DESIGN AND IMPLEMENTATION OF AN ONLINE CHATBOT FOR ADMISSION ENQUIRY IN MOUNTAIN TOP UNIVERSITY

By

MESHE DAMILOLA PETER
19010301087

**A PROJECT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND MATHEMATICS, COLLEGE OF BASIC AND APPLIED SCIENCES, IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE**

2023

DECLARATION

I hereby declare that this project has been written by me and is a record of my own research work. It has not been presented in any previous application for a higher degree of this or any other University. All citations and sources of information are clearly acknowledged by means of reference.

MESHE, DAMILOLA PETER

DATE

CERTIFICATION

This is to certify that the content of this project titled, **DESIGN AND IMPLEMENTATION OF AN ONLINE CHATBOT FOR ADMISSION ENQUIRY IN MOUNTAIN TOP UNIVERSITY** was prepared and submitted by **MESHE, DAMILOLA PETER** in partial fulfillment of the requirements for the degree of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE.** The original research work was carried out by him under his supervision and is hereby accepted

______________________________ (Signature and Date **)**

Mr. J.A Balogun

Supervisor

______________________________ (Signature and Date **)**

Dr. A. A. Mebawondu

Coordinator, Department of Computer Science and Mathematics

DEDICATION

I dedicate this work firstly to God who has made it possible to have achieved such great and splendid feat amidst significant challenges. I also dedicate it to my father and mother Mr and Mrs Meshe and my sponsor Mummy Abraham Rose, who have put in their all to ensure that I get the best possible in my academic pursuit. Finally, I dedicate this to my very good friend, Obe Israel and brother, Abolude Peter who supported me in the delivering of this work.

ACKNOWLEDGEMENT

The success and final outcome of this study is all thanks to God for His wisdom and understanding. I also appreciate the entire staff and management of Mountain Top University for the immeasurable impact they have had on my life academically and spiritually from the person of the chancellor, Dr. D.K Olukoya and the chaplain. I specially recognize my supervisor Mr J.A. Balogun whom under his supervision and guidance I was able to surmount my fears and approach this project with the right mindset. I also recognize Dr. Funmilayo Kasali and a host of other lecturers who have made special contributions to the success of my academic pursuit. Finally, I duly appreciate my sister, brothers, close friends and colleagues without whom this journey could not have been a success. May the God of heaven water your lives.

ABSTRACT

The study's primary objective was to create an online chatbot admission system to assist university applicants in obtaining information related to the institution's admission procedures. To achieve this, the research encompassed a comprehensive review of existing chatbot systems, identification of user and system requirements through informal interviews, and the specification of system design employing UML diagrams like use case, sequence, and activity diagrams. Implementation was carried out using an array of technologies including HTML5, CSS3, JavaScript, jQuery, Ajax API, Python, and Natural Language Processing (NLP), with data management facilitated by a PostgreSQL database. At the project's culmination, a web-based platform was delivered, furnishing applicants with standardized information concerning the university's admission processes and incorporating a database to store both user-bot interactions. The study concludes that this application will substantially benefit the university and applicants, providing rapid responses, consistent information, and 24/7 support. In summary, the research offers a valuable contribution to streamlining university admission procedures through modern technology, alongside recommendations for future endeavors in this field.

Keywords: *Artificial Intelligence, Admission, Chatbot, Query*

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER ONE
INTRODUCTION

1.1 Background to the study

The usage of chatbots in various business sectors has been on the rise in recent years, and higher educations like the university; is no exception. In higher educational context, applicants sends their queries through the university's email or the inquiry section on the website which most times are slowly responded to or left unanswered, while chatbots can provide an accurately and quick responses to common admission enquiries, reducing the workload of the admissions team and improving the overall experience for applicants.

However, to implement this chatbot which can be complex, requires the understanding of natural language processing (NLP) and machine learning (ML) technologies, as well as thorough understanding of the user's journey and the various touchpoints that prospective students have with the university during the enquiry process. Additionally making the interaction between the bot and the human more humanly with the use of conversational AI. A chatbot system simulate a discussion (or a chat) with a user in natural language using conversational artificial intelligence (AI) technology via messaging applications, websites, mobile apps or the telephone (Selig, 2022). A chatbot is artificially constructed software that uses natural language as input and output to talk to humans. Chatbots can act as a personal assistant on mobile devices to provide users with personalized information, enable real-time social interaction media, and can even be used in health consultations (M. Yamada, 2016).

According to (Court Bishop, 2022) Customer service teams that handles upto 20,000 support requests on a monthly basis could save more than 240 hours per month by just using chatbots. Chatbots are used increasingly in instant messaging and they are implemented in people's regular lives, shopping experiences, and education courses (Ferrell, 2020). In general bots are just machines that is intelligent enough to understand human request and then formulate the request in a way that is understandable by other software systems to request the data you need (Sumit, 2019). This study aim to address the gap of prompt response between the university and prospective students and evaluating its effectiveness in improving the overall experience for applicants and streamlining the enquiry process for the admissions.

1.2 Statement of problem

For every new academic session, the university receives many applications which are processed into enrolment, applicants may need to ask questions ranging from courses, the university's environment, and fees to accommodation and learning systems. With so much queries from the students, putting in much workload on the admission teams, most queries may be left unanswered. There is a need for a system which can provide accurate and very quick responses to enquiries, and guide applicants on what to do, by handling multiple enquiries at once.

1.3 Aim and objectives of study

The aim of this study is to develop a system that provides quick and accurate response to queries from applicants and reducing the workloads of the admissions team. The specific objectives are to:

i. collect data relevant to the student applications and it features
ii. formulate the model using the features in (i)
iii. design the model
iv. implement the system.
v. test the system.

1.4 Methodology of the study

To accomplish the aforementioned objectives fully, the waterfall model is used. The waterfall model contains series of processes, that are used duing development. The stage usually will require:

a. Gathering of requirement for the proposed system which will involve understanding of. the user journey and the various touchpoints that prospective students have with the university during the enquiry process, like answers to the queries, logs, feedback, keywords.
b. A file called intents will be created which will contain every features extracted in (a) . These features will be grouped according to how the bot will be trained. That is; Interactive FAQ, question answering, dialogue planning etc
c. Design of the proposed system will be specified using UML diagram such as Use case, Sequence diagram, Activity diagram.

d. The implementation will be done using HTML, CSS, jQuery and Ajax api, JavaScript as the frontend and Python Language and Natural Language Processing (NLP) as the backend with PostgreSQL as the database.
e. The system will be tested by users via a user assessment test questionnaire.

1.5 Scope and Limitation of the study

This scope of this study is to develop a web-based chatbot that will be used to manage the common enquiries of the applicants, increase efficiency and reduce response time during multiple enquiries, availability 24/7 etc. Due to time availability for this study, the chatbot is an Information-retrieval bot. Therefore, the chatbot cannot self-learn or generate it own newly response.

1.6 Significance of the study

This study will reduce the workload of the admissions team by providing quick and accurate responses to common enquiries, it will handle multiple enquiries at once and be available 24/7. It will also reduce the cost of operation and increase the reach of the University to many student who are looking for information.

1.7 Defination of terms

a. **Applicants:** An applicant is one who formally applies for or requests something, especially a job or to study at a university or college.
b. **Artificial Intelligence:** It is the effort to automate intellectual tasks typically performed by humans. It can be complex or simple. A computer-controlled robot can perform duties normally related to intelligent beings.
c. **Bootstrap:** Bootstrap is a CSS Framework used in developing responsive and mobile-first websites.
d. **Chatbot:** A chatbot is a compound word of a chat and a bot. It is an application that simulates and strategies human verbal exchange (both written and spoken), permitting them to engage with virtual gadgets as if they had been speaking with an actual individual.
e. **Conversational AI:** It is a combination of machine learning and natural language processing that allows people to have huma-like interactions with computer.

f. **Machine Learning:** It is a subset of Artificial Intelligence, mainly focusing on building systems that learn, and enhance overall performance based on the data being consumed.

g. **Natural Language Processing:** It refers to the branch of computer science and more especially, the branch of artificial intelligence concerned with giving computers the capacity to recognize textual content and spoken words in the almost identical way people can.

h. **Query:** Queries in computer science, are requests for data or information from a database table.

1.8 Organisation of study

Chapter one of this study, discusses the background of the project work, the aim and objective, problem statement and the methodology which were carried out. Chapter two layout the literature review of the work and some related works. Chapter three explains how the study was carried out in details. Chapter four shows the implementation details of the methodology described in chapter three of the project. Chapter five concludes the project and provides recommendations for further development of the study.

CHAPTER TWO
LITERAURE REVIEW

2.1 Chatbot

Many terms have been used to refer to the chatbot technology, the ones widely known and common are: Chatbot (can be spelled chat bot, or chat-bot), Conversational Agent, Chatterbot, Conversational System and Pedagogical Agent (or Intelligent Pedagogical Agent, IPA). The last is solely used in educational settings or educational papers. (Doering et al, 2008). Chatbots are intelligent processors that are used to convey knowledge by mentors to a beginner or a professional by analysing the learning patterns and comprehension thereby adapting to their speed through a series of messages. (Thomas, 2020).

According to (Wollny et al, 2021) Digital systems that can be engaged with, either orally or through text interfaces using natural language can be referred to as a Chabots. They can be built into softwares like; digital assistants, or through messaging platforms since they are made to automate discussions by impersonating a human conversation partner. Chatbots are smart conversational computer programs that stimulate human conversations in their natural form (Guendalina Caldarini et al, Introduction, 202). To converse in human language through text or orally with people or other chatbots, chatbot uses Natural Language Processing (NLP) and sentiment analysis. Smart bots, artificial conversation entities, and digital assistants, interactive agents are also knwn as chatbot. (Adamopoulou & Lefteris, 2020).

2.1.1 History of Chatbot

“ Can machine think?. ” Alan Turing back in 1950 forwarded this question. Turing wondered if computer program could communicate in a way that is identical to humans. This proposed question named Turing Test, brought about the idea of chatbot and how it has become today (Adamopoulou & Lefteris, 2020). In 1966, the first chatbot was created and was named Eliza, it depended primarily on linguistic norms and pattern matching algorithm whose purpose is to serve as a psychotherapist. Through the use of keyword matching program, communication with the user could be possible. To reorganize the input so as to offer an answer to the user, it looks for a suitable transformation rule. Eliza was a landmark system that inspired additional research in the field. (Guendalina Caldarini et al, 2022).

In 1995, ALICE (Artificial Linguistic Internet Computer Entity) was the first online chatbot to be created and it was inspired by ELIZA (Wallace, 2009). ALICE was developed using Artificial Intelligence Markup Language (AIML), which is the most critical difference between ALICE and ELIZA. ALICE knowledge Base consists of approximately 41,000 templates and associated patterns, which has a huge number compared to ELIZA having only 200 keywords and rules. (Heller, B et al, 2005).

The development of Smarterchild in 2001, brought about a real evolution in the history of chatbot technology. (György et al, 2018). It could retrieve information from databases about weather, sport scores, movie times, making it the first time chatbot ease human task. The creation smarterchild-like chatbots started, that is, from Siri in 2010 to ChatGPT in 2021.

2.2 Implementation Approaches to Chatbot

As several implementation method are used in creating different chatbot either being Information-Retriever Chatbot, Generative Chatbot or the use of Keyword Matching technique. These approaches can be distinguished to be Artificial Intelligence based chatbot and Rule-based Chatbots.

2.2.1 Rule-based Chatbot

Chatbots on rule-based uses a series of defined rules. Conversation chat with this type of bot are through some predefined rules and different set of questions. These chatbots uses a tree-like flow method, thereby making them referred to as a decision-tree bot, the user is given a set of options that are predefined which leads to their desired answer.

2.2.2 AI-Based Chatbot

AI-based chatbots are train using Machine learning models that makes them to self-learn thereby generating almost accurate answers accordingly. The key that gives AI chatbots the ability to understand and provide answers to human queries is called Natural Language Processing(NLP). AI-Based chatbots uses machine learning and the potentialities of AI that makes bots smart with time. (Saxena, 2021).

a. Information Retrieval Chatbots

Retrieval-based chatbot uses empirical to select a response from predefined responses. It can handle context of the query and selects the best answer from predefined answer. (Wayesa, 2021). Information retrieval chatbots works on the principle of directed flows. The bot trained using this technique, provides the best response possible from a database of pre-defined responses.

To identify the most relevant response, retrieval-based chatbots use techniques such as keywords matching, machine learning or deep learning. No matter the type of technique used, these chatbots only provide answers that are predefined and do not generate new output. (Fainchtein, 2020). In case of retrieval-based chatbot, techniques like machine learning, deep learning, and keywords matching are used in selection of relevant responses. This model won't generate new output and can only give it outputs based on predefined responses. (H. Akkineni et al., 2022).

b. Generative Chatbots

It also called cognitive based chatbots. It requires a much bigger data set and use deep learning methods for training. They have cognitive ability to generate different answers. (Wayesa, 2021). Chatbots that uses generative models, generate new responses word by word, based on the users' input. A generative chatbot is an open-domain chatbot program which creates it own unique combinations of language rather than choosing from pre-defined responses. Generative chatbots are built using seq2seq models, which is used for machine translation. (Leah, 2022). Multi-step training in generative chatbots uses different combination of adversarial learning, supervised learning, reinforcement learning, and unsupervised learning. (Fainchtein, 2020).

