c	"cf2538fef9764696ac40a35e5babdf	"Former empire in east Europe a	2020-05-16T07:55:52.980+00:00	true
93	5ebf9f270b4bd27ef605970d	"cf2538fef9764696ac40a35e5babdf	"Hello"	2020-05-16T07:55:52.980+00:00	false
94	5ebf9f2a0b4bd27ef605970e	"cf2538fef9764696ac40a35e5babdf	"What can I call you?"	2020-05-16T07:55:52.980+00:00	true

Figure 3.10: User queries collection



```
class Channels(channelsdb.Document):
    """ Channel Model for storing fAQ related details """
    channels_id    = channelsdb.StringField(max_length=255,required=True)
    message        = channelsdb.StringField(max_length=255,nullable=True)
    created_on     = channelsdb.DateTimeField(default=datetime.utcnow())
    is_bot         = channelsdb.BooleanField(default=False)
```

Figure 3.11: Class for user queries collection

Above Figure 3.11 shows the class written for Channels collection.

3.5 UI Designs

When designing the UI, user friendliness should be considered as the main priority. With the nature of this project, it should consist of an attractive Graphical User Interface (GUI) for the chatbot application, to attract students to build up a conversation with the system. And it should be mobile friendly and compatible with main browsers and operating systems.

When considering the administrator portal, the system design should be done in such a way that only authorized users can access the admin portal. The user interface should be interactive and user-friendly.

University Web Page

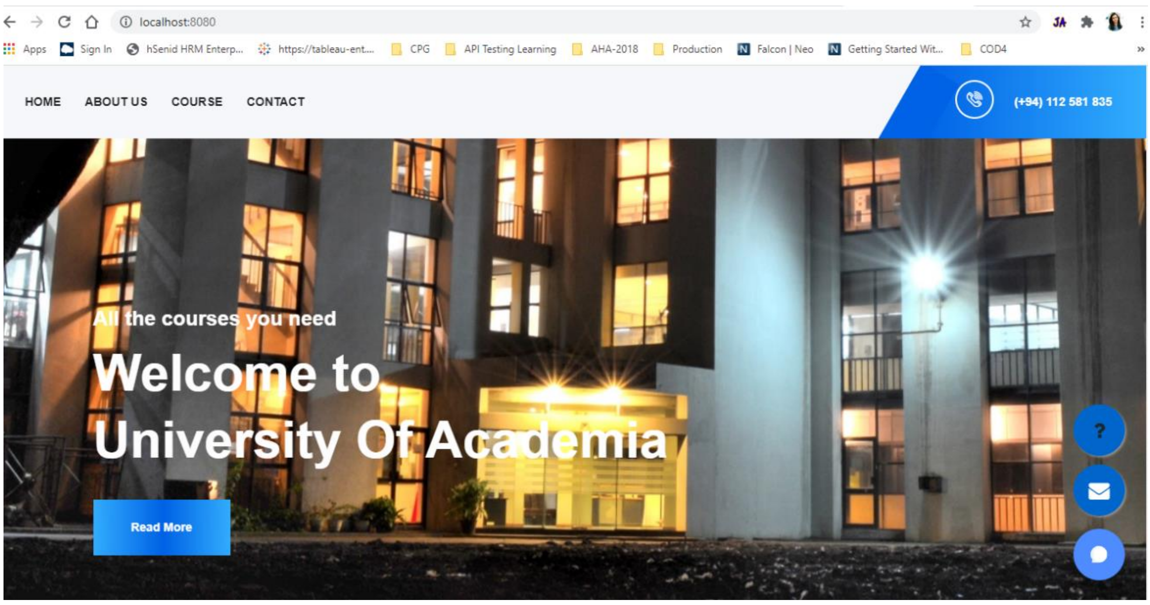
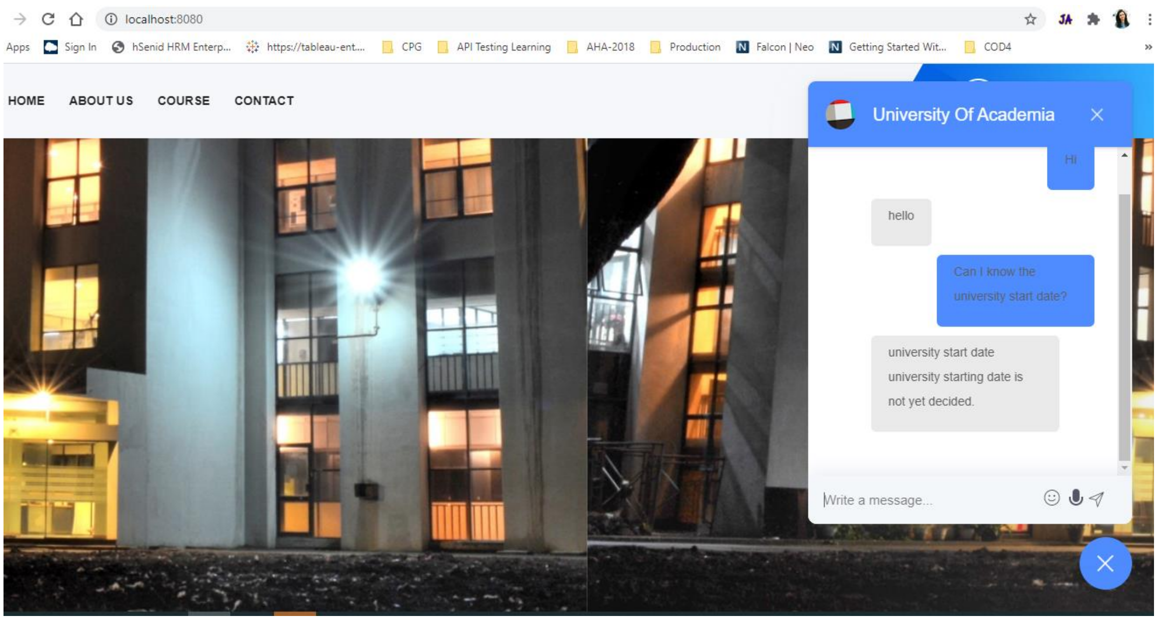


