SERVER PROGRAM

```python
import socket
import threading


def client_conn(client, address):
    client_name = client.recv(1024).decode()
    print("\nconnected with", client_name, address)
    print(f"Active Connections : {threading.activeCount()-1}")
    client.send(bytes(f"Data sent from server to {client_name}", "utf-8"))
    if client.recv(1024).decode() == 'X' or 'x':
        print(f"\n{address} {client_name} Disconnected")
        print(f"Active Connections : {threading.activeCount()-2}")
        client.close()


def start():
    s.listen(3)  # can connect utmost to 3 clients
    while True:
        c, add = s.accept()
        thread = threading.Thread(target=client_conn, args=(c, add))
        thread.start()
```

```python


s = socket.socket()  # by default ipv4 and TCP
print("Socket created\n")
s.bind(('localhost', 9999))  # socket = ip + port_no
print("Waiting for connection\n")
start()
```

CLIENT PROGRAM

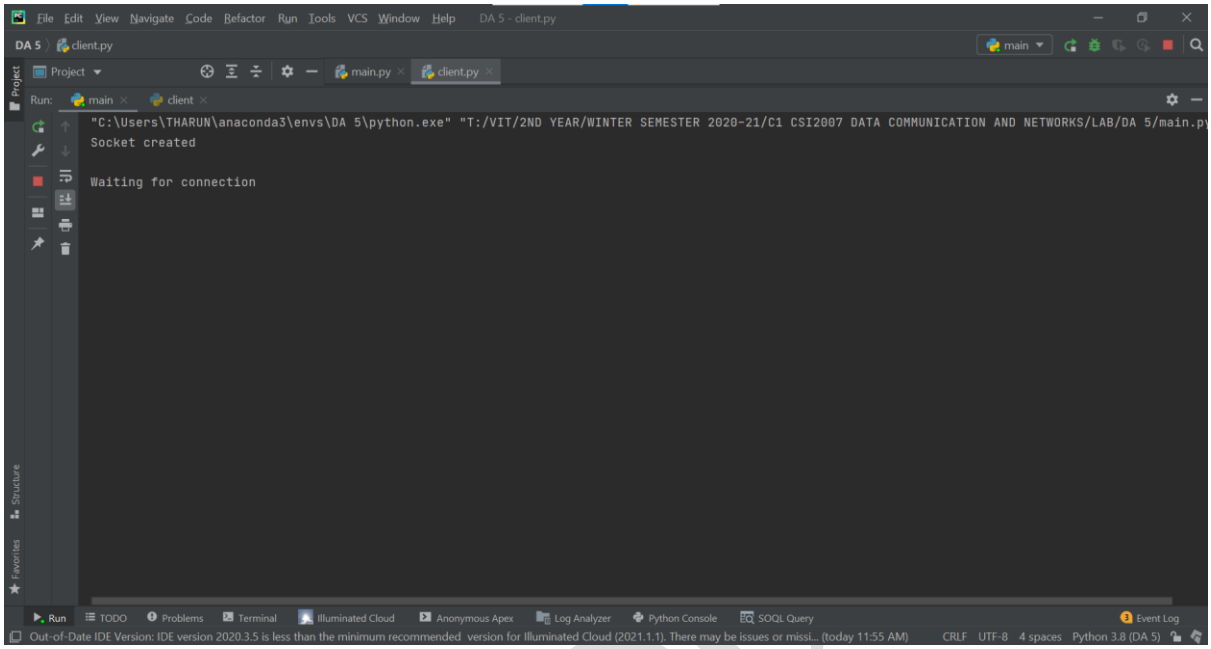
```python
import socket

c = socket.socket()
c.connect(('localhost', 9999))
name = input("Enter name of the client : ")
c.send(bytes(name, 'utf-8'))

print("\nData received from server ==> ", end="")
print(c.recv(1024).decode())
print("Press X to disconnect : ", end='')
if input() == 'X' or 'x':
    c.send(bytes(f"{name} Disconnected", 'utf-8'))
```

OUTPUT

Initially server doesn't have any active client connections



Three clients will be connecting to the server. The corresponding output in the consoles of server as well as client will be shown in each step.

