Assignment # 1 – Report
CS-3001 COMPUTER NETWORKING

Group Members:

Sohaib Akhter [20K-0292]

Abdul Hannan [20k-0389]

Section: BCS-6C

Submitted to:

Dr. Farrukh Salim Shaikh

FAST-NUCES Karachi

SOCKET PROGRAMMING:

Socket programming with TCP (Transmission Control Protocol) establishes communication between two devices over a network. In TCP, data is sent as a stream of bytes, with the protocol providing a reliable, ordered, and error-checked delivery of data. Here is a brief overview of how to use socket programming with TCP in Python:

1) Create a Host Socket on the server side with IP address “127.0.0.1”, and bind it to the port number given by the user.

```
def hostSocket():
    port=getport()

    if port==-1:
        return
    address="127.0.0.1"
    server=socket(AF_INET,SOCK_STREAM)
    #Here, AF_INET is the address family that is used for IPv4 addresses,
    #and SOCK_STREAM is the type of socket that is used for TCP connections.
    server.bind((address,port))
    server.listen(1)
    button_hostsocket.configure(state=DISABLED)
    input_portno.configure(state=DISABLED)
    mainthread=Thread(target=main,args=(server,5))
    mainthread.start()
```

2) On the client side, create a Socket and connect it to the server's IP address and port number.

```
def connectToSocket():

    address, port = getIPPort()

    client = socket(AF_INET, SOCK_STREAM)
    client.connect((address, port))

    button_connect.configure(state = DISABLED)
    text_PortIP.configure(state = DISABLED)
    input_sendMsg.configure(state = NORMAL)
    input_PortIP.configure(state = DISABLED)
    button_sendMsg.configure(state = NORMAL)
    text_connStatus.configure(text = 'STATUS: Connected')

    mainthread = Thread(target = main, args = (client, 5))

    text_receivedMsg.configure(text = 'Connected')
    mainthread.start()
```

3) On the server side, use the accept () method of the Server Socket to wait for an incoming connection from the client. This method blocks the program until a connection is made.

```
connection,address=server.accept()
```

4) On the server side, we use “SendMsg()” and “receiveMsg()” to send and receive messages to and from client while on the client side, we use “sendMsg()” and “receiveMsg()” methods for the same purpose.

Server:

```
def SendMsg(connection,address):
```

```
def receiveMsg(connection,address):
```

Client:

```
def sendMsg(connection, add):
```

```
def receiveMsg(connection, add):
```

6) Once communication is complete, close the Socket objects on both the client and server sides by entering “!” symbol on either side as a message.

```
if re:
    text_recvMsg.configure(text=f'CLIENT:{re}')
    if re=="!":
        global flag
        flag=1
        re+='!'
        connection.close()
        break
```

```
if re:
    text_receivedMsg.configure(text = f'Server: {re}')
    if re == '!':
        global flag
        flag = 1
        re += '!'
        connection.close()
        break
```

7) For simultaneous communication between the server and client we have used two threads.

Server:

```
thread1=Thread(target=SendMsg,args=(connection,address))
thread2=Thread(target=receiveMsg,args=(connection,address))

thread1.start()
thread2.start()
thread1.join()
thread2.join()
```

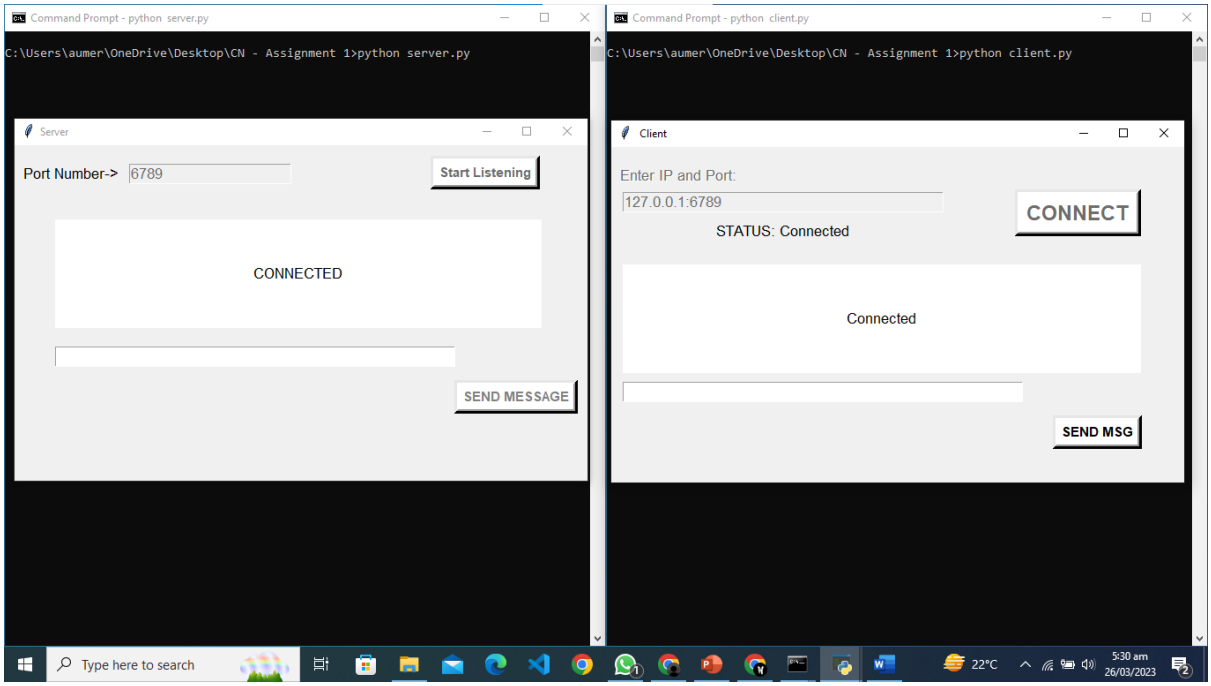
Client:

```
thread1 = Thread(target = sendMsg, args = (client, address))
thread2 = Thread(target = receiveMsg, args = (client, address))

thread1.start()
thread2.start()

thread1.join()
thread2.join()
```