Figure 3.12: University web page

Chat Application Interface



Hi

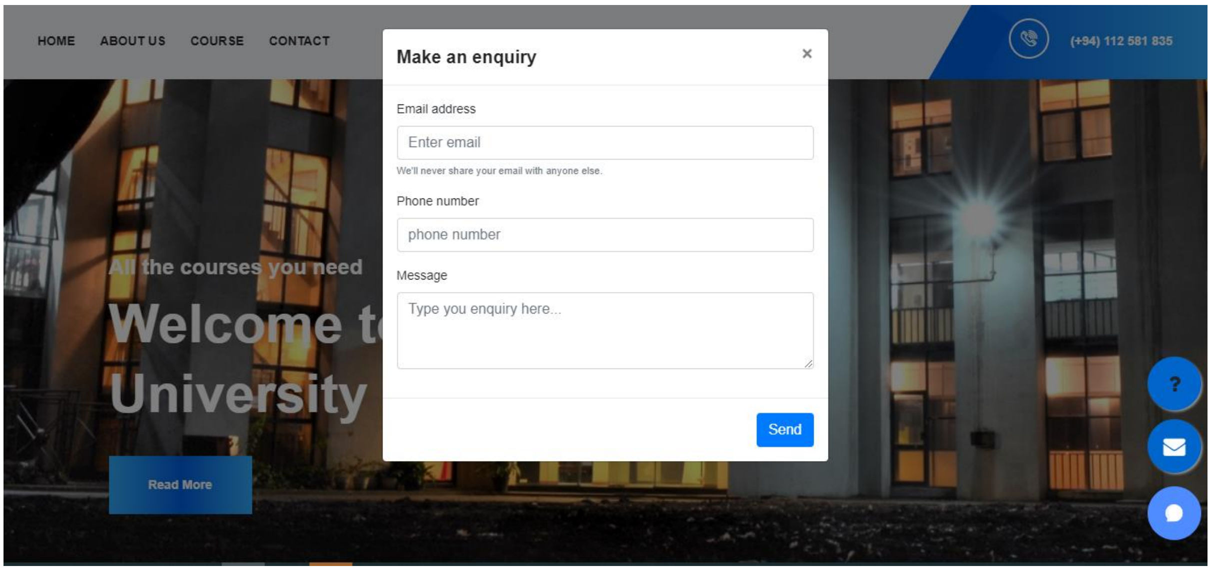
hello

Can I know the university start date?

university start date
university starting date is not yet decided.

Figure 3.13: Chatbot application interface

Student Enquiry Interface



Make an enquiry ×

Email address

Enter email

We'll never share your email with anyone else.

Phone number

phone number

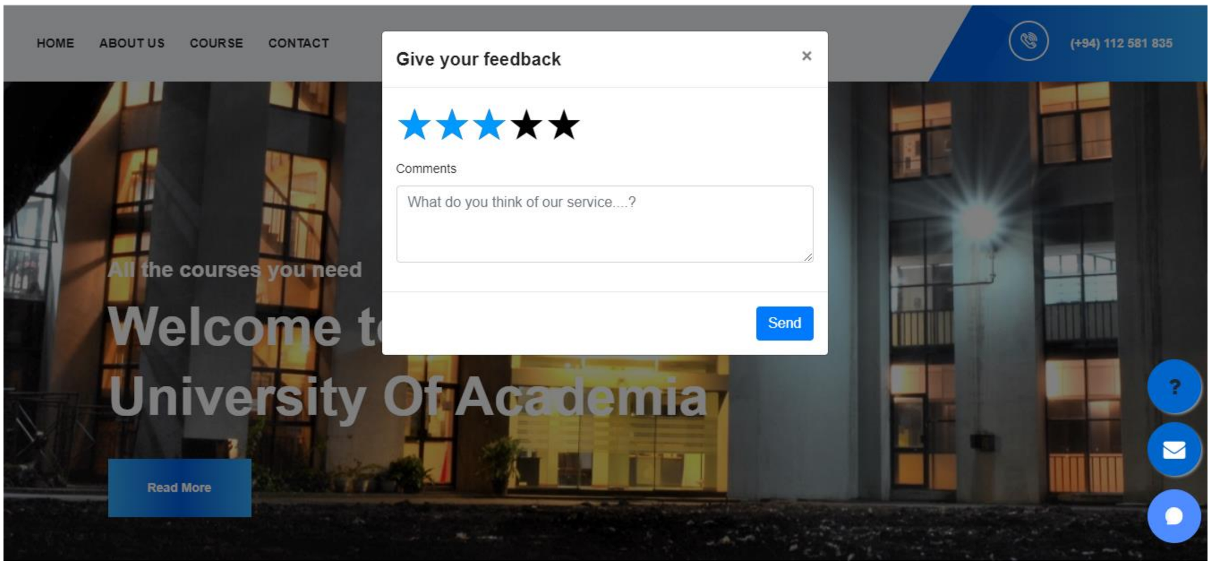
Message

Type you enquiry here...

Send

Figure 3.14: Student enquiry interface

Student Feedback Interface



Give your feedback ×

★★★★★

Comments

What do you think of our service....?

