A black background with red text

Description automatically generated

FUE chatbot system for  
students’ inquiries

|  |  |
| --- | --- |
| Name | Id |
| Abdelrhman Montasser | 20181003 |
| Hassan Morsy | 20185004 |
| Ahmed Ibrahim | 20180678 |
| Ishak Magdy | 20181398 |
| Mina Emil | 20191353 |
| Abdelrhman Abdelhadi | 20192854 |

Supervised by Professor: Samah Zaki

**Table of content**

**Abstract**…………………………………………………………………………………..5

1. **Chapter 1: Introduction and motivation……….…………….….6**
   1. Background of the project…………………………………….……….……….7
   2. Problem………………………………….…………………………..……….…………7
   3. Objectives……………………………………………….…………….…………….…7
   4. Purpose………………………………………………….…………………..………….8
   5. Assumption…………………………………………….………………………..…….8
   6. Limitation…………………………………………………….……………………..….8
   7. Structure…………………………………………………………….………………….9
2. **Chapter 2: Literature review………………………………………….10**
   1. Project 1 (Online Chatting System for College Enquiry using Knowledgeable Database) ……………………………………………………………………………………………………………..11
   2. Project 2 (ARTIFICIAL INTELLIGENCE CHATBOT SYSTEM FOR STUDENT INQUIRIES IN THE LKC\_FES WEBSITE) …………………………………………………………………………………….12
   3. Project 3 (Development of Admission ChatBot for UST, Bannu Using Machine Learning Techniques) ……………….…………………………………………………………………………..13
   4. Project 4 (Design and Implementation of Student Chat Bot

using AIML and LSA) …………………………………………………………………………………………..…15

* 1. Project 5 (Implementation of a Chat Bot System using AI and NLP) ……………….…17
  2. Project 6 (CHAT-BOT SYSTEM PROJECT BASED

ON THE NATURAL LANGUAHGE)……………………………………………………………..…………….18

1. **Chapter 3: System Analysis…………………………………………...21**
   1. Data Gathering……………………………………………………….…………….22
   2. User Requirement………………………………………………….………….…23
      1. Functional Requirement……….……………….……………….……….23
      2. Non-Functional Requirement……….……………….…………………23
   3. System Requirement………..………………………………………………..…23
      1. Functional Requirement……….……………….……………….…….…23
      2. Non-Functional Requirement……….……………….…………………24
   4. System Model……………………………………………………….………………25
      1. Use case diagram……….……………….……………….………………..25
      2. Use case specification table……….……………….……………….….25
      3. Use case scenario……….……………….……………….……………….26
      4. Sequence Diagram……….……………….……………….………………27
      5. Class Diagram……….……………….……………….………………….….28
2. **Chapter 4: System** **Design……………………………………………..29**
   1. System architecture……………………………………………………………..30
   2. Prototype……………………………………………………………………………..31
   3. Model Accuracy……………………………………………………………………31
   4. App UI Design……………………………………………………………………….33
   5. Graphical User Interface……………………………………………………….34
3. **Chapter 5: System Implementation……………………………….37**
   1. No-SQL Database………………………………………………………………….38
   2. Database Implementation…………………………………………………….40
   3. Software Implementation…………………………………………………….42
   4. Machine learning Model……………………………………………………….43
   5. Process Flowchart…………………………………………………………………47
   6. Development tools……………………………………………………………….49
4. **Chapter 6: System Testing……………………………………………..52**
   1. Functional Testing…………………………………………………………………53
   2. User Interface (UI) Testing……………………………………………………52
   3. Integration Testing……………………………………………………………….53
   4. Performance Testing…………………………………………………………….53
   5. Security Testing…………………………………………………………………….53
   6. Usability Testing……………………………………………………………………54
   7. Compatibility Testing……………………………………………………………54
   8. Error Handling Testing………………………………………………………….54
   9. Regression Testing………………………………………………………………..54
5. **Chapter 7: Conclusion & Future Work……………………………55**
   1. Conclusion…………………………………………………………………56
   2. Future Work………………………………………………………………57
6. **References……………………………………………………………………58**

**List of figures**

[Figure 1 - Structure 9](#_Toc138802541)

[Figure 2 - Project 1 System architecture 11](#_Toc138802542)

[Figure 3 - Project 2 System architecture 12](#_Toc138802543)

[Figure 4 - Project 3 System architecture 13](#_Toc138802544)

[Figure 5 - Project 4 System architecture 15](#_Toc138802545)

[Figure 6 - Project 5 System architecture 17](#_Toc138802546)

[Figure 7 - Project 6 System architecture 18](#_Toc138802547)

[Figure 8 - Use case diagram 25](#_Toc138802548)

[Figure 9 - Sequence Diagram 27](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802549)

[Figure 10 - Class Diagram. 28](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802550)

[Figure 11 - System architecture. 30](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802551)

[Figure 12 – Prototype 31](#_Toc138802552)

[Figure 13 - Model Accuracy. 31](#_Toc138802553)

[Figure 14 - App UI Design 1 32](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802554)

[Figure 15 - App UI Design 2 33](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802555)

[Figure 16 - Graphical User Interface 1. 34](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802556)

[Figure 17 - Graphical User Interface 2. 35](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802557)

[Figure 18 - Graphical User Interface 3. 36](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802558)

[Figure 19 - No-SQL Database tree architure. 40](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802559)

[Figure 20 - Process Flowchart. 47](file:///D:\Computer%20science\2022%20-%202023%20_2\Project-2\Text%20Files\FUE%20chatbot%20system%20for%20student%20inquires.docx#_Toc138802560)

**List of Tables**

[Table 1 - Comparing between previously projects and our project. 20](#_Toc138802561)

[Table 2 - Ask question Use case table. 25](#_Toc138802562)

[Table 3 - Ask question Use case scenario. 26](#_Toc138802563)

[Table 4 - Ask for grade Use case scenario. 26](#_Toc138802564)

[Table 5 - Ask for Schedule Use case scenario. 26](#_Toc138802565)

**Abstract**

Chatbot is a computer program designed to simulate conversation with human users, especially over the Internet. Chatbots provide assistance and access to information quickly and efficiently. Chatbots fuel conversions and enhance social experiences. Chatting with bots also helps to avoid loneliness, gives a chance to talk without being judged. This project will be helpful for new students and also for any others who is asking general question about the university.

The communication of students with a university department is performed manually and it is a very time-consuming procedure. The opportunity to communicate with on a one-to-one basis is highly valued. The communication will require a member of academic staff to expend several hours to find suitable answers and contact each student. It would be useful to reduce his costs and time. The project aims to reduce the burden on the head of admissions, and potentially other users, by developing a convincing chatbot.

Chatbots are an intelligent system being developed using artificial intelligence (AI) and natural language processing (NLP) algorithms. The algorithm must be devised to search through the set of data and find a potential answer. The program then replies to the user and provides a relevant web link.

It has an effective user interface and answers the queries related to examination cell, admission, academics, users’ attendance, grades, GPA, and other miscellaneous activities.

