Software Engineering
Hackathon

SRS Documentation
For
AI based Chatbot for FAQs

Prepared by:
(29) Aditi Uphade
(68) Divyansh Sawant
(126) Om Ghogare
(135) Chaitanya Mane
(172) Sanket Bhoite
(184) Tejas Jadhav

Index

Abstract

This project aims to develop an AI-based chatbot, available as a mobile and web application, that generates answers to queries based on frequently asked questions (FAQs). Traditionally, answering queries required manual intervention, leading to delays and increased workload. By leveraging the latest AI technologies, this project seeks to automate the process by generating answers automatically based on the given questions. Additionally, the chatbot will store every question asked and continually expand its database of FAQs, enhancing the mapping of questions to answers.

To achieve this, machine learning algorithms will be employed to augment the dataset of questions and answers. These algorithms will ensure that the mapping between questions and answers is maintained as the dataset expands. The ultimate goal is to alleviate the workload for both users and service providers, enabling quick responses and suggestions, and fostering better economic market growth.

By implementing this AI-based chatbot, organizations and individuals will benefit from reduced manual effort in addressing queries, faster response times, and the ability to scale the FAQ database effortlessly. The chatbot will serve as an efficient tool for delivering accurate information to users while enhancing user experience and overall productivity.

Problem Statement

AI BASED CHATBOT WEB APP TO CREATE ANSWERS TO QUERIES BASED ON E-COMMERCE FAQS

- **Problem Statement Description:**
  Develop an AI based Chatbot web app to create answers to queries based on FAQs and more FAQs getting automatically added including categorization.You can use machine learning algorithms to increase the dataset of questions and answers and ensure that the mapping of questions and answers remain maintained every time there is an increase in the dataset.

- **Overview:**
  There have always been queries of people regarding various topics to which answers are provided manually by people. In order to make this process fast the FAQs can be answer using the latest AI technologies where answers can be automatically generated according to the questions. Also, for every question asked it can be stored and the list of questions can be increased to provide better mapping of question and answers. This process can reduce lot of work pressure for both the consumer and the government. Also providing quick responses and suggestions can help in the better economical market growth.

Process Model

The selected approach for software development is Agile, specifically leveraging the Scrum framework. This methodology has proven to be highly effective in delivering high-quality software in an efficient manner. It achieves this by placing emphasis on flexibility, iterative development, and active customer collaboration throughout the development process.

1. **Agile Methodology Overview:**

   Agile methodologies, like Scrum, prioritize adaptability, customer collaboration, and incremental delivery of valuable features. By following an Agile process model, the development team can effectively manage the project, accommodate changing requirements, and deliver a functional chatbot within the given timeframe.

2. **Process Model Application**:
   The Agile process model, specifically Scrum, will be applied to the development of the AI-based chatbot as follows:

   a. Product Backlog:
   - Create a comprehensive list of features and requirements for the chatbot.
   - Prioritize the most important and feasible features for initial development.
   - Features include generating answers based on FAQs, storing new FAQs, and categorizing questions and answers.

   b. Sprint Planning:
   - Select a subset of features from the product backlog for implementation within the one-day time frame.
   - Break down the selected features into smaller tasks that can be completed during the sprint.

   c. Sprint Development:
   - Begin the development sprint, focusing on implementing the selected features.
   - Utilize machine learning algorithms to expand the dataset of questions and answers and ensure proper mapping.
   - Design and develop the chatbot interface for web platforms.

d. Daily Scrum:

- Conduct brief daily meetings to discuss progress, challenges, and coordination among the development team members.
- Adjust the plan if needed to keep the project on track.

e. Sprint Review:

- Review the completed work with stakeholders at the end of the one sprint.
- Collect feedback and evaluate the chatbot's performance, accuracy, and usability based on the generated answers and categorization.

f. Sprint Retrospective:

- Reflect on the sprint process and identify areas for improvement.
- Make necessary adjustments to the development strategy for subsequent sprints based on lessons learned.

3. Conclusion :

By adopting the Agile development approach, specifically utilizing the Scrum framework, the project aims to rapidly develop an AI-based chatbot that can generate answers to queries based on FAQs. This iterative approach allows for flexibility, effective collaboration, and continuous improvement throughout the development process, ultimately ensuring the successful implementation of the chatbot within the one-day time constraint

Software Requirement Specification

3.1 External Interface Requirements

External Interface Requirements for an AI chatbot encompass the interfaces it interacts with to communicate and integrate with external systems or users. Here are some common external interface requirements for an AI chatbot:

1. **User Interface (UI):**
   - The chatbot should have a user-friendly interface for users to interact with.
   - The UI can be a web-based interface, a mobile app, or integrated into an existing platform.
   - It should provide an intuitive and responsive user experience.
2. **Input Interfaces:**
   - The chatbot should support various input methods, such as text input, voice input, or both.
   - It should be able to process user queries and requests from different input sources or channels.
3. **Output Interfaces:**
   - The chatbot should provide responses or output in a user-friendly format.
   - The output can be in the form of text messages, voice responses, or multimedia content, depending on the chatbot's capabilities and deployment environment.
4. **Integration Interfaces:**
   - The chatbot may need to integrate with external systems, databases, or APIs to fetch or update information.
   - It should have integration interfaces to communicate securely and efficiently with these external systems.
   - Integration interfaces may include APIs, web services, or data exchange protocols.
5. **Application Programming Interface (API):**
   - If the chatbot is designed to be integrated into other applications or platforms, it should expose APIs for developers to interact with.

- The API should provide clear documentation and specifications for developers to utilize the chatbot's functionality and integrate it into their own systems.

6. **Messaging Platforms:**
   - The chatbot may need to integrate with messaging platforms or communication channels, such as Facebook Messenger, Slack, or WhatsApp.
   - It should have the necessary interfaces and configurations to connect and communicate with these platforms.
7. **Database Interfaces:**
   - The chatbot may require access to databases or external data sources to retrieve or update information.
   - It should have the ability to connect to the required databases or data repositories securely and efficiently.
8. **Authentication and Security Interfaces:**
   - If the chatbot handles sensitive user data or integrates with systems that require authentication, it should provide appropriate authentication mechanisms.
   - It should follow secure communication protocols to protect user information and ensure data privacy.
9. **Error and Exception Handling Interfaces:**
   - The chatbot should have error and exception handling interfaces to log and report errors.
   - It should provide error messages or notifications to users or administrators when errors occur.
10. **Administrator Interface:**
    - The chatbot may require an interface for administrators or developers to manage its configuration, update FAQs, or monitor its performance.
    - The administrator interface should provide secure access and appropriate functionalities for managing the chatbot effectively.

3.1.1 User Interfaces

The user interface (UI) of an AI chatbot can vary depending on the platform or application where it is deployed. Here are some common elements and considerations for designing the user interface of an AI chatbot:

1. **Chat Window:**
   - The chat window is where the conversation between the user and the chatbot takes place.

- It should display the messages exchanged between the user and the chatbot in a visually distinguishable manner.
- Each message should be attributed to the sender (user or chatbot) for clarity.

2. **Input Field:**
   - The input field allows the user to enter their queries or messages.
   - It should be easily accessible and prominently displayed within the UI.
   - The input field may support text-based input, voice input, or a combination of both, depending on the capabilities of the chatbot.
3. **Send/Submit Button:**
   - The send/submit button triggers the chatbot to process and respond to the user's query.
   - It should be located near the input field and clearly labelled to indicate its purpose.
   - Alternatively, the chatbot may support real-time input processing without the need for a manual submission button.
4. **Chat History:**
   - The chat history section displays the previous messages exchanged between the user and the chatbot.
   - It allows the user to refer to previous responses and maintain context during the conversation.
   - The chat history may provide scroll functionality to accommodate long conversations.
5. **Typing Indicator:**
   - The typing indicator visually notifies the user when the chatbot is processing their query.
   - It gives the user feedback to indicate that the chatbot is actively working on their request.
6. **Visual Design and Branding:**
   - The UI should adhere to consistent visual design principles and branding guidelines.
   - It should reflect the branding elements, colors, and style of the organization or application it is associated with.
7. **Responsiveness and Adaptability:**
   - The UI should be responsive and adaptable to different screen sizes and devices.

- It should be optimized for both desktop and mobile environments to ensure a seamless user experience.

3.1.2 Hardware Interfaces

Hardware Interfaces for an AI chatbot typically involve the hardware components necessary for its deployment and operation. Here are some common hardware interfaces for an AI chatbot:

1. **Servers or Hosting Environment:**
   - The chatbot may require dedicated servers or a hosting environment to run its software components.
   - The hardware infrastructure should meet the requirements for processing power, memory, and storage capacity based on the chatbot's workload and user traffic.
2. **Network Interfaces:**
   - The chatbot needs network interfaces to connect to the internet or intranet.
   - It may require Ethernet ports, Wi-Fi adapters, or other networking hardware to establish network connectivity.