2.3 Artificial Intelligence

Modern systems today are all made with artificial intelligence. Being able to facilitate day to day human tasks by having simple conversation either through text or audio. One of the main goals of AI is automating tasks that previously would have required human intelligence. Significant efficiencies can be greatly increased by cutting down on the labour resources an organization must devote to completing a project, or the time a person must devote to routine tasks. (Robert et al, 2000). From the book written by (Stuart J. Russell and Peter Norvig, 2021), AI was defined by differentiating computing systems on the basis of rationality and thinking versus acting:

Human approach:

a. Systems that think like humans
b. Systems that act like humans

Ideal approach:

c. Systems that think rationally

d. Systems that act rationally

2.3.1 Conversional AI

Conversational AI are tools being used to make machines capable of understanding, processing and responding to human language. Conversational AI deals with a variety of technologies that work together thereby enabling an efficient, automated communication via text and oral by understanding customer’s intent, deciphering language and context, and give out answers in a human-like manner. (boost.ai, 2023). Technology like virtual assistants or bots that can stimulate conversation with people are referred to as the term conversational AI. (Tres Dean, 2022).

2.3.2 Natural Language Processing (NLP)

Natural language processing is a branch of Artificial Intelligence and Linguistics, aimed with making computers understand and interpret statements or words written in human languages. It came into existence to make user’s work easier and to satisfy the desire to have conversation with the computer in natural language. (Diksha Khurana et al, 2022). One of the goals of NLP research is to create computational models that mimic human linguistic skills (reading, writing, listening and speaking). (Danilo Cavalier et al, 2019). NLP aspects distinguish in low-level activities, such as:

a. **Tokenization:** Tokenization involve cutting raw text into tiny parts. Tokenization divides the original text into words and sentences known as tokens. These tokens aid in understanding the context or constructing the NLP model (Chakravarthy, 2020). It identifies the tokens of text. Tokens are sequence of characters with meaning, such as words, numbers or symbols. The three categories of tokenization are word, character, and subword (n-gram characters) tokenization.
b. **Lemma and stem:** These are algorithms used in NLP to standardize text and get words and documents ready for more deep processing in machine learning. They are both used to develop derived(inflicted) word’s root form
c. **Part-of-speech (POS):** This is the classification of gramma which mostly include; verbs, nouns, adverbs, adjectives. It’s also called POS taggng which is a method where each words in a sentence is labelled with their corresponding part of speech. It’s an important NLP applications used in question answering

parsing, machine translation and so on. They can either be rule-based labels (rules that are manually created to specify POS for the ambiguous words given their context) or labels created using stochastic models that are trained on sentences that have the proper POS labels. (Cahn, 2017)

d. **Sentence splitting:** The technique of breaking up free-flowing text into sentences is called sentence splitting. It is one of the initial steps in any application that uses natural language processing (NLP). The terms sentence tokenizer, sentence boundary detector, and sentence boundary disambiguator are all used to describe sentence splitters.

e. **Entity:** In natural language processing, entities are words or groups of words that all relate to the same object. Any real-world item or notion that can be represented in text, such as a person, organization, place, or product, can be referred to as a textual object.

2.3.3 Natural Language Understanding (NLU)

Natural language understanding is a subset of artificial intelligence that enables humans and computers interaction possible. NLU helps machines to understand and analyze natural language by extracting concepts, entities, sentiments, keywords and so on. It is used in customer service applications to comprehend problems reported either verbally or written by customers. (Diksha Khurana et al, 2022). According to NLU set a ladder of classification models to analyse text or input:

a. **Domain Classifier:** Domain classifier is used for conversations about multiple topis that requires specialized vocabulary. It classifies input into one predetermined group of conversational domains.

b. **Intent Classifier:** This is a part of NLU, where the machine learning learns to classify or analyzes texts data and categorizes them into customer’s intent. It automatically classify them into intents such as Downgrade, demo request, and so on.

c. **Entity Recogniser:** It is also called Named Entity Recognition. It is a wel-liked NLU techmique that entails identifying and extracting significant textual information in the text before categorizing it into groups of predetermined categories. Entity recognizer helps you identify key elements in a text document, like Tesla (Brands), Lagos (location), Barack Obama (name) and others.

2.4 Software development life cycle (SDLC)

Software development life cycle is a breakdown or series of steps involved in the creation of software. A team of software developers follows software development life cycle to develop and maintain software. SDLC is an organized approach in completing software development process within the time and maintain the beauty of the software. The software development life cycle includes distinct phases designed to give software engineers a clearly defined goal for how to work. It aid developers and other project personnel in developing a system that meets all technical and user requirements while exceeding customer expectations. All these phases revolve around a central theme of quality assurance (Lemke, 2018). According to (Murray, 2022) SDLC is the software engineering process used for developing, designing, testing and deploying software. Every stage of the SDLC is intended to provide software teams control over their software development with predictable results and visibility into budgets and timelines.

2.4.1 Software development life cycle phases

SDLC Phases (also called SDLC Processes) involves the various stages in the development of high-quality software product. Below are the given stages summarized into six(6) which are:

a. Requirement gathering and analysis:

Information that are relevant are collected from the end users so as to meet their possible expectations. It is carried out by the senior team members. This stage draws a clear picture of what the entire project is about and the anticipated issues, directives which triggered the project. The SRS (Software Requirements Specification) is created after the project is understood.

b Design:

In this phase, the detailed requirements is to be turned into detailed specifications that will be used during the Development stage by the engineers. The SRS document that contained the requirements gathered is used as input and turned into more logical architecture which is implemented in a programming language.

c Implementation or Coding:

In the implementation stage, the design architecture is converted into a functional system. Different programming languages like; C++, Phython is used in this phase to achieve the desired system. During the coding stage, tasks are broken down into units or modules and given to different developers. This is the most time-consuming stage of the Software Development Life Cycle process. Developer must adhere to certain predefined coding principles. They must also make use of programming tools like compilers, interpreters, debuggers to develop and implement the code. (Martin, 2023).

d Testing:

In the testing phase, the developed software is tested if it meets the desired requirements of the users. The entire functionality is tested by the testing team, making sure the software is free of bugs, stable and meets the functionalities according to the business needs of the system. The testing phase aims to determine if the requirements specified in SRS document are met. Three (3) different types of testing are ideal to include in this phase: integration testing of subsystems, security testing, user(acceptance) testing, and unit testing. Each of these types of tests serves different purpose and help all stakeholders determine flaws in the system prior to deployment. A unit test, which tests a small section of code and is generally done throughout the development phase. (Lemke, 2018).

e Deployment:

In this phase once the software is tested and confirm free of bugs and meets the business needs then, the deployment process of the software start. The tested version of the software is shipped to the market for beta testing. The support team collects feedback from the first users, and if any bugs come up during this stage, software developers fix them. After that, a new, improved version is rolled out. (Demchenko, 2021).

f Maintenance:

This can be handling residual bugs that were not able to be patched before launch or resolving new issues that comes up due to user(customers') reports. Larger systems may require longer maintenance stages compared to smaller systems. (Preston, 2022). More than half the cost of the development life cycle is due to the Operations and Maintenance phase. This phase may include identifying system operations, maintaining data, identifying problems, and revising documentation that received additional analysis (e.g. security audit/analysis). (Lemke, 2018).

2.4.2 Software development life cycle models

a Waterfall Model

Waterfall model is the oldest SDLC Model, but not used in recent years. It use a linear sequential process in which progress are seen flowing downward while in development stage. It is the first software development process model. Waterfall model is a sequential process model which does not overlap. It means that until the one phase is not completed then next phase cannot start. (Shylesh S, 2022). Waterfall technique spends the most of its time in the first phase, documenting the project, and the implementation phase also consumes a substantial amount of time, whereas extreme programming approach spends the most time in the testing phase (Pooja and Nitasha, 2016). According to (Gagan Gurung et al, 2020), Waterfall model is used for small projects, as there is little room for revisions once a stage is completed. In waterfall model problems can't be fixed until you get to the maintenance stage.

b Iterative Model

Iterative model help patches up waterfall model weekness by using the iterative process. Each iteration evolves a small version of product and it is repeated until the final version is developed. Iterative process model starts implementation with a subset of requirement specifications (Shylesh S, 2022). Unlike the waterfall model where the requirement was required once, iterative model requirements are gathered in every phase. The project is divided into smaller components so that the result can be utilized in the next phase (Gagan Gurung et al, 2020). In every phase, a new version of the software is produced and this is repeated until the system is completely ready.

c V Shaped Model

The V-shaped model, which is also called verification and validation model. It mainly focus on the testing functionality in every phase or steps of the project. The V shaped model is an extension of the waterfall model. The V-shaped model shows the relationships between each phase of development and the associated phase of testing. (Gagan Gurung et al, 2020). While it is still very difficult to go back and make changes, v-shaped is very useful when the requirements are not fully known.

d Agile Model

The agile methodology prioritizes fast and ongoing release cycles, utilizing small but incremental changes between releases. This results in more iterations and many more tests compared to other models. (Preston, 2022). In agile model, continuous interaction of development and testing of the project is allowed. Agile Model is a combination of the Iterative and incremental model. This model focuses more on flexibility while developing a product rather than on the requirement. (Sotehp, 2023).

In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed. The Agile model adopts Iterative development. Each incremental part is developed over an iteration (Gagan Gurung et al, 2020). The software is always tested no matter how many changes aew done, so it can be accepted. Some of agile model that are used are; Feature-driven development, Scrum, Extreme programming (XP).

2.5 Unified Modelling Language (UML)

Unified Modelling Language was originally released in the late 1990s, as an extension of the object-oriented programming language(cite). It arose from the convergence of three popular of the 1980s and 1990s methodologies: the Booch method, Rum-baugh's Object-Modeling Technique (OMT), and Jacobson's Object Oriented Software Engineering (OOSE). The Object Management Group (OMG) formally adopted and standardized the first version of UML (UML 1.x) in 1997 and it combined the object-modeling technique of James Rumbaugh, Grady Booch's component notation, Ivar Jacobson's use case notation, Archie Bowen's timing analysis, and David Harel's statecharts (Xiaocong, 2015).

The Unified Modeling Language (UML) is a general-purpose modeling language used in the discipline of object-oriented software engineering. UML is a graphical language used to visualize, describe, build, and document software-intensive system artifacts. UML create a standard for developing system blueprints (Padmanabhan, 2012). Business processes and system operations, as well as programming language statements, database schemas, and reusable software components are all included in these blueprints.

According to (Booch et al, 1999). Class diagram, Object diagram, Component diagram, Deployment diagram, Use Case diagram, State-chart diagram, Activity diagram, Sequence diagram, and Collaboration diagram are the nine diagramming techniques defined in UML 1.x. While the UML 2.x specifies thirteen diagramming techniques. The latest version of the unified modeling diagram is the UML 2.5.1 and the diagrams discussed in this review are a part of that version.

2.5.1 UML Diagrams

The UML diagram is a visual depiction of classes, objects, and their relationships. It is a picture of a system model that shows some of the design and implementation. UML diagram are divided into three main categories. They include; Behaviour diagrams, Structure diagrams, and Interaction diagrams. In addition, there are 14 other types of UML diagrams that span these three categories

2.5.2 Types of UML diagrams

a. Class diagrams

This UML class diagrams explore domain concepts in the form of a domain model, analyze requirements in the form of a conceptual/analysis model, and portray the detailed design of object-oriented or object-based software. It describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes.

b Component diagrams

This illustration is used to represent the system's physical components. These physical components include things that are stored in a node, such as executables, libraries, files, documents, etc. The system division into components and their interdependencies are displayed in component diagrams.

c Deployment diagrams

Deployment diagrams are a subset of class diagrams. In this scenario, the graphic illustrates the coupling and cooperation of the run-time processing units. The primary difference between component and deployment diagrams is that component diagrams lay focuses on software components, whereas deployment diagrams describe the hardware structure of the proposed system.

d Object diagrams

This is a type of class diagram that represents the detailed status of a system at a specific point in time. It is a subset of a class diagram or a communication diagram that comprises items and their relationships. Object diagram focuses on the properties of a group of items and how they interact with one another.

e Composite structure

A UML structural diagram that combines classes, interfaces, packages, and their interactions, as well as their links, and that provides a logical perspective of all or a section of a software system is known as a composite structure diagram. It shows the internal structure of a structured classifier or collaborator (containing pieces and connectors). This diagram, like a class diagram, allows you to illustrate the underlying structure of numerous classes in greater detail and demonstrate how they interact. Inner classes and parts, as well as interactions between and within classes, can all be represented visually

f **Package diagrams**

A package diagram, a sort of structural diagram, shows the layout and organization of model pieces in a middle to large-scale project. Package diagrams can show the organization of a system, the relationships between its components, as well as several viewpoints on the system, such as a multi-layered (also known as multi-tiered) application or multi-layered application model. Package diagrams are structural representations that show how different model components are grouped and arranged to generate packages.

g **Activity diagrams**

Activity diagrams are graphical representations of actions and activities in step-by-step workflow processes. Activity diagrams show the workflow from beginning to end while emphasizing the various choice pathways that might be taken as the activity's sequence of events develops. They can be used to describe characterize scenarios in which certain tasks can be completed in simultaneously**.**

h **State machine diagrams**

A state diagram, often known as a state machine diagram, is a type of Unified Modeling Language (UML) behavioral diagram that depicts object transitions. The sequence of states that an object in a system goes through is depicted by the state machine diagram. It keeps track of the software system's activity. State machine diagrams are used to explain the behavior of an application.