**Chapter 1**

**Introduction and motivation**

* 1. **Background:**

The usual way of finding information on a university’s website is usually done by navigating through the website. It takes a huge effort just to find an answer to a user query as websites contain vast amounts of information.

However, in this era of the fourth industrial revolution, Artificial Intelligence is revolutionizing the structure of many organizations by providing an easy solution at much lower cost. One such technology that stands on a huge wave of progress is chatbots. Chatbots are conversation machine system that interacts with users using human communication language.

Chatbots are able to provide students with a quicker solution to resolve their queries. The use of chatbots offers a new opportunity for organizations at providing personalized user experience at scale and encourages more interactions between user and organization. In the context of education, there will be more interaction between students and the university. Such a benefit will enhance the university’s performance at a much lower cost.

* 1. **Problem:**

At the beginning we have a serious problem that we face many questions from students to advisors and the admission, and we don’t have enough people power to respond to those inquiries in real time.

* 1. **Objectives:**
* To minimize the time required to solve the queries.
* To simplify communication between user and machine.
* To provide 24 hours Availability.
* To reduce the cost for the organization.
* To answer all student questions about grades, information about university.
* To give the student easy access to the registration and model page.
  1. **Purpose:**

To establish real-time engagement with student

* 1. **Assumption:**
* Assume that the conversation between the user and the chatbot will be in English language.
* Assume that system database is not exist and will design and implement database and use it to simulate and get data from it.
* Assumes that students have access to a device with internet connectivity, such as a computer or smartphone, to interact with it effectively.
* Assumes that it needs to maintain user privacy and ensure data security, protecting any personal information shared by students during their interactions.
  1. **Limitation:**
* The chatbot will be just for students’ inquiries and guests.
* The Input query must contain proper key words.
* If query mis-spelled by user, then he should correct it, or it’ll not be able to provide proper response.
* The length of chatbot replies to one or two sentences.
  1. **Structure:**

**Graphical user interface, text, application

Description automatically generated**

Figure 1 - Structure

**Chapter 2**

**Literature review**

**System architecture:**

**2.1 Project 1: (Online Chatting System for College Enquiry using Knowledgeable Database)**

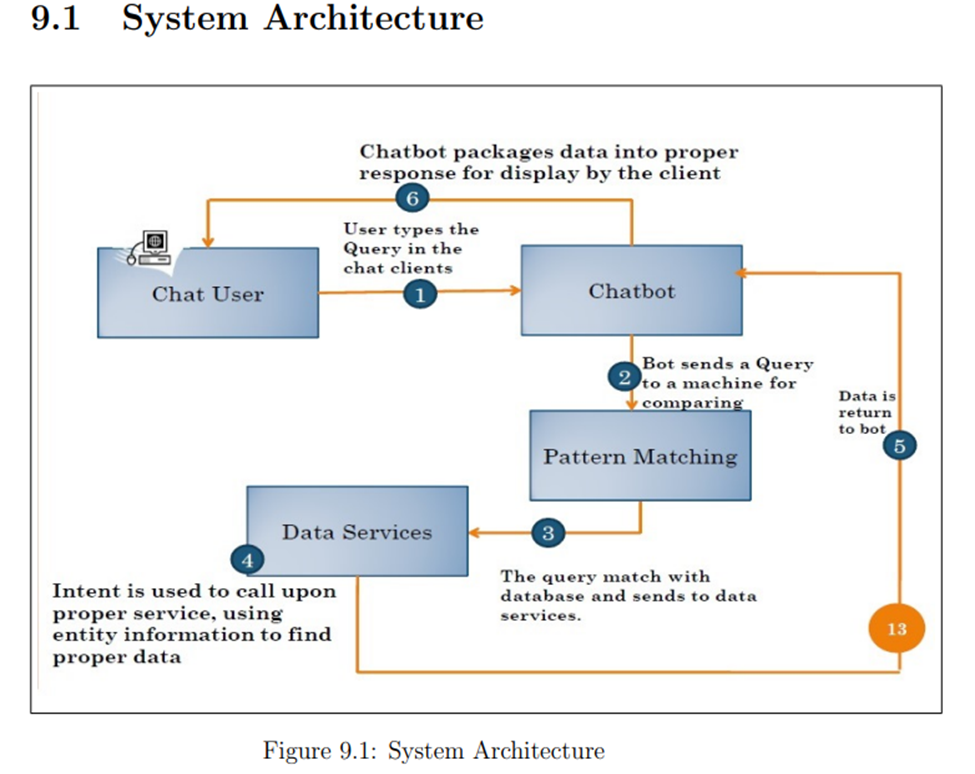
****

Figure 2 - Project 1 System architecture

**Modules**

* **Client-Server (chat user):**

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

* **Chatbot:**

A chatbot is a technology that allows users to have natural conversations to access content and services. Chatbots typically take the form of a chat client, leveraging natural language processing to conduct a conversation with the user. Chatbots control conversation flow based on the context of the users requests and respond with natural language phrases to provide direct answers, request additional information or recommend actions that can be taken.

* **Pattern matching:**

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

* **Data Services:**

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

**2.2 Project 2: (ARTIFICIAL INTELLIGENCE CHATBOT SYSTEM FOR STUDENT INQUIRIES IN THE LKC\_FES WEBSITE)**

**A picture containing text, screenshot, diagram, font

Description automatically generated**

Figure 3 - Project 2 System architecture

**System Architecture Design:** The system architecture design used in this project is the three-tier architecture.

Three-tier architecture divides the entire application into three different layers mainly Presentation Tier, Application Tier and Data Tier (JReport, 2020). The overview of the three-tier architecture is shown in Figure 5.21 below.

1. **Presentation Tier** – Front-end of the application that consists of

graphical user interface.

1. **Application Tier** – Contains business logic and is the core

functionality of the system.

1. **Data Tier** – Data tier consist of database or data storage.

**2.3 Project 3: (Development of Admission ChatBot for UST, Bannu Using Machine Learning Techniques)**

**A diagram of a chatbot

Description automatically generated with medium confidence**

Figure 4 - Project 3 System architecture

The basic algorithm that will be implemented for working of this proposed system is as follows:

**Step 1:** Start.

**Step 2:** Get the input query from the user.

**Step 3:** The query is pre-processed. E.g. suppose there is this query “what are the project domains for CSE fourth year major projects.” So, we are going to remove these stop words like “are”, “the” using pre-processing technique.

**Step 4:** Fetch the remaining keywords from the query.

**Step 5:** Match the fetched keywords with the keywords in Knowledge base and provide an appropriate response.

**Step 6:** Further the Database module is used to call proper services using entity information to find proper data.

**Step 7:** The keywords will be matched with the help of keyword matching algorithm.

**Step 8:** It returns the query response to the bot.

**Step 9:** Chat-bot packages the data into proper response for display by the client.

**Step 10:** Exit

**2.4 Project 4: (Design and Implementation of Student Chat Bot using AIML and LSA)**

**A picture containing text, diagram, screenshot, font

Description automatically generated**

Figure 5 - Project 4 System architecture

In Middleware encodes the layout into the JSON design and sends the answer to the client. In the wake of accepting the reaction web disentangle the JSON and gives the reaction to the client. On the off chance that the example isn't coordinated utilizing both AIML and LSA contents it pings to the administrator’s telephone as a message. The figure1 demonstrates the diagram of our proposed model.