3. **Input Devices:**
   - If the chatbot supports voice input, it may require microphones or audio input devices to capture user voice commands or queries.
   - For text input, standard input devices such as keyboards or touchscreens can be used.

4. **Output Devices:**
   - The chatbot should be able to deliver output to users through various devices.
   - Common output devices include displays (monitors, screens), speakers or headphones for voice responses, and printers (if applicable).

5. **Storage Devices:**
   - The chatbot may require storage devices such as hard drives or solid-state drives to store data, logs, or other information.

- The storage capacity should be sufficient to handle the expected workload and data requirements of the chatbot.

3.1.3 Software Interfaces

Software Interfaces for an AI chatbot involve the software components and interfaces required for its development, deployment, and integration. Here are some common software interfaces for an AI chatbot:

1. Application Programming Interface (API):

- The chatbot may expose APIs that allow other applications or systems to interact with it.
- These APIs define the methods and protocols for communication and data exchange with the chatbot.

2. Natural Language Processing (NLP) APIs:

- The chatbot may utilize external NLP APIs or libraries to process and understand user queries.
- These APIs provide functionalities such as text parsing, entity recognition, sentiment analysis, or language translation.

3. **Web Service Interfaces:**

- The chatbot may need to integrate with external web services or APIs to fetch or update information.
- It should communicate with these services using web service interfaces, such as RESTful APIs or SOAP.

4. Database Interfaces:

- The chatbot may interact with databases or data storage systems to retrieve or update information.
- It should use appropriate database interfaces, such as SQL or NoSQL database connectors, to perform database operations.

5. Messaging Platform APIs:

- If the chatbot is integrated with messaging platforms like Facebook Messenger or Slack, it should utilize their respective APIs.

- These APIs provide methods for sending and receiving messages, managing conversations, and accessing user profiles.

3.2 Functional Requirements

Functional Requirements for an AI chatbot typically involve the core functionalities and capabilities it should possess to effectively interact with users and provide valuable assistance. Here are some common functional requirements for an AI chatbot:

1. **Natural Language Understanding (NLU):**
   - The chatbot should be able to understand and interpret user queries in natural language.
   - It should utilize natural language processing (NLP) techniques to extract the meaning, intent, and entities from user inputs.
2. **Intent Recognition and Classification:**
   - The chatbot should accurately recognize the intent or purpose behind user queries.
   - It should classify user intents into predefined categories or actions to determine appropriate responses.
3. **Contextual Understanding and Memory:**
   - The chatbot should maintain contextual understanding during the conversation.
   - It should remember past interactions and utilize the context to provide relevant and coherent responses.
4. **Answer Generation:**
   - The chatbot should generate accurate and informative responses based on user queries.
   - It should be able to retrieve information from a knowledge base or database and generate concise and relevant answers.
5. **Multi-turn Conversation:**
   - The chatbot should engage in multi-turn conversations, allowing back-and-forth interactions with users.
   - It should maintain the conversational flow and coherence across multiple user queries.
6. **Personalization:**
   - The chatbot may incorporate personalization elements to enhance the user experience.

- It should adapt its responses and recommendations based on user preferences, history, or demographic information.

7. **Multi-language Support:**
   - The chatbot should support multiple languages to cater to a diverse user base.
   - It should be able to understand and respond to user queries in different languages accurately.
8. **FAQs and Knowledge Base Management:**
   - The chatbot should have a mechanism to manage a frequently asked questions (FAQs) database or knowledge base.
   - It should provide answers to common queries by retrieving relevant information from the knowledge base.
9. **Integration with External Systems and APIs:**
   - The chatbot may need to integrate with external systems or APIs to fetch or update information.
   - It should have the ability to connect to external databases, services, or APIs securely.
10. **Error Handling and Fall-back Mechanisms:**
    - The chatbot should handle errors, exceptions, and unexpected user inputs gracefully.
    - It should provide meaningful error messages or prompts to guide users when queries cannot be processed.
    - The chatbot should have fall-back mechanisms in place to handle unknown or ambiguous queries.

Non-functional requirements:

Performance: The chatbot should be able to generate answers quickly and provide a responsive user experience, even with a large number of concurrent users.

Reliability: The chatbot should be highly reliable and available for users to access at all times, with minimal downtime or disruptions.

Scalability: The system should be scalable to handle an increasing number of users and a growing dataset of questions and answers without compromising performance.

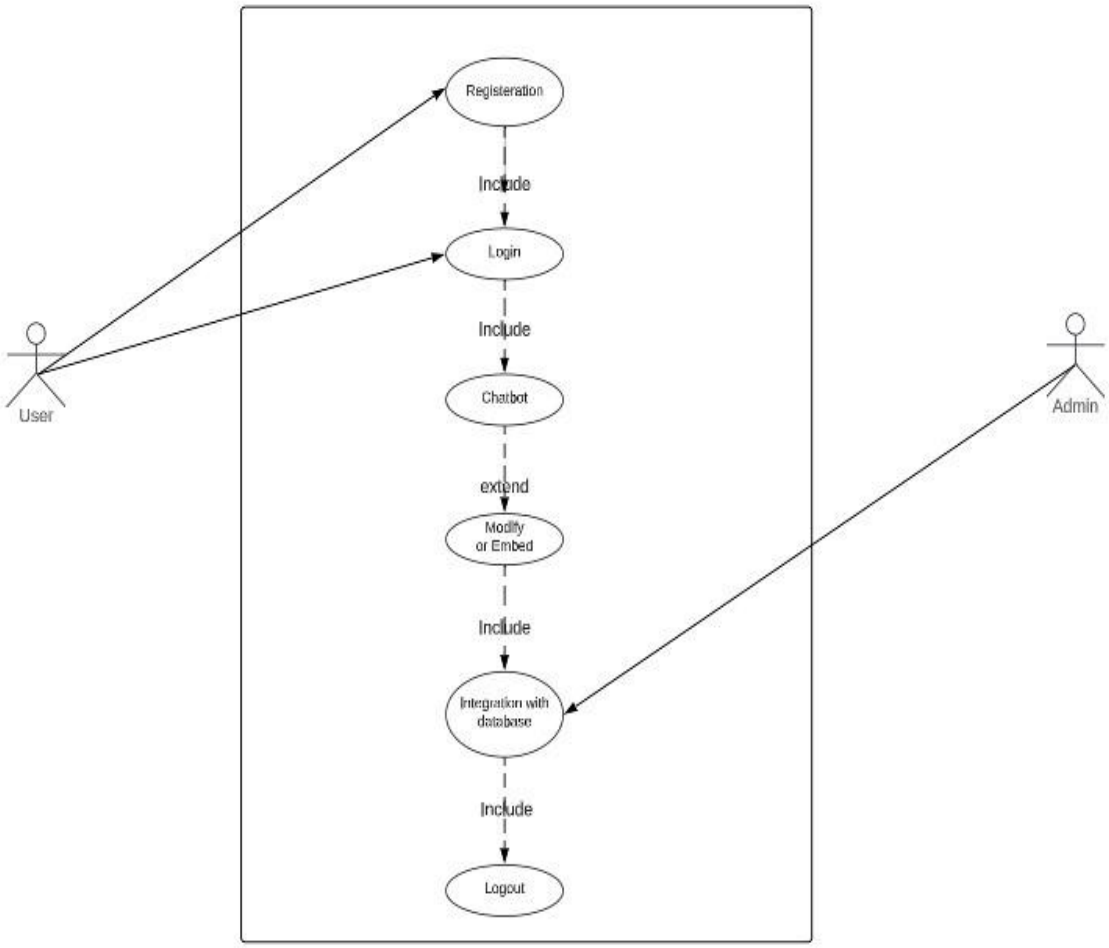
Security: The chatbot should implement appropriate security measures to protect user data and ensure confidentiality, integrity, and availability.

Accuracy: The generated answers should be accurate and relevant to the user's queries, providing reliable information based on the FAQ dataset.

Adaptability: The chatbot should be capable of learning and adapting to new questions and answers over time, improving its performance and effectiveness.

