St. Francis Institute of Technology, Mumbai-400 103

A.Y. 2023-24

Class: SE-ITA/ITB, Semester: IV

Subject: Python Lab.

Experiment – 14: Python program to implement Client Server Communication using TCP and UDP Sockets.

1. **Aim:** To implement a python program to demonstrate the following:
   a. Client-Server Chat Application using TCP.
   b. Client-Server Chat Application using UDP.

2. **Prerequisite:**. Knowledge of Networking in python

3. **Objective:** Knowledge of Web Programming and Client Server Communication using Sockets

4. **Requirements:** Personal Computer (PC), Windows /Linux Operating System, IDLE 3.6 for Python3.

5. **Pre-Experiment Exercise:**

Theory:

- **Socket Programming:**

  Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket (node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server. They are the real backbones behind web browsing. Socket programming is started by importing the socket library and making a simple socket.

  ```
  import socket
  s = socket. socket (socket.AF_INET, socket.SOCK_STREAM)
  ```

  Here we made a socket instance and passed it two parameters. The first parameter is **AF_INET** and the second one is **SOCK_STREAM**. AF_INET refers to the address family ipv4. The SOCK_STREAM means connection oriented TCP protocol.

- **TCP Sockets:**

TCP/IP socket are connected, communication is bi-directional.

```
import socket
import sys
sock = socket.socket (socket.AF_INET,
socket.SOCK_STREAM)
```

- **UDP Sockets:**

  UDP or user datagram protocol is an alternative protocol to its more common counterpart TCP. UDP like TCP is a protocol for packet transfer from 1 host to another, but has some important differences. UDP is a connectionless and non-stream oriented protocol. It means a UDP server just catches incoming packets from any and many hosts without establishing a reliable pipe kind of connection.

  A UDP socket is created like this:

```
import socket
import sys
s=socket.socket(socket.AF_INET,

socket.SOCK_DGRAM)
```

5. Laboratory Exercise

A. Procedure

i. Open Idle for python
ii. Open editor in Idle from menu file-new
iii. Type python code with proper syntax
iv. Save file with .py extension
v. Execute the code inside the saved file using shortcut key F5 or using menu: Run-Run module

B. Program code with comments:

Write and execute your program code to achieve the given aim and attach it **with your own comments with neat indentation**.

6. Post-Experiments Exercise

A. Extended Theory:

1. Explain File Server and File Client in Python.

B. Questions/Programs:

1. Write a Python program to copy a file from client to the server using TCP Sockets.

C. Conclusion:

1. Write what was performed in the experiment/program.
2. What is the significance of experiment/program?

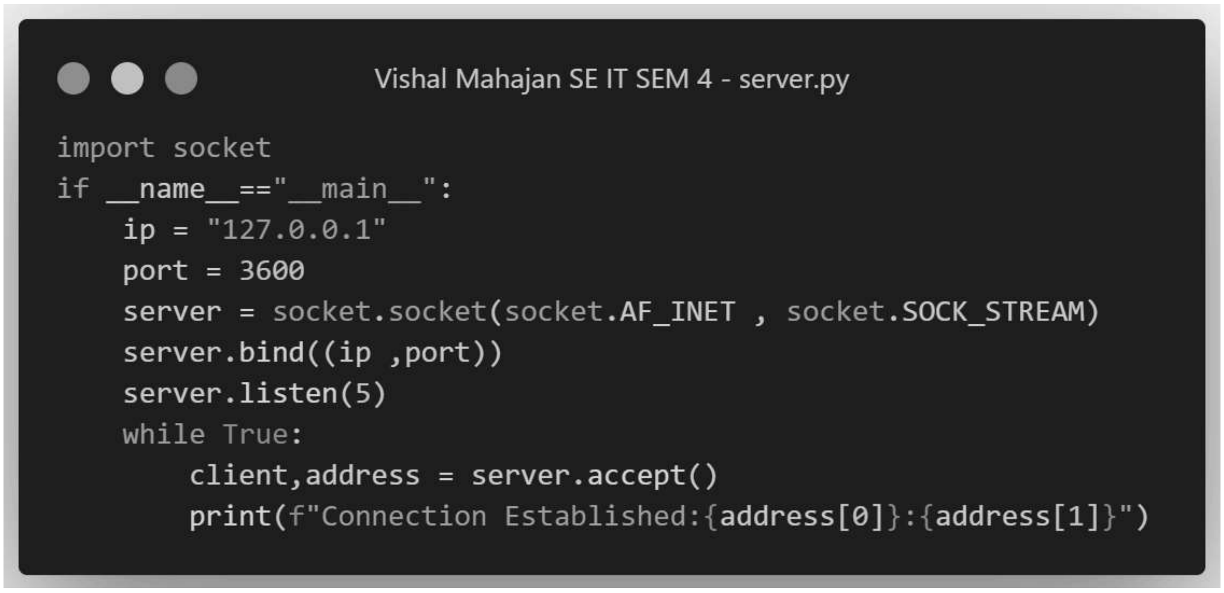
D. References

[1]James Payne, ”Beginning Python: Using Python 2.6 and Python 3.1”,WroxPublication.
[2]Dr.Nageswara Rao,”Core Python Programming”,Wiley Publication.
[3] https://www.python.org/
[4] www.pythonforbeginners.com