i **Use case diagrams:**

A use case illustrates the functionality of a system in terms of its actors, their use cases for expressing their aims, and any connections between those use cases. An illustration of a system's intended uses and environment is called a use-case model. Additionally, It serves as a binding agreement between the developer and the consumer.

j **Interaction diagrams:**

An interaction is a behavior that consists of a series of messages delivered among a group of objects inside a context in order to accomplish a task. A message is a description of inter-object communication that sends information with the intention that the action will be successful. The interaction diagram shows the relationships between the various model entities. In this diagram, the activity and sequence diagrams are integrated. The term "interaction" describes a group of messages that are exchanged between entities in order to complete particular tasks in the system. Any classifier characteristic that is available to it can be used. The interaction diagram's most crucial components are the messages and the lifeline

k Communication/ Collaboration diagrams:

These diagrams offer yet another way to show how objects interact in order to carry out a use case's functionality, or a portion of a use case. In contrast to sequence diagrams, which emphasize how things interact across time, communication diagrams are intended to emphasize the connections between objects. According to (Singh and Dr. Sidhu, 2018), a collaboration diagram represents interactions between objects as sequenced messages, where each message transmitted between the objects is given a number.

l Sequence diagrams:

In-depth, sequence diagrams illustrate and validate the logical stages of use cases (Ambler, 2000). Sequence diagrams represent the temporal ordering of communications between items in the system, which include lifelines for the objects in the sequence as well as the concentration of control at various moments in time (Rumbaugh et al, 1999).

m Interaction overview:

It depicts the movement of information across diagrams while also offering an overview of the interaction's control flow. It can show a control flow with nodes having interaction diagrams that show how a collection of fragments could be launched in various scenarios. Interaction overview diagrams have nodes that indicate interactions (sd) or interaction usage and display the overall flow of control (ref).

n Timing diagrams

A subset of the interaction diagram that focuses on time constraints. Timing diagrams show the order of events and how each one affects the states and property values. Horizontally, from left to right, time is measured. They can be utilized to provide method execution profiling or concurrency-related circumstances (Padmanabhan, 2012).

2.6 System development tools

In this section, the following technologies were used to implement the system; HTML5, CSS, jQuery and Ajax api, Rasa Open Source, Python, PostgreSQL.

a Rasa Open Source:

The Rasa core and Rasa NLU are open source python libraries for creating conversational software. Their purpose is to make machine-learning based dialogue management and language understanding accessible to non-specialist software developers. Developers has full control of their software while using rasa unlike DialogFlow which is a closed source platform.

b Hyper-Text Markup Language 5 (HTML5):

Hyper-Text Markup Language (HTML) is a sort of global publishing language used by the World Wide Web. Tim Berners-Lee developed it at CERN in the beginning. HTML 2.0 was designed in late 1994 under the authority of the Internet Engineering Task Force (IETF) to standardize best practices. HTML 3.0 and HTML+ (1993) both proposed much richer HTML versions.

c Cascading Style sheet (CSS):

CSS is a way of distributing stylesheets over the World Wide Web (WWW). CSS enables the separation of content and style, as well as the generic application of style to content. CSS was initially used in October 1995, and it was included in the World Wide Web Consortium's recommendation in March 1998. CSS originally included layout, color, fonts, and simple presentation techniques that allowed for information structuring and highlighting, providing basic depth to the textual level of the website. It enables developers to present information in a way that conveys its nature and importance. Absolute positioning using the z index, which allows for element overlapping in CSS2, is one of the powerful techniques in CSS.

d jQuery:

Jquery is a subset of javascript used for both client-side and server-side, that allows for the development of an interactive web page. Jquery simplifies the interactions between a Javascripts and HTML/CSS documents or Document

Object Model (DOM). Jquery is an open source javascript library which makes HTML document traversing and manipulation, Ajax interaction, event handling in browser easier.

e **Ajax Api:**

The term Ajax(Asynchronous JavaScript and XML) was formed by Jesse James Garett in 2005. It is used in web development to create asynchronous web applications. It allows exchange of data between the web browser and the server without page refreshing. It makes web applications to send request to server and receive response asynchronously without user interaction page being disrupted. This was possible by the convergence of several technologies like; JavaScript, XMLHttpRequest(XHL) object, XML(Extensible Markup Language), CSS etc.

f **Javascripts:**

JavaScript (commonly abbreviated as "JS") is a dynamic programming language with full capability that can be used to improve the interactivity of a website. Brendan Eich created JavaScript while working for Netscape Communications Corporation in 1995. The term Mocha was given to the first version of JavaScript, which was eventually superseded by LiveScript before becoming JavaScript. The JavaScript API allows scripts that be uploaded with the web page and run in the client's browser to provide client-side functionality. Javascript can be used to alter and interact with the Document Object Model. The Document Object Model (DOM) offers several features, including the ability to change and create style, tags, and element layout. It is a first-class programming language that may be interpreted or compiled just-in-time.

g **Python Language:**

Python is a general-purpose language. It is frequently used to create software and websites, automate processes, and analyze data. It was developed by Guido van Rossum, and was published in 1991. It is widely used as it syntax enables programmers to construct applications with fewer lines of code which makes it differ from other programming languages. The most recent version of python is python 3.11.

h **PostgreSQL:**

PostgreSQL is a free and open source relational database that provides different support for SQL like subqueries, different user-defined types and functions. Originally called POSTGRES, as an object-oriented database which later on was named Posgres95, after SQL capabilities is added. The name was changed from Posgres95 to PosgreSQL in 1996. Its technical features includes; Multi-Version Concurrency Control (MVCC), ACID(Atomicity, Consistency, Isolation, Durability) complaint.

2.7 Related work

(J. Thakkar, 2018) worked on the AI-based chatbot named Eramus, which answers questions on university information. The authors designed Erasmus as an end-to-end system using cloud services, starting from api.ai (Dialogflow), Mlab (MongoDB cloud), IBM Bluemix (webhook API). However, their chatbot took quite a long latency in responding to the users because it uses too many cloud services. (B.P. Prashant, 2017) proposed an online chatting system for college inquiries using a knowledgeable database which did pattern matching to perform chatbots' information retrieval. All the detailed working steps in this research are clear with UML and various process diagrams. However, their bot did not use a machine learning approach, so that is still too rigid by rules for students and parents to make inquiries. The patterns were built manually, so it does not scale in the real use case. (Akande, 2020) proposed a university enquiry chatbot system using Google DialogFlow framework. Keyword matching algorithm and string distance-comperison algorithm so as retrieve best possible result. Every detailed steps to make the system successful was clear using UML diagrams. However, the prosposed system could not guide the user or ask the user different options on what they would like to achieve unless otherwise stated by the user.

CHAPTER THREE
METHODOLOGY OF THE STUDY

3.1 Method of identification of user and system requirements

Informal interview and reviewing of other existing systems in the related context were used to identify the user and the system requirements during this study. The technique used to identify the system requirements include; Functional requirements, nonfunctional requirements, hardware requirements and software requirements. During the course of development of the system, different observations were carried out on systems with same functionality. From the observation, different features needed to be implemented were identified. A prototype version of the system was developed and made available to the users to experiment with. This help to refine the system and add new features that has not yet been identified.

3.1.1 Identification of the system requirements

In this section, the functional and nonfunctional requirements of the system will be discussed. It identify the essential features that the system should provide. It covers the data to be used by the system as well as the operations to be performed.

a. **Functional Requirements**

These are the functionalities provided by the proposed software. They include:

i. The system shall provide a screen for a user’s input to be seen.
ii. The system guides the user after greetings from the user.
iii. The system shall provide direct access so it can be used by the user without signing in.
iv. The system give answers to users queries based on admission
v. The system give answers to users queries based on tuition fees
vi. The system shall provide an avenue for the developer to perform regular system check to ensure more efficiency in response between the bot and the user.

b **Nonfunctional Requirements**

The non-functional requirements include:

i. The system shall be able to respond to any query regarding admission in a text or sentence even if the words are misspelled.
ii. The system shall provide an interface that is easy to use and understand to the users.

c **Hardware Requirements**

Processor and processor speed: Core i3(minimum) and 1.20GHz(minimum), RAM: 4gb(minimum), Hard disk: 255gb(minimum for SDD) and 500gb(minimum for HDD).

d Software Requirements

The required software to implement the system include; Visual studio code (for desktop C++ redistribution), Windows 7+, Rasa Open Source(rasa[full]), Python , VS code, Postgresql.

3.1.2 Identification of user requirements

This section gives an overview of the functionalities the system is supposed to make available to the users of the system. They include;

a. It will provide a text area or text input for the user to input his/her text.
b. It will provide a submit button for the user to send his/her text after input.
c. It will provide a reply message to be displayed to the user when his/her text has been submitted which align to the query sent by the user.
d. It shall be very user friendly, interoperable and flexible.

3.2 System design methods

The system was designed using several UML diagrams.These UML diagrams provides graphical representation of the interaction between the system and users of the system, the hierarchy of activities in the system and the flow of control between these activities. This system architecture was also designed and described in this section. The followings are illustrated below;

3.2.1 System architecture

The architectural diagram describes the architectural structure of all the components in the system. It also helps to illustrates how the components of the system communicate with each other. The proposed software consists majorly of five components. The users of the system, the user interface they communicate with, HTML and CSS UI components, application logic implemented in Jquery, Rasa with Python backend, and an API. The architectural diagram of the proposed system is graphically illustrated in Figure 3.1.

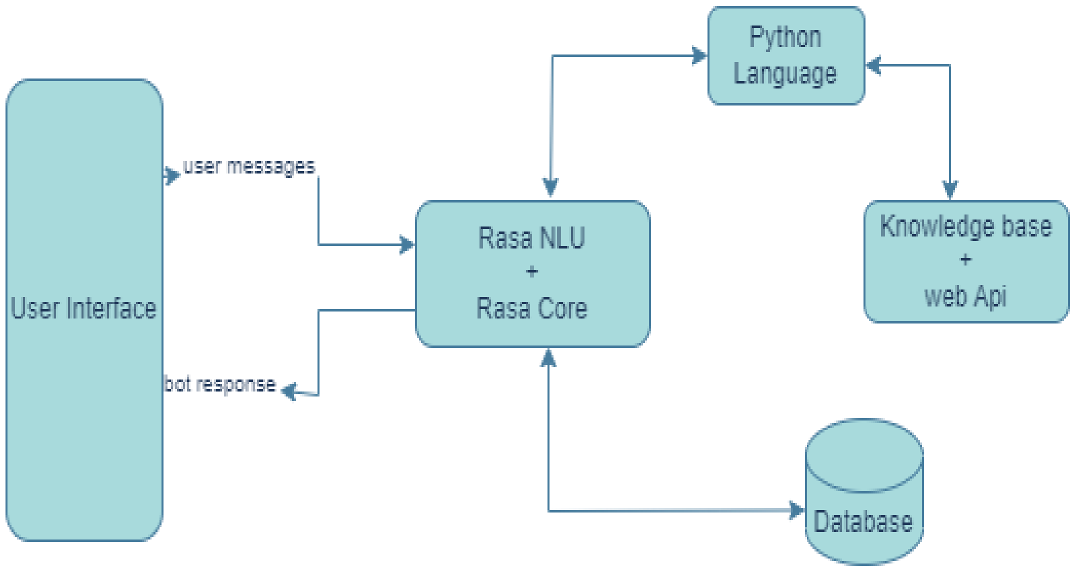


Figure 3.1 System Architecture

3.2.2 Use case diagram

The use case diagram as shown in figure 3.2 and figure 3.3 represents a workflow of the activities that takes place in the system and the various actions that they can perform. The users that make use of the system are:

a User use case diagram

In Figure 3.2, the role of the user is to send messages or text through the system and receive responses from the system relating to the query being sent. The figure below illustrates the user use case scenarios.

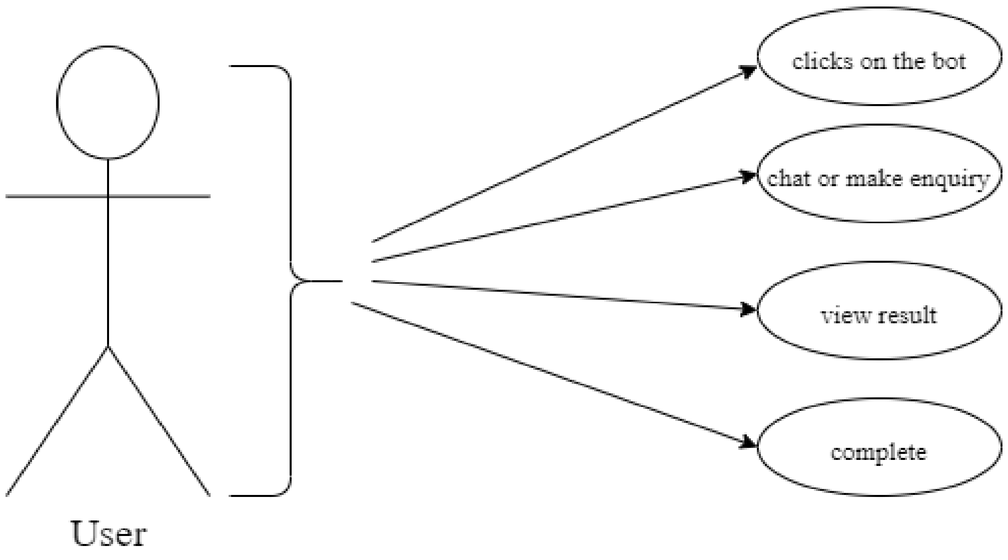


Figure 3.2 User Use Case Diagram

b **Admin use case diagram**

In Figure 3.3, the role of the admin is to manage the interactions between the system and the user. They are responsible for monitoring the system and receiving usage report of the system. The figure below illustrates the administrator use case scenarios.

