

CHAT APPLICATION (CHAT BOT)

Using Client - Server
Communication

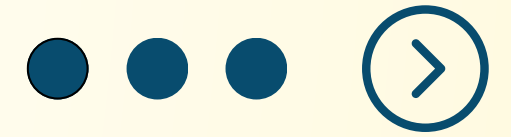
1.Raunak Naik(RA2211003010180)

2.Adya Singh(RA2211003010181)



PROBLEM STATEMENT

In today's digital age, communication is a critical aspect of our personal and professional lives. With the widespread use of the internet and mobile devices, chat applications have become an integral part of our daily interactions. The objective of this project is to design and develop a chat application with a client-server architecture that allows users to exchange text-based messages in real-time.



CLIENT SERVER COMMUNICATION

Client-server communication is a fundamental architectural model in computing and networking, where two separate entities, known as the client and the server, interact and exchange information.

1.Client:

The client is the end-user or application that initiates requests to the server. It could be a web browser, a mobile app, a desktop application, or any device or software that seeks services or resources from the server

2.Server:

The server is a specialized computer or software application responsible for processing client requests and providing the requested services or resources.



CHAT BOT

A chatbot, short for "chat robot," is a computer program or artificial intelligence (AI) application designed to simulate human conversation and interact with users through text or voice-based messaging interfaces. Chatbots are also known as conversational agents, virtual assistants, or interactive agents.

Chatbots are also known as conversational agents, virtual assistants, or interactive agents. They can be used for a variety of purposes, including customer support, information retrieval, automation of tasks, entertainment, and more.



FEATURES CHAT BOT

1. Conversational Interface:

Chatbots have a natural language interface, allowing users to interact with them through text-based messages or spoken language.

2. Answering Questions:

They can answer frequently asked questions and provide information on a wide range of subjects.

3. E-commerce and Transactions:

Transactional chatbots can guide users through online purchases, bookings, or other financial transactions.

4. Integration with Other Systems:

They can integrate with external systems and databases to provide real-time data or access information from other sources.



ALGORITHM

1.Initialization:

Import the required Python modules for socket communication, threading, tkinter, and the OpenAI API. Set up the OpenAI API key.

2.Create a User Login Window (Code 1):

Initialize a GUI window using tkinter, create a login window. Allow users to enter their names.

3.User-to-User Interaction (Code 1):

If the user selects to log in as a user, initialize a chat window for user-to-user interactions.

4.User-to-Chatbot Interaction (Code 2):



If the user selects to continue as a chatbot, create a chat window for user-to-chatbot interactions.

5.Message Handling (Code 1 and Code 2):

Users enter messages in an entry widget.

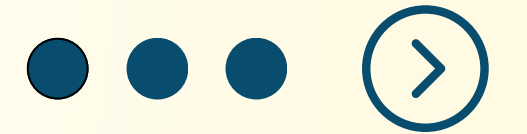
When a user sends a message, display it in the chat window (with an appropriate prefix) and clear the entry widget for the next message.

6.Error Handling (Code 1 and Code 2):