Send

Figure 3.15: Student feedback interface

Login Page for both admin and student

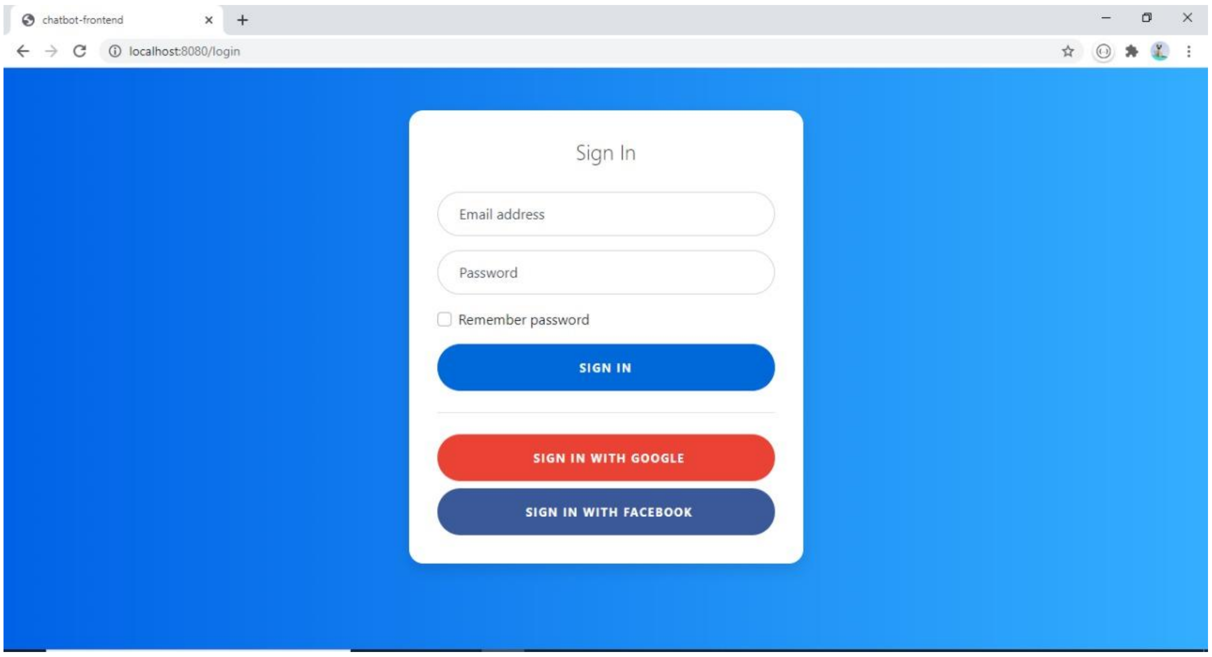


Figure 3.16: Administrator portal - Login page

Signup page for both admin and student

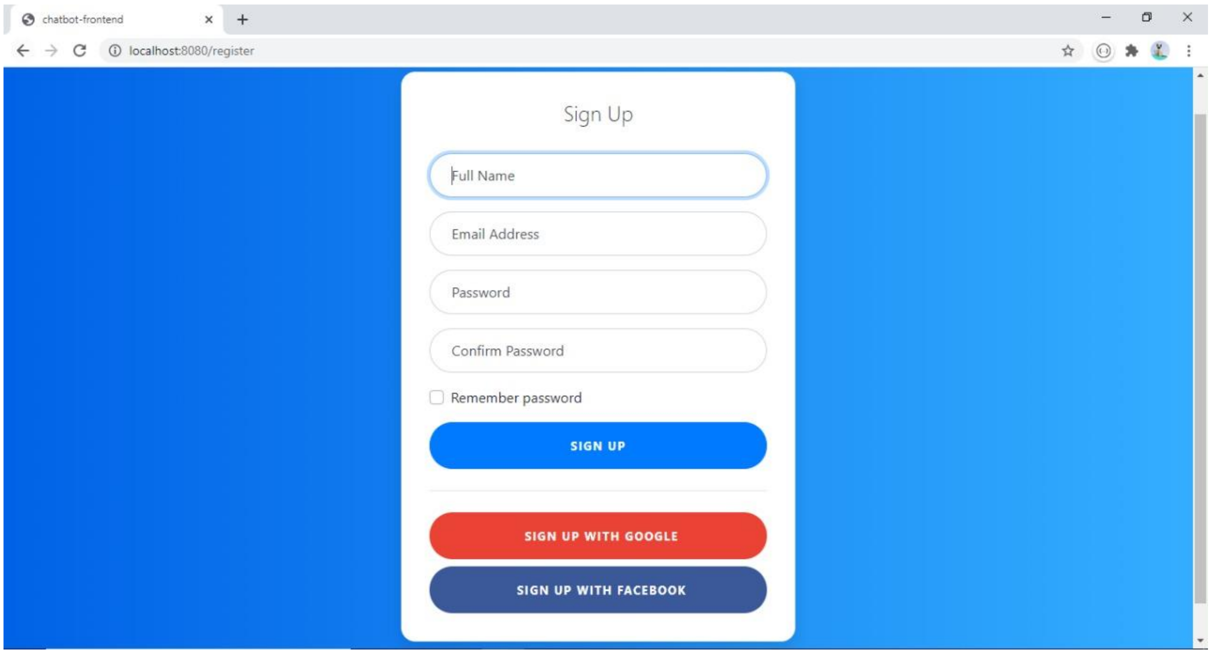


Figure 3.17: Administrator Portal - Signup page

Administrator Portal - Dashboard Page

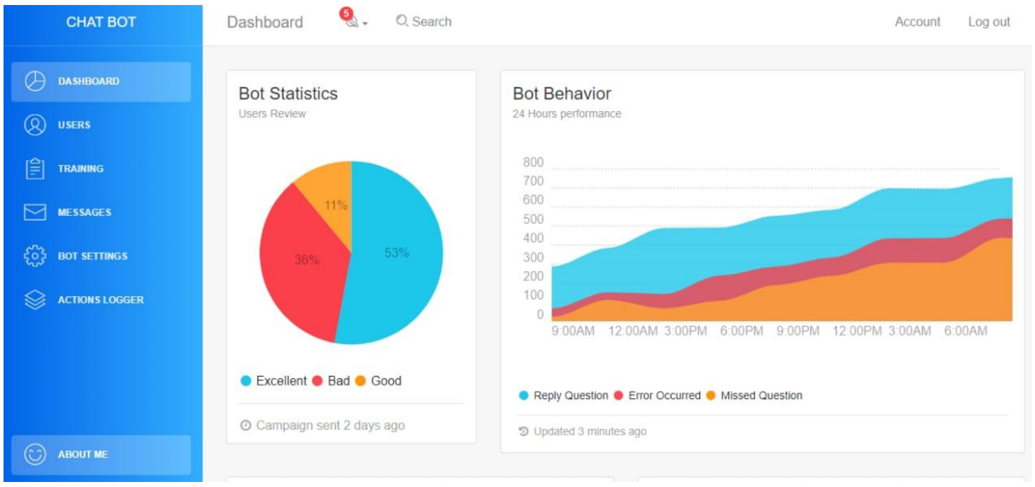
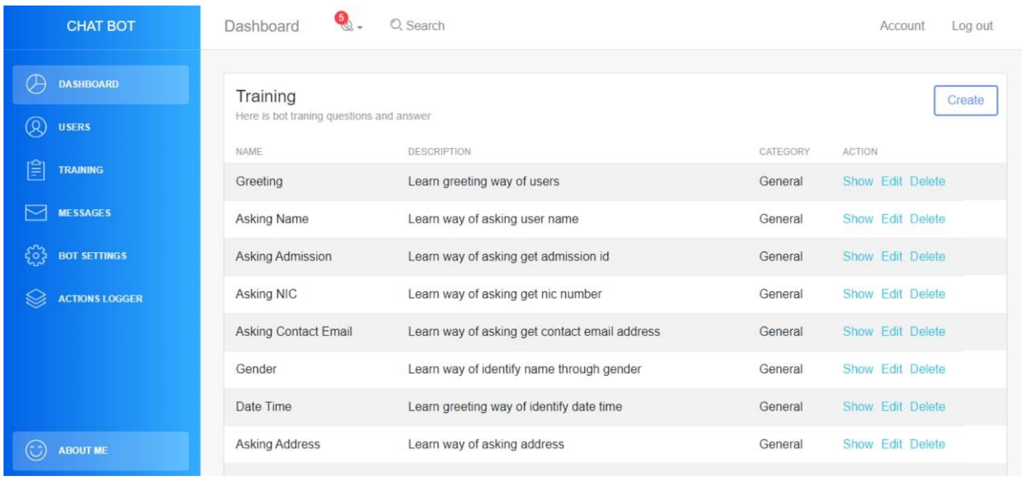


Figure 3.18: Administrator portal - Dashboard page

Administrator Portal – Chatbot Training



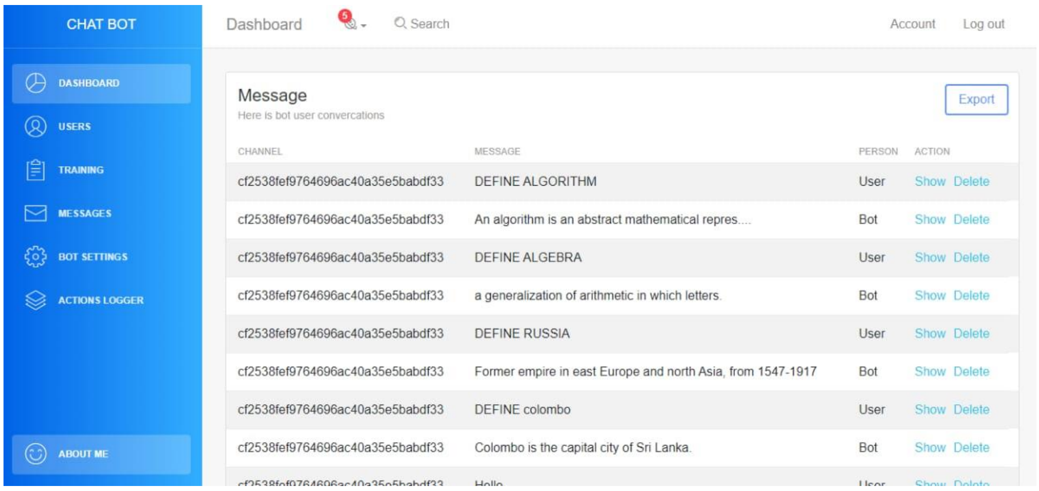
Training

Here is bot traning questions and answer

NAME	DESCRIPTION	CATEGORY	ACTION
Greeting	Learn greeting way of users	General	Show Edit Delete
Asking Name	Learn way of asking user name	General	Show Edit Delete
Asking Admission	Learn way of asking get admission id	General	Show Edit Delete
Asking NIC	Learn way of asking get nic number	General	Show Edit Delete
Asking Contact Email	Learn way of asking get contact email address	General	Show Edit Delete
Gender	Learn way of identify name through gender	General	Show Edit Delete
Date Time	Learn greeting way of identify date time	General	Show Edit Delete
Asking Address	Learn way of asking address	General	Show Edit Delete