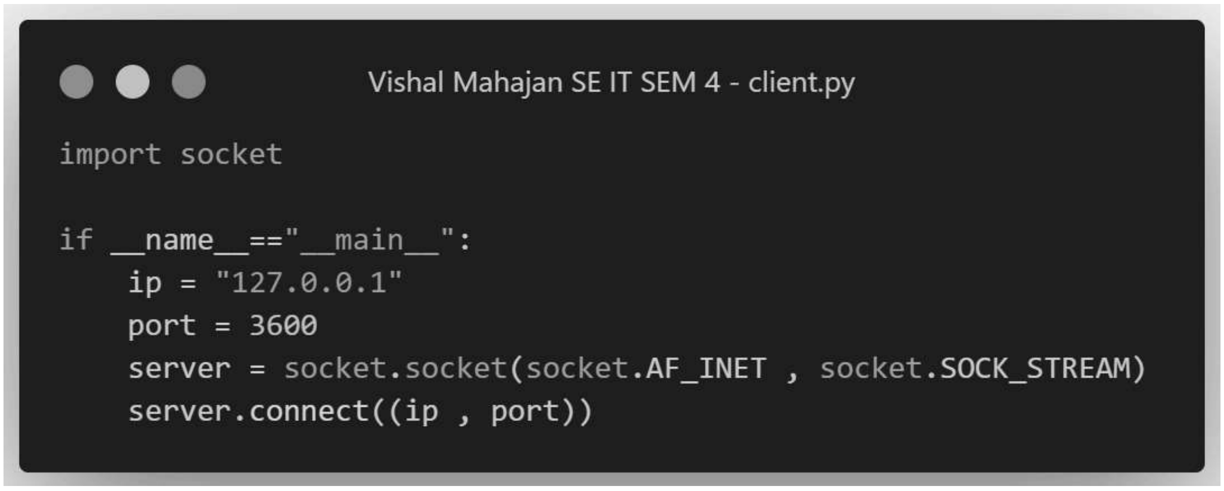
In- Lab Exercise:

A. Client-Server Chat Application using TCP.

1. To connect from server to client

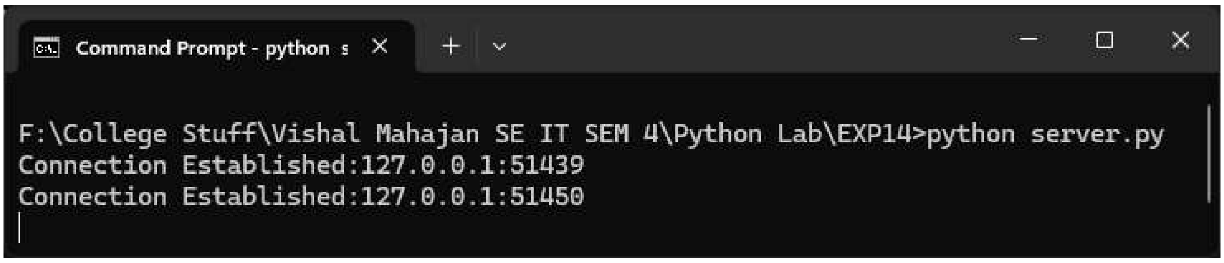

Vishal Mahajan SE IT SEM 4 - server.py

```python
import socket
if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    server = socket.socket(socket.AF_INET , socket.SOCK_STREAM)
    server.bind((ip ,port))
    server.listen(5)
    while True:
        client,address = server.accept()
        print(f"Connection Established:{address[0]}:{address[1]}")
```

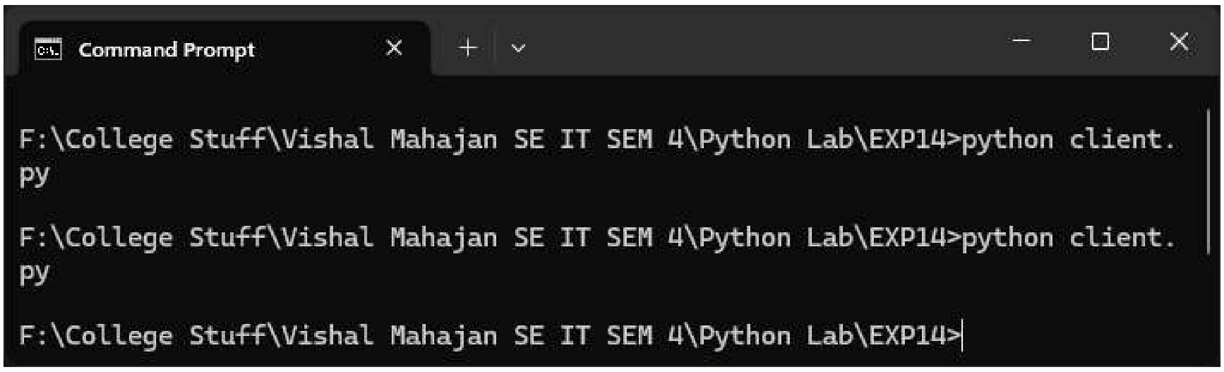

Vishal Mahajan SE IT SEM 4 - client.py

```python
import socket

if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    server = socket.socket(socket.AF_INET , socket.SOCK_STREAM)
    server.connect((ip , port))
```