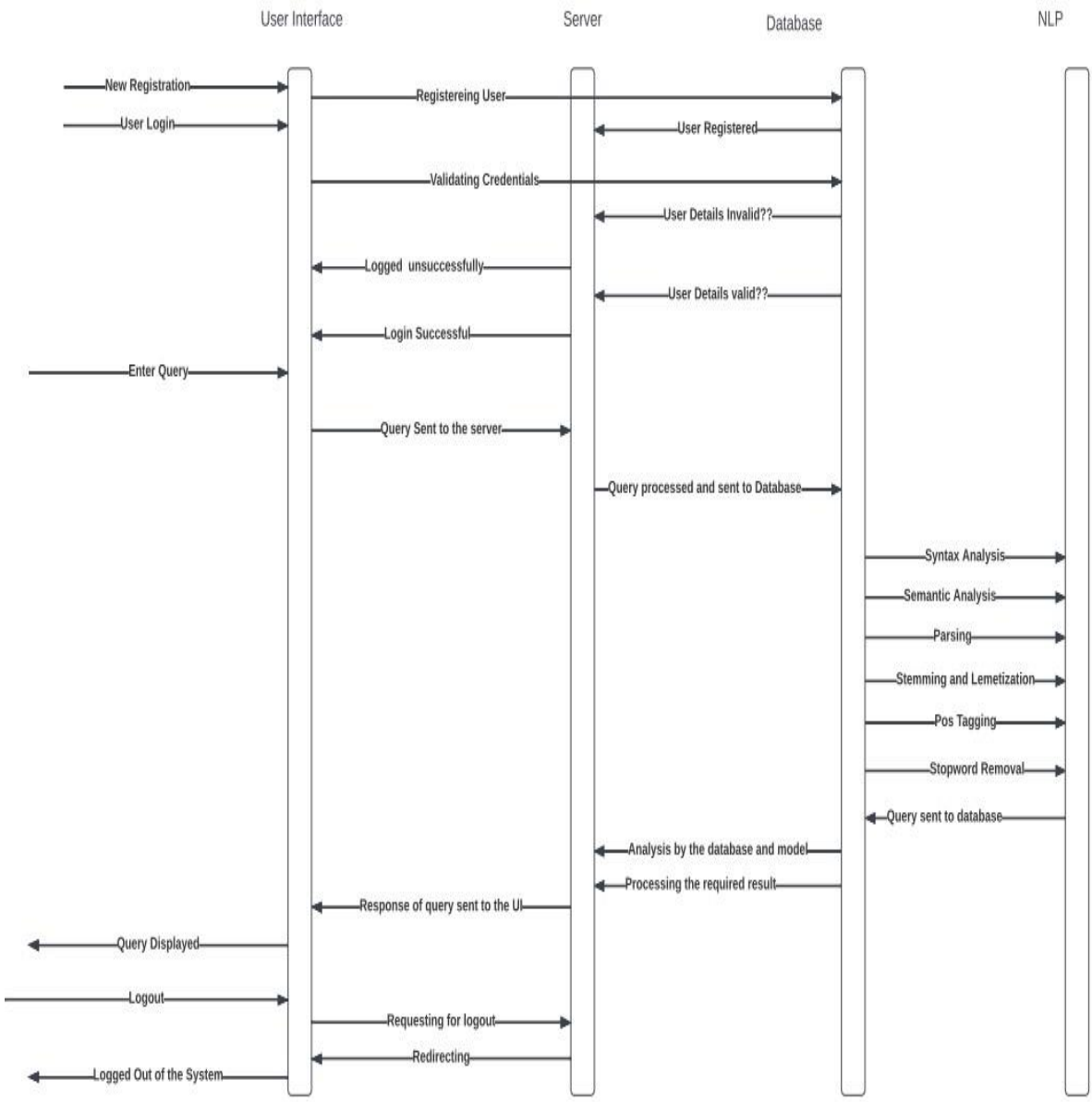
UML DIAGRAMS

1. Usecase Diagram

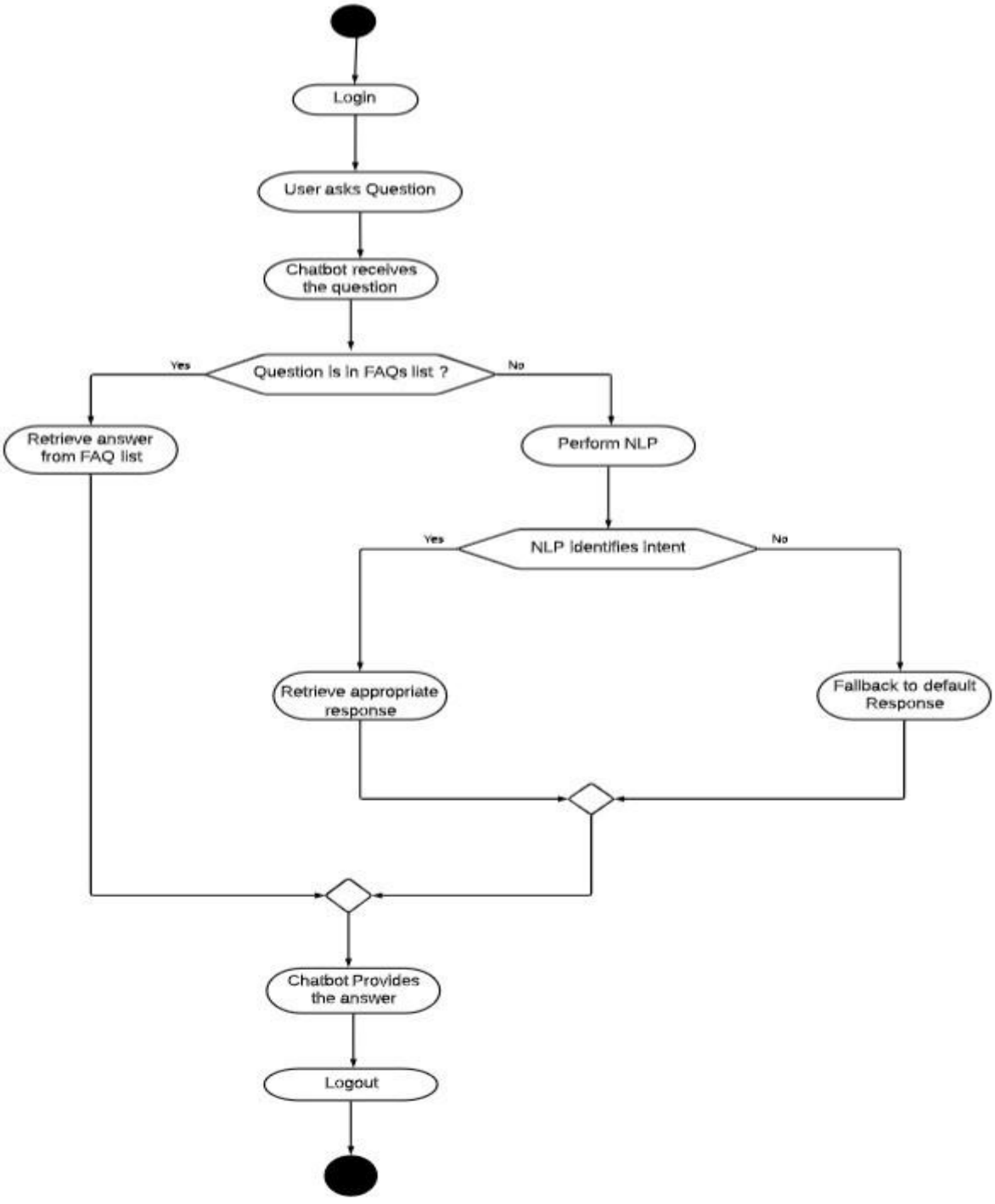


Usecase Diagram

2. Sequence Diagram

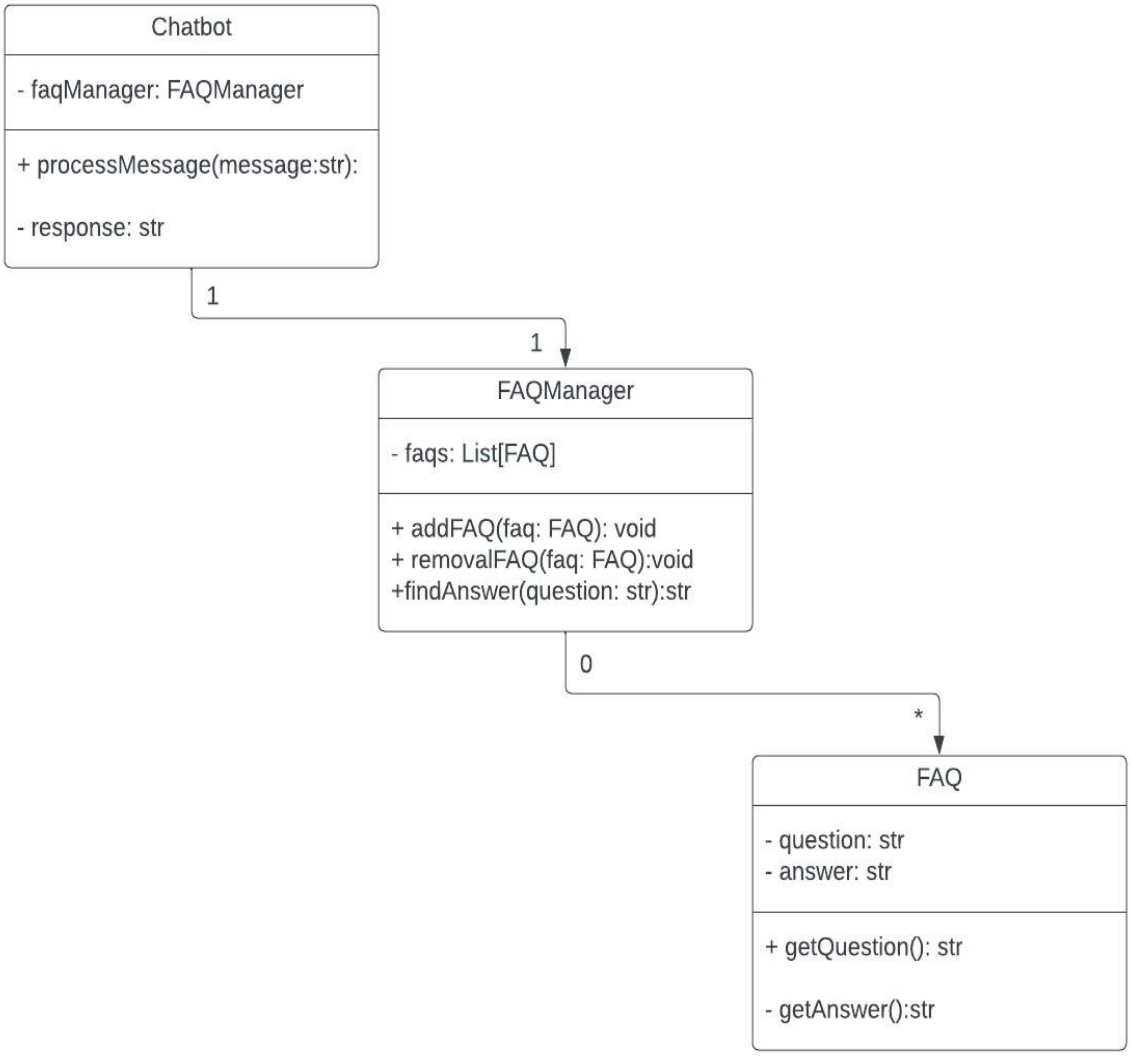


3. Activity Diagram

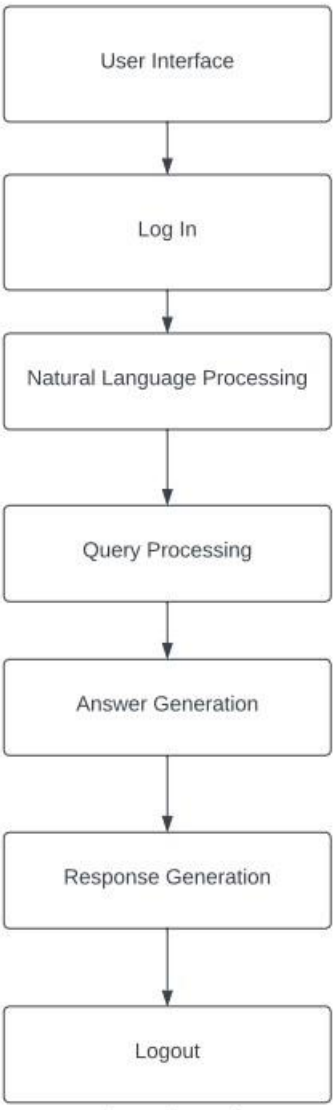


Activity Diagram

4. Class Diagram

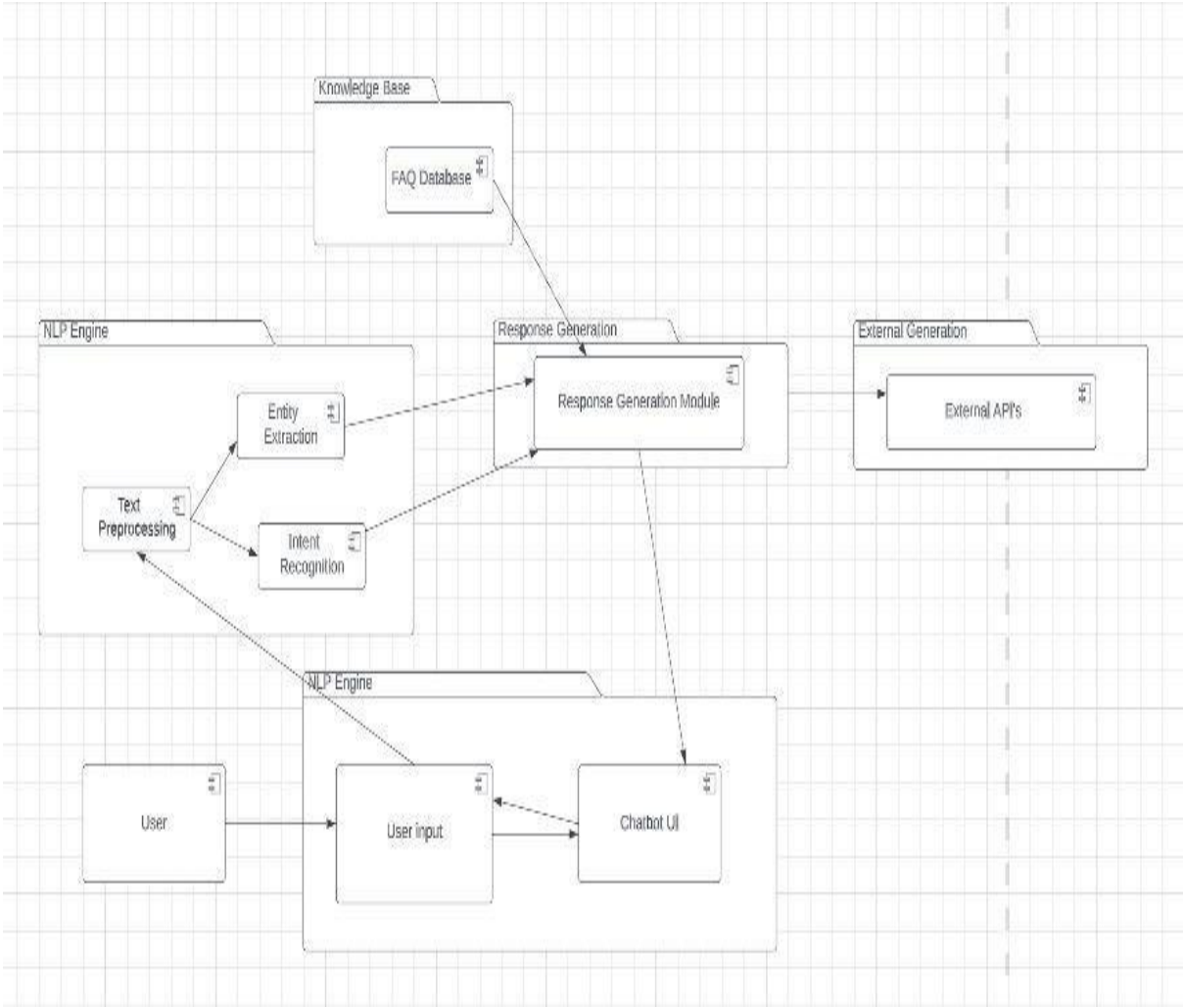


5.Low Level Design

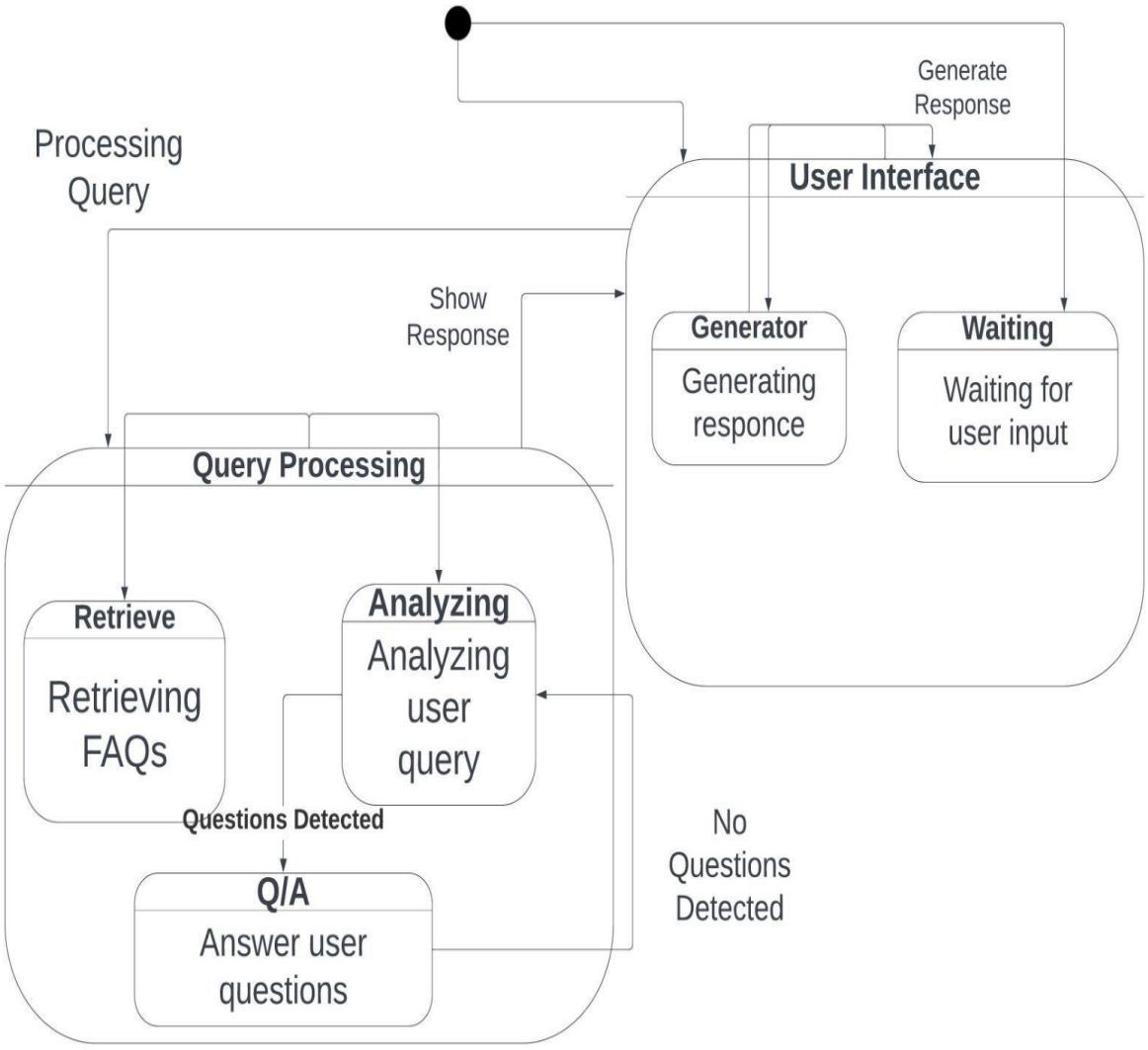


Low Level Design

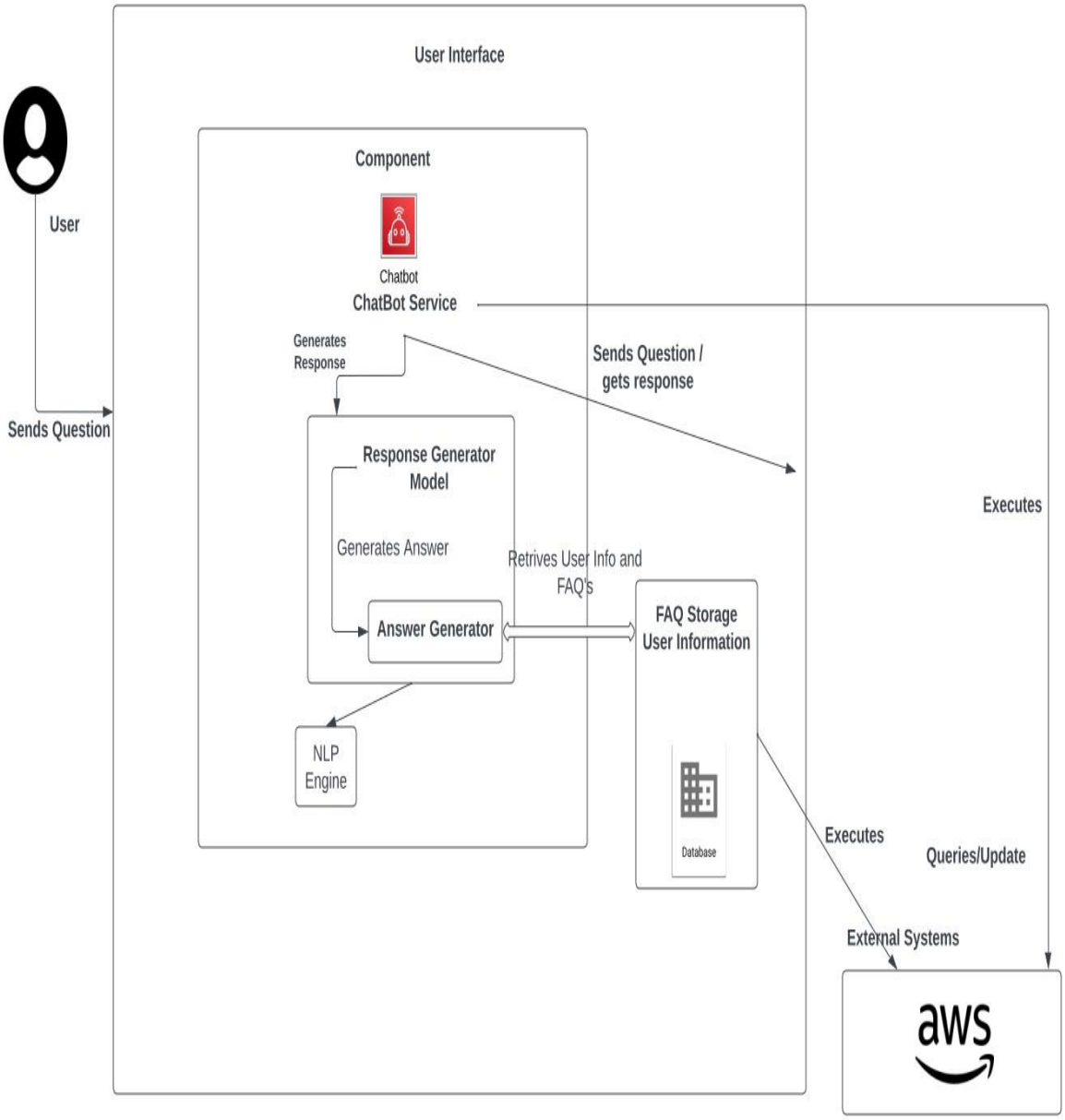
6. High Level Design



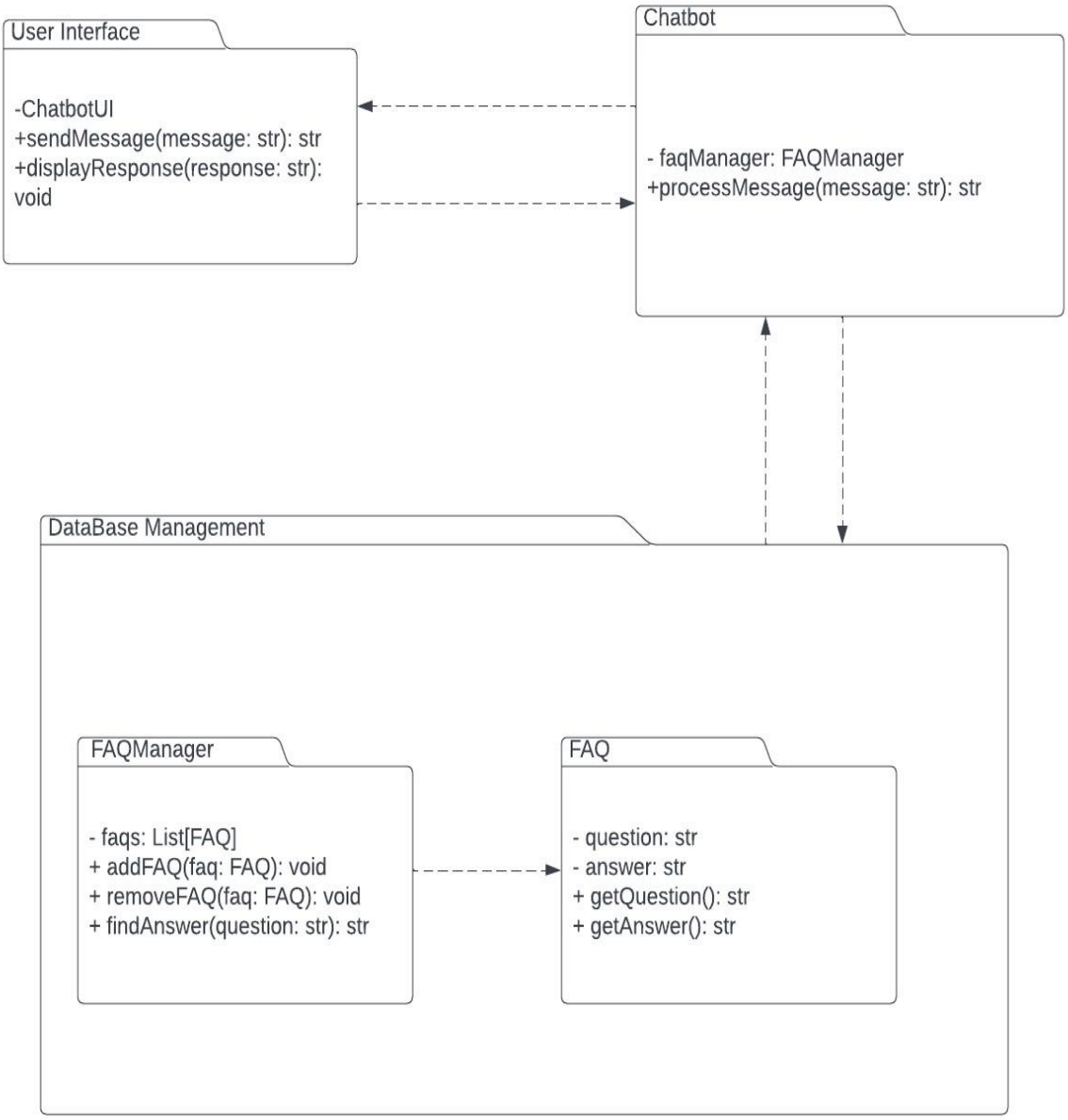
7. State Diagram



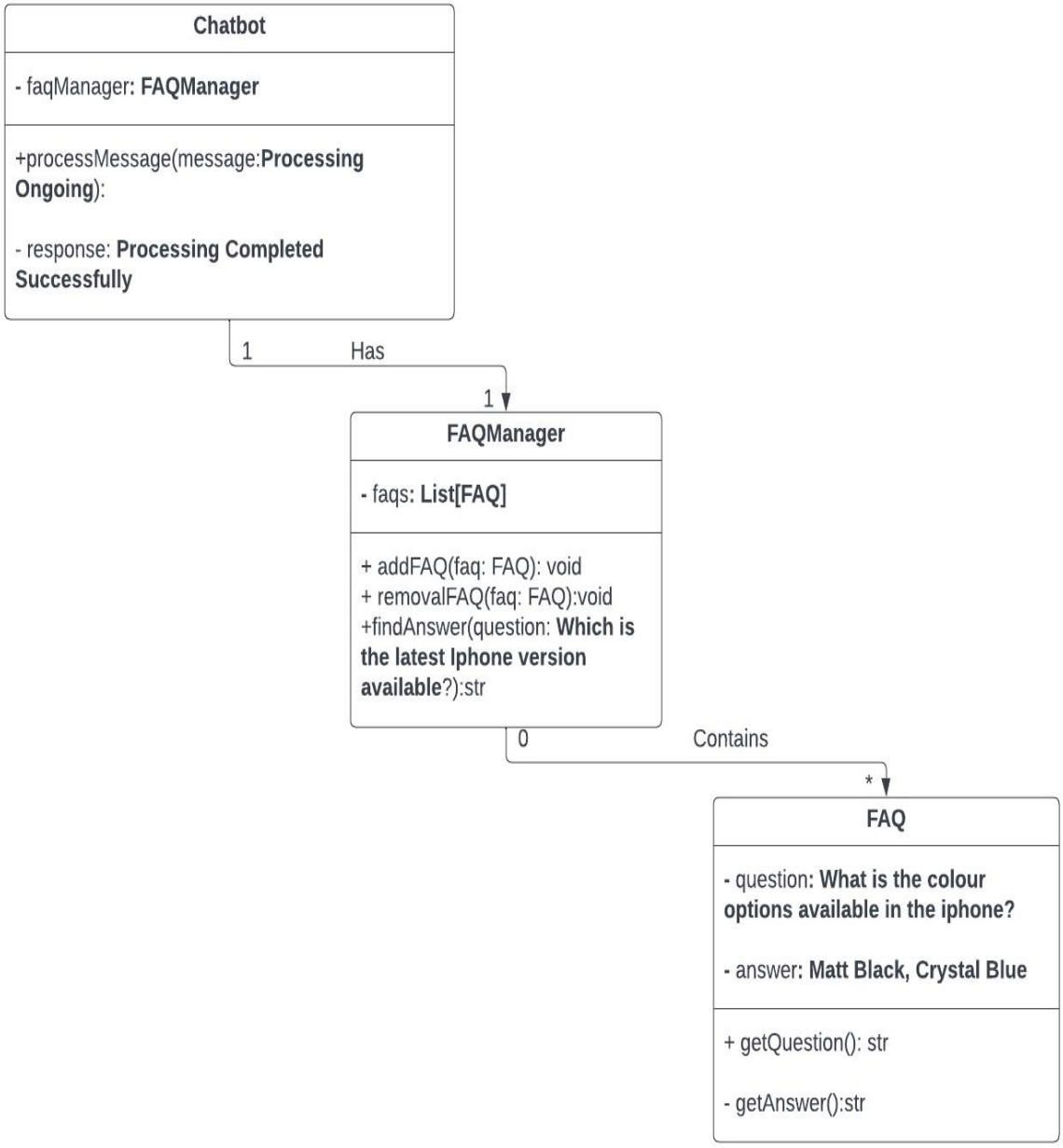
8. Component Diagram



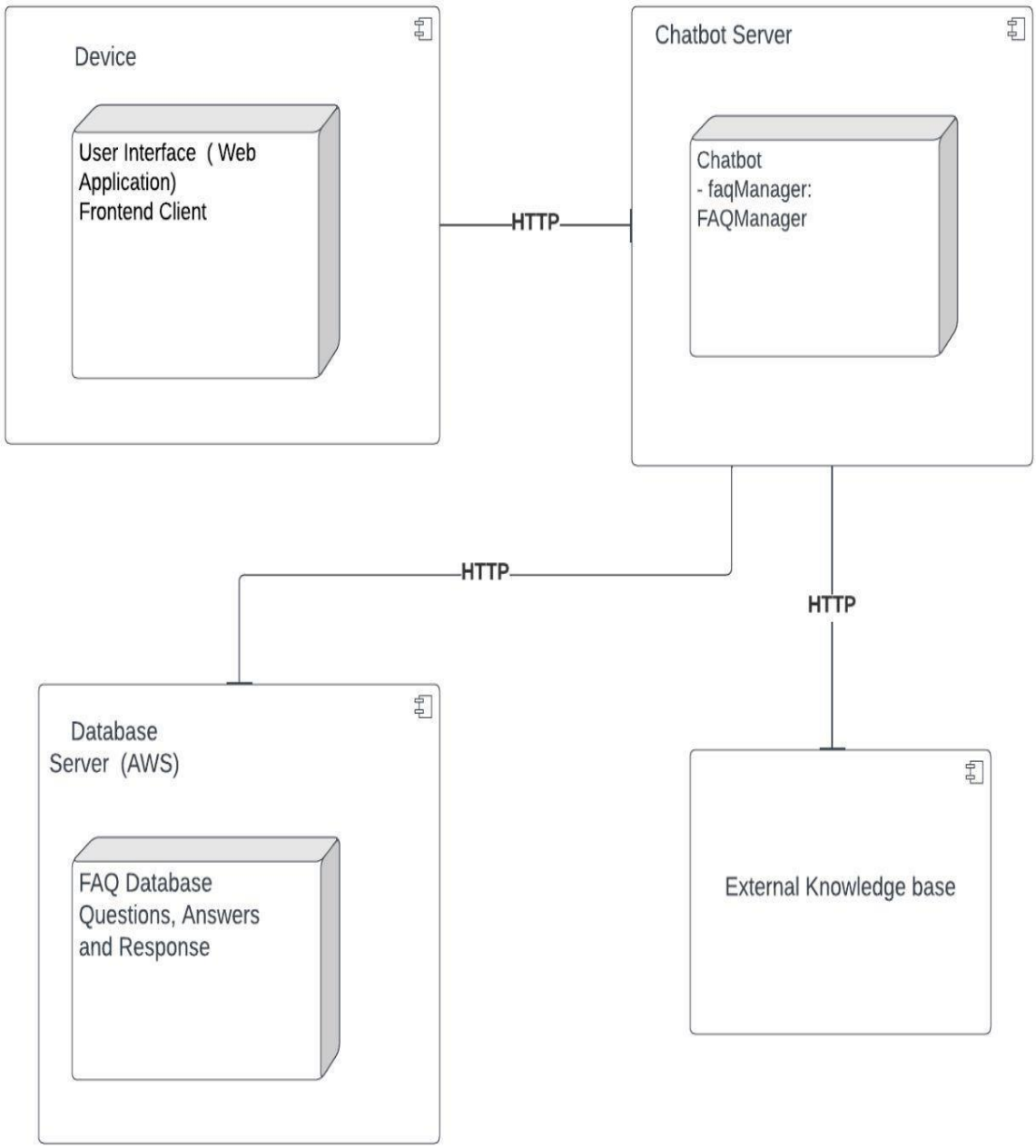
9. Package Diagram



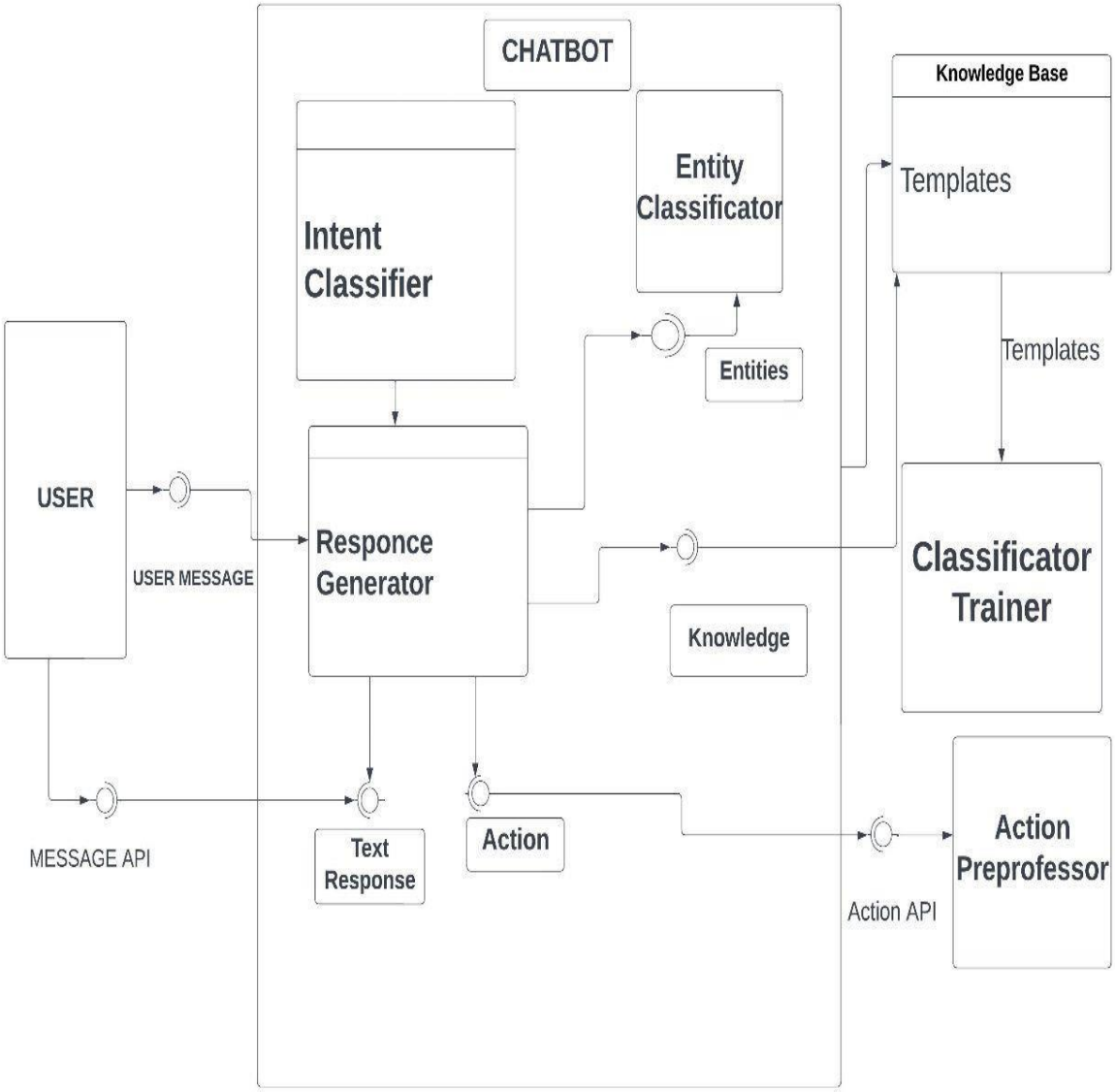
10. Object Diagram



11. Deployment Diagram

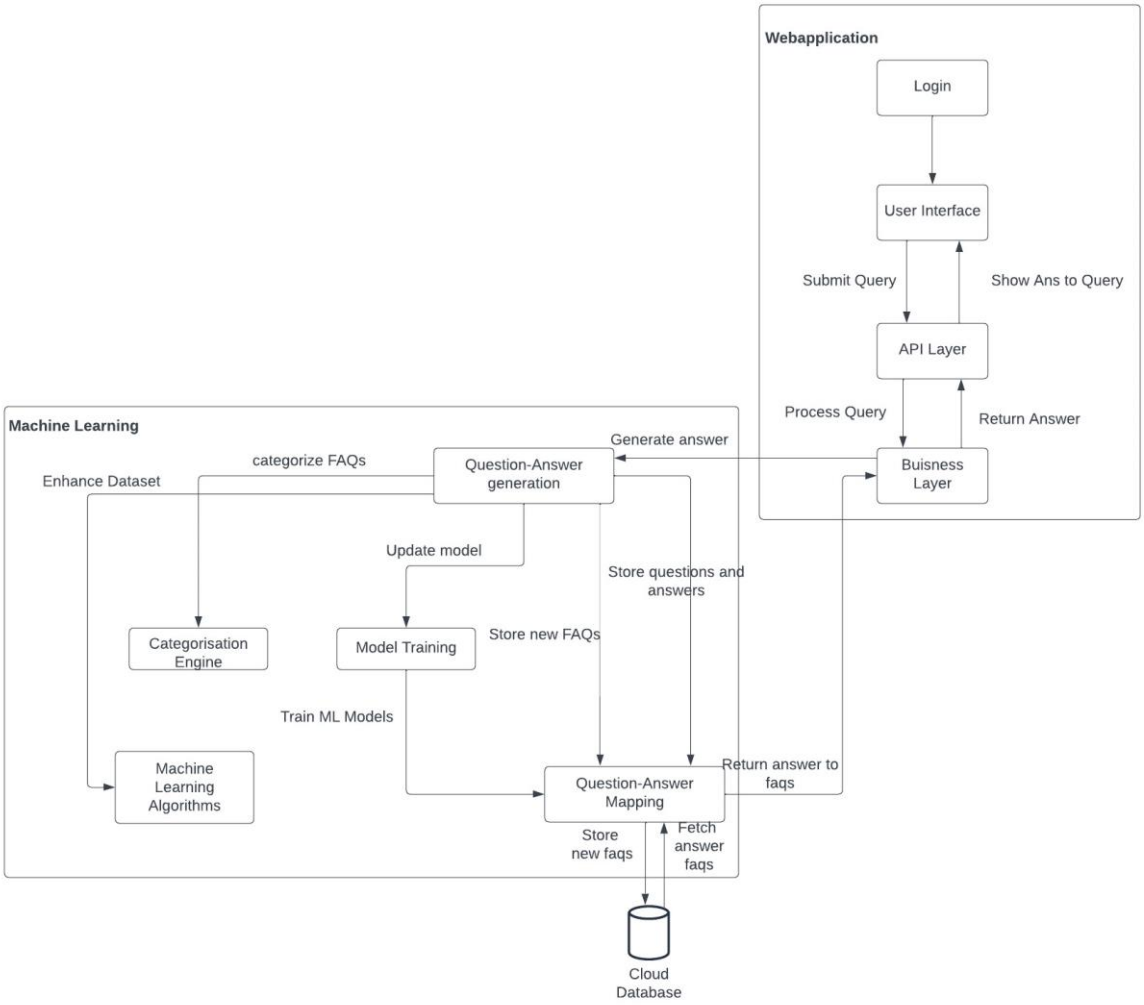


12. Interaction Diagram



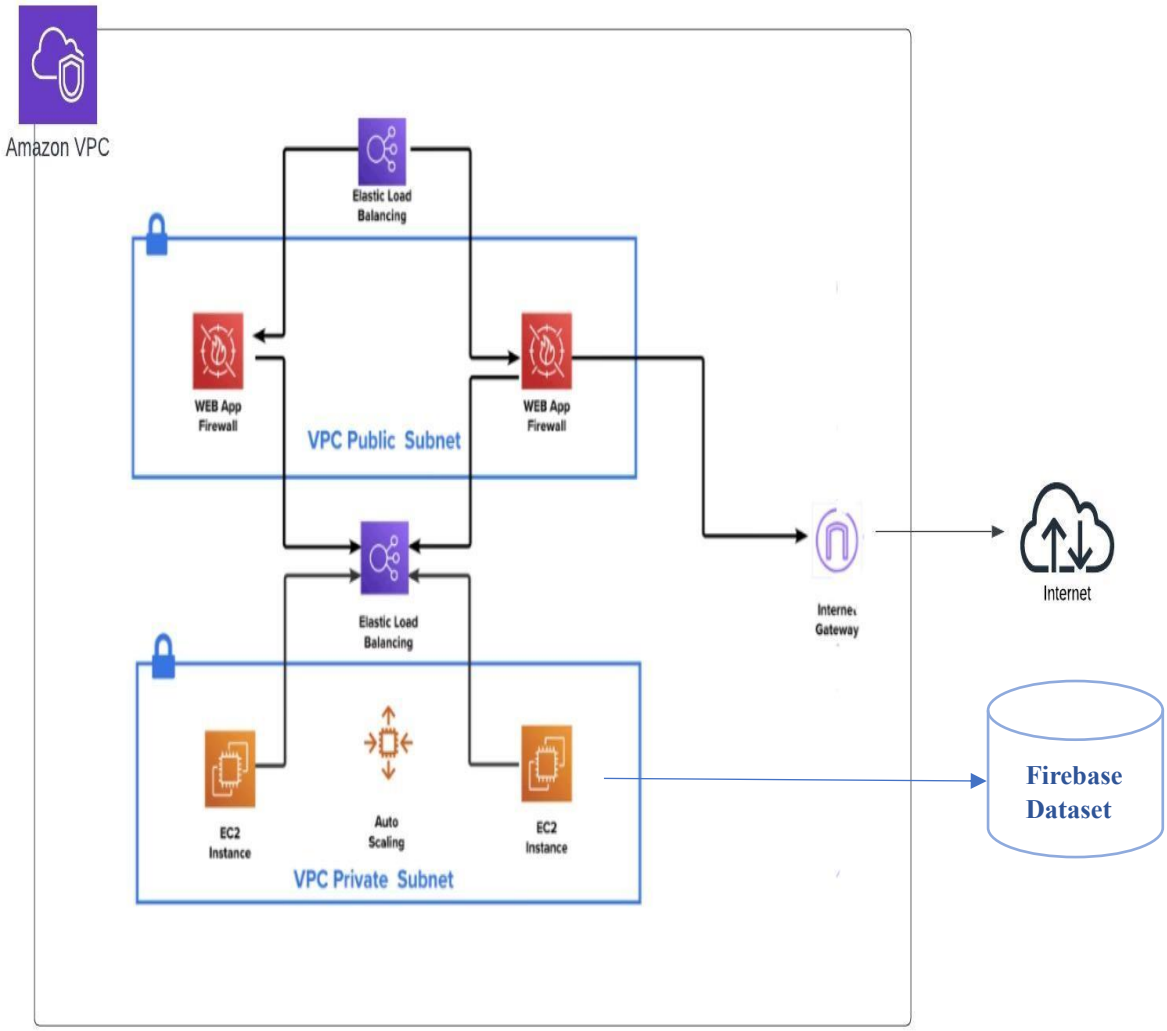
Design Modeling

1. Architectural Diagram



Application Architectural

2. AWS Infrastructural Diagram



AWS Infrastructure Diagram

Cost Estimation Using COCOMO Model