Figure 3.19: Administrator Portal - Chatbot training

Administrator Portal – View User queries and feedbacks



Message

Here is bot user convercations

CHANNEL	MESSAGE	PERSON	ACTION
cf2538fef9764696ac40a35e5babdf33	DEFINE ALGORITHM	User	Show Delete
cf2538fef9764696ac40a35e5babdf33	An algorithm is an abstract mathematical repres....	Bot	Show Delete
cf2538fef9764696ac40a35e5babdf33	DEFINE ALGEBRA	User	Show Delete
cf2538fef9764696ac40a35e5babdf33	a generalization of arithmetic in which letters.	Bot	Show Delete
cf2538fef9764696ac40a35e5babdf33	DEFINE RUSSIA	User	Show Delete
cf2538fef9764696ac40a35e5babdf33	Former empire in east Europe and north Asia, from 1547-1917	Bot	Show Delete
cf2538fef9764696ac40a35e5babdf33	DEFINE colombo	User	Show Delete
cf2538fef9764696ac40a35e5babdf33	Colombo is the capital city of Sri Lanka.	Bot	Show Delete
cf2538fef9764696ac40a35e5babdf33	Hello	User	Show Delete

Figure 3.20: Administrator Portal - View user queries and feedbacks

3.6 Technologies

- HTML/CSS and Bootstrap
  These languages are used to design and implement the UI pages of this system. HTML (Hypertext Markup Language), gives content structure and meaning by defining the content. CSS (Cascading Style Sheets), is a used to style the appearance of content with fonts and colors. Bootstrap is used to design the webpages responsively.

- NodeJS
  Node.js is a JavaScript runtime environment. It can execute JavaScript code outside of a web browser. It helps to create dynamic web page content before the web page is sent to the user's web browser [12]. Here NodeJS is used to do the client-side scripting.

- Python
  Python is the programming language used as the base for chatbot implementation. Python is the mostly used language to create Applications related to Artificial Intelligence. It is simple to use. Python has a straightforward syntax and it is object-oriented.

- Chatterbot
  Chatterbot is a python library specially designed to create chatbots. It uses a set of machine learning algorithms to fabricate the varying responses of users as per their requests. Chatterbot makes it easier to develop chatbots that can engage in conversations. It starts by creating an untrained chatterbot that has no prior experience or knowledge regarding how to communicate. When the number of instances increases in chatterbot, the accuracy of the responses made by are also getting increased.

- Flask
  Flask is a web application framework. It’s written in Python. It is a microframework that does not require tools or libraries. Flask supports extensions that can add application features like APIs. Flask is used as the framework when developing this system.

- MongoDB
  MongoDB is a cross-platform and a document-oriented database program. Known as a NoSQL database program. It uses JSON-like documents with schema. It is a widely used database to deal with machine language related projects.

- Google Text to Speech API
  This API is used as a third-party in the system to implement the text-to-speech and speech-to-text features. This API is release by Google cloud platform as a voice assistant.

- Visual Studio Code
  Visual Studio Code is the code-editor used for this project implementation. It’s a lightweight application supported by main operating systems.

- Pusher (Stun server)
  Pusher is a hosted service. It is used to add real-time data and functionality to web and mobile applications. It works as a real-time layer between servers and clients. Major functionality to have a persistent connection with the clients.

- Socket.io
  Socket.IO is a JavaScript library for real-time web applications. It enables real-time, bi-directional communication between web clients and servers.

Chapter 4

4. EVALUATION

4.1 Testing

In this chapter the system is tested against its outputs with comparison of its expected results. Here it describes the testing of the system with testing strategies used and the identified possible test scenarios. Since this system is developed based on Incremental developed on with agile, it consists of testing activity in each of its development phase.

4.1.1 Unit and Integration Testing

Here the system is divided into separate modules and the functionalities of each module will be tested with both positive and negative scenarios. Then perform an integration testing combining all the modules.

- University website home page
- Chatbot module
- Administrator portal
- REST API testing

University website home page

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
T01	Website loads successfully in main web browsers	1.Load the main university website in the web browser	Website should get loaded successfully	Website is loading successfully	PASS
T02	Verify whether the chatbot is loading when chatbot icon is clicked	1.Click the chatbot icon	Chatbot should get loaded successfully	Chatbot is loading successfully	PASS
T03	Verify whether the admin portal is displaying when the admin button is clicked	1.Click the admin button	User should be redirected to the admin portal	User is redirected to the admin portal	PASS

Table 4.1: Test cases of University Website Page

Chatbot Module

Current system responses to the messages in 3 different response categories.

- Salutation - This response to the greetings of the user in an addressable manner and it is user friendly

- Domain Responses - This includes the responses regarding specific questions asked by students.
- Apologetic Responses - This includes response to the queries which are tough to retrieve and difficult to answer.

Test scenarios are executed covering all these 3 response types.

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
T04	Verify the response of the chatbot when the user greets using text input	1.Open the chatbot 2.Type “Hello”	Chatbot should greet the user in text format	Chatbot greets the user successfully in text format	PASS
T05	Verify the response of the chatbot when the user greets using voice input	1.Open the chatbot 2.Click voice icon 3.Say “Hello”	Chatbot should greet the user in voice format	Chatbot greets the user successfully in voice format	PASS
T06	Verify the response of the chatbot when the user asks a question related to admission procedure in text format	1.Open the chatbot 2.Type “When is the next intake for BSc. Engineering degree?”	Chatbot should provide the relevant answer in text format	Chatbot gives the relevant answer in text format	PASS
T07	Verify the response of the chatbot when the user asks a question related to admission procedure as a voice input	1.Open the chatbot 2. Click voice icon 3.Say “When is the next intake for BSc. Engineering degree?”	Chatbot should provide the relevant answer in voice format	Chatbot gives the relevant answer in voice format	PASS
T08	Verify the response of the chatbot when the user asks a question	1.Open the chatbot	Chatbot should provide the relevant answer as “Sorry. I cannot process	Chatbot gives the relevant apologetic answer in text format	PASS

	irrelevant to the admission procedure in text format	2.Type “What is your favorite song?”	your request” in text format		
T09	Verify the response of the chatbot when the user asks a question irrelevant to the admission procedure in voice format	1.Open the chatbot 2.Click voice icon 3.Say “Singer Song”	Chatbot should provide the relevant answer as “Sorry. I cannot process your request” in voice format	Chatbot gives the relevant apologetic answer in voice format	PASS
T10	User submits a feedback rating and a message	1.Rate the service 2.Write a feedback	Should be stored in DB	Record is stored in DB	PASS
T11	User submits a message with wrong spellings	1.Open chatbot 2.Write a message with incorrect spellings	Spell checker should identify it	Spell checker identifies it	PASS