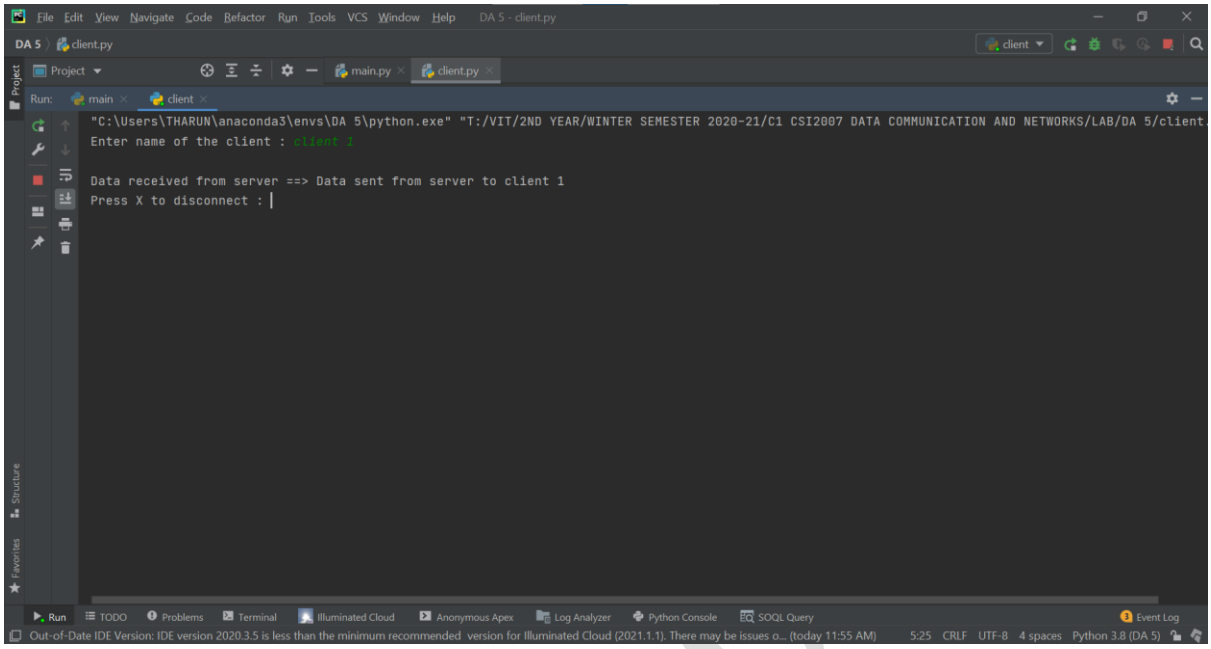
All the three clients have been manipulated as localhost. And server has also been manipulated as localhost. So, in a single machine all the three clients have been connected to server which itself is running in the same machine.

One client is running in terminal of PyCharm IDE itself and the other two clients are made to run in two instances of Command Prompt. Client is running in terminal of PyCharm IDE.