1. **Pattern matching:**

It is the demonstration of checking a given grouping of tokens for the nearness of the constituents of some examples.

In qualification to design acknowledgment, the match commonly must be definite: "it is possible that it will or will not be a match." The normally for the most part have the state of either arrangements or tree structures.

Employments of example coordinating grasp yielding the areas (assuming any) of an example inside a token succession, to yield some piece of the coordinated example, and to substitute the coordinating example with some another token arrangement.

1. **AIML:**

Artificial Intelligence Mark-up Language (AIML) is derived from Extensible Mark-up Language (XML) which is used to build up a conversational agent (chatbot) artificially. In this we use “program-O‟ which is an AIML interpreter for generation of the responses of user’s input. We have used this method for developing an android application chatbot. Which will interact with user using text and voice responses. Among s Natural Language Processing begins when the client submits an inquiry to Visit bot. This inquiry is gotten by the AIML mediator that answers utilizing a learning base (Jindal et al, 2004) and there is a detailed task at Figure 4. This information base was executed in AIML language comprising of labels on class structure. In every class we have the inquiry and the appropriate response, utilizing individually, the example and layout labels. To improve the collaboration, making it as reasonable as could be expected under the circumstances, we can utilize more than one response to one inquiry when we have a similar inquiry, done in various ways. In the event that the inquiry isn't in the learning base chatbot, a reaction is produced haphazardly by the framework, keeping the client dependably find a similar solution. We can all the more likely comprehend the utilization of these labels by above figure.

1. **LSA:**

Latent Semantic analysis (LSA) is a thought and approach for separating and speaking to the relevant utilization which methods for words by method for factual calculations actualized to an enormous corpus of literary substance. The fundamental idea is that the totality of certainties pretty much the majority of the word settings wherein a given word does and does not appear bears a rigid of shared constraints that in extensive part decides the likeness of importance of expressions and set of words to each extraordinary. The ampleness of LSA's reflected picture of human information has been snared in a repercussion of techniques. for instance, its rankings cover the ones of people on in vogue vocabulary and issue matter appraisals, it copies human expression arranging and classification decisions, reenacts word-expression and section word lexical preparing data and as expressed in organization Papers, effectively gauges entry soundness, learnability of entries by means of individual understudies and the fine and amount of mastery contained in an article.

**2.5 Project 5: (Implementation of a Chat Bot System using AI and NLP)**

**A diagram of a login

Description automatically generated with medium confidence**

Figure 6 - Project 5 System architecture

**2.6 Project 6: (CHAT-BOT SYSTEM PROJECT BASEDON THE NATURAL LANGUAHGE)**

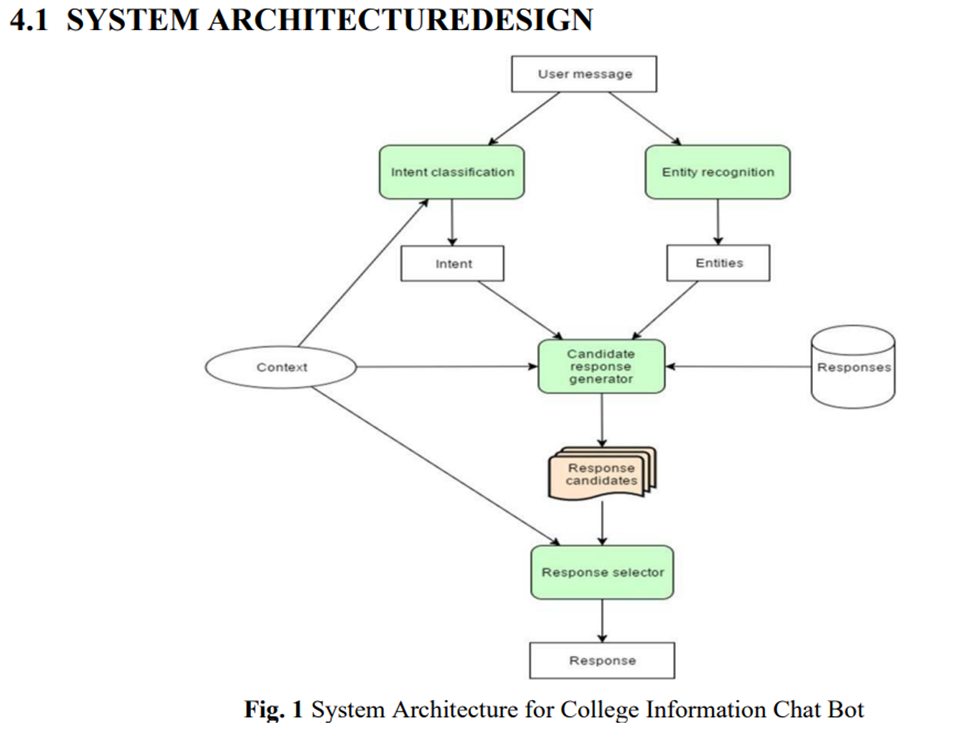
****

Figure 7 - Project 6 System architecture

There are 7-Modules according to the system architecture for college information chat bot.

**system which are explained below as follows:**

* **Add User:**

This module is responsible for adding user to the system. Each user is assigned a unique ID and Password to get into the system.

* **Database Server:**

It contains record of all the users login credentials, like College Data, User Queries etc...

* **Manage Course:**

In this module the admin performs the various tasks to fetch into the database various college information like Placement Cell Information, Department Information, Class Schedule etc.

* **View Chat:**

In this, User asks the query, and the bot replies to the user query accordingly. So, User can view the chat which the user had with the bot system.

* **Upload:**

In this module admin uploads the general notices like Schedule, Exam Dates, Fee Structure, Events and Fests, Seminar-Conference Notice etc.

* **Forum:**

In this module, If the user founds that answer does not satisfy regarding the user query, then he can mark that answer as invalid. Later, It can be viewed by the admin, And can delete or add or modify specific answer.

* **Exit:**

This is the phase where user after finishing his work signs out from the system.

Table 1 - Comparing between previously projects and our project.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Features | Project 1 | Project 2 | Project 3 | Project 4 | Project 5 | Project 6 | Our Project |
| Text-based inquiries |  |  |  |  |  |  |  |
| Voice-based inquiries |  |  |  |  |  |  |  |
| Text output |  |  |  |  |  |  |  |
| Voice output |  |  |  |  |  |  |  |
| API |  |  |  |  |  |  |  |
| Knowledge base |  |  |  |  |  |  |  |
| Database |  |  |  |  |  |  |  |
| Log file |  |  |  |  |  |  |  |
| Admin notification |  |  |  |  |  |  |  |

**Chapter 3**

**System Analysis**

**3.1 Data Gathering:**

Here are many ways to collect data, but two main sources have been chosen for data collection, and they are as follows:

**Primary**

As the name implies, this is original, first-hand data collected by the data researchers. This process is the initial information gathering step, performed before anyone carries out any further or related research. Primary data results are highly accurate provided the researcher collects the information. However, there’s a downside, as first-hand research is potentially time-consuming and expensive.