Table 4.2: Test cases of Chatbot

Administrator Portal

Test Case ID	**Test Scenario**	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
T12	Verify whether the admin can sign into the system with correct credentials	1.Open the administrator portal 2.Enter valid credentials 3.Click sign in	Admin should be redirected to the admin dashboard	Admin is redirected to the admin dashboard	PASS
T13	Verify whether the admin cannot sign into the system with	1.Open the administrator portal	Admin should receive a login error	Admin receives a login error	PASS

	incorrect credentials	2.Enter incorrect credentials 3.Click sign in			
T14	Verify the form validations in the sign in form	1.Open the administrator portal 2.Click sign in button without entering username or password	System should display the relevant error messages	System displays the relevant form validation error messages	PASS
T15	Verify a successful admin user registration	1.Open the administrator portal 2.Click signup button 3.Enter relevant details in the registration form 4.Click on Submit button	1.Admin should successfully register with the system 2.Admin should be redirected to the sign in page	1.Admin successfully registers with the system 2.Admin is redirected to the sign in page	PASS
T16	Verify the form validations in the registration form	1.Open the administrator portal 2.Click submit button without filling out the mandatory fields in the registration form	System should display the relevant error messages	System displays the relevant form validation error messages	PASS
T17	Administrator can view the queries asked by users so far	1.Open the administrator portal	System should display the queries asked by users so far	System displays the user queries to admin	PASS

		2. Sign in with valid credentials 3.Go to user queries tab			
T18	Administrator can view the user feedbacks	1.Open the administrator portal 2. Sign in with valid credentials 3.Go to user feedbacks	System should display the user feedbacks	System displays the user feedbacks to the admin	PASS
T19	Administrator can enter information to the system	1.Open the administrator portal 2. Sign in with valid credentials 3.Go to admission details tab	Administrator should be able to enter details to the aiml files	Details entered by admin are getting saved successfully in aiml files	PASS

Table 4.3: Test cases of Administrator Portal

API Testing

REST APIs used in implementing the system were tested manually using postman. The response is verified with the response codes and a database verification was done to ensure the data saving mechanisms.

4.1.2 Web Accessibility Testing

Web accessibility for this application is implemented in compliance with the WCAG 2.0 guidelines. Here the expectation is that a differently abled student with visual impairments also can use the chatbot seamlessly. In order to test web accessibility feature, screen reader called Jaws is used with Firefox browser. To test the color contrast ratio, WCAG Color contrast checker, a Chrome browser plugin is used.

4.1.3 Device Compatibility Testing

A browser compatibility testing was carried to test whether this application is working properly in widely used browsers like Google Chrome, Firefox, Edge and Safari and Operating systems like Windows and MacOS. To test the mobile combability, this application was tested in android device and an iOS device. Saucelabs was used to test the application in different browsers.

4.1.4 Performance Testing

Students needs to have an active web services to use this system. Since this system is a web-based application, it will be hosted in AWS cloud services. Users can access this system from anywhere at any time. But the response time of the system will depend on the internet speed. With a good internet connection, user can get observe a response time of 2,3 second. This response time is calculated from fetching the keywords from the user’s query, searching it in the knowledge base and then showing the output. Worst response time will be 5seconds for a poor internet connection.

4.1.5 Security Testing

Security testing is ensuring that the system will protect its data from unauthorized access and modification. Authentication is used to confirm the identity of someone tries to access the Administrator portal. Authorization is used to ensure that the administrator has the overall control of the system. Session management plays a significant role in this project. When many users access the system at the same time, their session details should be managed separately.

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
T20	User knows the exact link of the administration dashboard page and tries to submit it directly	1.Go to administrator dashboard page link	System should direct the user to the login page	User is directed to the login page	PASS
T21	Verify whether a user cannot sigin to the administrator portal with random credentials	1.Open the administrator portal 2.Enter random credentials 3.Click sign in	System rejects the login	System rejects the login	PASS
T22	Verify a unique session is maintained for each user	1.Mulitiple users chats with the system	Conversations should be unique with each other	Chatbot handles unique conversations with each user	PASS

Table 4.4: Test cases related to security testing

4.2 Evaluation

In this chapter it determines if the developed software application aligns with the end goals of the users. In order to validate whether this system satisfies the end user requirements, several strategies were followed. Questions that can be asked from the system by users were categorized and the system was tested against those question types. User acceptance testing was carried out with the help of several prospective students and their feedbacks were collected. Then the system was modified in a way to align with user feedbacks.

4.2.1 Question Categories

In order to analyze the responses of chatbot, the types of potential questions a user could ask, have been categorized into four main categories.

1. Relevant question: A valid question related to the domain which has the answer in the knowledgebase and able to retrieve.
2. Irrelevant question: A question that is not related to the admission process and that is not included in the knowledgebase.
3. No response question: A valid and a relevant question related to the admission domain. But the answer is not there in the knowledgebase.
4. Poor response question: In this case answer exists in the knowledgebase and couldn't retrieve it or gives an incorrect reply

Examples of each of these categories are as below.

