CN JACKFRUIT

Topic: Multi User Chatroom with raw sockets

Sujoy Sen: PESUG23CS621

Syed Kaifuddin: PES2UG23CS636

SECTION: J

Client.py

```
import socket
import threading
import tkinter as tk
from tkinter import scrolledtext, messagebox, simpledialog
import datetime

# === CONFIG ===
DEFAULT_HOST = 'localhost'
DEFAULT_PORT = 12345
DEFAULT_USERNAME = 'Guest'
THEME = 'dark'  # Options: 'dark' or 'light'

# === COLOR SCHEME ===
colors = {
    'dark': {
        'bg': '#1e1e1e',
```

```
        'fg': '#ffffff',
        'input_bg': '#2d2d2d',
        'input_fg': '#ffffff',
        'system': '#888888',
        'username': '#4FC3F7',
        'timestamp': '#9CCC65'
    },
    'light': {
        'bg': '#f2f2f2',
        'fg': '#000000',
        'input_bg': '#ffffff',
        'input_fg': '#000000',
        'system': '#555555',
        'username': '#1565C0',
        'timestamp': '#558B2F'
    }
}

class ChatClient:
    def __init__(self, master):
        self.master = master
        self.master.title("Chat Client")
        self.master.geometry("700x500")
        self.master.protocol("WM_DELETE_WINDOW", self.disconnect)

        self.theme = colors[THEME]
```

```
        self.sock = None
        self.username = None
        self.running = False

        self.build_gui()
        self.prompt_login()

    def build_gui(self):
        self.master.configure(bg=self.theme['bg'])

        self.chat_area = scrolledtext.ScrolledText(self.master, bg=self.theme['bg'],
fg=self.theme['fg'], state='disabled', wrap=tk.WORD)
        self.chat_area.pack(padx=10, pady=5, fill=tk.BOTH, expand=True)

        self.entry_frame = tk.Frame(self.master, bg=self.theme['bg'])
        self.entry_frame.pack(fill=tk.X, padx=10, pady=(0, 10))

        self.msg_entry = tk.Entry(self.entry_frame, bg=self.theme['input_bg'],
fg=self.theme['input_fg'])
        self.msg_entry.pack(side=tk.LEFT, fill=tk.X, expand=True, padx=(0, 5))
        self.msg_entry.bind("<Return>", lambda event: self.send_message())

        self.send_btn = tk.Button(self.entry_frame, text="Send",
command=self.send_message)
        self.send_btn.pack(side=tk.RIGHT)
```

```
        self.user_list = tk.Listbox(self.master, bg=self.theme['input_bg'],
fg=self.theme['input_fg'], width=25)
        self.user_list.pack(side=tk.RIGHT, fill=tk.Y, padx=(0, 10), pady=5)


        self.setup_tags()


    def setup_tags(self):
        self.chat_area.tag_config('system', foreground=self.theme['system'],
font=('Arial', 10, 'italic'))
        self.chat_area.tag_config('username', foreground=self.theme['username'],
font=('Arial', 10, 'bold'))
        self.chat_area.tag_config('timestamp',
foreground=self.theme['timestamp'], font=('Arial', 9))


    def prompt_login(self):
        host = simpledialog.askstring("Server IP", "Enter server IP:",
initialvalue=DEFAULT_HOST)
        port = simpledialog.askinteger("Port", "Enter port number:",
initialvalue=DEFAULT_PORT)
        username = simpledialog.askstring("Username", "Choose a username:",
initialvalue=DEFAULT_USERNAME)


        if not host or not port or not username:
            self.master.destroy()
            return


        self.username = username
        self.connect(host, port)
```

```
    def connect(self, host, port):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        try:
            self.sock.connect((host, port))
            self.sock.send(self.username.encode('utf-8'))
            self.running = True
            threading.Thread(target=self.receive_messages, daemon=True).start()
            self.display_system_message("Connected to the server.")
        except Exception as e:
            messagebox.showerror("Connection Failed", f"Failed to connect to
server: {e}")
            self.master.destroy()


    def receive_messages(self):
        while self.running:
            try:
                message = self.sock.recv(1024).decode('utf-8')
                if message.startswith("USERLIST:"):
                    users = message.split(":", 1)[1].split(",")
                    self.update_user_list(users)
                elif message.startswith("KICKED:"):
                    self.display_system_message("You were kicked from the chat by an
admin.")
                    break
                elif message == "SERVER_SHUTDOWN":
                    self.display_system_message("Server has shut down.")
```

```
                break
            else:
                self.display_message(message)
        except:
            break
    self.disconnect()

def send_message(self):
    message = self.msg_entry.get().strip()
    if message:
        try:
            self.sock.send(message.encode('utf-8'))
            self.msg_entry.delete(0, tk.END)
        except:
            self.display_system_message("Failed to send message. Server might be down.")

def update_user_list(self, users):
    self.user_list.delete(0, tk.END)
    for user in users:
        self.user_list.insert(tk.END, user)

def display_message(self, message):
    self.chat_area.config(state='normal')

    # Parse timestamp and username if possible
    if ']:' in message:
```

```
            try:
                username_part, msg_part = message.split(']: ', 1)
                username = username_part.split(' [')[0]
                timestamp = username_part.split('[')[-1]
                self.chat_area.insert(tk.END, f"[{timestamp}] ", 'timestamp')
                self.chat_area.insert(tk.END, f"{username}: ", 'username')
                self.chat_area.insert(tk.END, f"{msg_part}\n")
            except:
                self.chat_area.insert(tk.END, f"{message}\n")
        else:
            self.chat_area.insert(tk.END, f"{message}\n")

        self.chat_area.config(state='disabled')
        self.chat_area.yview(tk.END)

    def display_system_message(self, message):
        timestamp = datetime.datetime.now().strftime("[%H:%M]")
        self.chat_area.config(state='normal')
        self.chat_area.insert(tk.END, f"{timestamp} * {message}\n", 'system')
        self.chat_area.config(state='disabled')
        self.chat_area.yview(tk.END)

    def disconnect(self):
        if self.running:
            self.running = False
            try:
```

```
            self.sock.close()
        except:
            pass
        self.display_system_message("Disconnected from server.")
    self.master.destroy()

if __name__ == "__main__":
    root = tk.Tk()
    app = ChatClient(root)
    root.mainloop()
```