* Specific Data Collection Techniques:

Interviews.

Delphi Technique.

**Secondary**

Secondary data is second-hand data collected by other parties and already having undergone statistical analysis. This data is either information that the researcher has tasked other people to collect or information the researcher has looked up. Simply put, it’s second-hand information. Although it’s easier and cheaper to obtain than primary information, secondary information raises concerns regarding accuracy and authenticity. Quantitative data makes up a majority of secondary data.

* Specific Data Collection Techniques:

The internet.

Previous student reports.

**3.2 User Requirement**

**3.2.1 Functional Requirement:**

1. User shall ask a question to retrieve an answer about specific information. Ex**:**

* Ask about Results.
* Ask about Schedule.
* Ask about general information about university.

1. User shall ask with correct sentence.
2. Bot shall check knowledge base for similar question to retrieve best response.
3. Bot shall respond with the correct answer.

**3.2.2 Non-Functional Requirement:**

1. Bot must respond in 3 second of user question.
2. The chatbot interface should be intuitive and user-friendly, requiring minimal effort for users to interact with and understand.

**3.3 System Requirement:**

**3.3.1 Functional Requirement:**

* 1. Provide basic information:
* The system must answer user question about faculties.
* The system must answer user question about all general information about university.
  1. Provide Student information:
* The system must answer user question about courses.
* The system must answer user question about schedules.
* The system must provide to user the results of exams.

**3.3.2 Non-Functional Requirement:**

* 1. Performance Requirements:
* The system should be compatible with all modern browsers and all platforms.
* The system should response the messages to the users within 3 seconds.
* The system should be reliable.
  1. Usability:
* The system should have user-friendly interface.
  1. Availability:
* The system must be available 24/7.

**3.4 System Model**

**3.4.1 Use case diagram:**

Graphical user interface, application

Description automatically generated

Figure 8 - Use case diagram

**3.4.2 Use case specification table:**

Table 2 - Ask question Use case table.

|  |  |
| --- | --- |
| UC ID | 1 |
| UC Name | Ask question |
| Actors | Student |
| Input | Text query |
| Description | This use case designates the flow of event of a student asking the  chatbot system question |
| Flow of event | 1- Student submit enquiry via chat prompts to the chatbot system.  2- If chatbot doesn’t understand input, “Ask for simplify input again”.  3- If chatbot understand the input, “Respond with correct answer”.  4- The System displays the most suitable response. |

**3.4.3 Use case scenario:**

Table 3 - Ask question Use case scenario.

|  |  |
| --- | --- |
| SC ID | 1 |
| Target UC | Ask question |
| SC name | Ask for final results |
| Scenario event | After the student “Ahmed” input the query asking for final results in exams, the Bot will ask him to enter his id like “20180678” and then the bot send to him a link to the results page of student portal like “https://services.fue.edu.eg/fuestudentportal/StudentResult.aspx”. |

Table 4 - Ask for grade Use case scenario.

|  |  |
| --- | --- |
| SC ID | 2 |
| Target UC | Ask for grade |
| SC name | Ask for final grade |
| Scenario event | After the student “Ahmed” input the query asking for final grade in subject , the Bot will ask him to enter his id like “20180678” and then the bot send to him a link to the results page of student portal like “https://services.fue.edu.eg/fuestudentportal/StudentResult.aspx”. |

Table 5 - Ask for Schedule Use case scenario.

|  |  |
| --- | --- |
| SC ID | 3 |
| Target UC | Ask for Schedule |
| SC name | Ask for Schedule |
| Scenario event | After the student “Ahmed” input the query asking for his Schedule, the Bot will ask him to enter his id like “20180678” and then the bot send to him a link to the Schedule page of student portal like “https://services.fue.edu.eg/fuestudentportal/StudentAcademicData.aspx”. |

**3.4.4 Sequence Diagram:**

Diagram

Description automatically generated

Figure 9 - Sequence Diagram

**3.4.5 Class Diagram:**

Graphical user interface, application, Teams

Description automatically generated

Figure 10 - Class Diagram.

**Chapter 4**

**System Design**

**4.1 System architecture:**

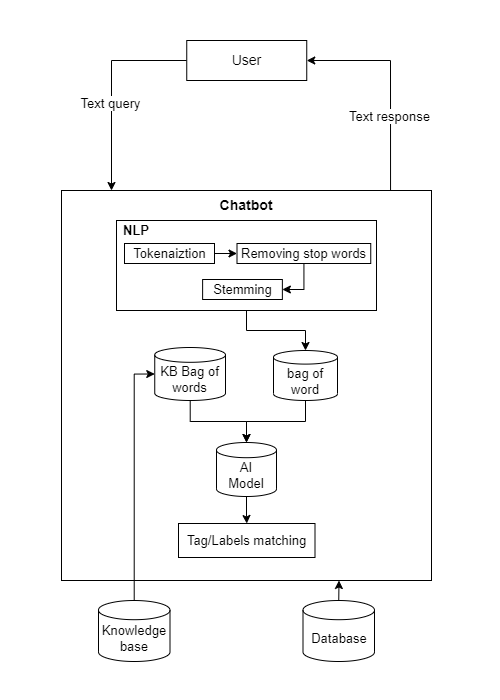


Figure 11 - System architecture.