Output:

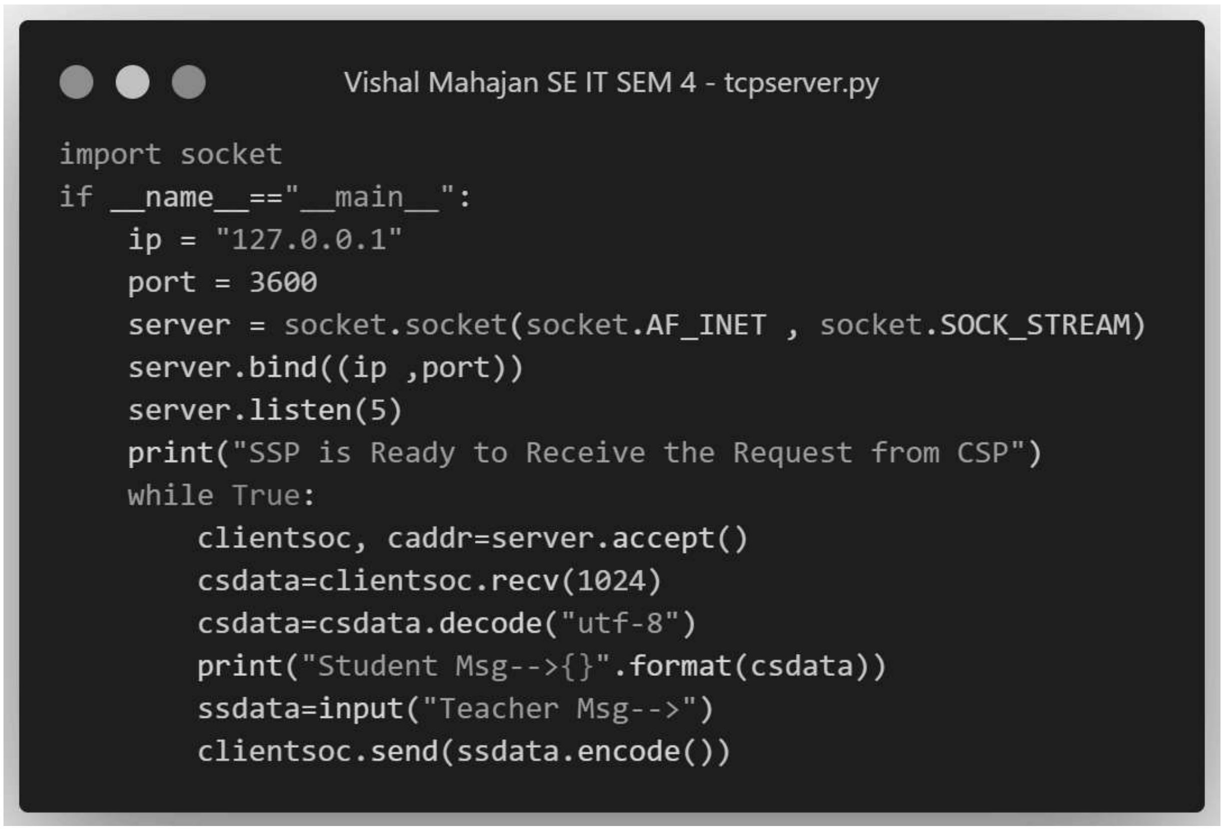


```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python server.py
Connection Established:127.0.0.1:51439
Connection Established:127.0.0.1:51450
```