Server.py

```
import socket
import threading
import datetime
import time
import sys
import os

def get_input(prompt, default=None):
    user_input = input(prompt)
    if not user_input and default is not None:
        return default
    return user_input
```

```
HOST = get_input("Enter the HOST IP (default: 0.0.0.0): ", "0.0.0.0")
PORT = int(get_input("Enter the Port number (default: 12345): ", "12345"))

clients = {}
server_running = True

def broadcast(message, sender_socket=None):
    disconnected_clients = []
    for client_socket in clients:
        if client_socket != sender_socket:
            try:
                client_socket.send(message.encode('utf-8'))
            except:
                disconnected_clients.append(client_socket)

    for client_socket in disconnected_clients:
        if client_socket in clients:
            client_socket.close()
            del clients[client_socket]

def send_user_list_to_all():
    users = list(clients.values())
    user_list_message = "USERLIST:" + ",".join(users)
    for client_socket in clients:
        try:
```

```
            client_socket.send(user_list_message.encode('utf-8'))
        except:
            print(f"Error sending user list to {clients.get(client_socket, 'unknown')}")

def kick_user(username, sender_socket):
    user_socket = None
    for sock, name in clients.items():
        if name == username:
            user_socket = sock
            break

    if not user_socket:
        return False

    try:
        user_socket.send(f"KICKED:{username}".encode('utf-8'))
    except:
        pass

    current_time = datetime.datetime.now().strftime("%H:%M")
    kick_message = f"{username} has been kicked from the chat.
[{current_time}]"
    broadcast(kick_message)
    print(kick_message)

    if user_socket in clients:
        del clients[user_socket]
```

```
    send_user_list_to_all()

    try:
        user_socket.close()
    except:
        pass

    return True

def handle_client(client_socket):
    global server_running
    try:
        username = client_socket.recv(1024).decode('utf-8')
        clients[client_socket] = username
        current_time = datetime.datetime.now().strftime("%H:%M")
        join_message = f"{username} has joined the chat! [{current_time}]"
        broadcast(join_message, client_socket)
        print(join_message)

        send_user_list_to_all()

        while server_running:
            try:
                message = client_socket.recv(1024).decode('utf-8')
                if not message:
```