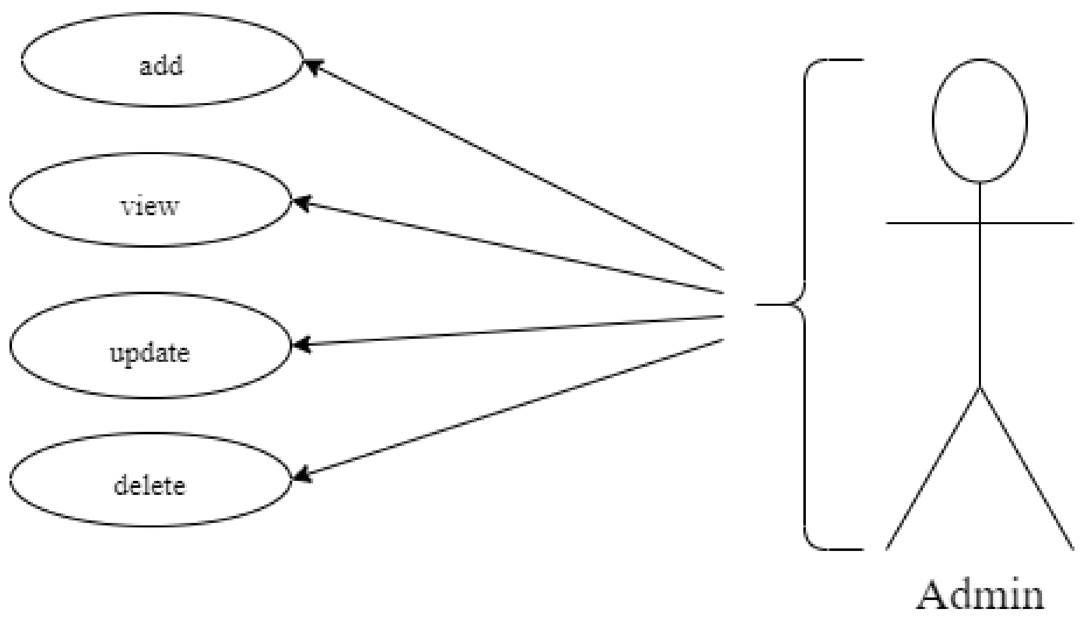


Figure 3.3 Admin Use Case Diagram

3.2.4 Sequence diagram

The sequence diagram represents a workflow of the activities that takes place in the system. It shows the internal component and the user involved in the process. Some of the processes designed with this diagram are User and Bot converations, Webhook sequence and so on.

a User-Bot Conversation Sequence:

Figure 3.4 below, describe the activities between the user and the bot. When the user send a query from the user interface to the chatbot, the webhook receives the input as a plain text then, it being converted to json which is then send to te chatbot database. The message is being processed by the chatbot database and resulted output is sent to the webhook in json which is inturn converted to plain text and shown to the user.

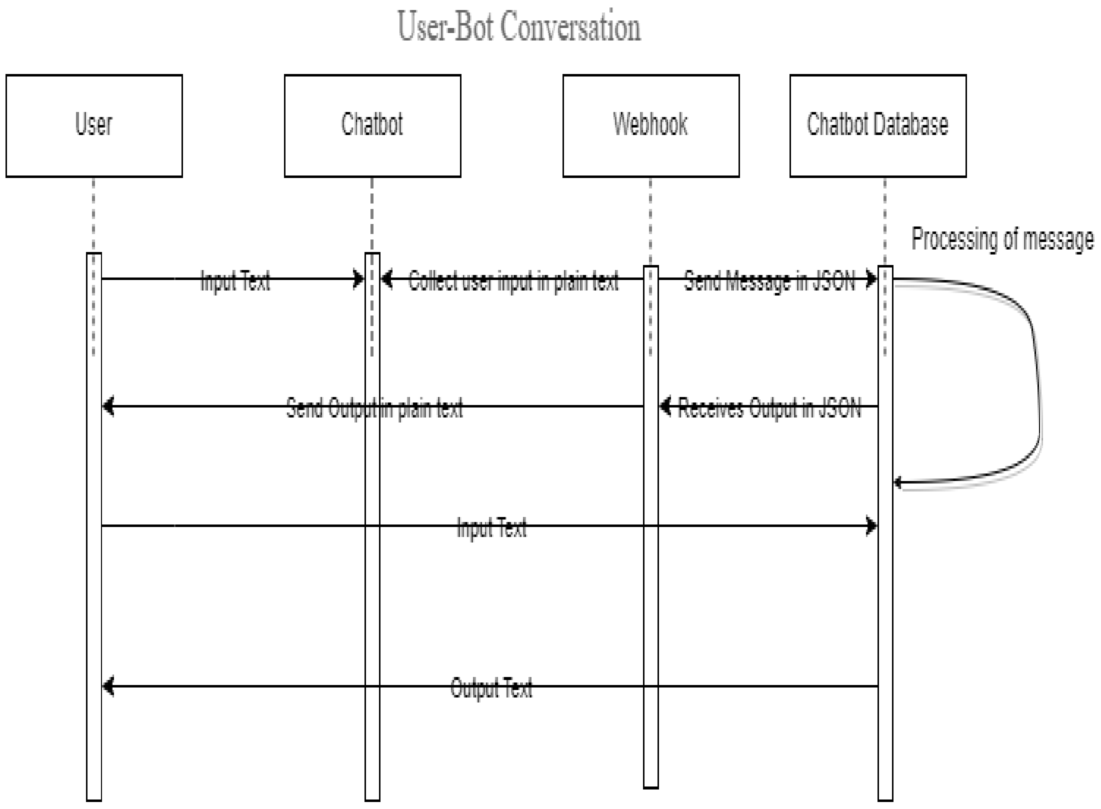


Figure 3.4 Sequence Diagram for User-Bot Conversation

b Webhook Sequence:

Figure 3.5 below, describe the activites between the webhook and the chatbot database. When a text in json is sent to the chatbot database and it been processed, response output by the chatbot database is based on the confidence level of the response relating to the query. When the confidence level of a response is greater than 0.6, the response chosen as the accurate response, when it is equal to 0.5, memory cache is checked to get response if similar input is present and when it is less than 0.5, inaccurate response is generated.

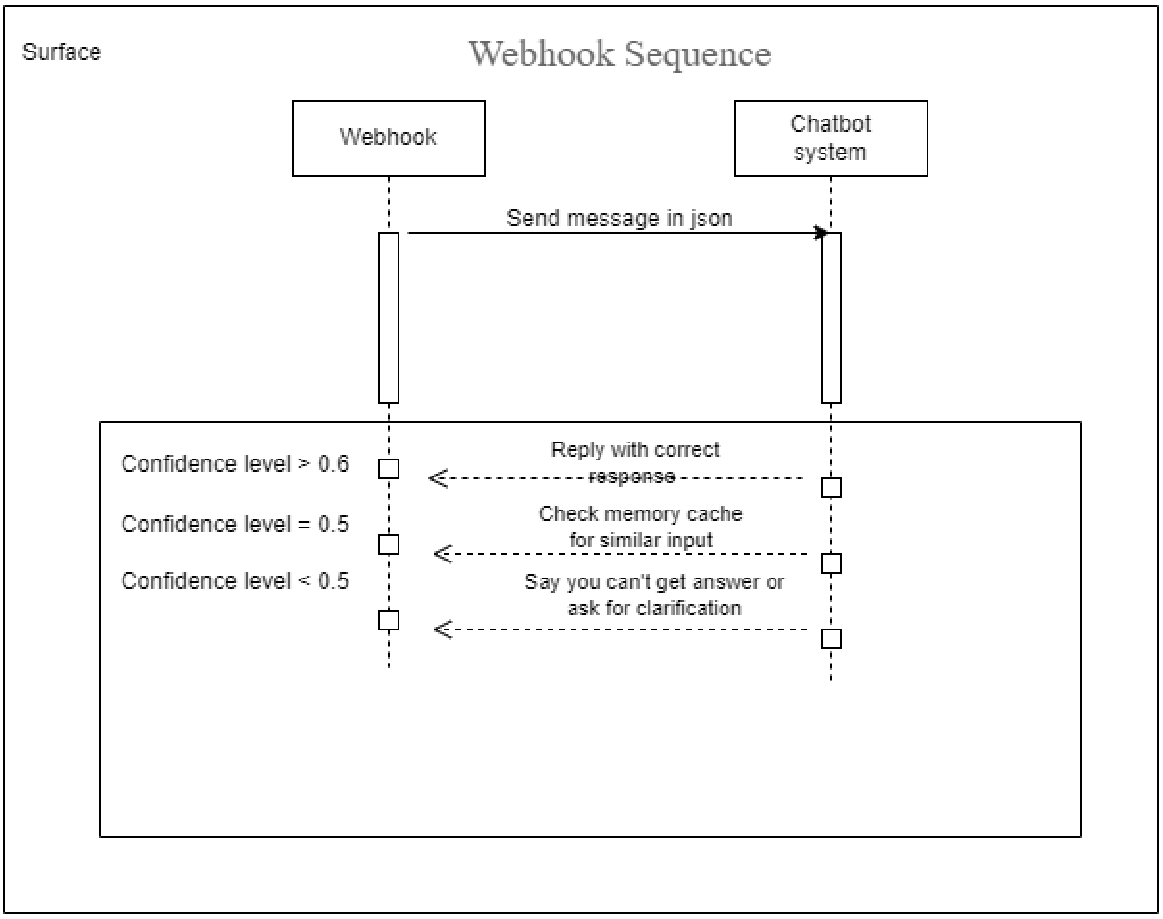
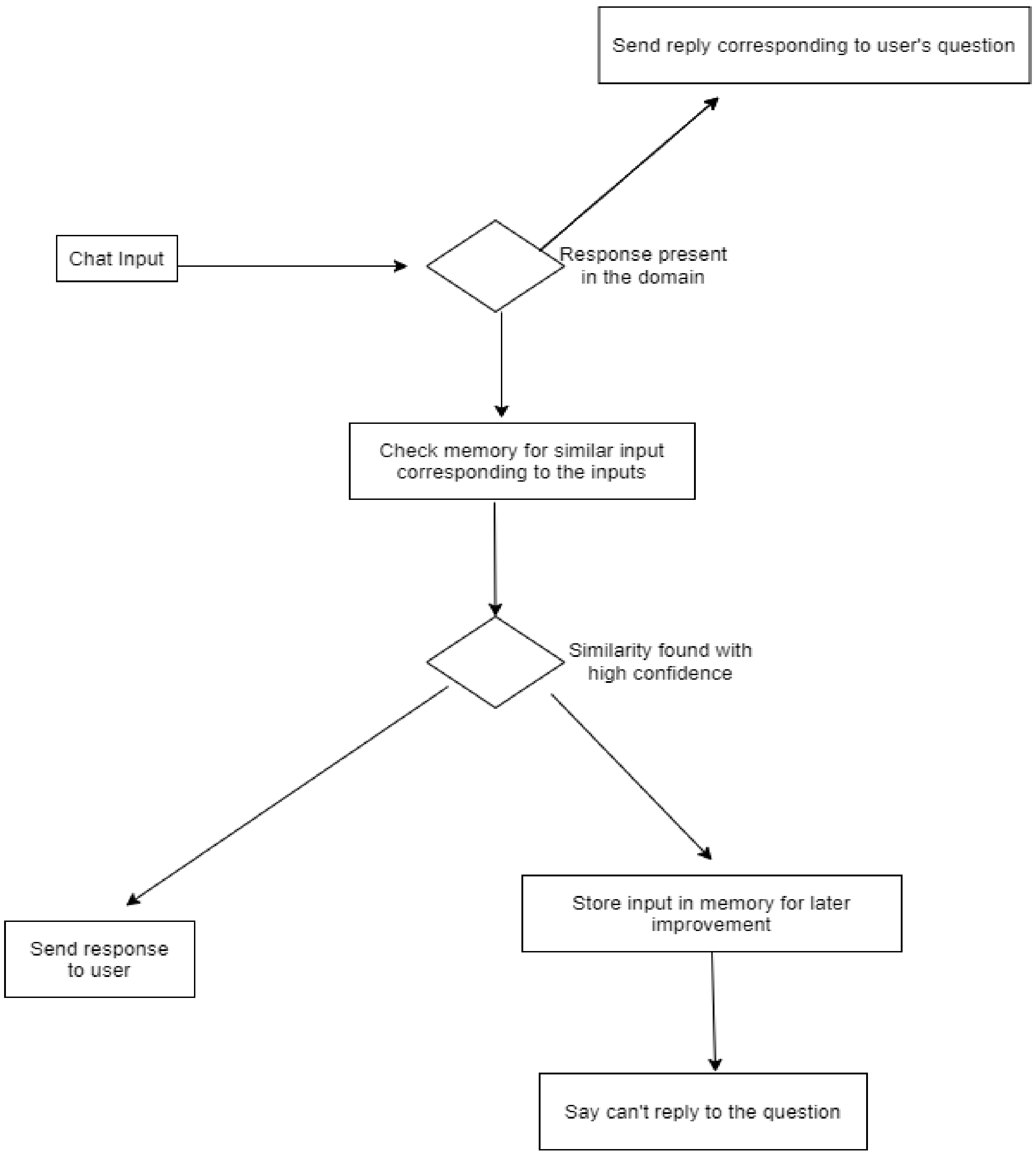


Figure 3.5 Sequence Diagram for Webhook and Chatbot Database

3.2.4 Activity diagram

Figure 3.6, illustrates all the activities that occur within the system and the flow of control between these activities. It shows the standard order of activities in the software system.



