A blue circle with white text and blue letters

Description automatically generated

**K214549 Aahil Ashiq Ali**

**K213328 Khuzaima Ahsan**

**K214596 Mohammad Khubaib Khan Lodhi**

**MR KASHAN HUSSAIN**

**CN PROJECT :**

**CHAT APPLICATION WITH USER AUTHENTICATION USING SOCKET PROGRAMMING**

**INTRODUCTION**

Introducing a secure chat system built on Python's socket programming, this application provides robust user authentication, ensuring the confidentiality and integrity of conversations. Users can register with unique credentials and securely log in to engage in private chats with other authenticated users. Utilizing a client-to-client restricted conversation model, communication is established only after specifying the recipient's username, enhancing privacy and control over interactions. Through seamless socket communication, this system offers efficient messaging with built-in blocking mechanisms, enabling users to initiate, conduct, and conclude chats securely and intuitively.

**METHODOLOGY**

**Secuirity**

When a user registers, their password undergoes a one-way hashing process. This is achieved using the hashlib library, where the password string is converted into its hashed representation.This hashed password is then stored in a dictionary on the server, associating it with the user's username.

When a user attempts to log in, the provided password is hashed using the same hash function, and the resulting hash is compared with the stored hash for that username. If the hashes match, the user is granted access.

**Message Encryption**

Messages exchanged between clients are encrypted before transmission and decrypted upon receipt to ensure confidentiality.

**Login and Register**

Users provide their username and password. The server verifies the credentials by checking if the provided username exists and if the hashed password matches the stored hash. Upon successful login, the server sends a confirmation message to the client, allowing them to proceed.

If the user is not registered or the credentials are incorrect, an error message is sent back to the client.

**Start Chat and End Chat**

To initiate a chat, a user sends a command to the server indicating the username of the recipient. The server verifies if the recipient is online and available for chat. If the recipient is available, the server establishes a connection between the two clients and informs them that the chat has started. During the chat, messages are exchanged between the clients until either user decides to end the chat. Upon ending the chat, both clients receive a notification, and the server closes the connection between them.