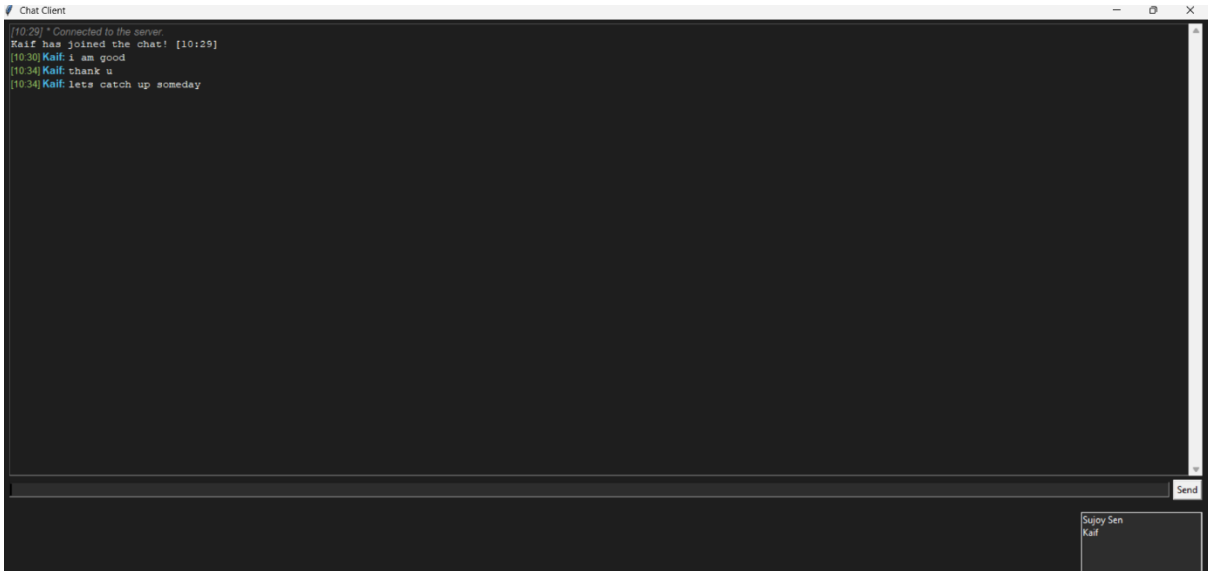
```
                break

            if message.startswith("KICK:"):
                user_to_kick = message.split(":", 1)[1]
                if kick_user(user_to_kick, client_socket):
                    print(f"{clients[client_socket]} kicked {user_to_kick} from the
chat")
                continue

            current_time = datetime.datetime.now().strftime("%H:%M")
            formatted_message = f"{username} [{current_time}]: {message}"
            print(formatted_message)
            broadcast(formatted_message, client_socket)
        except:
            if not server_running:
                break
            raise

    except Exception as e:
        print(f"Error handling client: {e}")
    finally:
        if client_socket in clients and server_running:
            username = clients[client_socket]
            current_time = datetime.datetime.now().strftime("%H:%M")
            leave_message = f"{username} has left the chat. [{current_time}]"
            broadcast(leave_message, client_socket)
            print(leave_message)
```

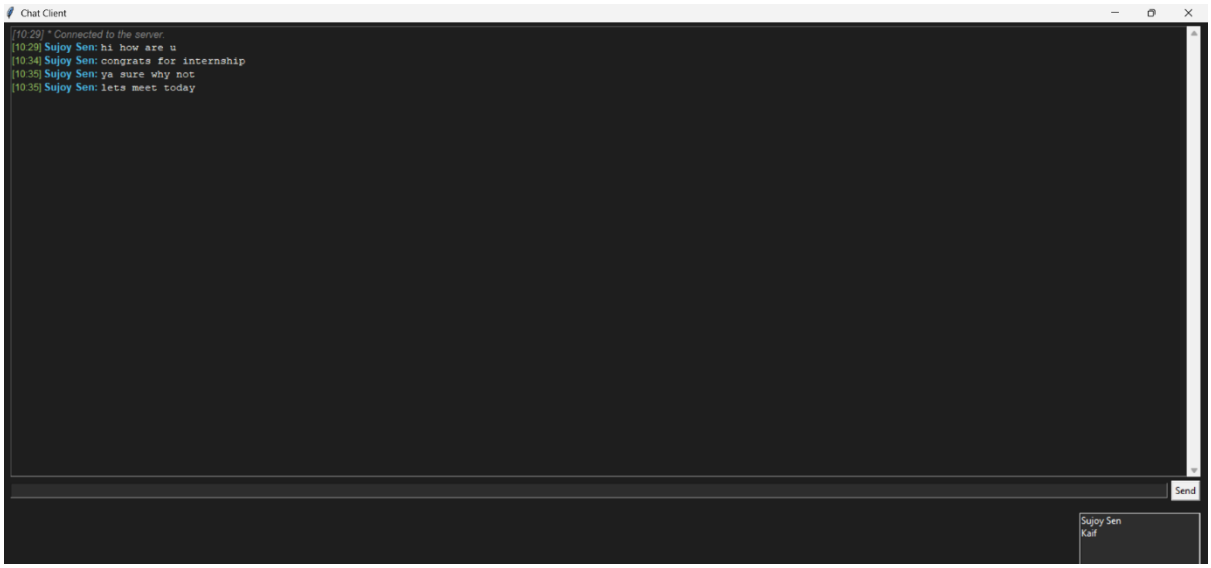
```
            del clients[client_socket]
            send_user_list_to_all()

        try:
            client_socket.close()
        except:
            pass

def shutdown_server():
    global server_running
    server_running = False
    print("Shutting down server...")

    broadcast("SERVER_SHUTDOWN")
    time.sleep(1)

    for client_socket in list(clients.keys()):
        try:
            client_socket.close()
        except:
            pass

    print("All clients notified. Server is shutting down.")
    os._exit(0)

def console_input():
```