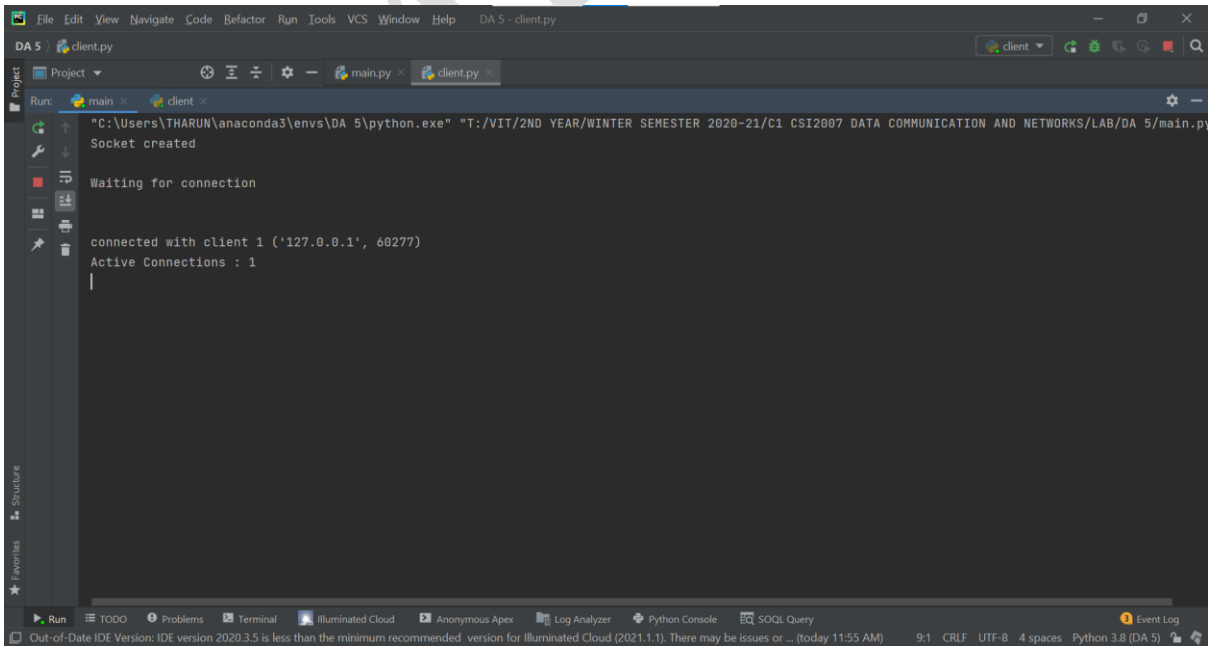
All the host and server will have same ip address as all are running in a local host. In case of different machine, the host must connect to the ip address of the server. Here clients got connected to the sever by specifying 'localhost' instead of specifying the ip address of the server.

Server-Host connection has been established. Now the protocol exchanges the message. Client communicates the "client name" to the sever and the server communicates "Data sent from server to {client name}" to the client.

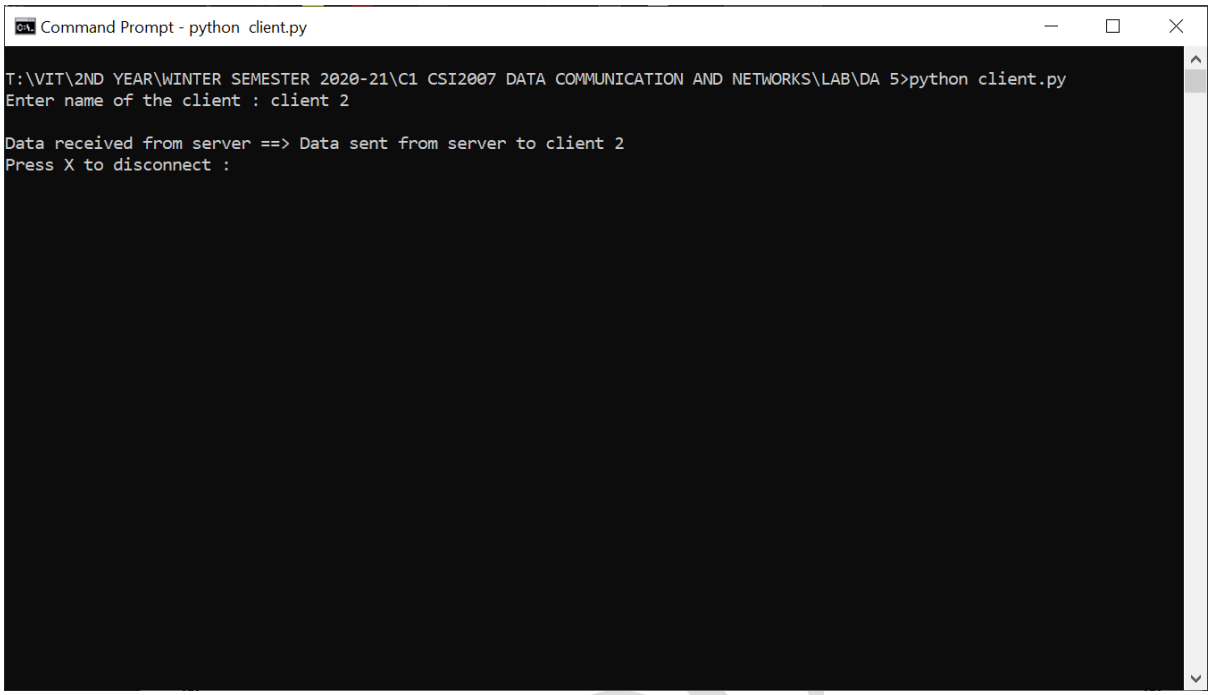
Now “client 1” gets connected. It does not wish to disconnect. So, it’ll remain connected unless client presses ‘X’ or ‘x’ to close the connection