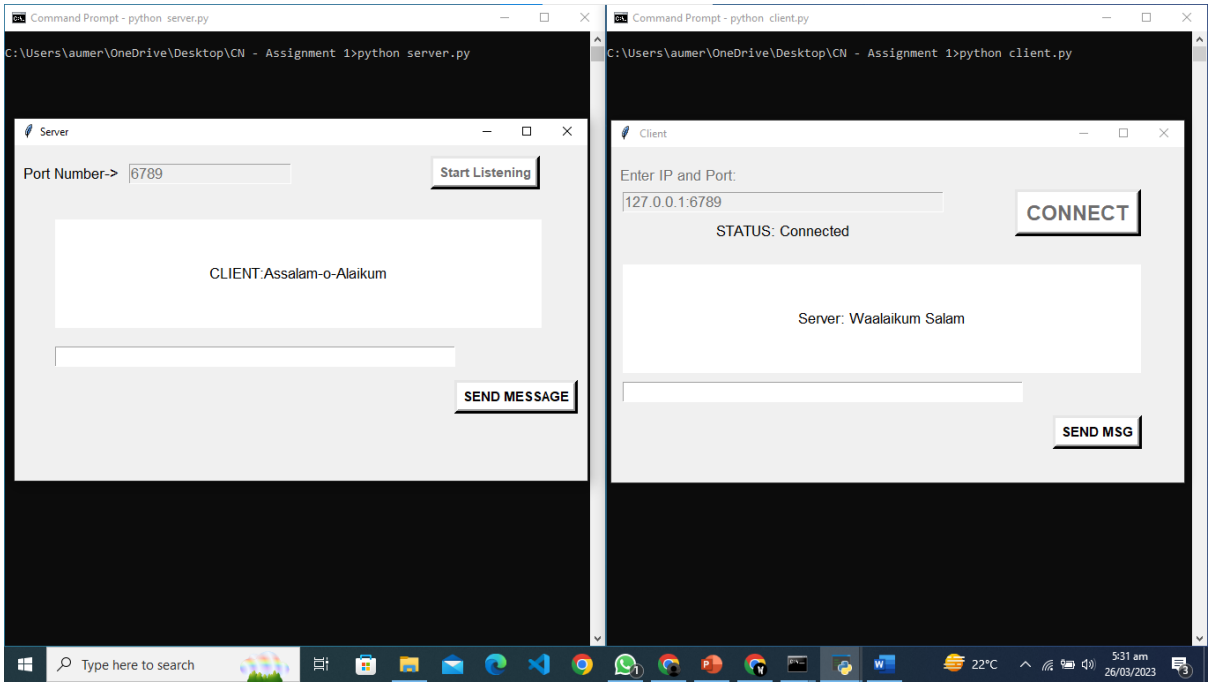
DEMO:

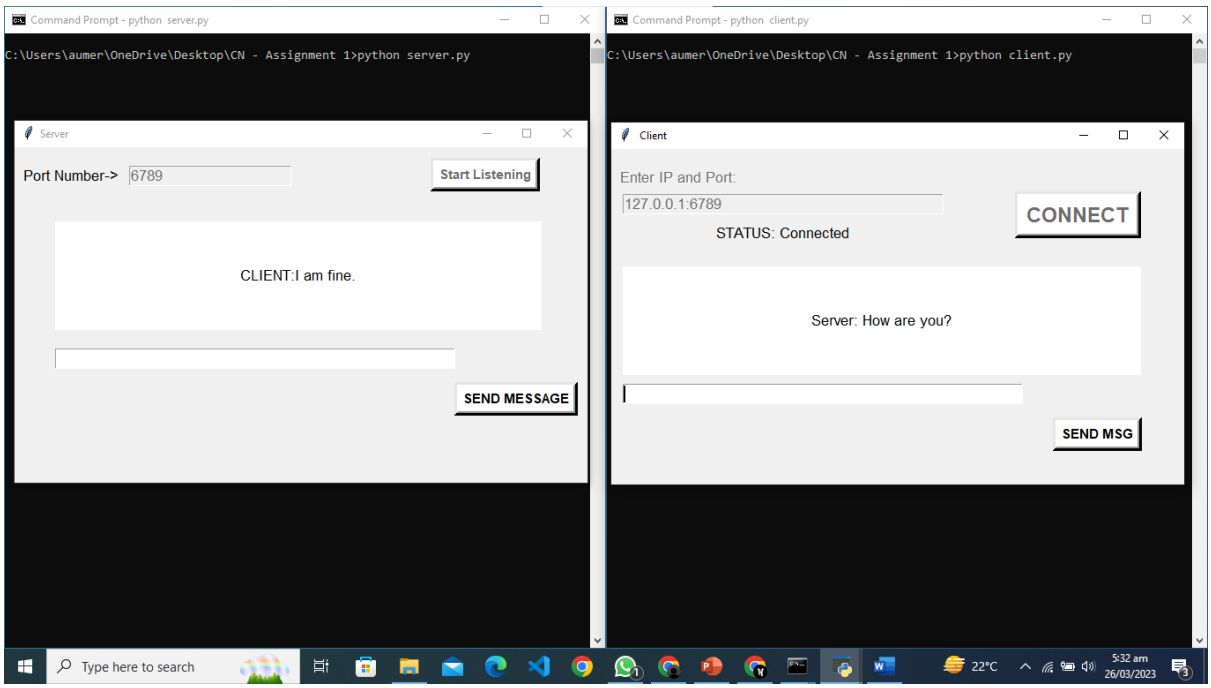
We use the Tkinter library of Python to create a simple GUI for the Client Server Application. The purpose of GUI is to make it easier for the user to understand the flow of the work.

1) Run both the python files and on the server side, enter the port number and click “Start Listening” then on the client side enter the IP address and port number “127.0.0.1: port number”.

