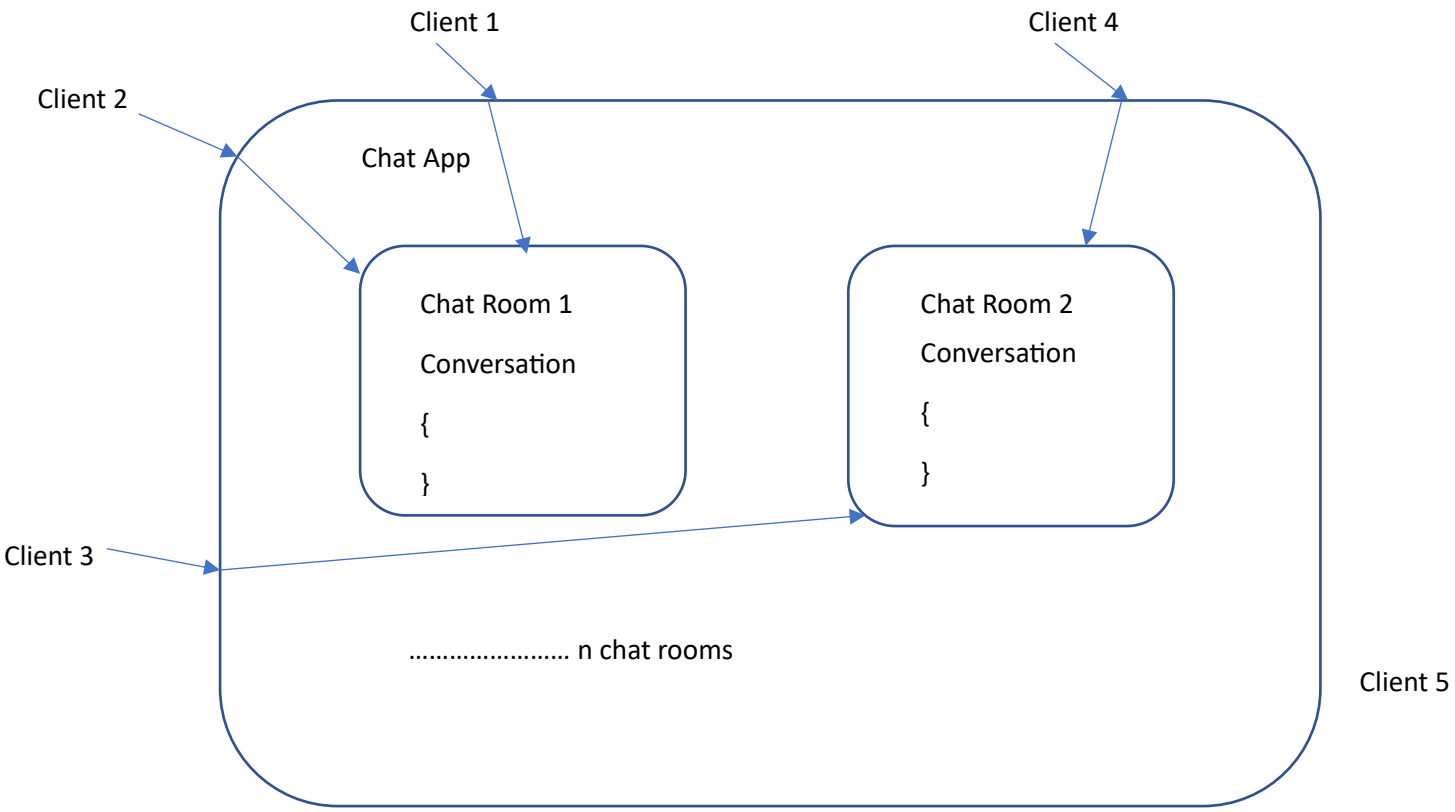
Computer Networks Assignment 4

Vinay Kumar 2020csb1141

Task2

ARCHITECTURE

- There is a multithreaded server which is on always on to cater the request of clients.
- Once a client is connected, a different to thread is made to handle that specific client.
- Once client is Logged in Successfully using correct credentials, he/she can create/join/leave a chat room and his/her information will be stored in *ActiveUsers* dictionary.
- The moment client sends a msg in a chat room he will view all the previous messages.
- All registered users are stored in a dictionary on server side.
- Once a chat room is created a new entry is made in *UsersInChatRoom* dictionary to store all the active users and a new entry in Conversations dictionary to store the chat of the chat room.
- Once a user leaves a chat room his/her information will be removed form *UsersInChatRoom* and once a user logout of chat app his/her information will be removed form *ActiveUsers* but he/she will stay a registered user.
- If the server is killed all data of registered users will be lost because no explicit database is used.