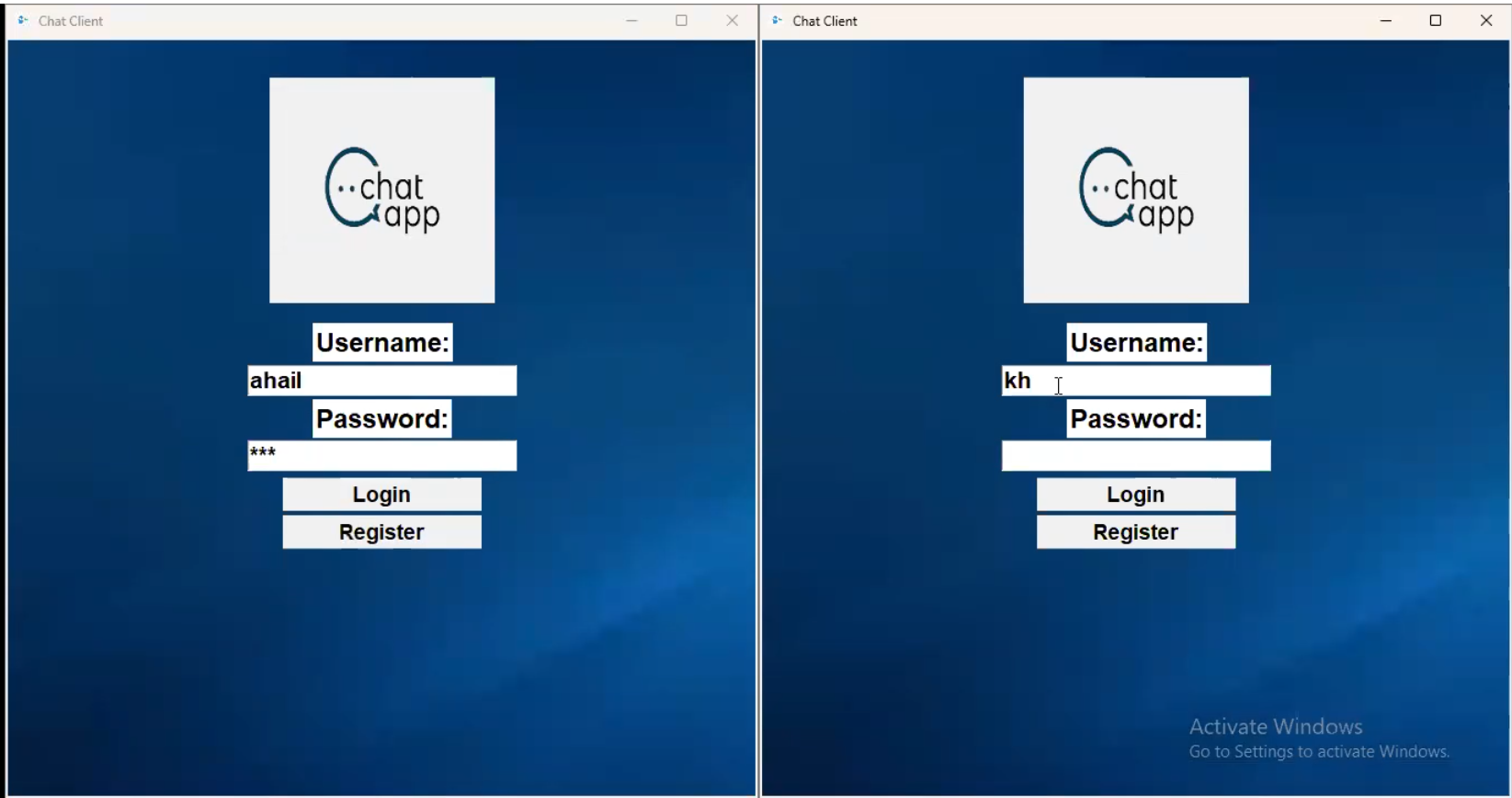
**Chatting**

Once a chat is initiated, clients can send messages to each other. The sender's client sends the message to the server along with the recipient's username. The server forwards the message to the recipient's client, which displays it in the chat interface. Messages are continuously exchanged between clients until the chat is ended.

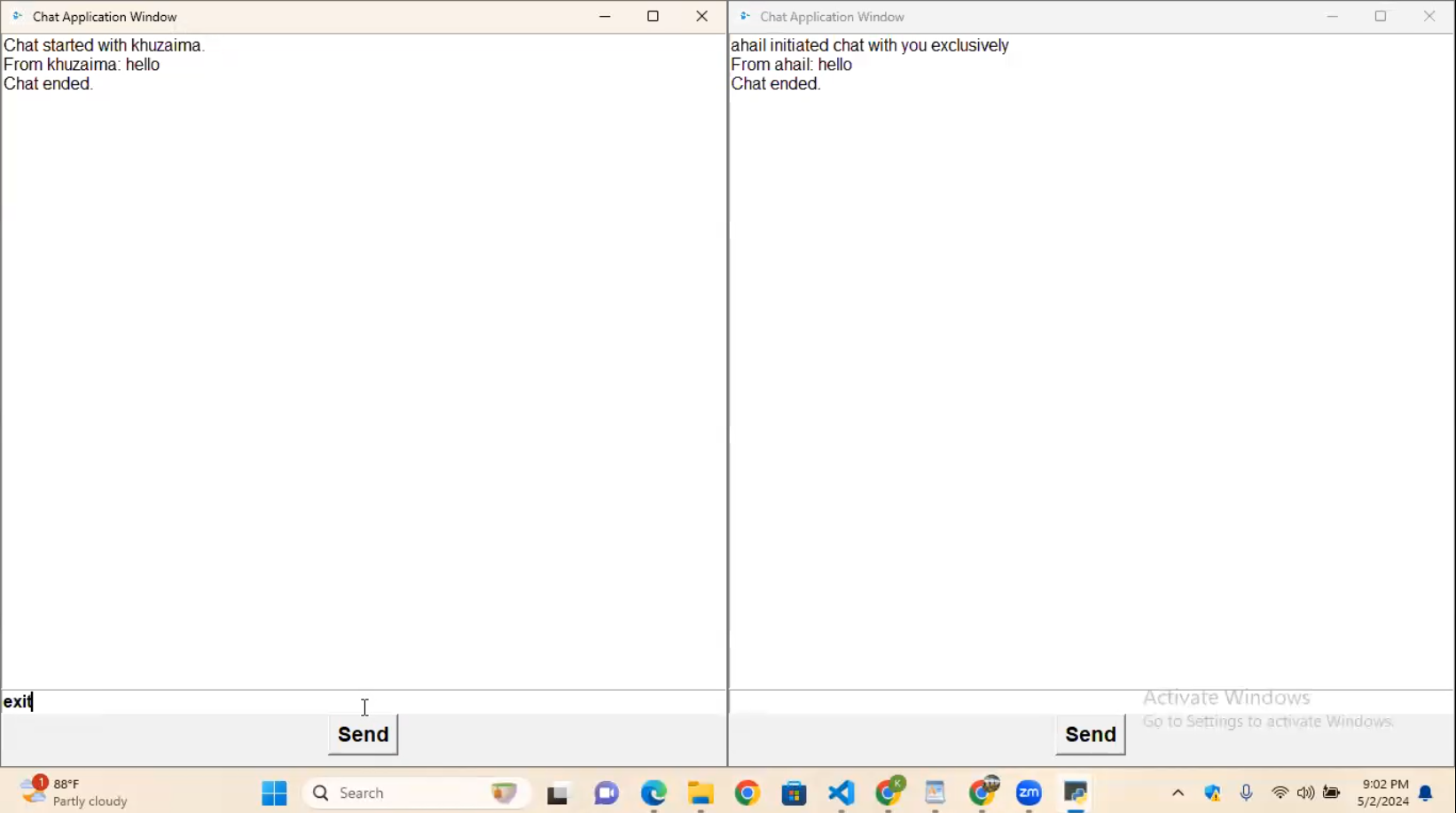
**Exit**

Users can gracefully exit the chat system by sending an exit command to the server. The server closes the connection with the client, freeing up resources and terminating the session. Before disconnecting, any ongoing chats are ended, and other users are notified about the client's departure.

**OUTPUT**



Login Page



Chatting between Aahil and Khuzaima

**REFERENCES**

<https://docs.python.org/3/library/tk.html>

<https://codedamn.com/news/java/how-to-make-a-java-chat-application-using-socket-on-both-side>

<https://www.youtube.com/watch?v=Ar94t2XhKzM>