Send reply corresponding to user's question
Chat Input
Response present in the domain
Check memory for similar input corresponding to the inputs
Similarity found with high confidence
Send response to user
Store input in memory for later improvement
Say can't reply to the question

Figure 3.6 Activity Diagram

3.3 System implementation

This system was implemented using the following technology and languages. The frontend was implemented using Html, CSS, jQuery, Javascripts and Ajax api and the backend was implemented using Rasa Open Source and Python Language with PostgreSQL as the database

3.3.1 Frontend Implementation

a. HTML

Html is a markup language that was use as the skeleton of the interface ui. It is used to develop the structure of the web. HTML5 is version used for this system.

b. Cascading Style Sheet (CSS)

It was used to style the view pages by accessing the style classes declared in the CSS files. The classes were made available by a declaration of the path of the styles sheet in the head tag of the html, using a link tag.

c. jQuery

Jquery is a subset of javascript used for both client-side and server-side, that allows for the development of an interactive web page. It is used for both frontend and backend implementation across the web development stack. It was used in this project for the user interface ui, for easy interaction between the user and bot.

d. Ajax API

Ajax api call was used with Jquery to make a post request in JSON format to the bot to send different queries from the user and get back response from rasa provided api link.

3.3.2 Backend Implementation

a. Rasa Open Source

Rasa is a conversational AI application. It contains both Rasa NLU and Rasa Core. It helps to easily build a simple chatbot using natural language processing and machine learning tools.

b. Python Language:

Python is a general-purpose language that may be used to make a wide range of programs and isn't tailored for any particular issues. Its versatility and beginner-friendliness have elevated it to the top of today's most popular programming languages. It’s used to retrieve information from an api.

c. **PostgreSQL:**

PostgreSQL is a free and open source relational database that provides different support for SQL like subqueries, different user-defined types and functions. It’s used to store conversations held between the user and the bot.

CHAPTER FOUR
IMPLEMENTATION AND RESULT

4.0 Overview

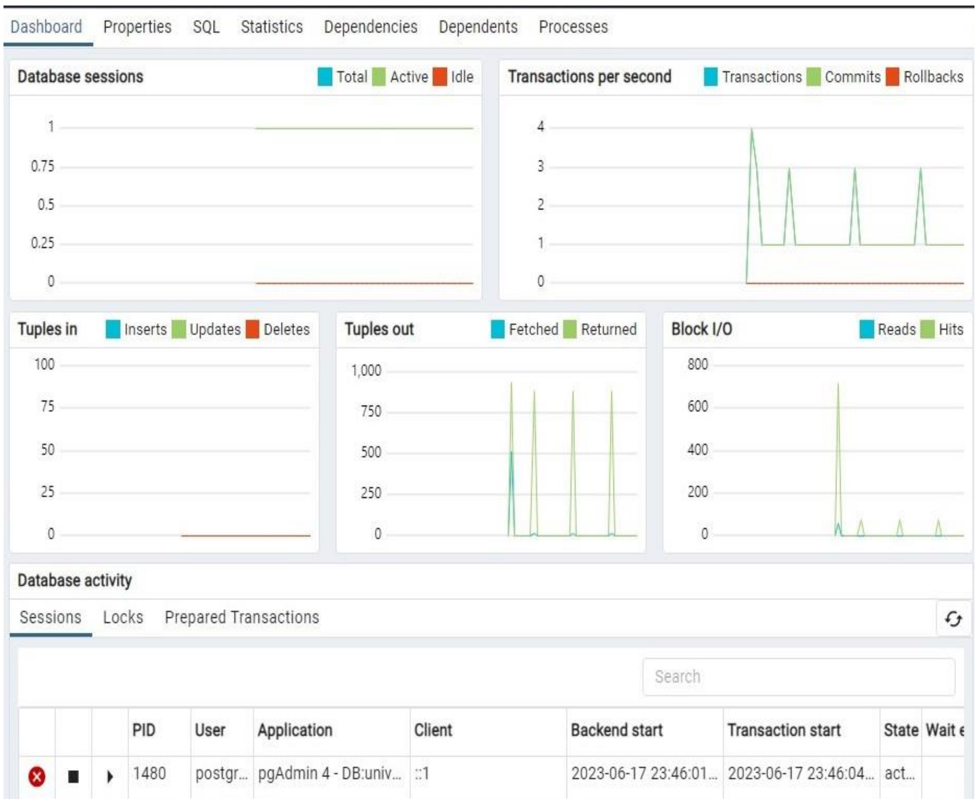
In this chapter, the implementation and the results of the Online Chatbot Admission Equiry that are obtained are illustrated. It covers the results of the database, implemented using Postgresql, and the frontend implementation is implemented using HTML, CSS, jQuery, JavaScript, and Ajax Api. The result will be a web application.

4.1 Database implementation

In figure 4.1, it shows the dashboard of the Online Chatbot Admission Enquiry database server. The dashboard consists of the Database sessions, Transactions per second, Turples in, Turples out, Block I/O and Database activity. Figure 4.2 shows the details of the database activity, which consist of the idle and active state of sessions connected to the backend of the application. Figure 4.3 shows the activities between the bot and the user in a table-like manner. It consists of the Id, Sender-id, Type_name, Intent_name, Timestamp, Action_name, Data. Figure 4.4 shows the column queried using SQL which consist just the Intent_name, Type_name and Data. Figure 4.5 shows the responses of the bot in respect to the user's input with column consisting of just the Type_name and Data queried using SQL. Figure 4.6 shows a Stacked Bar Chart illustration of the Intent_name and the Timestamp.

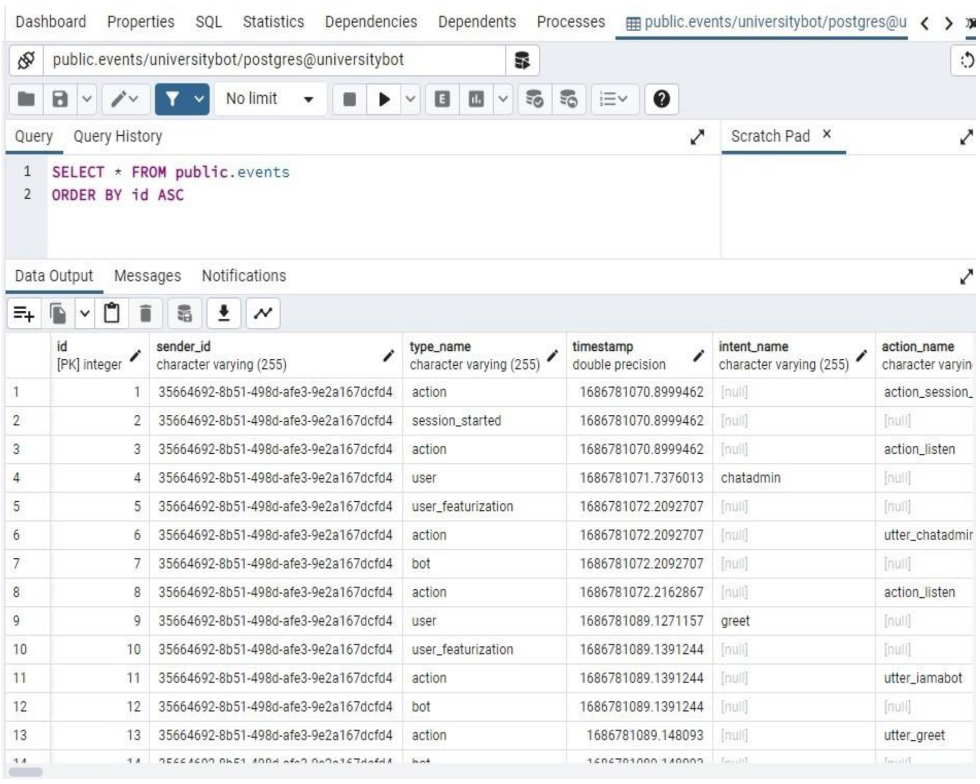
4.2 Frontend implementation of the online chatbot for admission enquiries

Figure 4.7 shows the starting icon of the chatbot. This is shown when the page is newly visited or refreshed. It consist of a pop-up text which notifies the user when the page is refreshed or newly visited. In Figure 4.8, it shows the bot directing the user with some given options on what the user can do. These options are Admission, Info and Others. These options or suggestions are given when the user start by greeting. Figure 4.8.1(a), shows the interaction between the bot and the user after the



			PID	User	Application	Client	Backend start	Transaction start	State	Wait e
			1480	postgr...	pgAdmin 4 - DB:univ...	::1	2023-06-17 23:46:01...	2023-06-17 23:46:04...	act...	

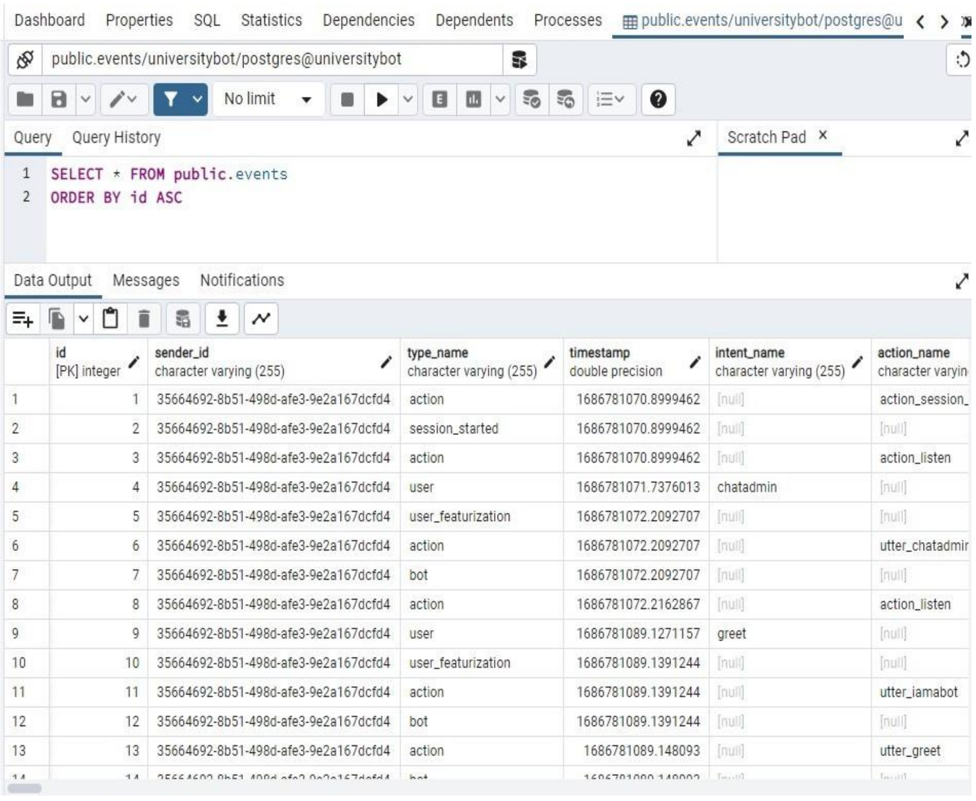
Figure 4.1 Dashboard of the Database Server



```
SELECT * FROM public.events
ORDER BY id ASC
```

	id [PK] integer	sender_id character varying (255)	type_name character varying (255)	timestamp double precision	intent_name character varying (255)	action_name character varyin
1	1	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781070.8999462	[null]	action_session_
2	2	35664692-8b51-498d-afe3-9e2a167dcfd4	session_started	1686781070.8999462	[null]	[null]
3	3	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781070.8999462	[null]	action_listen
4	4	35664692-8b51-498d-afe3-9e2a167dcfd4	user	1686781071.7376013	chatadmin	[null]
5	5	35664692-8b51-498d-afe3-9e2a167dcfd4	user_featurization	1686781072.2092707	[null]	[null]
6	6	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781072.2092707	[null]	utter_chatadmir
7	7	35664692-8b51-498d-afe3-9e2a167dcfd4	bot	1686781072.2092707	[null]	[null]
8	8	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781072.2162867	[null]	action_listen
9	9	35664692-8b51-498d-afe3-9e2a167dcfd4	user	1686781089.1271157	greet	[null]
10	10	35664692-8b51-498d-afe3-9e2a167dcfd4	user_featurization	1686781089.1391244	[null]	[null]
11	11	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781089.1391244	[null]	utter_iamabot
12	12	35664692-8b51-498d-afe3-9e2a167dcfd4	bot	1686781089.1391244	[null]	[null]
13	13	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781089.148093	[null]	utter_greet