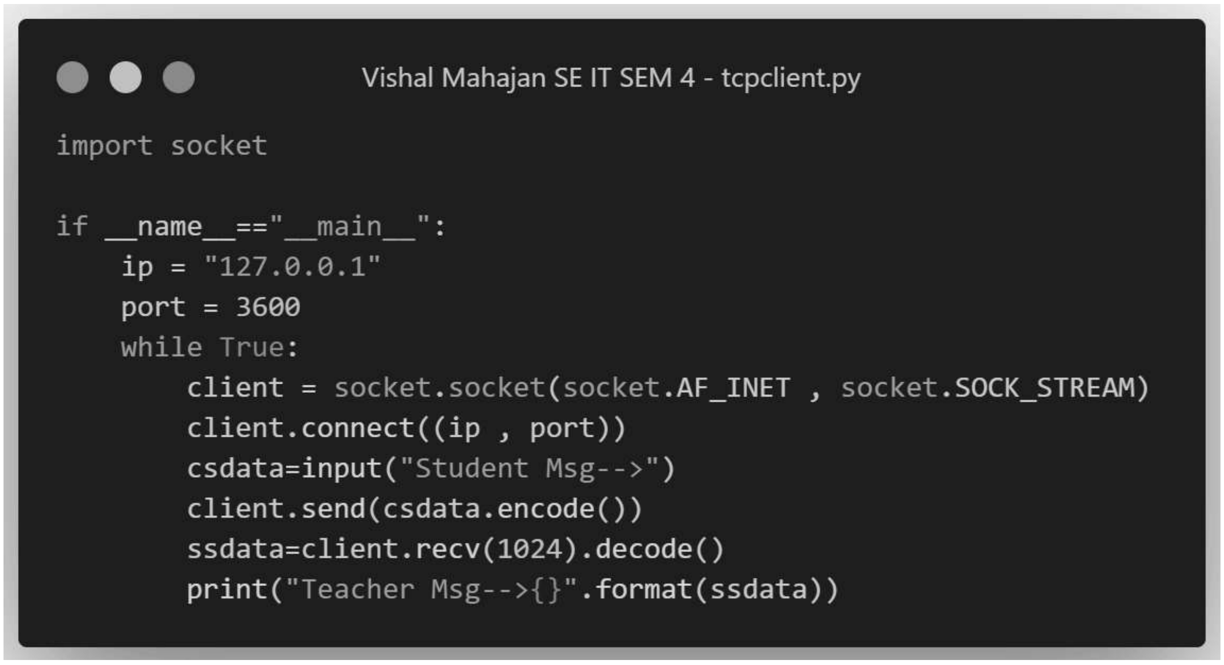


```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python client.
py

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python client.
py

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>
```

2. Chatting Using Earlier created Connection

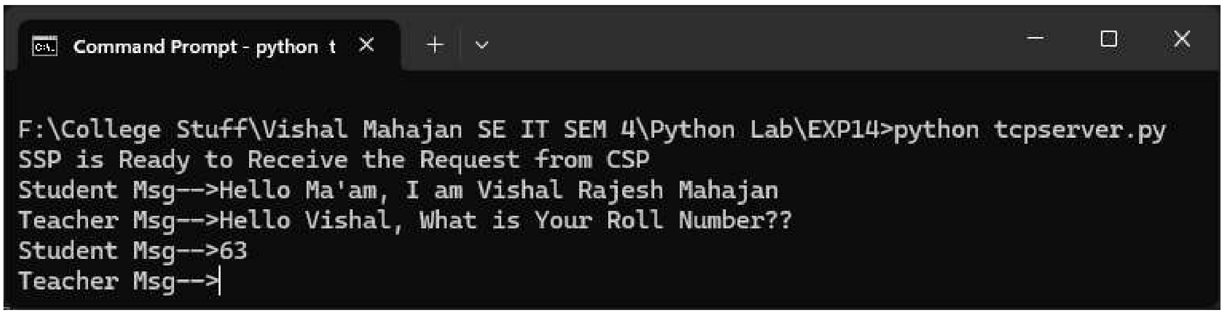


```python
import socket
if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    server = socket.socket(socket.AF_INET , socket.SOCK_STREAM)
    server.bind((ip ,port))
    server.listen(5)
    print("SSP is Ready to Receive the Request from CSP")
    while True:
        clientsoc, caddr=server.accept()
        csdata=clientsoc.recv(1024)
        csdata=csdata.decode("utf-8")
        print("Student Msg-->{}".format(csdata))
        ssdata=input("Teacher Msg-->")
        clientsoc.send(ssdata.encode())
```

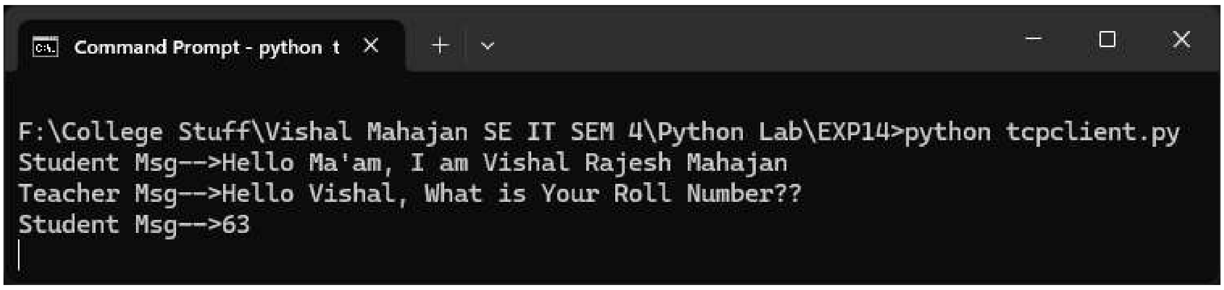

Vishal Mahajan SE IT SEM 4 - tcpclient.py

```python
import socket

if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    while True:
        client = socket.socket(socket.AF_INET , socket.SOCK_STREAM)
        client.connect((ip , port))
        csdata=input("Student Msg-->")
        client.send(csdata.encode())
        ssdata=client.recv(1024).decode()
        print("Teacher Msg-->{}".format(ssdata))
```