**4.2 Prototype:**

A screenshot of a computer

Description automatically generated with medium confidence

Figure 12 – Prototype

**4.4 Model Accuracy:**

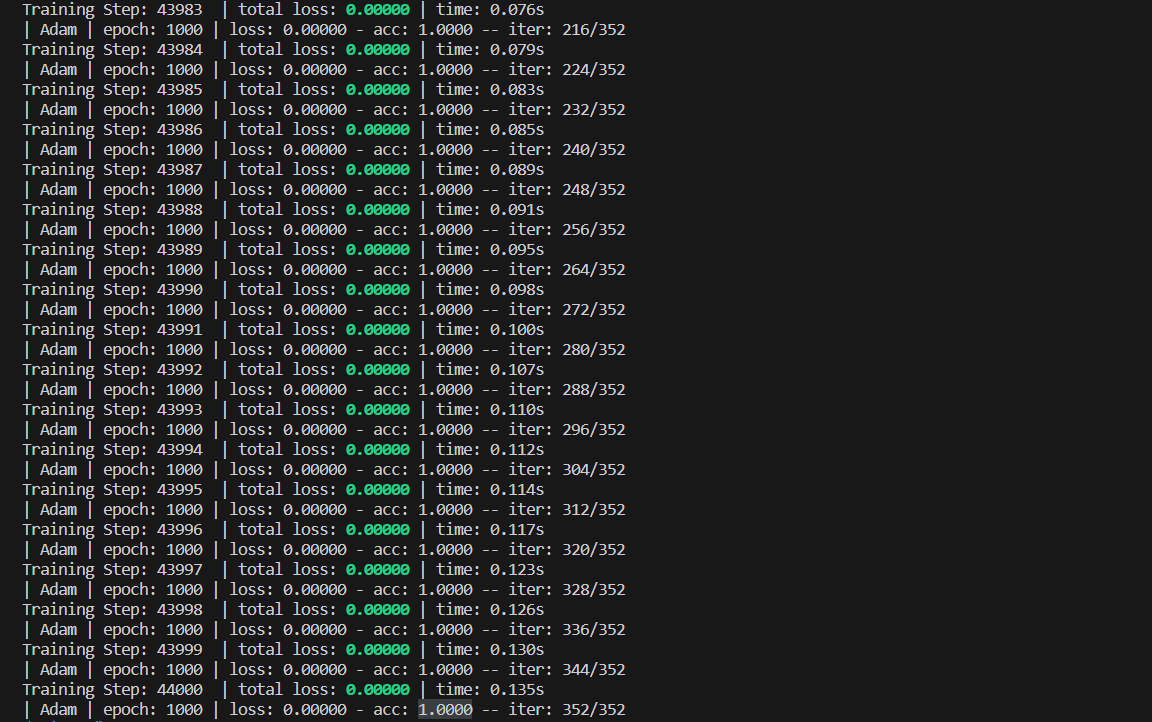


Figure 13 - Model Accuracy.