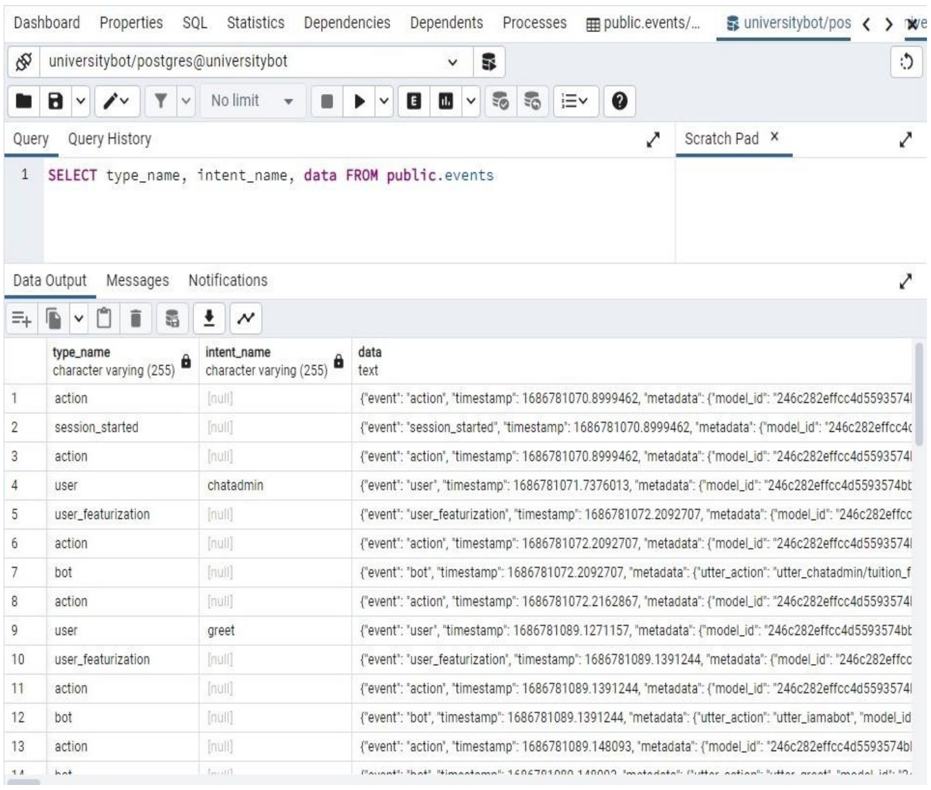
Figure 4.2 Database Activities Sessions



```sql
SELECT * FROM public.events
ORDER BY id ASC
```

	id [PK] integer	sender_id character varying (255)	type_name character varying (255)	timestamp double precision	intent_name character varying (255)	action_name character varyin
1	1	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781070.8999462	[null]	action_session_
2	2	35664692-8b51-498d-afe3-9e2a167dcfd4	session_started	1686781070.8999462	[null]	[null]
3	3	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781070.8999462	[null]	action_listen
4	4	35664692-8b51-498d-afe3-9e2a167dcfd4	user	1686781071.7376013	chatadmin	[null]
5	5	35664692-8b51-498d-afe3-9e2a167dcfd4	user_featurization	1686781072.2092707	[null]	[null]
6	6	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781072.2092707	[null]	utter_chatadmir
7	7	35664692-8b51-498d-afe3-9e2a167dcfd4	bot	1686781072.2092707	[null]	[null]
8	8	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781072.2162867	[null]	action_listen
9	9	35664692-8b51-498d-afe3-9e2a167dcfd4	user	1686781089.1271157	greet	[null]
10	10	35664692-8b51-498d-afe3-9e2a167dcfd4	user_featurization	1686781089.1391244	[null]	[null]
11	11	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781089.1391244	[null]	utter_iamabot
12	12	35664692-8b51-498d-afe3-9e2a167dcfd4	bot	1686781089.1391244	[null]	[null]
13	13	35664692-8b51-498d-afe3-9e2a167dcfd4	action	1686781089.148093	[null]	utter_greet

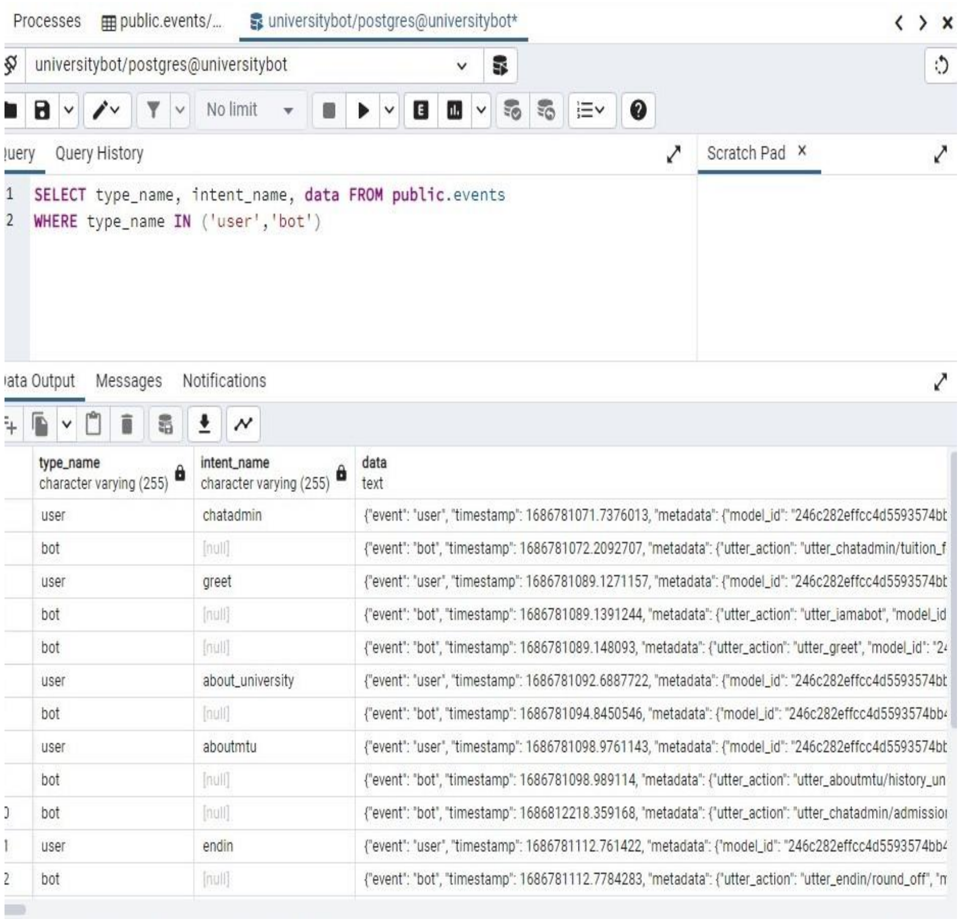
Figure 4.3 Activity Table



```sql
SELECT type_name, intent_name, data FROM public.events
```

	type_name character varying (255)	intent_name character varying (255)	data text
1	action	[null]	{"event": "action", "timestamp": 1686781070.8999462, "metadata": {"model_id": "246c282effcc4d5593574l
2	session_started	[null]	{"event": "session_started", "timestamp": 1686781070.8999462, "metadata": {"model_id": "246c282effcc4c
3	action	[null]	{"event": "action", "timestamp": 1686781070.8999462, "metadata": {"model_id": "246c282effcc4d5593574l
4	user	chatadmin	{"event": "user", "timestamp": 1686781071.7376013, "metadata": {"model_id": "246c282effcc4d5593574bt
5	user_featurization	[null]	{"event": "user_featurization", "timestamp": 1686781072.2092707, "metadata": {"model_id": "246c282effcc
6	action	[null]	{"event": "action", "timestamp": 1686781072.2092707, "metadata": {"model_id": "246c282effcc4d5593574l
7	bot	[null]	{"event": "bot", "timestamp": 1686781072.2092707, "metadata": {"utter_action": "utter_chatadmin/tuition_f
8	action	[null]	{"event": "action", "timestamp": 1686781072.2162867, "metadata": {"model_id": "246c282effcc4d5593574l
9	user	greet	{"event": "user", "timestamp": 1686781089.1271157, "metadata": {"model_id": "246c282effcc4d5593574bt
10	user_featurization	[null]	{"event": "user_featurization", "timestamp": 1686781089.1391244, "metadata": {"model_id": "246c282effcc
11	action	[null]	{"event": "action", "timestamp": 1686781089.1391244, "metadata": {"model_id": "246c282effcc4d5593574l
12	bot	[null]	{"event": "bot", "timestamp": 1686781089.1391244, "metadata": {"utter_action": "utter_iamabot", "model_id
13	action	[null]	{"event": "action", "timestamp": 1686781089.148093, "metadata": {"model_id": "246c282effcc4d5593574bl
14	bot	[null]	{"event": "bot", "timestamp": 1686781089.148093, "metadata": {"utter_action": "utter_greet", "model_id": "2

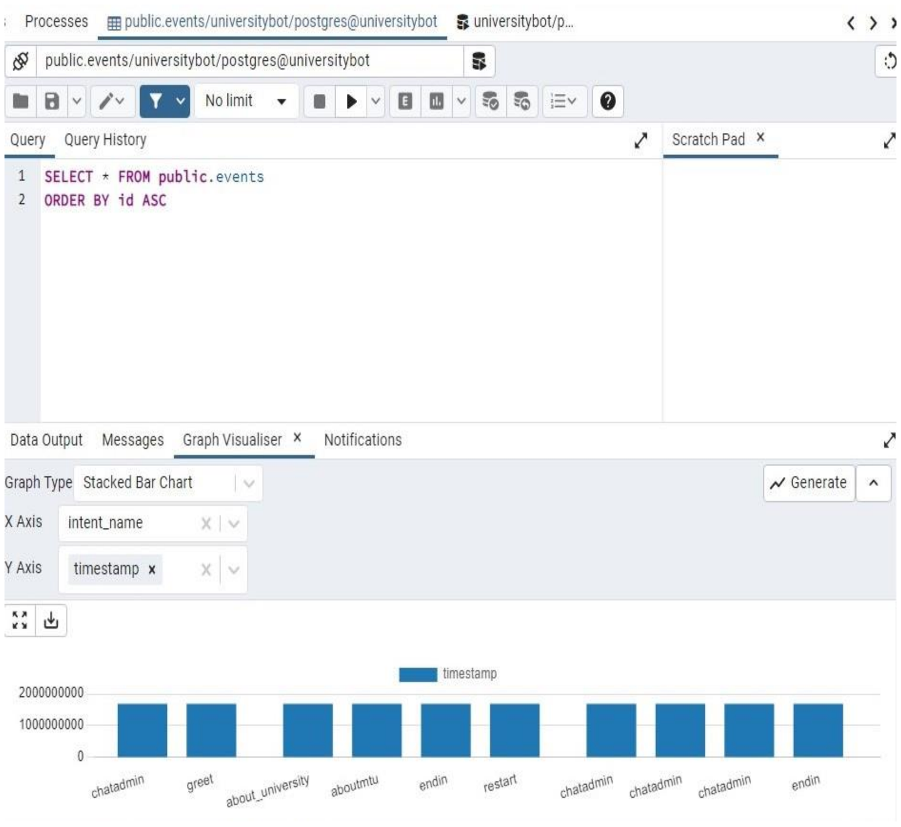
Figure 4.4 Custom Table using SQL



```sql
SELECT type_name, intent_name, data FROM public.events
WHERE type_name IN ('user','bot')
```

	type_name character varying (255)	intent_name character varying (255)	data text
	user	chatadmin	{"event": "user", "timestamp": 1686781071.7376013, "metadata": {"model_id": "246c282effcc4d5593574bt
	bot	[null]	{"event": "bot", "timestamp": 1686781072.2092707, "metadata": {"utter_action": "utter_chatadmin/tuition_f
	user	greet	{"event": "user", "timestamp": 1686781089.1271157, "metadata": {"model_id": "246c282effcc4d5593574bt
	bot	[null]	{"event": "bot", "timestamp": 1686781089.1391244, "metadata": {"utter_action": "utter_iamabot", "model_id
	bot	[null]	{"event": "bot", "timestamp": 1686781089.148093, "metadata": {"utter_action": "utter_greet", "model_id": "2
	user	about_university	{"event": "user", "timestamp": 1686781092.6887722, "metadata": {"model_id": "246c282effcc4d5593574bt
	bot	[null]	{"event": "bot", "timestamp": 1686781094.8450546, "metadata": {"model_id": "246c282effcc4d5593574bb
	user	aboutmtu	{"event": "user", "timestamp": 1686781098.9761143, "metadata": {"model_id": "246c282effcc4d5593574bt
	bot	[null]	{"event": "bot", "timestamp": 1686781098.989114, "metadata": {"utter_action": "utter_aboutmtu/history_un
0	bot	[null]	{"event": "bot", "timestamp": 1686812218.359168, "metadata": {"utter_action": "utter_chatadmin/admissio
1	user	endin	{"event": "user", "timestamp": 1686781112.761422, "metadata": {"model_id": "246c282effcc4d5593574bb
2	bot	[null]	{"event": "bot", "timestamp": 1686781112.7784283, "metadata": {"utter_action": "utter_endin/round_off", "n

Figure 4.5 User and Bot Conversation



```
SELECT * FROM public.events
ORDER BY id ASC
```