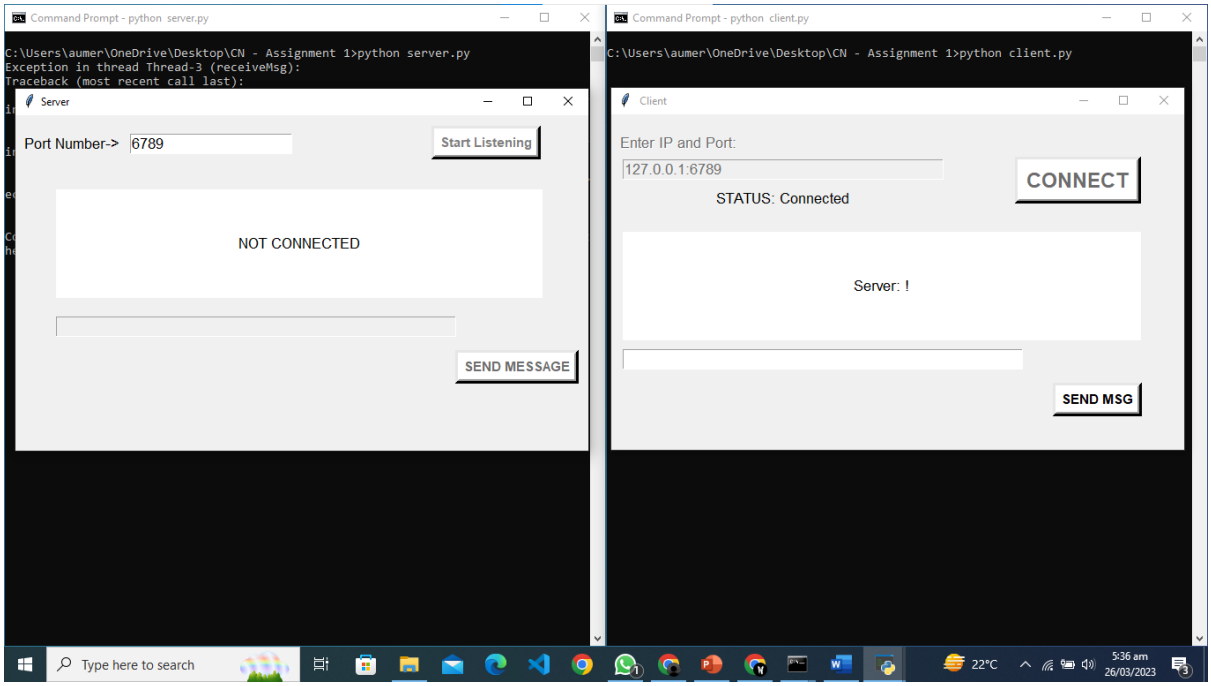


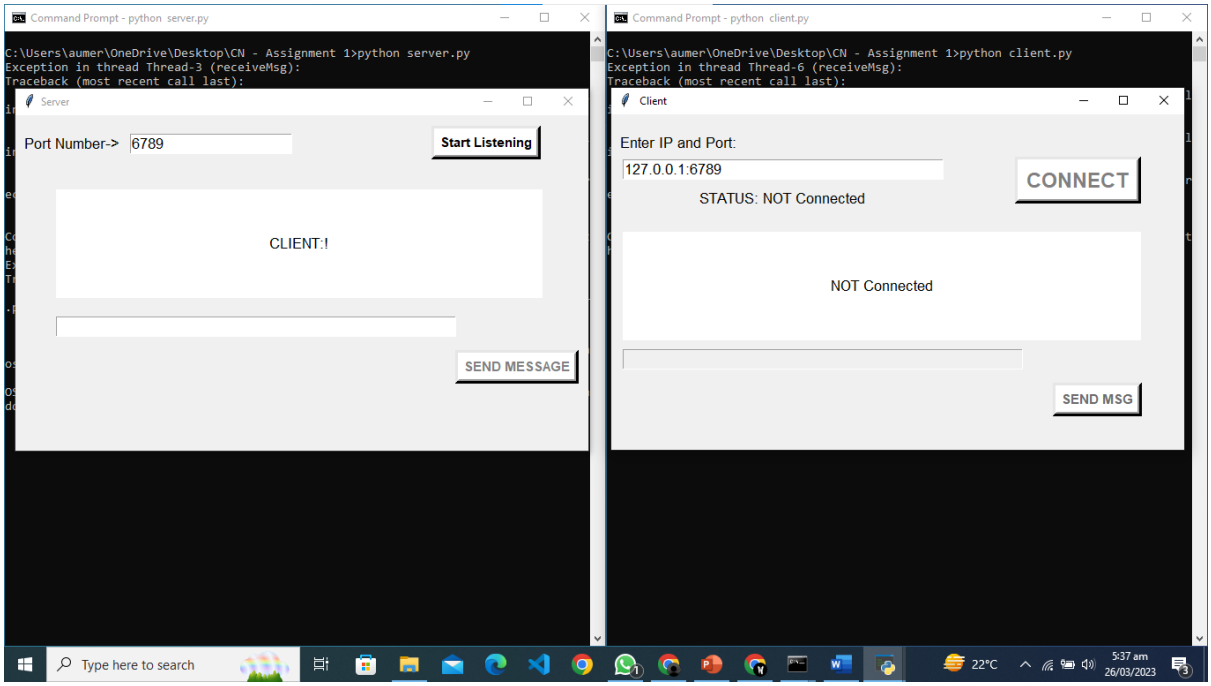
2) Now they can communicate with each other.





3) To end the communication, enter “!” as the message.


Command Prompt - python server.py
C:\Users\aumer\OneDrive\Desktop\CN - Assignment 1>python server.py
Exception in thread Thread-3 (receiveMsg):
Traceback (most recent call last):
Server
Port Number-> 6789
Start Listening
NOT CONNECTED
SEND MESSAGE
Command Prompt - python client.py
C:\Users\aumer\OneDrive\Desktop\CN - Assignment 1>python client.py
Client
Enter IP and Port:
127.0.0.1:6789
STATUS: Connected
CONNECT
Server: !
SEND MSG
Type here to search
22°C
5:36 am
26/03/2023


Command Prompt - python server.py
C:\Users\aumer\OneDrive\Desktop\CN - Assignment 1>python server.py
Exception in thread Thread-3 (receiveMsg):
Traceback (most recent call last):
Server
Port Number-> 6789
Start Listening
CLIENT:!
SEND MESSAGE
Command Prompt - python client.py
C:\Users\aumer\OneDrive\Desktop\CN - Assignment 1>python client.py
Exception in thread Thread-6 (receiveMsg):
Traceback (most recent call last):
Client
Enter IP and Port:
127.0.0.1:6789
STATUS: NOT Connected
CONNECT
NOT Connected
SEND MSG
Type here to search
22°C
5:37 am
26/03/2023