**Confusion Matrix**:

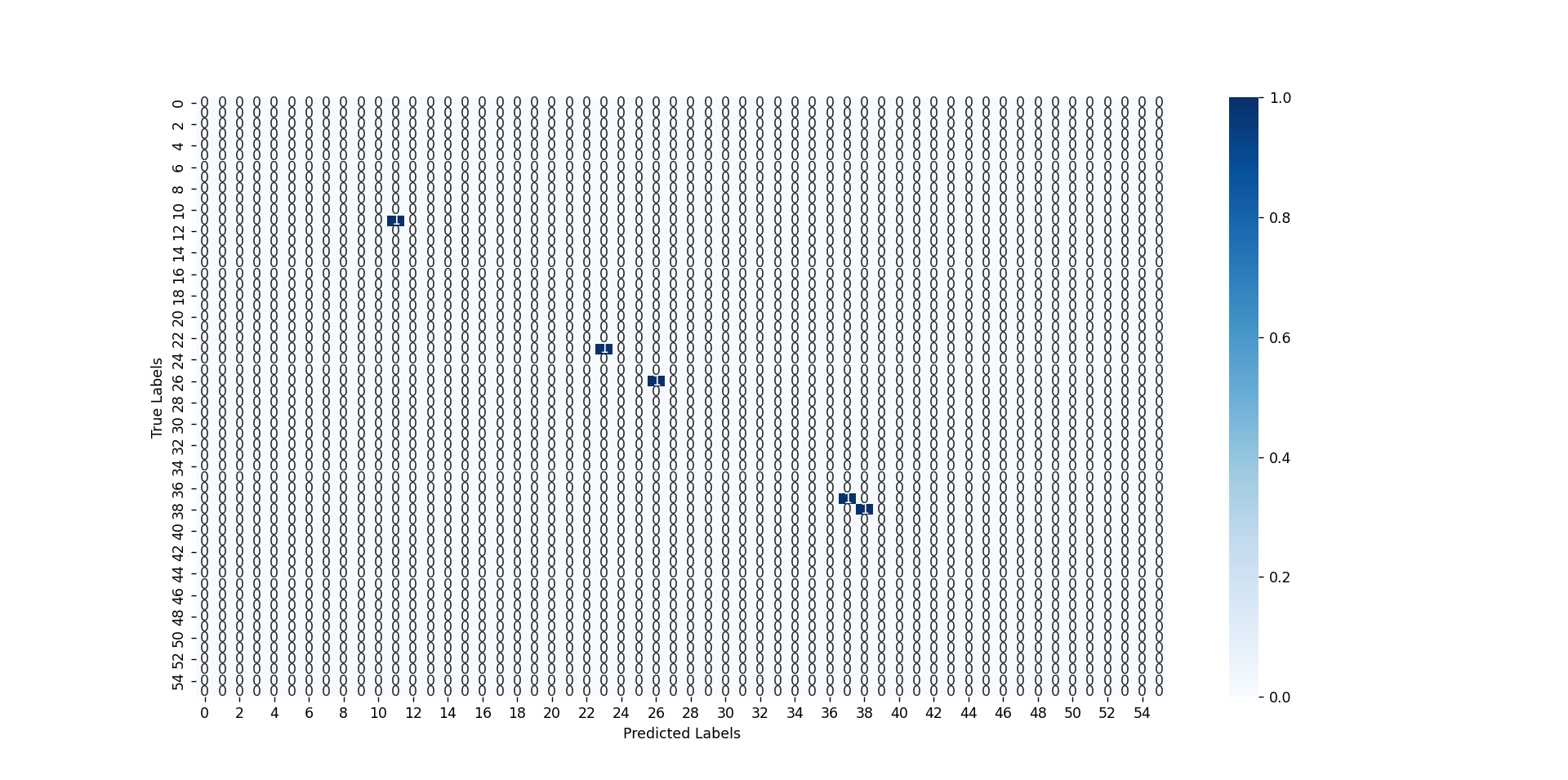


Figure 15 - Confusion Matrix

**4.3 App UI Design:**

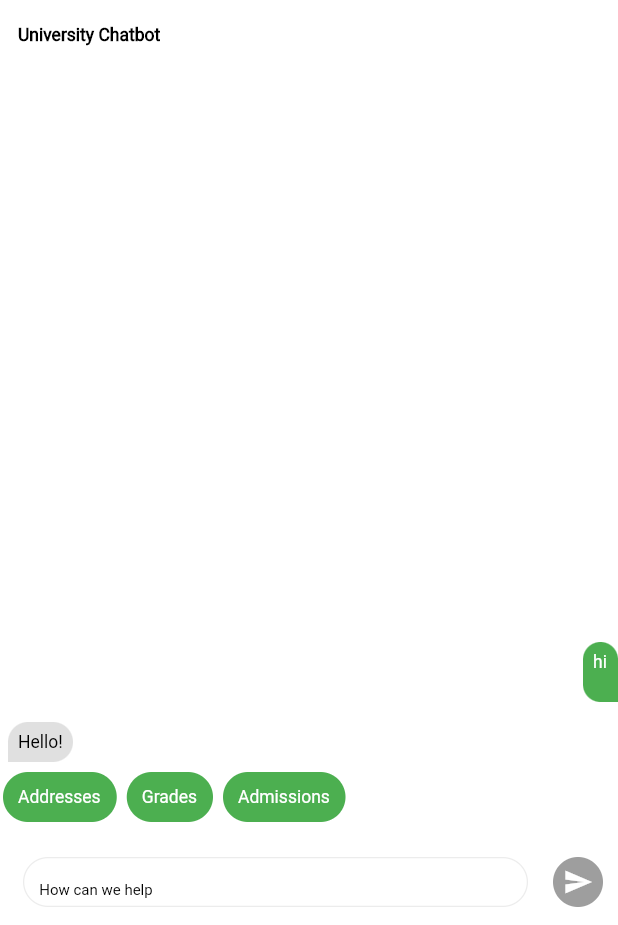


Figure 14 - App UI Design 1

**4.5 Graphical User Interface:**

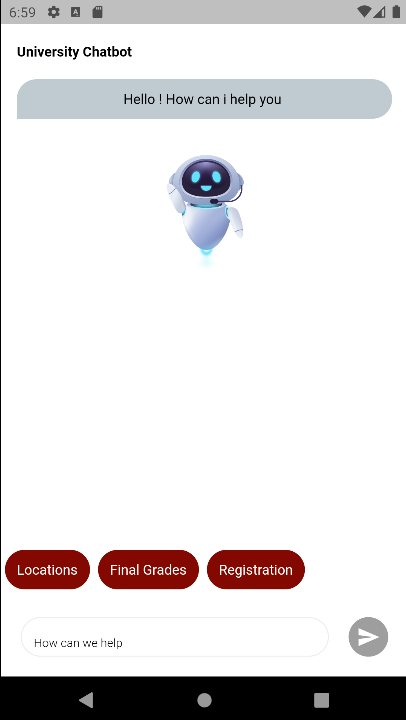


Figure 16 - Graphical User Interface 1.



Figure 17 - Graphical User Interface 2.

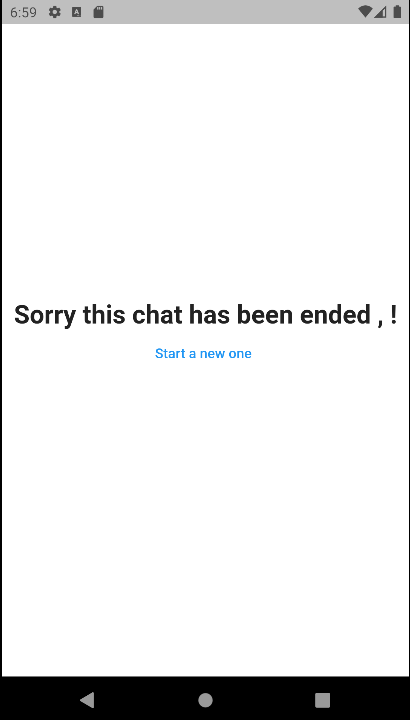


Figure 18 - Graphical User Interface 3.

**Chapter 5**

**System Implementation**

**5.1 No-SQL Database:**

A NoSQL (not only SQL) database is a type of database management system that provides a non-relational approach to storing and retrieving data. Unlike traditional relational databases, which organize data into tables with predefined schemas, NoSQL databases offer a flexible and scalable way to handle large amounts of unstructured or semi-structured data.

**There are different types of NoSQL databases, each with its own data model:**

1. Document databases: Store and retrieve data in flexible, JSON-like documents. Examples include MongoDB and Firebase.
2. Key-value stores: Use a simple key-value data model, where data is stored and retrieved based on unique keys. Examples include Redis and Riak.
3. Column-family databases: Organize data into column families or column groups rather than tables, allowing for efficient querying of specific columns. Examples include Apache Cassandra and HBase.
4. Graph databases: Optimize for storing and traversing relationships between entities, making them suitable for applications that heavily rely on complex data relationships. Examples include Neo4j and Amazon Neptune.

**Why we choose Firebase database:**

1. Real-time synchronization: Firebase offers real-time data synchronization, allowing changes made in the database to be instantly propagated to all connected clients. This feature is crucial for chat applications where messages need to be delivered and displayed in real-time.
2. Scalability: Firebase is a serverless platform that automatically scales to handle high traffic loads. It can accommodate a large number of concurrent users and seamlessly scale as the application grows without requiring manual configuration or infrastructure management.
3. Offline support: Firebase provides offline support, allowing chat applications to continue functioning even when the device is offline. Messages can be stored locally and synchronized with the server once the connection is restored, ensuring a seamless user experience.
4. Easy integration: Firebase offers SDKs and APIs for various platforms and programming languages, making it easy to integrate with different client applications. It provides libraries for iOS, Android, web, and other platforms, simplifying the development process and allowing developers to focus on building the chat application logic.

A picture containing diagram, sketch, text, plan

Description automatically generated**5.2 Database Implementation:**

Figure 19 - No-SQL Database tree architure.

**5.3 Software Implementation:**

The software implementation of a student chatbot would involve developing a system that can interact with students and provide relevant information or support in an automated manner.

Here are some key aspects of the software implementation for a student chatbot:

**Natural Language Processing (NLP):** The chatbot should be able to understand and interpret natural language input from students. NLP techniques such as tokenization, part-of-speech tagging, and named entity recognition may be used to extract relevant information from user queries.

**Intent Recognition:** The chatbot needs to recognize the intent behind a student's query to provide an appropriate response. This can be achieved using techniques like intent classification or machine learning algorithms trained on labeled data.

**Dialog Management:** The chatbot should be able to maintain a conversation flow with the student, handling context and managing multiple turns. Dialog management techniques, such as maintaining conversation state or using dialogue state trackers, can be employed.

**Knowledge Base:** The chatbot should have access to a knowledge base or database containing relevant information for students. This could include FAQs, course information, campus resources, academic policies, and more. The knowledge base can be built using a structured database or utilizing a knowledge graph.

**Integration with APIs and Services:** The chatbot may need to integrate with external APIs or services to fetch real-time data or perform specific actions. For example, it could integrate with a student information system to retrieve course schedules or grades or connect to a library system to check book availability.

**User Interface:** The chatbot can have a user interface where students interact with it, such as a web-based chat interface or integration with messaging platforms like Slack or Facebook Messenger. The interface should be intuitive and user-friendly.

**Testing and Continuous Improvement:** Testing is crucial to ensure the chatbot functions correctly and provides accurate responses. It may involve unit testing, integration testing, and user acceptance testing. Additionally, feedback from users should be collected and analyzed to continuously improve the chatbot's performance and accuracy.

**5.4 Machine learning Model:**

There are several machine learning models that can be used to build chatbot systems. Here are some of the commonly used models:

* Rule-based models
* Retrieval-based models
* Generative models
* Sequence-to-Sequence models
* Transformer models

Since the project is not conversational but more on retrieving information needed by the user or the student, so the one we used for our project is the “Retrieval-based model”.

**5.4.1 Retrieval-based models:**

These models use a predefined database or Knowledge base of responses and retrieve the most suitable response based on the input. They often rely on techniques such as keyword matching, vector similarity, or sequence matching to find the best-matching response. Examples of retrieval-based models include “Bag-of-Words (BoW)”, “Term Frequency-Inverse Document Frequency (TF-IDF)”, and “word embeddings” like Word2Vec or GloVe.

We used the "Bag of Words (BoW)" as defined in the system architecture.

**5.4.2 Bag of Words (BoW):**

The Bag of Words (BoW) model is a basic technique in natural language processing (NLP) that represents text as a collection or "bag" of individual words, disregarding grammar and word order but keeping track of their frequency of occurrence. It is a simple and commonly used method for text representation and feature extraction.