- Relevant question: "What are the entry requirements for the MIT?"
  Reply: "To enter the program, you need to have a Bachelor's Degree from a University or Institution recognized by the University Grants Commission or Any other academic or professional qualification equivalent to a Bachelor's Degree"

- Irrelevant question: "How many buildings are there in the university?"
  Reply: "There are about 4-degree programs we conduct this year"

- No response question: "Where can I find the admission form?"
  Reply: "Admission deadline is 7th July 2020"

- Poor response question: "Can I pay course fees online using American Express card?"
  Reply: "Yes, you need to visit BOC bank Thibirigasyaya to do the course fees payments"

4.2.2 User Acceptance Test

User acceptance test was carried out in two phases with 60 prospective students. First the developed application was hosted in cloud for 7days and a sample of 30 students were encouraged to use the system. Then their feedback ratings and feedbacks were considered and incorporated their feedbacks and did some improvements to the system. Then the second phase of testing was carried out with the rest of testing was carried out with the rest of 30 students and same as before their feedbacks were considered and further improvements were done to the system.

<u>Phase 1 – User Ratings</u>

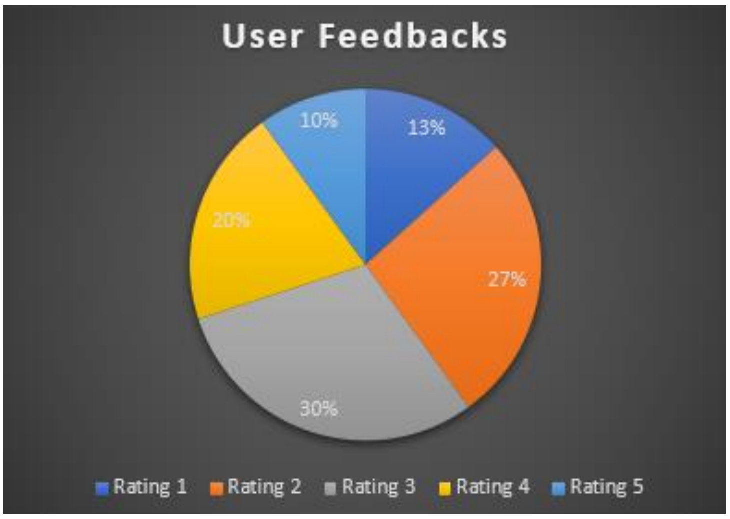


Figure 4.1: Phase 1 user ratings

<u>Phase 1- User Feedbacks</u>

- After the initial test, it was clear that the data was not enough in the knowledgebase to answer some domain related questions. So that more data was entered to the knowledgebase.
- Answers must be more specific and should include more details.
- User interface to be more attractive.

<u>Phase 2 – User Ratings</u>

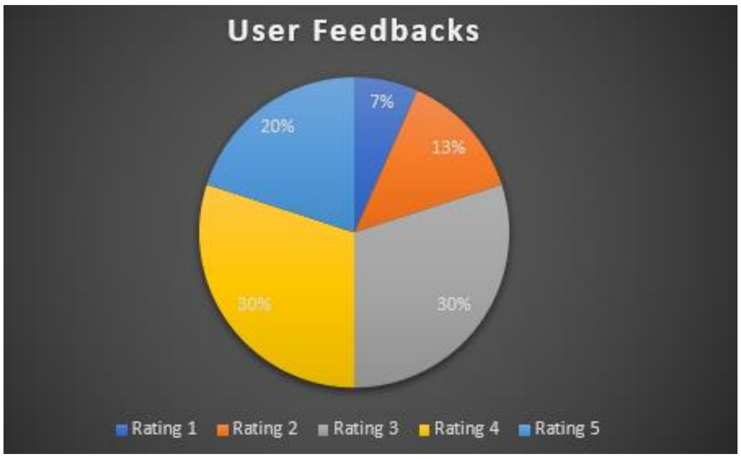


Figure 4.2: Phase 2 user ratings

<u>Phase 2 – User Feedbacks</u>

- With phase 2 user feedbacks also, it was clear that the system should be fed with more relevant domain data for more convenient responses from the chatbot.
- Users suggested to build this as a mobile application too. It can be considered as a expansion.
- Users suggested to expand this chatbot to other domain areas as well. It also can be considered as a future enhancement.

Chapter 5

5. CONCLUSION

5.1 Achieving the Objectives of the Project

This project was developed using new technologies and a lot of research had to be done in order to finalize the technologies and to finalize an approach. To develop this system knowledge in technologies like python and AIML was necessary and studying those was a mandatory fact. With that individual skills in those languages were developed along with the implementation of this project.

Finally, a working system was developed in Python and AIML along with other web development technologies. This system was uploaded in an AWS server for two weeks and acceptance testing was done with the participation of some prospective university students. Their comments and feedbacks were highly considered, and improvements were done to the system to align with those. Since, user queries were stored in database as logs, referring to them could give a realistic idea of what the real-world users expect from this system.

Since this web application is dynamic, it has an administrator portal. An administrator can to log in and make appropriate changes to the system. He also can add information to the knowledgebase, such as questions, answers and keywords. User queries and feedbacks can be viewed. This system is secure from unauthorized access according to the measure taken during implementation process.

Moreover, this system is developed in compliance with Web accessibility and a text-to-speech feature is introduced. Therefore, prospective students with visual impairments also can use this system without any hassle.

Device compatibility and performance testing is done with this system and all are in good shape.

The results gathered from the users was used to identify the limitations of the system and do more improvements. The evaluation results and the overall testing strategy used, could make the system more effective to use.

5.2 Problems Encountered in the Project

A main challenge encountered was to identify an approach to implement this system. Researches and studies were done on different approaches and finally decided to use python with AIML as the core of the chatbot. In order to implement this with Python and AIML technology, first it was necessary to get some hands-on experience with these technologies. So, had to spent quite some time to learn them.