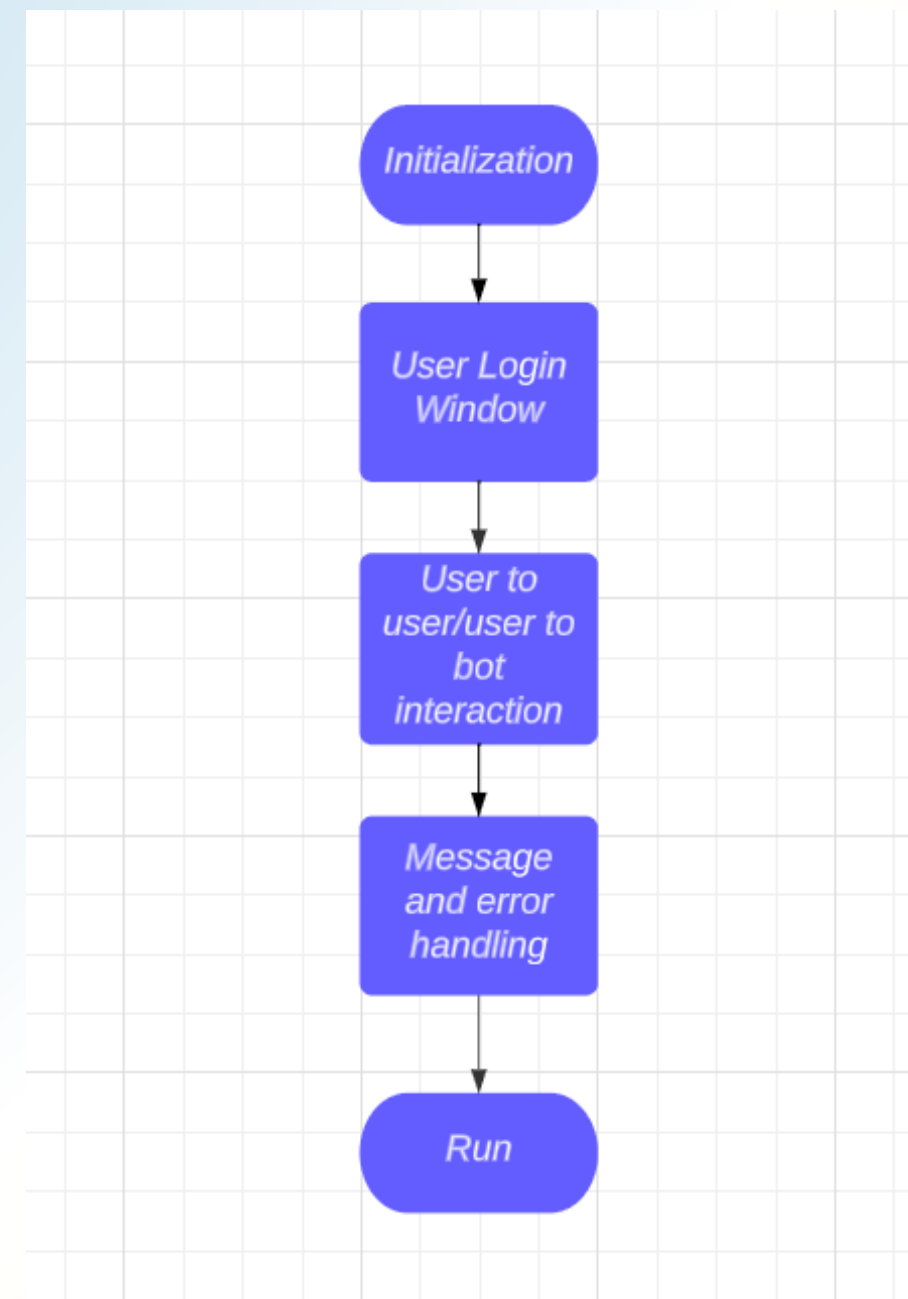
Implement error handling for network communication (user-to-user) and API requests (user-to-chatbot).

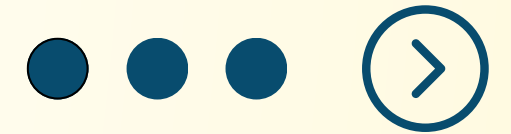
7.Run the Application:

Launch the main application window and start the GUI event loop using tkinter



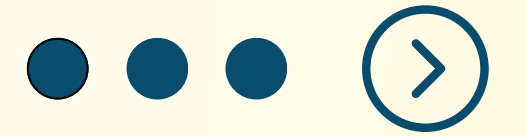
FLOWCHART





MAIN FUNCTIONS OF THE CODE

1. **_init_(self)**: This is the constructor method of the **GUI** class, responsible for initializing the chat application's GUI components.
2. **goAhead(self, name)**: This function is called when the user logs in with their name. It initiates the chat layout.
3. **layout(self, name)**: This function sets up the main chat layout, including the chat window, input field, and send button.
4. **sendButton(self, msg)**: This function handles sending messages from one user to another.
5. **receive(self)**: This function runs in a separate thread to continuously receive and display incoming messages from other users.



6.sendMessage(self): This function sends the user's message to the chat server to be distributed to other users.

7.send(): This function is called when the user clicks the "Send" button. It handles sending user messages to the chatbot.

8.generate_response(message): This function utilizes the OpenAI GPT-3 API to generate responses for the chatbot based on the user's input.



OUTPUT

Welcome

You: hi
Bot: Hi, my friend! Do you want me to tell you a joke?
You: how are you
Bot: I'm fine. Always cheered up when I see you. How are you?
You: im hungry
Bot: What will you be eating?
You: what should i eat
Bot: Healthy food! I recommend you eating mostly vegetables, fruit, and whole grains, healthy fats, and healthy proteins. And drink water instead of sugary beverages.
You: okay thankyou for your suggestion
Bot: Sure. I'm always glad to help.
You: bye
Bot: Have a great journey until next time.

Send

THANKYOU