In the BoW model, a text document is transformed into a numerical vector representation. The process involves the following steps:

1. Tokenization: The text document is split into individual words or tokens. Punctuation and other non-alphanumeric characters are usually removed, and the text may be converted to lowercase.
2. Vocabulary creation: All unique words in the corpus (collection of documents) are collected to form a vocabulary or dictionary. Each word is assigned a unique index or ID.
3. Vector representation: For each document, a vector is created with dimensions equal to the size of the vocabulary. The value at each dimension represents the frequency or occurrence of the corresponding word in the document. In other words, each dimension of the vector represents a specific word, and the value in that dimension indicates how many times that word appears in the document.
4. Encoding: The vector representation can be encoded in different ways. The simplest approach is the binary encoding, where each dimension is represented by a binary value indicating whether the word is present or absent in the document. Another common encoding is using the raw frequency of words, where the actual count of each word is used as the value in the vector. Additionally, techniques like term frequency-inverse document frequency (TF-IDF) can be applied to weigh the importance of words based on their occurrence across the entire corpus.

The resulting BoW representation is a sparse vector, meaning most of its dimensions are zero because only a small subset of the vocabulary appears in any given document. This representation can then be used as input to various machine learning models, such as classifiers or clustering algorithms, for text classification, sentiment analysis, topic modeling, and other NLP tasks.

One limitation of the BoW model is that it ignores the contextual relationships between words and their ordering in a sentence, which can limit its ability to capture the semantic meaning of text. However, it remains a widely used and effective technique for many text analysis applications.

**5.4.3 DNN Model (Text Classifier):**

A DNN is a collection of neurons organized in a sequence of multiple layers, where neurons receive as input the neuron activations from the previous layer, and perform a simple mathematical computation (e.g., a weighted sum of the input followed by a nonlinear activation). The neurons of the network jointly implement a complex nonlinear mapping from the input to the output. This mapping between the input and output is learned from the data by adapting the weights of each neuron using a technique called error backpropagation. In simple words, DNN is a feedforward multilayer neural network architecture, or it is an artificial neural network (ANN) with multiple layers between the input and the output layers. We will train it with a set of labeled data in order to perform classification.

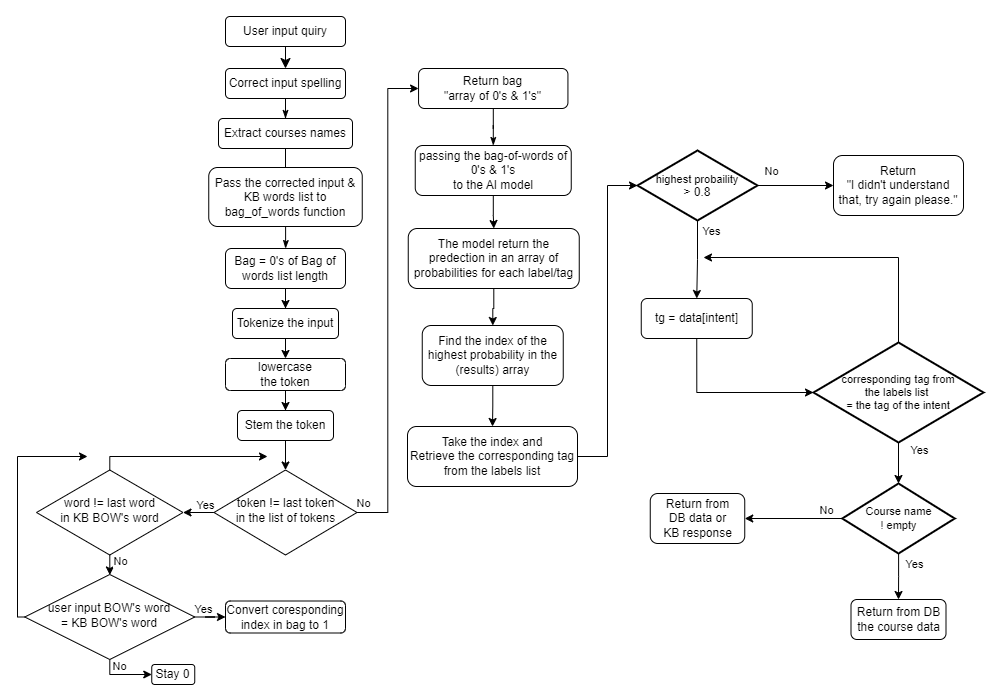
**5.5 Process Flowchart:**

Figure 20 - Process Flowchart.

Run sequence:

1. Import the necessary libraries:
2. Load the service account credentials from the "serviceAccountKey.json" file.
3. Initialize the Firebase Admin SDK with the loaded credentials.
4. Create a Firestore client to interact with the Firestore database.
5. Initialize the Lancaster stemmer for word stemming.
6. Initialize the Speller object for spell checking (assuming the correct language is specified).
7. Load the intents from the "intents.json" file.
8. Load the processed data (words, labels, training, output) from the "data.pickle" file.
9. Define the neural network architecture using (tflearn).
10. Create a (tflearn) DNN model.
11. Load the trained model weights from the "model.tflearn" file.

User input “Bag of word”:

1. Create a “bag\_of\_words” function that takes in a sentence “s” and a list of words as input.
2. It initializes an empty list called “bag” with the same length as the “words” list.
3. Tokenize the sentence “s” into words using (nltk.word\_tokenize).
4. Stemming and lowercasing each word from using (stemmer.stem(word.lower())).
5. Iterate over each stemmed word in “s\_words”.
6. For each stemmed word, iterate over the words list using enumerate.
7. If a stemmed word matches a word in the “words” list, set the corresponding index in “bag” to 1.
8. Finally, return the “bag” as a NumPy array.

Process the User input and make the prediction:

1. Takes a message as input.
2. Assigns the message to “inp”.
3. Applies spell correction to the “inp” using the spell function.
4. Extracts course names from the corrected input using a function called “extract\_course\_names”.
5. Makes a prediction using the loaded neural network model “model.predict” by passing the “bag-of-words” representation of the (“corrected input” and the “words” list) to the “bag\_of\_words” function.
6. The prediction results in an array of probabilities for each label/tag.
7. Find the index of the highest probability in the “results” array using (numpy.argmax).
8. Retrieve the corresponding tag from the labels list using the index obtained in the previous step and assign it to the variable tag.

**5.6 Development tools:**

1. **Tools and IDEs:**

* Visual Studio Code IDE:
  + Visual studio code is created by Microsoft as source-code editor software. Visual Studio codes have much functionality which includes debugging, syntax highlighting, snippets, code refactoring and many more.
* Android Studio:
  + Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems.