Graph Type: Stacked Bar Chart

X Axis: intent_name

Y Axis: timestamp

Figure 4.6 Graphical Visualization

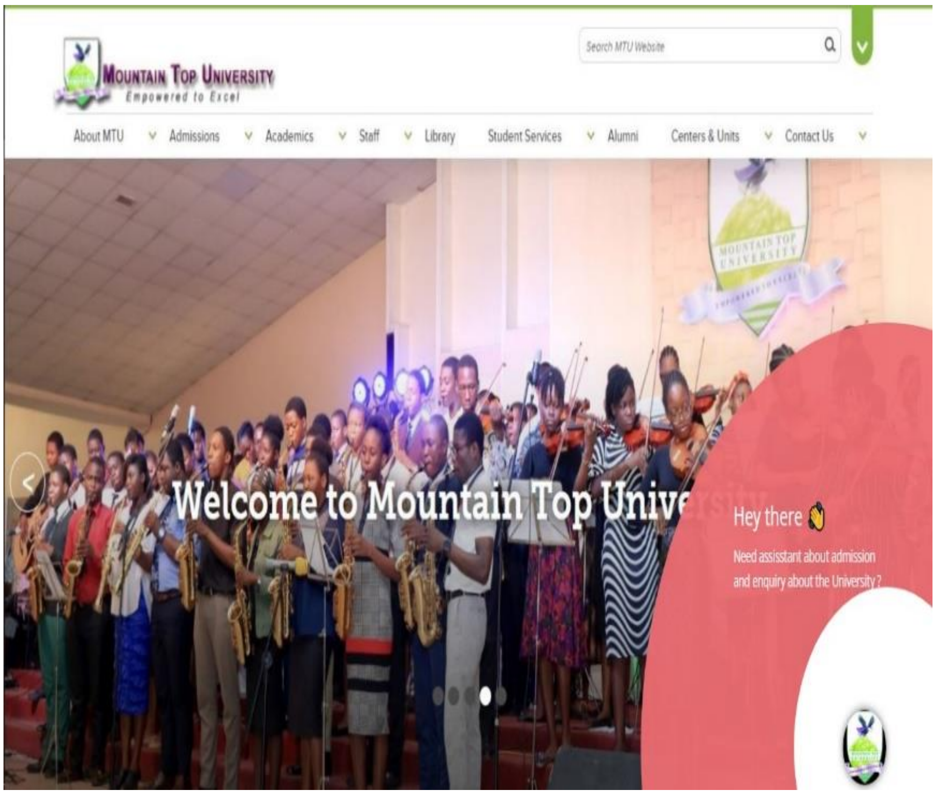
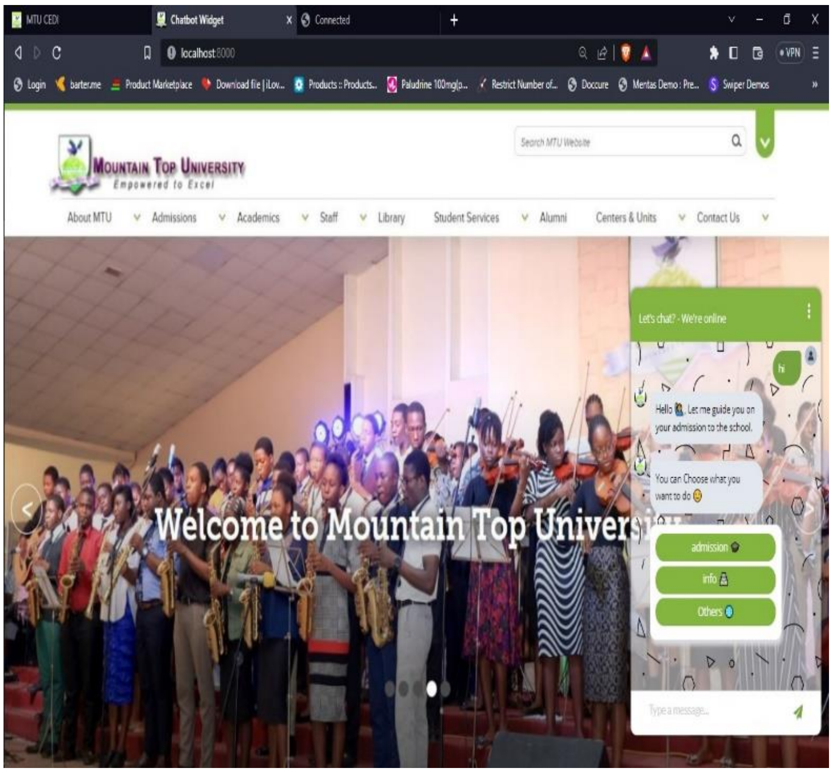


Figure 4.7 Starting Icon

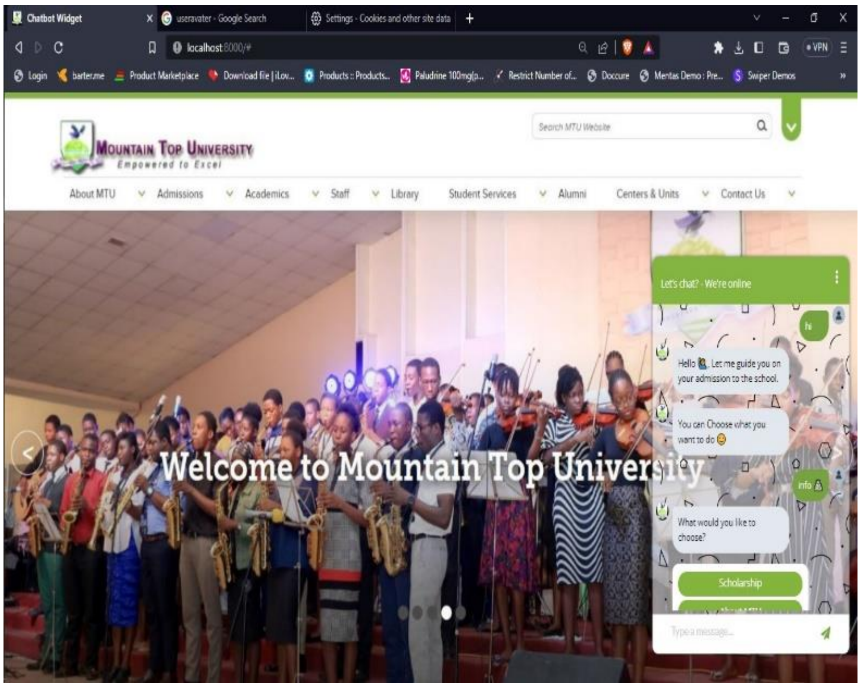


hi

Hello 🙋. Let me guide you on your admission to the school.

You can Choose what you want to do 🙂

Figure 4.8 Chatbot Directing the User



hi

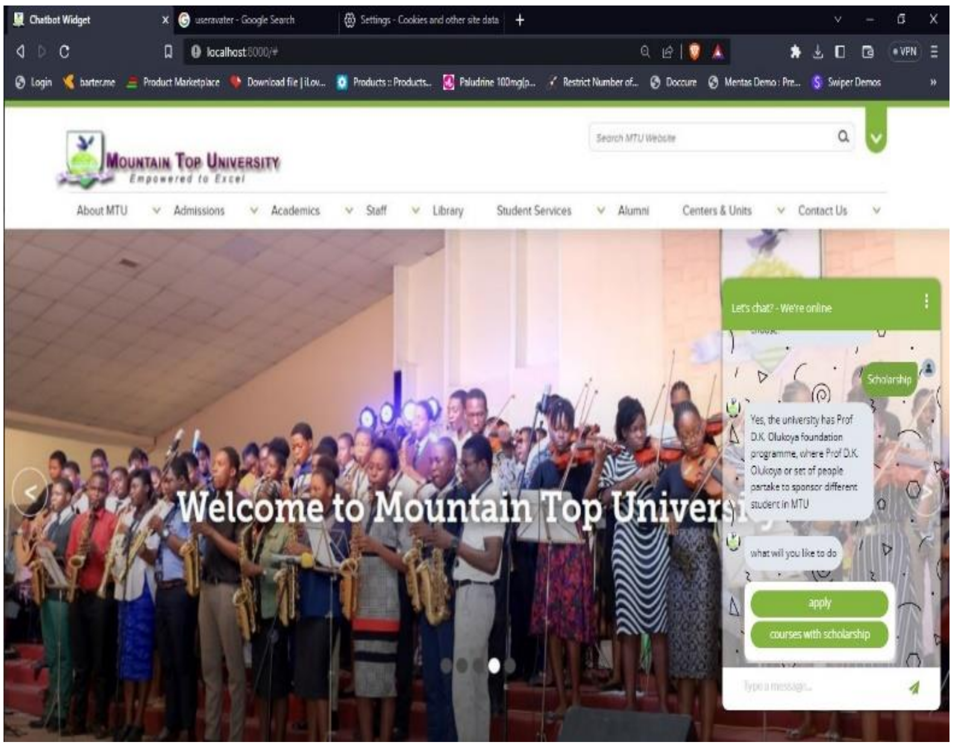
Hello 🙋. Let me guide you on your admission to the school.

You can Choose what you want to do 🙂

info ℹ️

What would you like to choose?

Figure 4.8.1(a) Chatbot giving response to user's choice

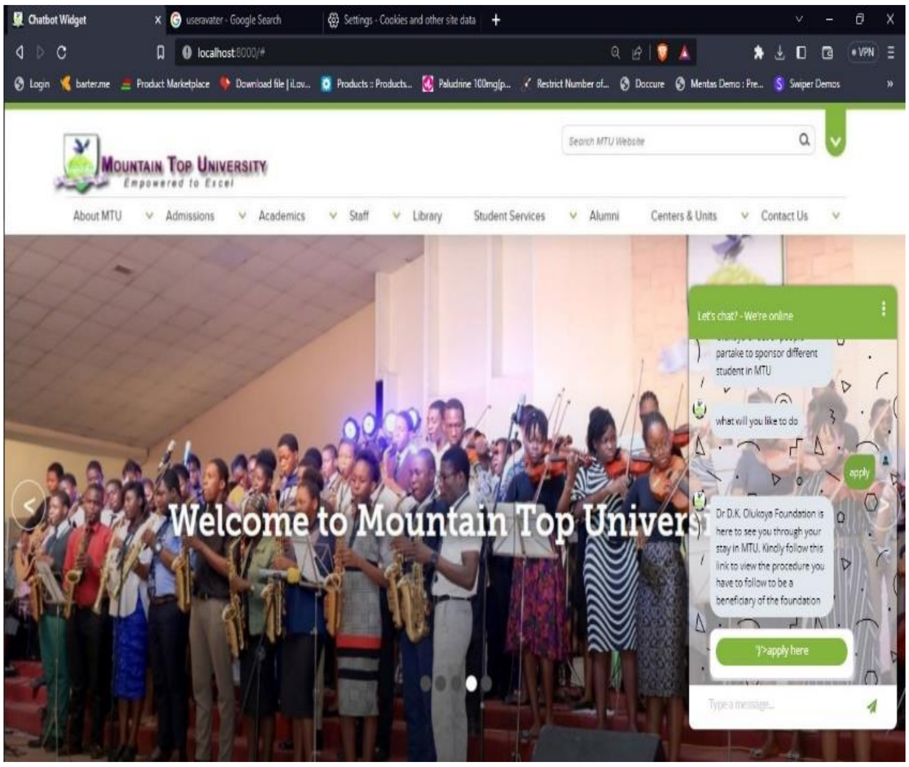


Scholarship

Yes, the university has Prof D.K. Olukoya foundation programme, where Prof D.K. Olukoya or set of people partake to sponsor different student in MTU

what will you like to do

Figure 4.8.1(b) Chatbot giving response to user's choice



partake to sponsor different student in MTU

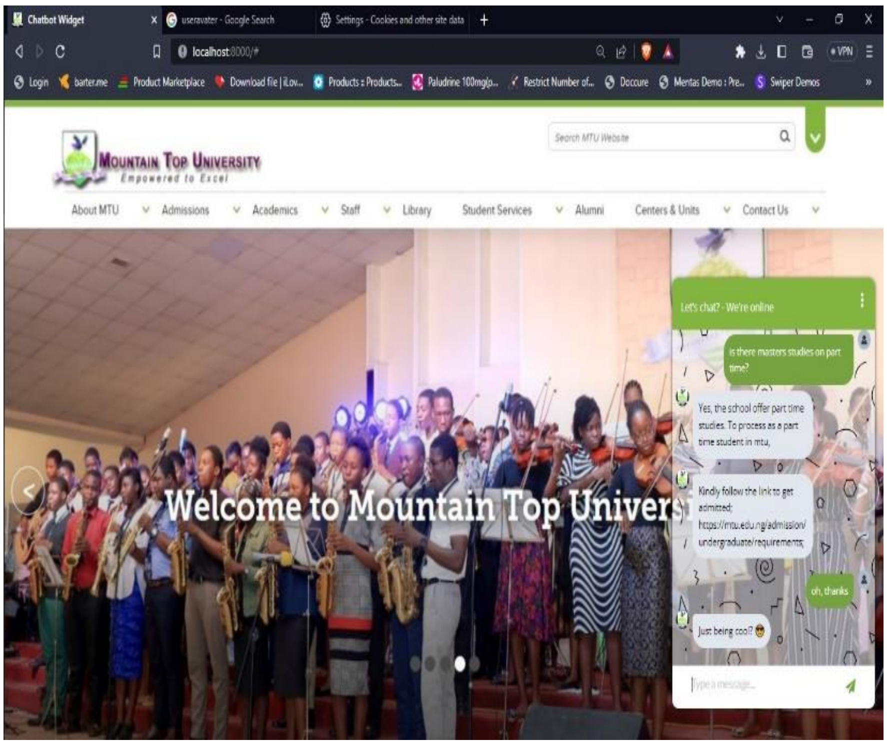
what will you like to do

apply

Dr D.K. Olukoya Foundation is here to see you through your stay in MTU. Kindly follow this link to view the procedure you have to follow to be a beneficiary of the foundation

'}'>apply here

Figure 4.8.1(c) Chatbot giving response to user's choice



Let's chat? - We're online

is there masters studies on part time?