Active Users = {Client1, Client2, Client3, Client4}

UsersInChatRoom[Chat Room 1] = { Client1, Client2}

UsersInChatRoom[Chat Room 2] = { Client3, Client4}

CODE

- SERVER

handle_connection()

The function is used to registed/login the user. If the request id for register, server adds it to the local database or if there is a request for login it checks the credentials and sends the response to the client. If the login is successful client is sent the list of active users and is allowed to join/create a chat room by calling *roomActions()*.

```
def handle_connection(client_socket,client_address):
    id = ""
    while True:
        msg = client_socket.recv(1024).decode()
        if msg.split("_")[0] == "register":
            registerUser(msg)
            SEND("User Registered Successfully, Now you can Login",client_socket)
        elif msg.split("_")[0] == "login":
            if login(msg):
                id = msg.split("_")[1]
                SEND("Login Successful",client_socket)
                time.sleep(0.02)
                actusr = ""
                for x in ActiveUsers:
                    actusr = actusr + x + ","
                actusr = actusr[:-1]
                SEND(actusr,client_socket)
                time.sleep(0.1)
                roomActions(client_socket,id)
                ActiveUsers.pop(id)
            else :
                SEND("Wrong Credentials",client_socket)
                client_socket.close()
            break

    client_socket.close()
```

roomActions()

This function simply gives the client three option logout, join chat room, create chat room. If the user want to create a chat room it will call *createRoom()* or if the user want to join a chat room it will call *joinRoom()*.

```
def roomActions(client_socket,id):
    while True:
        msg = client_socket.recv(1024).decode()
        print(msg)
        if msg == "logout":
            SEND("Logged Out Successfully",client_socket)
            return
        verb = msg.split("_")[0]
        roomID = msg.split("_")[1]
        if verb == "join":
            joinRoom(id,roomID,client_socket)
        elif verb == "create":
            createRoom(id,roomID,client_socket)
```

createRoom()

If the room with a provided room id does not exist a new entry will be made in Conversations and UsersInChatRoom corresponding to that room id.

```
def createRoom(id,roomID,client_socket):
    if roomID not in UsersInChatRooms:
        UsersInChatRooms[roomID] = []
        Conversations[roomID] = []
        msg = "Chat Room "+roomID+" Created Successfully"
        SEND(msg,client_socket)
    else:
        msg = "Chat Room "+roomID+" already exist"
        SEND(msg,client_socket)
```

joinRoom()

If the room with a provided room id exist users will be added to the room and the list of active user in the room is sent to the client. Now user is given access to chat in the room by calling *chat()*.

```
def joinRoom(id,roomID,client_socket):
    if roomID in UsersInChatRooms:
        UsersInChatRooms[roomID].append(id)
        SEND("1",client_socket)
        time.sleep(0.2)
        actusr = ""
        for x in UsersInChatRooms[roomID]:
            actusr = actusr + x + ","
        actusr = actusr[:-1]
        print(actusr)
        SEND(actusr,client_socket)
        time.sleep(0.2)
        print(client_socket.recv(1024).decode())
        chat(id,roomID,client_socket)
    else:
        msg = "Chat Room "+roomID+" does not exist"
        SEND(msg,client_socket)
```

c

chat()

Function receives messages from a client and stores them in conversations of chat room and the msg will be sent to everyone connected in the chat room. If the user sends “exit” he/she will leave the chat room.

```
def chat(id,roomID,client_socket):
    while True:
        message = client_socket.recv(1024).decode()
        if message == "exit":
            UsersInChatRooms[roomID].remove(id)
            SEND("Leaved Room Successfully",client_socket)
            return
        message = id+"["+datetime.now().strftime("%I:%M:%S %p")+"]: "+message
        Conversations[roomID].append(message)
        convo = ','.join(Conversations[roomID])
        SEND(convo,client_socket)
        time.sleep(0.5)
```

- CLIENT

chatRoomOptions()

The function is used to show the chat room options to the user. The user can either join a chat room or create a chat room. The user can also logout from the chat room. The user can also send/receive messages to/from the chat room by calling *chat()*. The user can also see the active users in the chat room

```
def chatRoomOptions(id,client_socket):
        while True:
                print("1. Join a chat room")
                print("2. Create a chat room")
                print("3. Logout")
                choice = input("Enter your choice: ")
                if choice == "1":
                        roomID = input("Enter Chat Room ID to join: ")
                        msg = "join_"+roomID
                        SEND(msg,client_socket)
                        response = client_socket.recv(1024).decode()
                        if(response == "1"):
                                print("Joined Chat Room")
                                print("----Active Users in Chat Room "+roomID+"-
                                print(client_socket.recv(1024).decode())
                                print("----------------------------------------
                                SEND("Received",client_socket)
                                chat(id,client_socket)
                        else:
                                print(response)    #Print response
                elif choice == "2":
                        roomID = input("Enter Chat Room ID to create: ")
                        msg = "create_"+roomID
                        SEND(msg,client_socket)
                        response = client_socket.recv(1024).decode()
                        print(response)
                elif choice == "3":
                        msg = "logout"
                        SEND(msg,client_socket)
                        response = client_socket.recv(1024).decode()
                        print(response)
                        break
```

chat()

The function is used to chat with other users in the chat room. It takes two parameters, the id of the user and the client socket. It is called when the user joins a chat room. It is used to send/receive messages to/from the chat room.

```
def chat(id,client_socket):
        print("Now you can start messaging.")
        while True:
                print("Type 'exit' to exit from chat room")
                message = input(id+": ")
                SEND(message,client_socket)
                time.sleep(0.5)
                if message == "exit":
                        print(client_socket.recv(1024).decode())
                        return
                conversation = client_socket.recv(1024).decode()
                convo  = conversation.split(",")
                os.system('clear')
                for x in convo:
                        print(x)
```