1. **Languages:**

* Python:
  + is a high-level, interpreter and general-purpose programming language. The versatility is extremely beneficial, allowing beginners to pick up with its consistent syntax and language that mirrors humans. Furthermore, the prebuilt libraries made available in python are an added advantage to building AI projects.
* Dart:
  + Dart is a programming language designed by Lars Bak and Kasper Lund and developed by Google. The programming language can be used to develop web and mobile apps as well as server and desktop applications. It is an object-oriented, class-based, garbage-collected language with C-style syntax.
* JSON:
  + is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays. It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers.
  + Also, we are using JSON to build our knowledge base that have tag, patterns and response, and that to train our machine learning model.

1. **Software Frameworks:**

* Flask:
  + is a web service framework written in Python. Flask is very minimal coming with an inbuilt light-weighted web server that requires minimal configuration. Likewise, it is easy to be controlled from Python code making it versatile and popular to developers.
  + Also, we are using flask to build our API that we can use to get and post data from the back-end (Python) to the front-end (Flutter)
* Flutter:
  + is an open-source UI software development kit created by Google. It is used to develop cross-platform applications for Android, iOS, Linux, macOS, Windows, Google Fuchsia, and the web from a single codebase. First described in 2015, Flutter was released in May 2017.
  + Mainly we use flutter to design our user interface also flutter provide a flexible and expressive way to design and style our user interface.
* Firebase:
  + Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.

1. **Testing Software:**

* Postman:
  + Postman is an API Platform for developers to design, build, test and iterate their APIs. As of February 2023, Postman reports having more than 25 million registered users and 75,000 open APIs, which it says constitutes the world's largest public API hub.
  + Mainly we use postman to test and debugging our flask API that allow us to send HTTP requests to APIs and receive responses.

**Chapter 6**

**System Testing**

**6.1 Aspects of chatbot system testing:**

System testing for a chatbot typically involves several aspects to ensure its functionality, performance, and user experience. Here are some common types of system testing conducted for a chatbot:

1. **Functional Testing:** This verifies that the chatbot performs its intended functions correctly. It includes testing features such as user authentication, conversation flow, natural language processing (NLP), response generation, error handling, and integration with external systems or APIs.
2. **User Interface (UI) Testing:** This focuses on testing the chatbot's user interface to ensure it is intuitive, visually appealing, and responsive. It involves checking the layout, formatting, navigation, and consistency across different devices and platforms.
3. **Integration Testing:** This validates the chatbot's integration with other systems, databases, or APIs it relies on for data or functionality. It ensures that data is transferred correctly, requests are processed, and responses are received as expected.
4. **Performance Testing:** This evaluates the chatbot's performance under different loads and stress conditions. It includes testing its response time, scalability, concurrent user handling, and resource utilization to identify and address performance bottlenecks.
5. **Security Testing:** This ensures the chatbot's data and interactions are secure. It involves testing for vulnerabilities, authentication and authorization mechanisms, data encryption, protection against malicious input or attacks, and compliance with privacy regulations.
6. **Usability Testing:** This focuses on the chatbot's user experience and ease of use. It involves gathering feedback from real users to evaluate its intuitiveness, clarity of instructions, error recovery, personalization, and overall satisfaction.
7. **Compatibility Testing:** This verifies that the chatbot works correctly across different platforms, browsers, and devices. It ensures compatibility with various operating systems, screen sizes, and browser versions to provide a consistent experience to users.
8. **Error Handling Testing:** This examines how the chatbot handles and recovers from errors or unexpected user inputs. It includes testing for error messages, fallback responses, graceful degradation, and error logging for troubleshooting purposes.
9. **Regression Testing:** This ensures that existing functionalities remain intact after new updates or changes to the chatbot. It involves retesting previously tested features to detect any unintended side effects or regressions.

**Chapter 7**

**Conclusion & Future Work**

**7.1 Conclusion:**

In conclusion, the chatbot's ability to answer questions, provide explanations, and offer support promotes self-directed learning.

Also, chatbot is a valuable tool designed to assist students in their educational journey. It serves as a digital assistant, providing support, resources, and guidance across various subjects and learning needs.

The chatbot assumes that students have access to technology and possess a basic level of language proficiency. It creates a non-judgmental environment where students can freely ask questions, seek clarification, and receive personalized assistance.

**7.2 Future Work:**

* Adding a GIS service to the chatbot to answer students for any quires about location in university.
* Adding a second language to chatbot like Arabic or French.
* Implement a recommendation system that analyzes user preferences, learning patterns, and performance to provide personalized recommendations for study courses to take, materials, practice exercises, educational resources, and additional subjects to explore.
* Enable students to form virtual study groups or join collaborative learning spaces within the chatbot. This feature would allow students to interact, share notes, discuss concepts, and work on projects together, fostering a sense of community and peer learning.

**References**

1. “Online Chatting System for College Enquiry using Knowledgeable Database”, Savitribai Phule Pune University, 2017

[1a269a70-6cb0-4c5d-bd5b-4549a22d2d10.pdf (youth4work.com)](https://www.static-contents.youth4work.com/y4w/Documents/Portfolio/1a269a70-6cb0-4c5d-bd5b-4549a22d2d10.pdf)

1. “ARTIFICIAL INTELLIGENCE CHATBOT SYSTEM FOR STUDENT INQUIRIES IN THE LKC\_FES WEBSITE”, University Tunku Abdul Rahman, April 2020

[FES Final Year Project Template (utar.edu.my)](http://eprints.utar.edu.my/4207/1/1604184_SIEW_YONG_JUN.pdf)

1. “Development of Admission ChatBot for UST, Bannu Using Machine Learning Techniques”, UST Bannu, January 2022

[(PDF) Development of Admission ChatBot for UST, Bannu Using Machine Learning Techniques (researchgate.net)](https://www.researchgate.net/publication/358145954_Development_of_Admission_ChatBot_for_UST_Bannu_Using_Machine_Learning_Techniques)

1. “Design and Implementation of Student Chat Bot using AIML and LSA”, International Journal of Innovative Technology and Exploring Engineering (IJITEE), April 2019

[(PDF) Design and Implementation of Student Chat Bot using AIML and LSA (researchgate.net)](https://www.researchgate.net/publication/341592140_Design_and_Implementation_of_Student_Chat_Bot_using_AIML_and_LSA)

1. “Implementation of a Chatbot System using AI and NLP”, International Journal of Innovative Research in Computer Science & Technology (IJIRCST), May 2018

[(PDF) Implementation of a Chat Bot System using AI and NLP (researchgate.net)](https://www.researchgate.net/publication/327456963_Implementation_of_a_Chat_Bot_System_using_AI_and_NLP)

1. “CHAT-BOT SYSTEM PROJECT BASED ON THE NATURAL LANGUAHGE”, Galgotias University, MAY- 2020

[17SCSE104110\_Vishvaketan Gaur\_Report - Vishvaketan Gaur.pdf](http://103.47.12.35/bitstream/handle/1/1422/17SCSE104110_Vishvaketan%20Gaur_Report%20-%20Vishvaketan%20Gaur.pdf?sequence=1&isAllowed=y)

1. “Text Classification: BERT vs DNN”, Dharm Raj Prajapat in Engineering@Zemoso, Feb 14, 2020

<https://eng.zemosolabs.com/text-classification-bert-vs-dnn-b226497c9de7>