1. **Size:** Considering a medium-sized Chatbot web app project.
    - **Size:** 1.0000 KLOC

2. **Cost Drivers:**

- Required software reliability extent: 1.00 (Nominal)
- Size of the application database:1.00 (Nominal)
- The complexity of the product:1.00 (Nominal)
- Run-time performance constraints: 0 Low
- Memory constraints:1.00 (Nominal)
- Software engineering capability:1.00 (Nominal)
- Applications experience: 0.70 (very high)
- Virtual machine experience: 0.90 (High)
- Programming language experience:0.90 (High)
- Use of software tools: 0.82 (Very High)
- Application of software engineering methods: 0.83 (Very High)
- Required development schedule: 0.04 (High)

3. **Effort Estimation:**
    - **COCOMO equation**: **Effort (E) =** a*(KLOC)^b *EAF PM
      D = c x (Effort)d

- **EAF =** It is an Effort Adjustment Factor, which is calculated by multiplying the parameter values of different cost driver parameters.

EAF=0.015436008

- Typical values for a and b:
- a = 2.4 (for organic projects)
- b = 1.05
- Effort = 2.4 * (1.0000)^1.05 *0.015436008 = 0.037 person-months •

  D= 2.5 *(0.037)*0.38=0.035 months approximately 25 Hours

4. **Cost Estimation**:
   - Considering an average hourly rate of $100 per person-hour.
   - Total Cost = Effort * Hourly Rate = 0.037 * $100 = $3.7 (approximately)

Test Cases and Test Scenario :

MANUAL TESTING : TEST CASES									
PROJECT NAME:	Chatbot								
Group Id :	30								
MODULE NAME:	Login								
TEST CASE ID	**TEST SCENARIO**	**TEST CASE**	**PRE-CONDITION**	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**POST CONDITION**	**ACTUAL RESULT**	**STATUS (PASS/FAIL)**
L_01	Verfiy the login of user	Enter the valid username and valid password	Need a valid username and password to login	1. Enter username 2. Enter password 3. Click "Login" button	<valid username> <valid password>	Successful Login	user Dasboard is visible	Successful Login	PASS
L_01	Verfiy the login of user	Enter the valid username and invalid password	Need a valid username and password to login	1. Enter username 2. Enter password 3. Click "Login" button 2. Enter password 3. Click "Login" button	