Output of Chatting Server:

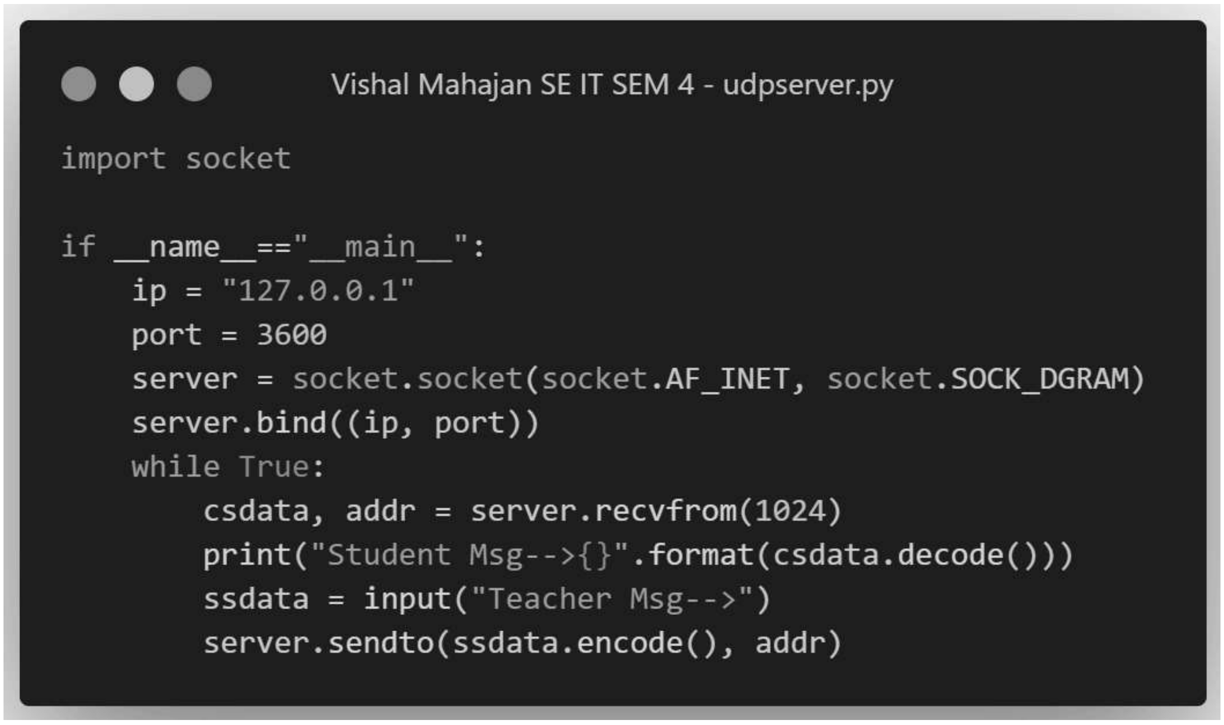


```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python tcpserver.py
SSP is Ready to Receive the Request from CSP
Student Msg-->Hello Ma'am, I am Vishal Rajesh Mahajan
Teacher Msg-->Hello Vishal, What is Your Roll Number??
Student Msg-->63
Teacher Msg-->
```

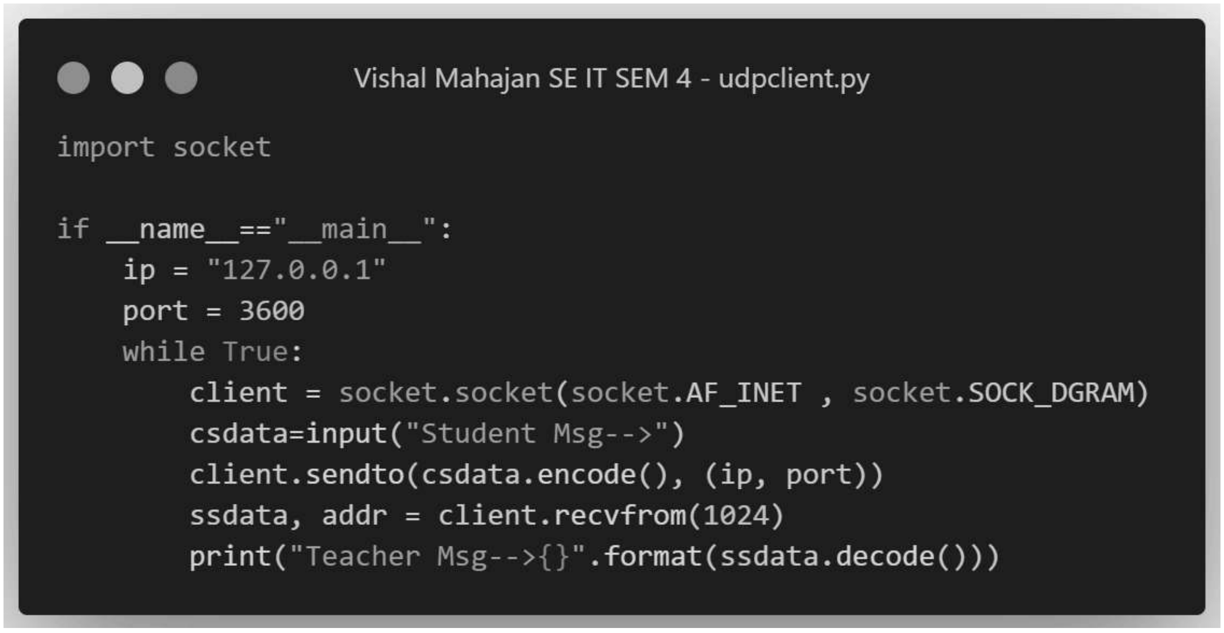


```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python tcpclient.py
Student Msg-->Hello Ma'am, I am Vishal Rajesh Mahajan
Teacher Msg-->Hello Vishal, What is Your Roll Number??
Student Msg-->63
```

