COMPUTER NETWORKS

(UE21CS252B)

Instant Messaging

(Socket Programming PROJECT)

Group Members:

Shrey - PES2UG21CS500

Sujal - PES2UG21CS548

Sudhanva Samaga - PES2UG21CS544

- The objective of this project is to create an instant messaging application using socket programming with the TCP protocol in Python.
- The program consists of a server and multiple clients that can connect to the server to send and receive messages in real-time over the internet.
- The application is built using Python's built-in socket module and the threading module, which allows multiple clients to connect to the server simultaneously without interfering with each other by using multithreads.
- TCP ensures reliable and secure communication betweenclients by establishing a connection between them, verifying the data received, and retransmitting any lost packets.
- The application allows clients to send messages to other clients by specifying their usernames or addresses. Clients can also send and receive files bysending them as data packets over the established TCPconnection.
- In conclusion, this project demonstrates the basic concepts of socket programming and how to use it to create a simple messaging application.
- The project also highlights the importance of using theTCP protocol to ensure reliable and secure communication between clients.
- With further development, this project can be expanded to include additional features such as file sharing, group messaging, and encryption to enhance the security of the messaging application.

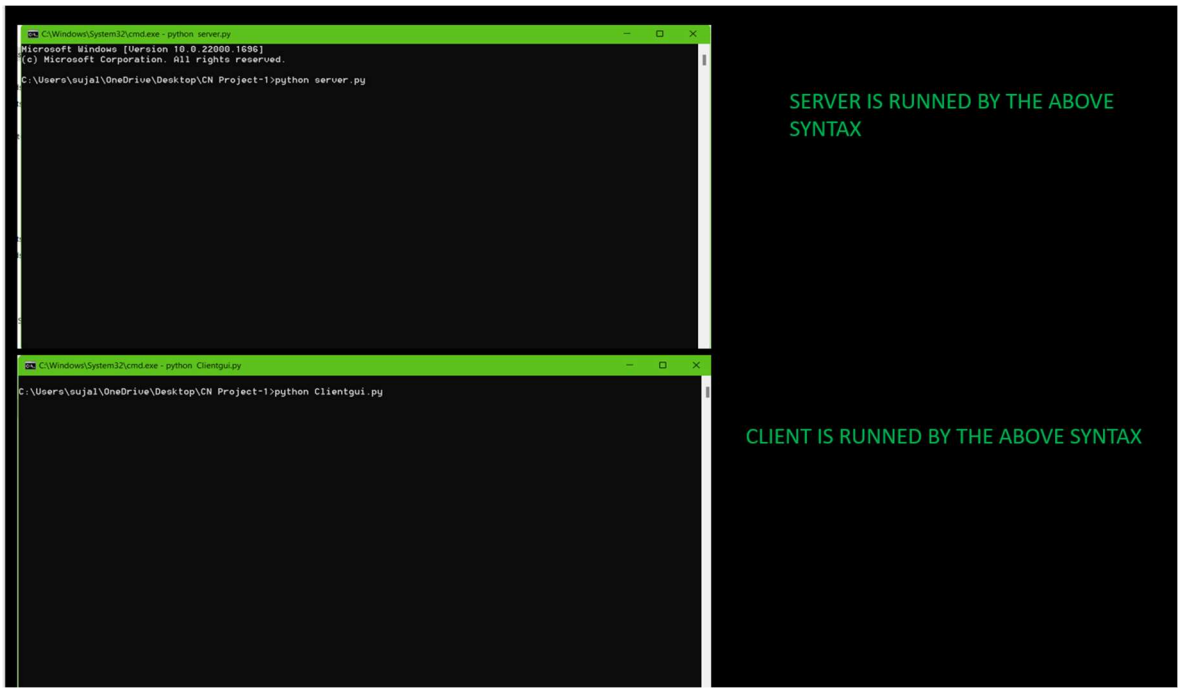
Our Server Code:

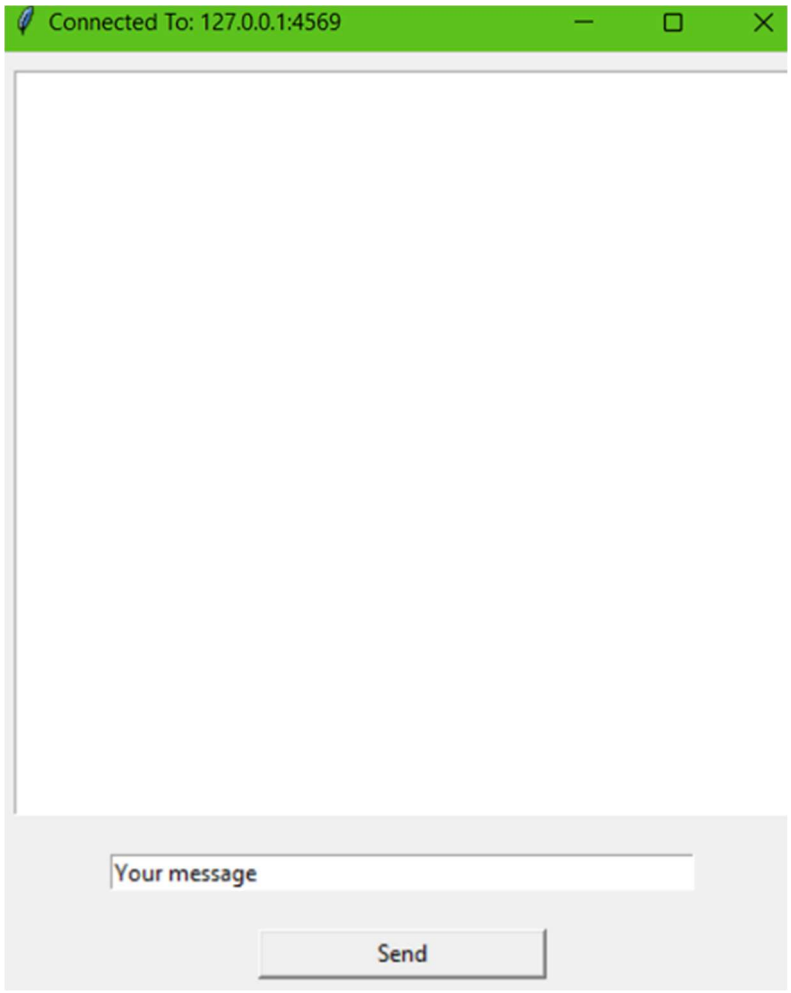
```
import socket

def server_program():
    # get the hostname
    host = "127.0.0.1"
    port = 4569  # initiate port no above 1024

    server_socket = socket.socket()  # get instance
    # look closely. The bind() function takes tuple as argument
    server_socket.bind((host, port))  # bind host address and port together

    # configure how many client the server can listen simultaneously
    server_socket.listen(2)
    conn, address = server_socket.accept()  # accept new connection
    print("Connection From: " + str(address))
    while True:
        # receive data stream. it won't accept data packet greater than 1024 bytes
        data = conn.recv(1024).decode()
        if not data:
            # if data is not received break
            break
        print("From Connected User: " + str(data))
        data = input(' -> ')
        conn.send(data.encode())  # send data to the client

    conn.close()  # close the connection


if __name__ == '__main__':
    server_program()
```

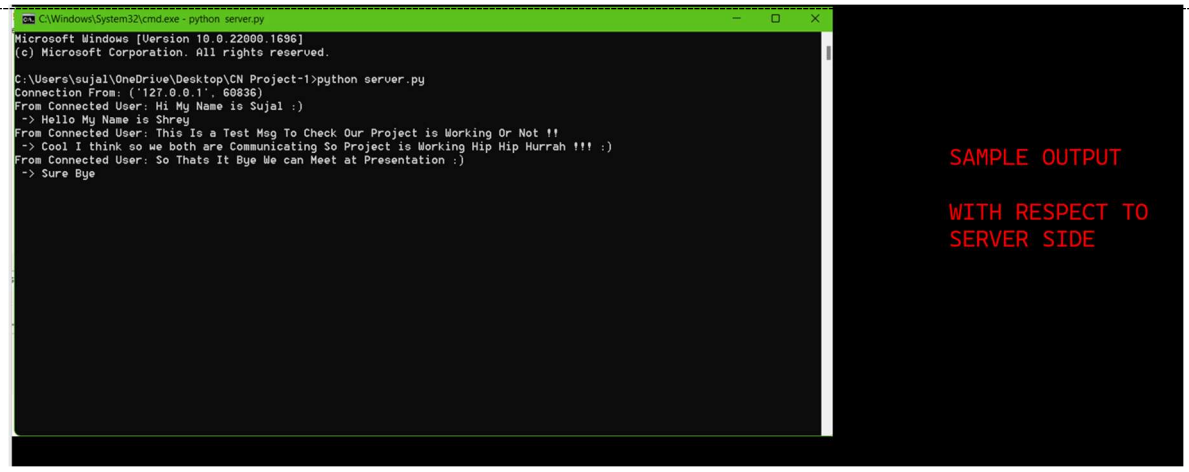
Our Client Code:

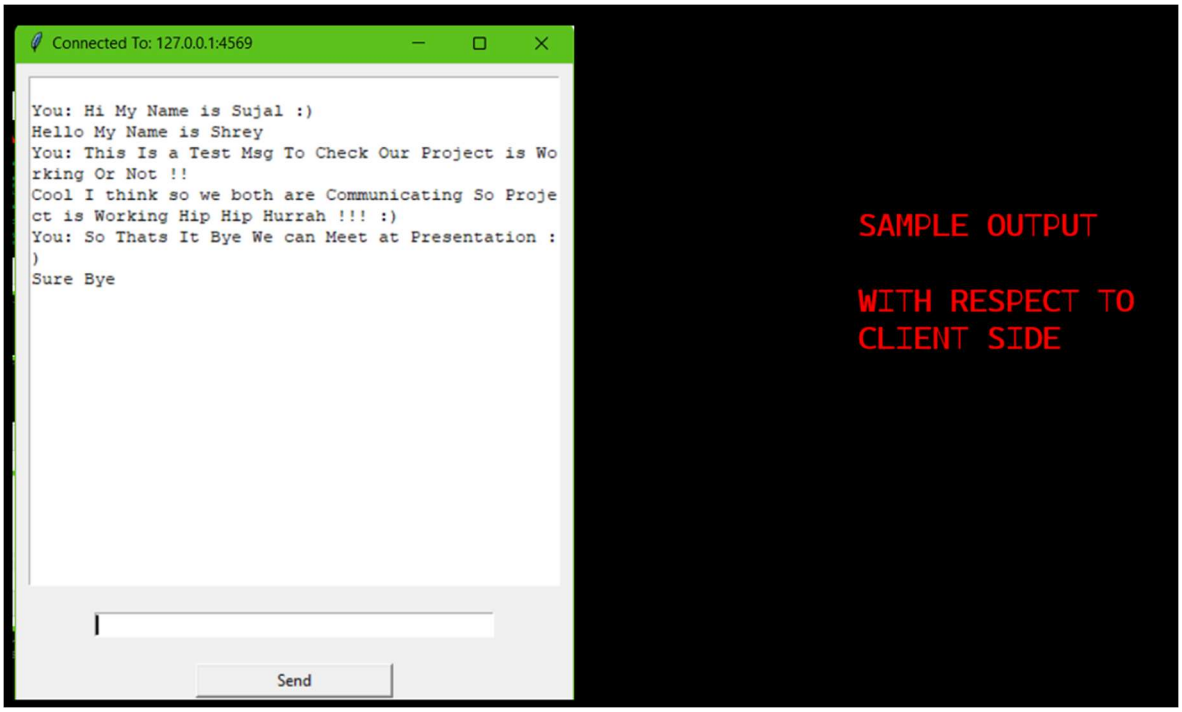
```
from socket import *
from threading import *
from tkinter import *

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

hostIp = "10.2.20.25"
portNumber = 4569

clientSocket.connect((hostIp, portNumber))

window = Tk()
window.title("Connected To: "+ hostIp+ ":"+str(portNumber))

txtMessages = Text(window, width=50)
txtMessages.grid(row=0, column=0, padx=10, pady=10)

txtYourMessage = Entry(window, width=50)
txtYourMessage.insert(0,"Your message")
txtYourMessage.grid(row=1, column=0, padx=10, pady=10)

def sendMessage():
    clientMessage = txtYourMessage.get()
    txtMessages.insert(END, "\n" + "You: "+ clientMessage)
    clientSocket.send(clientMessage.encode("utf-8"))

btnSendMessage = Button(window, text="Send", width=20, ceommand=sendMessage)
btnSendMessage.grid(row=2, column=0, padx=10, pady=10)

def recvMessage():
    while True:
        serverMessage = clientSocket.recv(1024).decode("utf-8")
        print(serverMessage)
        txtMessages.insert(END, "\n"+serverMessage)

recvThread = Thread(target=recvMessage)

recvThread.daemon = True
recvThread.start()

window.mainloop()
```

Our Output:


C:\Windows\System32\cmd.exe - python server.py
Microsoft Windows [Version 10.0.22000.1696]
(c) Microsoft Corporation. All rights reserved.
C:\Users\sujal\OneDrive\Desktop\CN Project-1>python server.py
C:\Windows\System32\cmd.exe - python Clientgui.py
C:\Users\sujal\OneDrive\Desktop\CN Project-1>python Clientgui.py
SERVER IS RUNNED BY THE ABOVE SYNTAX
CLIENT IS RUNNED BY THE ABOVE SYNTAX


Connected To: 127.0.0.1:4569
Your message
Send


C:\Windows\System32\cmd.exe - python server.py
Microsoft Windows [Version 10.0.22000.1696]
(c) Microsoft Corporation. All rights reserved.
C:\Users\sujal\OneDrive\Desktop\CN Project-1>python server.py
Connection From: ('127.0.0.1', 60836)
From Connected User: Hi My Name is Sujal :)
-> Hello My Name is Shrey
From Connected User: This Is a Test Msg To Check Our Project is Working Or Not !!
-> Cool I think so we both are Communicating So Project is Working Hip Hip Hurrah !!! :)
From Connected User: So Thats It Bye We can Meet at Presentation :)
-> Sure Bye
SAMPLE OUTPUT
WITH RESPECT TO
SERVER SIDE


Connected To: 127.0.0.1:4569
You: Hi My Name is Sujal :)
Hello My Name is Shrey
You: This Is a Test Msg To Check Our Project is Wo
rking Or Not !!
Cool I think so we both are Communicating So Proje
ct is Working Hip Hip Hurrah !!! :)
You: So Thats It Bye We can Meet at Presentation :
)
Sure Bye
Send
SAMPLE OUTPUT
WITH RESPECT TO
CLIENT SIDE