<valid username> <invalid password> <valid password>	A message "Username and Password is not valid!" is shown	Stay on login page	A message "Username and Password is not valid!" is shown	PASS
L_02	Empty field of username	Enter only password and not username	Need a non-empty valid username and valid password	1. Enter username 2. Enter password 3. Click "Login" button	<empty field of username> <valid password>	A message "Please enter Username! " is shown	Stay on login page	A message "Please enter Username! " is shown	PASS
L_03	Empty field of password	Enter only username and not password	Need a valid username and non-empty valid password	1. Enter username 2. Enter password 3. Click "Login" button	<valid username> <empty fiels of password>	A message "Please enter Password!" is shown	Stay on login page	A message "Please enter Password!" is shown	PASS

MODULE NAME:	**Asking FAQs**								
TEST CASE ID	**TEST SCENARIO**	**TEST CASE**	**PRE-CONDITION**	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**POST CONDITION**	**ACTUAL RESULT**	**STATUS (PASS/FAIL)**
F_01	Searching for FAQs	Enter the FAQS in the form of text	1. Logged-in into system 2. Click on chat option	Click on search button or Click enter		Chatbot provide response	Stay on chatbot	Response from chatbot	PASS

MODULE NAME :	Error Handling								

Test Case ID	TEST SCENARIO	TEST CASE	PRE-CONDITIONS	TEST STEP	TEST DATA	EXPECTED RESULT	POST CONDITIONS	ACTUAL RESULT	STATUS (PASS/FAIL)
E_01	Error Handling	Invalid input text	1. Logged-in into system 2. Click on Chat option	Click on search button		Chatbot prompts "error" message	Stay on chatbot	"Invalid input"	PASS
MODULE NAME	Checking Result								
Test Case ID	TEST SCENARIO	TEST CASE	PRE-CONDITIONS	TEST STEP	TEST DATA	EXPECTED RESULT	POST CONDITIONS	ACTUAL RESULT	STATUS (PASS/FAIL)
A_03	Checking for accurate result	Enter the FAQs in the form of text	1. Logged-in into system 2. Click on chat option	Click on search button		Chatbot prompts "Question is not in database"	Stay on chatbot	"Question is not related to domain!! Please try with specific domain"	PASS

MODULE NAME:	**Context and Conversation Flow**								

TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
C_01	Context and Flow Conversion	User Interrupts	1. Logged-in into system 2. Chat is active and ready to respond 3. Chatbot is in ongoing conversation with the user	User interrupts the conversation and ask the new question		Chatbot displays an error message asking the user to provide a valid input	"Error "message	Chatbot displays an error message asking the user to provide a valid input	PASS
MODULE NAME:	Performanance and Load								
TEST CASE ID	**TEST SCENARIO**	**TEST CASE**	**PRE-CONDITION**	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**POST CONDITION**	**ACTUAL RESULT**	**STATUS (PASS/FAIL)**
P_01	Performanance and Load	Peak load testing	1Logged-in into system. 2.Chatbot is active and ready to respond	Multiple user simultaneously send inquiries to the chat bot		Chatbot acknowledges the user interruption and provides a response to the new question	Stay on chat bot	Chatbot maintains its responsiveness and accuracy during peak load testing	PASS