B. Client-Server Chat Application using UDP.

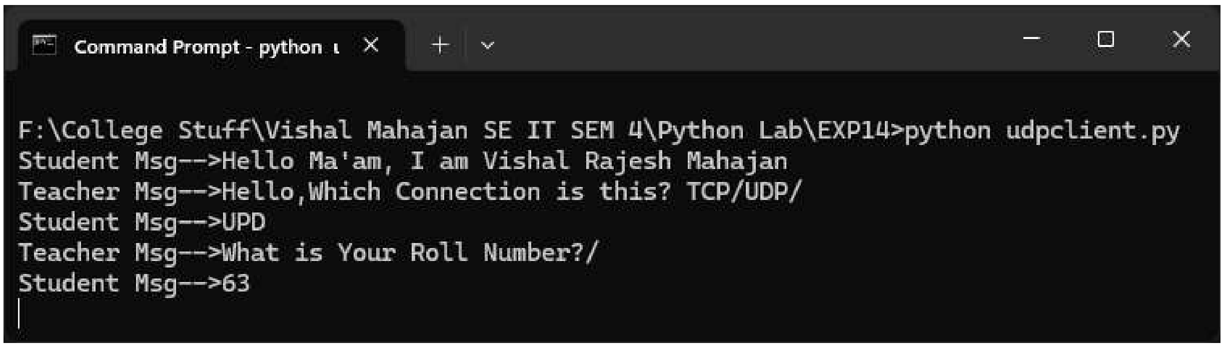

Vishal Mahajan SE IT SEM 4 - udpserver.py

```python
import socket

if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    server.bind((ip, port))
    while True:
        csdata, addr = server.recvfrom(1024)
        print("Student Msg-->{}".format(csdata.decode()))
        ssdata = input("Teacher Msg-->")
        server.sendto(ssdata.encode(), addr)
```

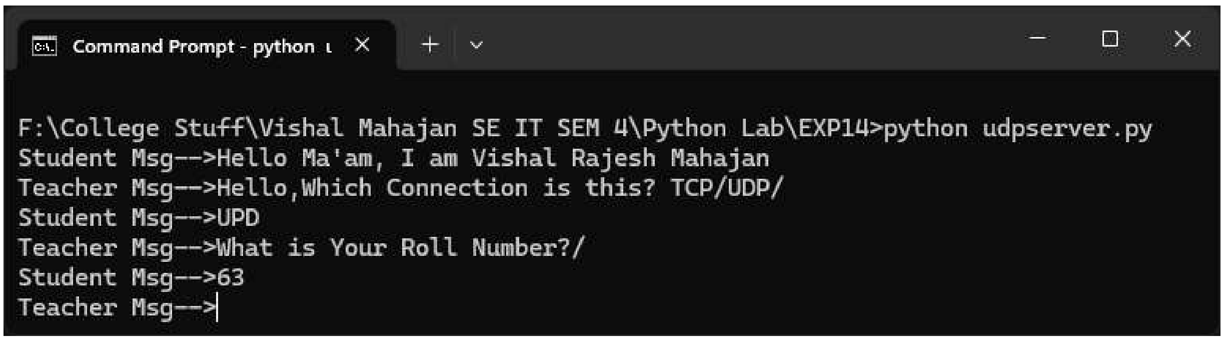

Vishal Mahajan SE IT SEM 4 - udpclient.py

```python
import socket

if __name__=="__main__":
    ip = "127.0.0.1"
    port = 3600
    while True:
        client = socket.socket(socket.AF_INET , socket.SOCK_DGRAM)
        csdata=input("Student Msg-->")
        client.sendto(csdata.encode(), (ip, port))
        ssdata, addr = client.recvfrom(1024)
        print("Teacher Msg-->{}".format(ssdata.decode()))
```