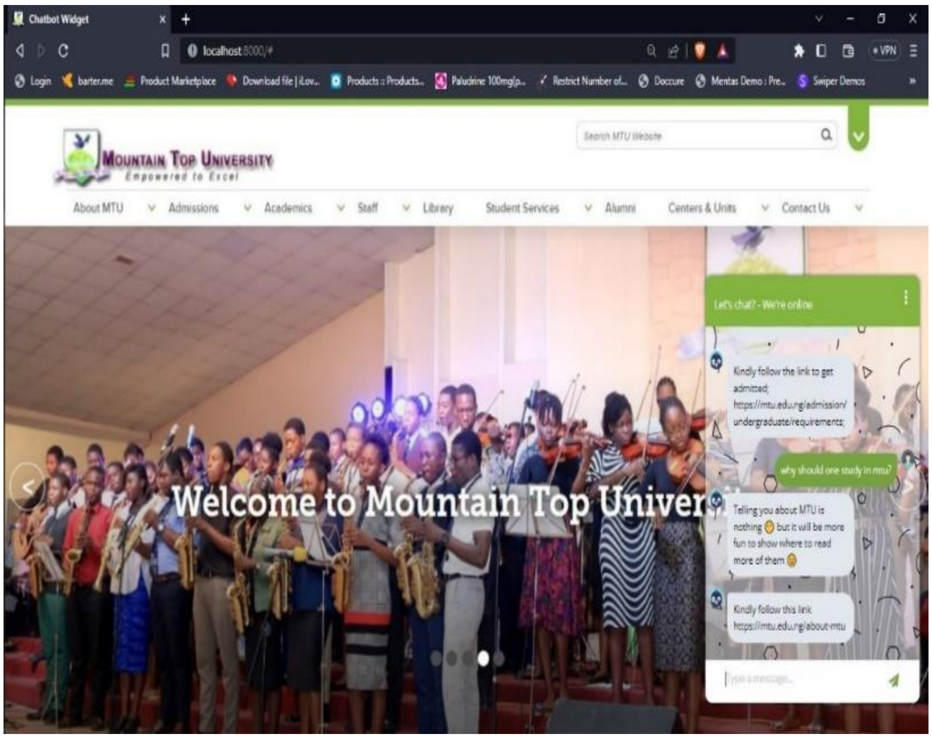
Yes, the school offer part time studies. To process as a part time student in mtu,

Kindly follow the link to get admitted; https://mtu.edu.ng/admission/undergraduate/requirements;

oh, thanks

Just being cool? 😎

Figure 4.9(a) Chatbot responding to user's query directly



Kindly follow the link to get admitted; https://mtu.edu.ng/admission/undergraduate/requirements;

why should one study in mtu?

Telling you about MTU is nothing 🙂 but it will be more fun to show where to read more of them 😉

Kindly follow this link https://mtu.edu.ng/about-mtu

Figure 4.9(a) Chatbot answering to user’s queries

user picked an option that was suggested. It consist of the bot response with other suggestions given which are Scholarship and About MTU. Figure 4.8.1(b), show the bot giving the user other choices after responding to the first query chosen by the user. Figure 4.8.1(c), shows another suggestion after the bot responding to the picked suggestion given by the bot before. Figure 4.9(a) and Figure 4.9(b), shows a user querying the bot directly and the bot responding to the user's queries without minding if the user greets or not.

4.3 Discussion of results

The result of the study on the topic Online Chatbot for Admission Enquiries presented an expected results based on the objectives that were initially stated on the topic. The result of user and system requirements aided the definition of the system users being admission officer as the primary user and the secondary user being the prospective students. The results shows the admission officer was responsible for the oversight function on the system being able to improve the friendliness of the bot with the user through conversation between the bot and the users in the database. And also generate a graphical representation of the timeframe at which most users visit or interact with the bot.

The results also shows that the prospective students can access the system without any need for authentication. The result of the web interface shows that the system was able to provide an interface efficiently suits the system and user requirement that were identified in the course of the study. The result revealed that implementing a chatbot admission enquiry is very possible. Hence, it is safe to say that the result of this system present a solution to the workload posed to admission staffs and slow and inaccurate response to user's queries.

CHAPTER FIVE
SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 Summary

This study developed an online chatbot for admission queries that allows prospective students, to gets quick and accurate responses to common enquiries about part time, fulltime studies or undergraduate or postgraduate studies and other related issues. This will reduce the workload of the admissions team and improving the overall experience for prospective students. This study identify the system and user requirements that were necessary to be met by the system which were identified alongside the software and hardware requirements of the system. The requirements were also specified using UML diagrams such as activity diagrams, sequence diagram and a use case diagram for the system modeling and user requirements respectively. The system was developed using HTML5, CSS3, jQuery, JavaScript, Ajax API as the frontend, Python, Natural Language Understanding (Rasa Open Source) for the backend and PostgreSQL for the database management on the system.

5.2 Conclusion

To conclude, this study has designed and implemented an online chatbot for admission enquiries in Mountain Top University that solves the challenges of response time and manual assistance and streamlining the admission process. The study was able to identify the system requirements, respective user and UML diagrams was used to specify these requirements so as suit to the expected functions of the proposed system.

5.3 Recommendation

The study recommends that future works in the area of chatbot should integrate various aspects of the university such as; cafeteria, chapel, accommodation, student handbook, which handles almost every information about the school and also integrate it to social media like whatsapp, telegram so it can be widely used.

References

Adamopoulou, E., & Lefteris, M. (2020). Introduction. *Chatbots: History, technology, and applications*, 1-18. From https://doi.org/10.1016/j.mlwa.2020.100006

Akande, O. F. (2020).

Ambler. (2000).

B.P. Prashant, M. A. (2017). Online chatting system for college Enquiry using knowledgeable database. *Online chatting system for college Enquiry using knowledgeable database*, 3-15. From https://scholar.google.com/scholar_lookup?title=Online%20chatting%20system%20for%20college%20Enquiry%20using%20knowledgeable%20database&publication_year=2017&author=B.P.%20Prashant&author=M.S.%20Anil&author=K.M.%20Dilip

Booch et al. (1999).

boost.ai. (2023, april 2). *KNOWLEDGE: What is conversational AI, anyway?*, 1. Retrieved april 2, 2023 from boost.ai: https://www.boost.ai/knowledge/what-is-conversational-ai#:~:text=Conversational%20AI%20is%20the%20synthetic,a%20virtual%20agent%20or%20chatbot.

Brush, K. (2021). chatbot. *Ultimate guide to customer service for businesses*, 1-3. From https://www.techtarget.com/searchcustomerexperience/definition/chatbot

Cahn, J. (2017). Architecture & Design. *CHATBOT: Architecture, Design, & Development* , 1-13.

Chakravarthy, S. (2020, june 19). *torwards.datascience.* From medium: https://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4

Court Bishop, C. W. (2022). What is a chatbot? *Chatbots vs. conversational AI: What's the difference?*, 3-6.

Danilo Cavalier et al. (2019). Data-Information-Concept Continuum From a Text Mining Perspective. From https://www.sciencedirect.com/referencework/9780128114322/encyclopedia-of-bioinformatics-and-computational-biology

Demchenko, M. (2021, juky 19). *Software Development Life Cycle: A Guide to Phases and Models.* From ncube: https://ncube.com/blog/software-development-life-cycle-guide

Diksha Khurana et al. (2022). Natural language processing: state of the art, current trends and challenges. *2*(3). From https://doi.org/10.1007/s11042-022-13428-4

Doering et al. (2008).

Doering et al. (2008).

Fainchtein, L. (2020, June 28). *Generative vs Retrieval Based Chatbots: A Quick Guide.* Retrieved June 28, 2020 from CloudBoost: https://blog.cloudboost.io/generative-vs-retrieval-based-chatbots-a-quick-guide-8d19edb1d645

Ferrell, O. C. (2020). TECHNOLOGY CHALLENGES AND OPPORTUNITIES FACING MARKETING EDUCATION. *TECHNOLOGY CHALLENGES AND OPPORTUNITIES FACING MARKETING EDUCATION*, 3-14. From https://doi.org/10.1080/10528008.2020.1718510

Gagan Gurung et al. (2020). THE SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) MODELS. *Software Development Life Cycle Models-A Comparative Study*, 1-3. From https://doi.org/10.32628/CSEIT206410

Guendalina Caldarini et al. (202). Introduction. *A Literature Survey of Recent Advances in Chatbots*, 1-10. From https://doi.org/10.3390/info13010041

Guendalina Caldarini et al. (2022). Chatbots Background. *A Literature Survey of Recent Advances in Chatbots*, 1-22. From https://doi.org/10.3390/info13010041

György Molnár; Zoltán Szüts. (2018). The Role of Chatbots in Formal Education. *0*(0). From https://ieeexplore.ieee.org/document/8524609/

H. Akkineni et al. (2022). Design and Development of Retrieval-Based Chatbot Using Sentence Similarity. *4*(8). From https://doi.org/10.1007/978-981-16-2919-8_43

Heller, B et al. (2005). Freudbot: An Investigation of Chatbot Technology in Distance Education. *2*(333). From https://www.learntechlib.org/primary/p/20691/

J. Thakkar, P. R. (2018). Erasmus–AI chatbot. *International Journal of Computational Science and Engineering*, 498-502. From https://doi.org/10.26438/ijcse/v6i10.498502

Leah. (2022). *deep-learning-and-generative-chatbots*. (Codecademy) From Codecademy web site: https://www.codecademy.com/learn/deep-learning-and-generative-chatbots/modules/generative-chatbots/cheatsheet

Lemke, G. (2018). THE SOFTWARE DEVELOPMENT LIFE CYCLE AND ITS APPLICATION. From https://commons.emich.edu/honors/589

M. Yamada, Y. G. (2016). A computer-supported collaborative learning design for quality interaction. In Y. G. M. Yamada, *A computer-supported collaborative learning design for quality interaction* (pp. 48-59). san franco: IEEE Ann. Hist. Comput.

Martin, M. (2023, February 25). *Software Development Life Cycle (SDLC) Phases & Models.* From GURU99: https://www.guru99.com/software-development-life-cycle-tutorial.html

Murray, A. (2022, june 25). *Software Development Life Cycle: Finding a Model That Works.* Retrieved june 25, 2022 from mend.io: https://www.mend.io/resources/blog/sdlc-software-development-life-cycle/

Padmanabhan. (2012).

Pooja and Nitasha. (2016). *SOFTWARE DEVELOPMENT LIFE CYCLES MODELS*.

Preston, M. (2022, jan 21). *System Development Life Cycle Guide.* From CloudDeffence.ai: https://www.clouddefense.ai/blog/system-development-life-cycle#:~:text=The%20new%20seven%20phases%20of,testing%2C%20implementation%2C%20and%20maintenance.

ROBERT & WALLACE, F. (2000, june 7). *cigionline.* Retrieved June 7, 2000 from Centre for Internation Governance Innovation: https://www.cigionline.org/articles/cyber-security-battlefield/

Rumbaugh et al. (1999).

Saxena, P. (2021, june 10). *Rule-Based Vs AI-Based Chatbots.* Retrieved june 10, 2021 from Medium: https://medium.com/predict/rule-based-vs-ai-based-chatbots-28613db3fe2c

Selig, J. (2022). What Is It? A Chatbot Definition. *The Power of Chatbots Explained*, 1-3.

Shylesh S. (2022). SOFTWARE DEVELOPMENT LIFE CYCLES MODELS. *A Study of Software Development Life Cycle Process Models*, 1-4. From https://ssrn.com/abstract=2988291

Singh and Dr. Sidhu. (2018).

Sotehp. (2023, march 14). *SDLC (Software Development Life Cycle) Phases, Process, Models.* From Software Testing Help: https://www.softwaretestinghelp.com/software-development-life-cycle-sdlc/

Stuart J. Russell and Peter Norvig. (2021). Artificial Intelligence A Modern Approach. *3*(4). From https://lccn.loc.gov/2019047498

Sumit, R. (2019). The Beloved Chatbots. In R. Sumit, *Building Chatbots with Python Using Natural Language Processing and Machine Learning* (pp. 1-2). Bangalore, Karnataka, India: Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013.

Thomas, H. (2020). Introduction. *Critical Literature Review on Chatbots in Education, 4*(6), 1-3. From www.ijtsrd.com

Tres Dean, A. G. (2022, october 31). *Hootsuite*. From Strategy: What Is Conversational AI: A 2023 Guide You'll Actually Use: https://blog.hootsuite.com/conversational-ai/

Wallace, R. S. (2009). The Anatomy of A.L.I.C.E. *4*(22). From https://dx.doi.org/10.1007/978-1-4020-6710-5_13

Wayesa, F. (2021). Design and Implementation of a Rule Based Afaan Oromoo Conversational Chatbots. *5*(8). From https://www.researchgate.net/publication/351945370

Wollny et al. (2021). Defination. *Are We There Yet? - A Systematic Literature Review on Chatbots in Education*, 1-18. From www.frontiersin.org

Xiaocong. (2015).