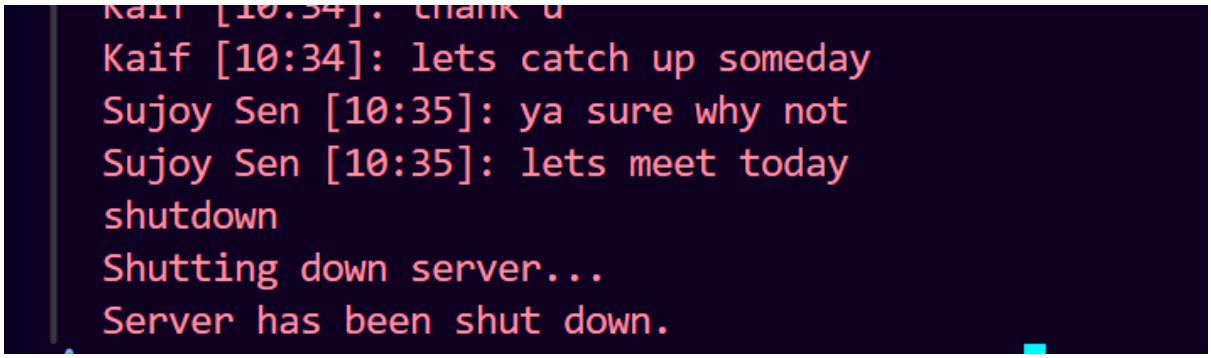
```
    global server_running
    while server_running:
        cmd = input("Enter 'shutdown' to stop the server: ")
        if cmd.lower() == "shutdown":
            shutdown_server()
            break

def start_server():
    global server_running

    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

    try:
        server.bind((HOST, PORT))
        server.listen(5)

        print(f"Server started on {HOST}:{PORT}")
        print("Type 'shutdown' to stop the server.")

        threading.Thread(target=console_input, daemon=True).start()

        while server_running:
            server.settimeout(1.0)
            try:
                client_socket, client_address = server.accept()
```

```
            print(f"New connection from {client_address}")

            threading.Thread(target=handle_client, args=(client_socket,), daemon=True).start()

        except socket.timeout:
            continue
        except Exception as e:
            if server_running:
                print(f"Error accepting connection: {e}")

    except KeyboardInterrupt:
        print("Keyboard interrupt detected.")
        shutdown_server()
    finally:
        server_running = False
        try:
            server.close()
        except:
            pass
        print("Server has been shut down.")

if __name__ == "__main__":
    start_server()
```

```
Enter the HOST IP (default: 0.0.0.0):
Enter the Port number (default: 12345):
Server started on 0.0.0.0:12345
Type 'shutdown' to stop the server.
Enter 'shutdown' to stop the server: New connection from ('127.0.0.1', 49295)
Sujoy Sen has joined the chat! [10:29]
New connection from ('127.0.0.1', 49299)
Kaif has joined the chat! [10:29]
Sujoy Sen [10:29]: hi how are u
Kaif [10:30]: i am good
Sujoy Sen [10:34]: congrats for internship
Kaif [10:34]: thank u
Kaif [10:34]: lets catch up someday
Sujoy Sen [10:35]: ya sure why not
Sujoy Sen [10:35]: lets meet today
```




Chat Client
[10:29] * Connected to the server.
[10:29] Sujoy Sen: hi how are u
[10:34] Sujoy Sen: congrats for internship
[10:35] Sujoy Sen: ya sure why not
[10:35] Sujoy Sen: lets meet today
Send
Sujoy Sen
Kaif


Kaif [10:34]: lets catch up someday
Sujoy Sen [10:35]: ya sure why not
Sujoy Sen [10:35]: lets meet today
shutdown
Shutting down server...
Server has been shut down.