Output:



```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python udpclient.py
Student Msg-->Hello Ma'am, I am Vishal Rajesh Mahajan
Teacher Msg-->Hello,Which Connection is this? TCP/UDP/
Student Msg-->UPD
Teacher Msg-->What is Your Roll Number?/
Student Msg-->63
```



```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python udpserver.py
Student Msg-->Hello Ma'am, I am Vishal Rajesh Mahajan
Teacher Msg-->Hello,Which Connection is this? TCP/UDP/
Student Msg-->UPD
Teacher Msg-->What is Your Roll Number?/
Student Msg-->63
Teacher Msg-->
```

Post-Experiment Exercise:

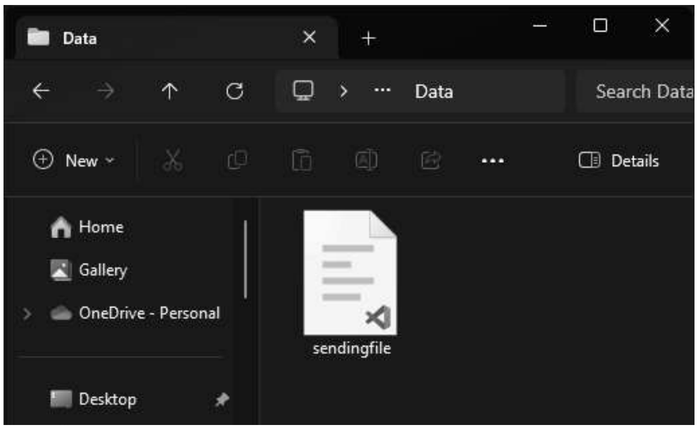
Write a Python program to copy a file from client to the server using TCP Sockets.

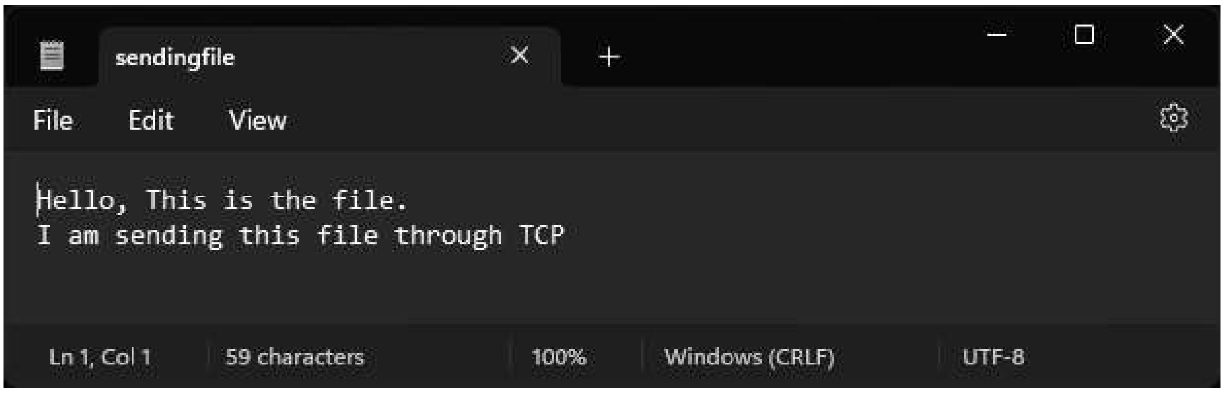
```
# File: FileTransferServer.py
import socket

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind(('localhost', 12345))
    server_socket.listen(1)
    print("Server is started and listening for the client to connect")

    client_socket, addr = server_socket.accept()
    print(f"Client {addr} connected")

    with open('./Data/recievedfile.txt', 'wb') as file:
        print("File opened")
        while True:
            data = client_socket.recv(1024)
            if not data:
                break
            file.write(data)

    print("File has been received successfully.")
    client_socket.close()

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

```
# File: FileTransferClient.py
import socket

def start_client():
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect(('localhost', 12345))
    print("Connected to the server...")
    with open('.\Data\sendingfile.txt', 'rb') as file:
        print("Sending file...")
        data = file.read(1024)
        while data:
            client_socket.send(data)
            data = file.read(1024)

    print("File has been sent successfully.")
    client_socket.close()

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

Output:

Before Sending File:

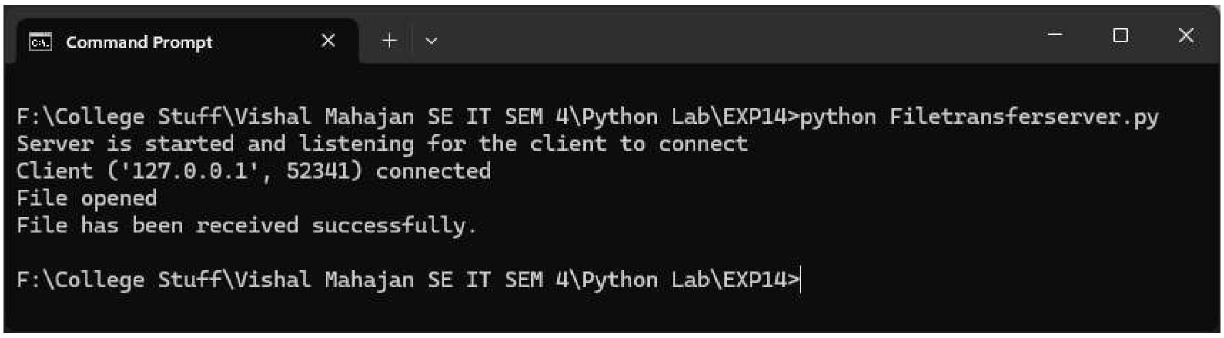




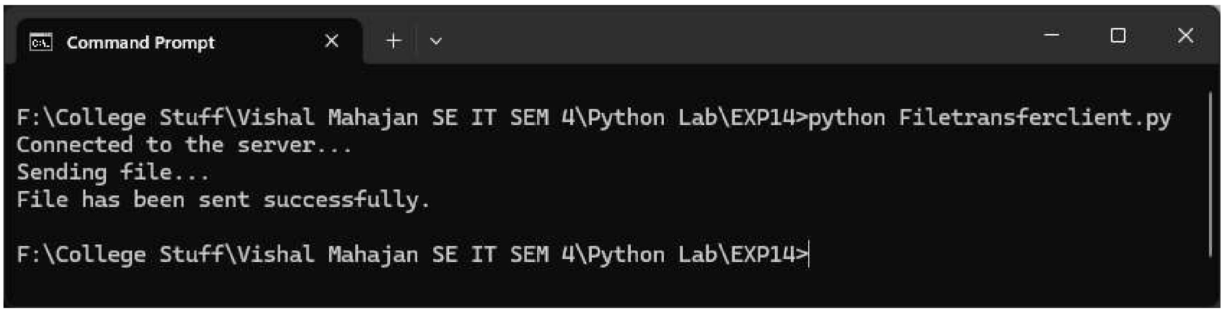
```
Hello, This is the file.
I am sending this file through TCP
```

Ln 1, Col 1 | 59 characters | 100% | Windows (CRLF) | UTF-8

Sending File:

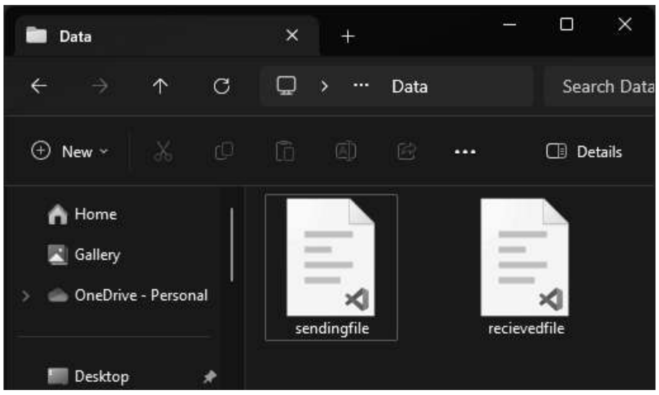


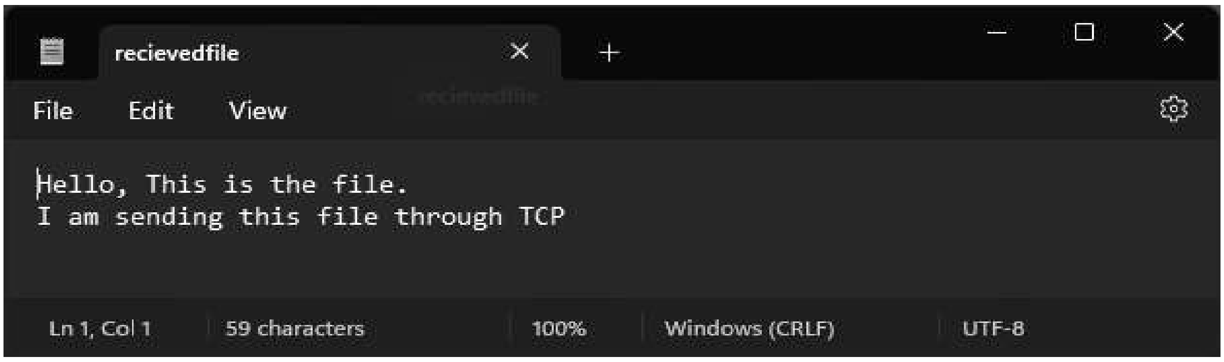
```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python Filetransferserver.py
Server is started and listening for the client to connect
Client ('127.0.0.1', 52341) connected
File opened
File has been received successfully.

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>
```



```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>python Filetransferclient.py
Connected to the server...
Sending file...
File has been sent successfully.

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP14>
```

After Sending File:





```
Hello, This is the file.
I am sending this file through TCP
```

Ln 1, Col 1 | 59 characters | 100% | Windows (CRLF) | UTF-8