Had to face lots of technical barriers while implementing this. The database design was quite complex when compared with other applications. With the usage of AIML files it was not necessary to save information related to the chatbot queries in a database. Because saving data both in AIML files and database files is just wasting the system memory. Therefore, it was decided to use AIML files as the knowledgebase of the system and database to store session details and administrator related information.

Proper session management of the system was a challenge. It was a main priority of the system to handle separate sessions for each user, so that the chatbot can have personalized chat conversations with multiple users at once. With the help of internet blogs, a solution to that could be found.

When there are number AIML files, it can take a long time to learn. This could lead to a performance degrade of the system. After searching a solution in internet, it was found that brain files can be used to achieve that. After the chatbot learns all the AIML files it can save its brain directly to a file which will drastically speed up load during subsequent runs.

Compatibility issues occurred when rendering the chatbot application in different devices with various resolutions. To overcome that had to use bootstrap tags to align with screen sizes.

Apart from the above technical barriers, many compile errors and syntax errors occurred while implementing the system. Analyzing helped to find the root cause and to fix all those errors successfully.

5.3 Limitations of the Currently Developed Solution

This system is developed only to guide the students throughout the admission procedure of the university. So, the chatbot can answer if the questions are only related to admission process. But there are number of other areas where the students need the assistance. Those are not addressed with this system.

An active internet connection is necessary to access this system as it is a web based one.

This is not developed as a mobile application. So only can be accessed via a web browser.

Some more AI algorithms should be implemented to improve the accuracy of chatbot’s responses and some more data should be feed into the system.

If overseas students wish to apply to the university, they must use English to get the service of this chatbot. Because at the moment this supports only English language.

5.4 Further extendibility of the system

This chatbot system can be improved as a college guide system for perspective students. This will help students in the process of admission and then after they enter college, it can help the students to get used with the college aspects and environment.

Expandability of this chatbot will not take a huge effort. This can be even integrated with university LMS systems through APIs and even the students can check their assignments due dates and exam results.

This chatbot can be enhanced as a pal to students. This way universities can develop a better relationship with their students when they use AI bots for automated tasks. In the process, they also will be able to reduce the workload that their staff handles

Moreover, this system can be developed to improve the service of chatbot to have conversations with multiple languages. It will help to attract overseas students to the colleges.

This web-based application can be converted into a mobile application that the students can download free of charge in mobile play stores. That will drastically improve the usability of this system. Since, mobile phones are always in their hands.

REFERENCES

[1] G. Ahern, "What Are Chatbots?," Ometrics, 2020. [Online]. Available: https://www.ometrics.com/blog/what-are-chatbots/. [Accessed 06 05 2020].

[2] "ELIZA," Wikepedia, 2020. [Online]. Available: https://en.wikipedia.org/wiki/ELIZA. [Accessed 08 05 2020].

[3] "Artificial Linguistic Internet Computer Entity," Wikepedia, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Artificial_Linguistic_Internet_Computer_Entity. [Accessed 08 05 2020].

[4] A. Storman, "5 Chatbot Challenges and How to Overcome Them," Chatbots Magazine, 2016. [Online]. Available: https://chatbotsmagazine.com/5-chatbotchallenges-and-how-to-overcome-them- caccc3a26d7c. [Accessed 08 05 2020].

[5] A. A. M. a. A. I. A. M. Rahman, "Programming challenges of chatbot:Current and future prospective," in *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, Dhaka, 2017.

[6] "Problems with Chatbots Today," Attunix, 2018. [Online]. Available: https://attunix.com/problems-chatbots-today/. [Accessed 07 05 2020].

[7] C. Phillips, "The 3 Types of Chatbots & How to Determine the Right One for Your Needs," chatbotsmagazine, 30 04 2018. [Online]. Available: https://chatbotsmagazine.com/the-3-types-of-chatbots-how-to-determine-the-right-one-for-your-needs-a4df8c69ec4c. [Accessed 10 05 2020].

[8] "What is Flask used for?," Dev, 16 11 2019. [Online]. Available: https://dev.to/amigosmaker/what-is-flask-used-for-2do5. [Accessed 15 05 2020].

[9] "Django vs Flask," Educba, 2020. [Online]. Available: https://www.educba.com/django-vs-flask/. [Accessed 12 05 2020].

[10] "Django vs Flask: Which is the best for your Web Application?," Edureka, 06 02 2020. [Online]. Available: https://www.edureka.co/blog/django-vs-flask/. [Accessed 13 05 2020].

[11] M. Cohn, "Agile Needs to Be Both Iterative and Incremental," Mountain Goat Software, 11 11 2014. [Online]. Available: https://www.mountaingoatsoftware.com/blog/agile-needs-to-be-both-iterative-and-incremental. [Accessed 08 05 2020].

[12] D. Banerjee, "Natural Language Processing (NLP) Simplified : A Step-by-step Guide," Data Science Foundation, 14 April 2020. [Online]. Available: https://datascience.foundation/sciencewhitepaper/natural-language-processing-nlp-simplified-a-step-by-step-guide. [Accessed 2 10 2020].

[13] "Node.js," Wikipedia, 14 05 2020. [Online]. Available: https://en.wikipedia.org/wiki/Node.js. [Accessed 15 05 2020].

[14] "WHAT IS THE ARTIFICIAL INTELLIGENCE MARKUP LANGUAGE?," Imarticus, 13 01 2019. [Online]. Available: https://blog.imarticus.org/what-is-the-artificial-intelligence-markup-language-artificial-intelligence-course-blog/. [Accessed 16 05 2020].

[15] "Visual Studio Code," Wikipedia, 16 05 2020. [Online]. Available: https://en.wikipedia.org/wiki/Visual_Studio_Code. [Accessed 16 05 2020].