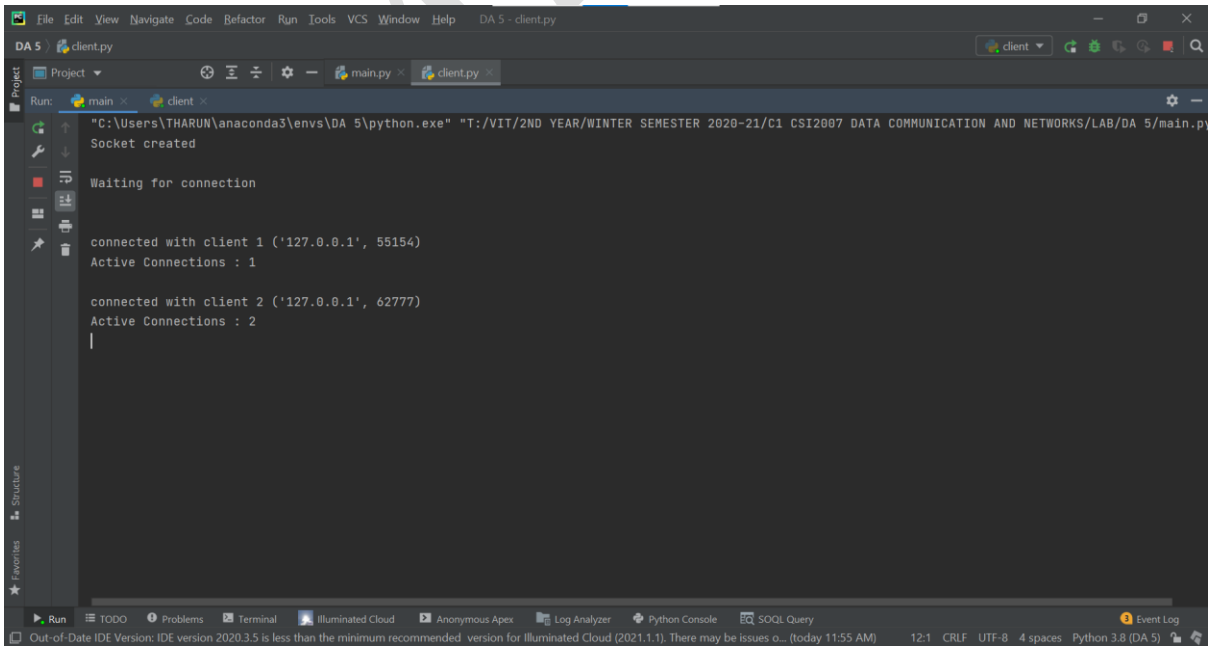
Now the server output is



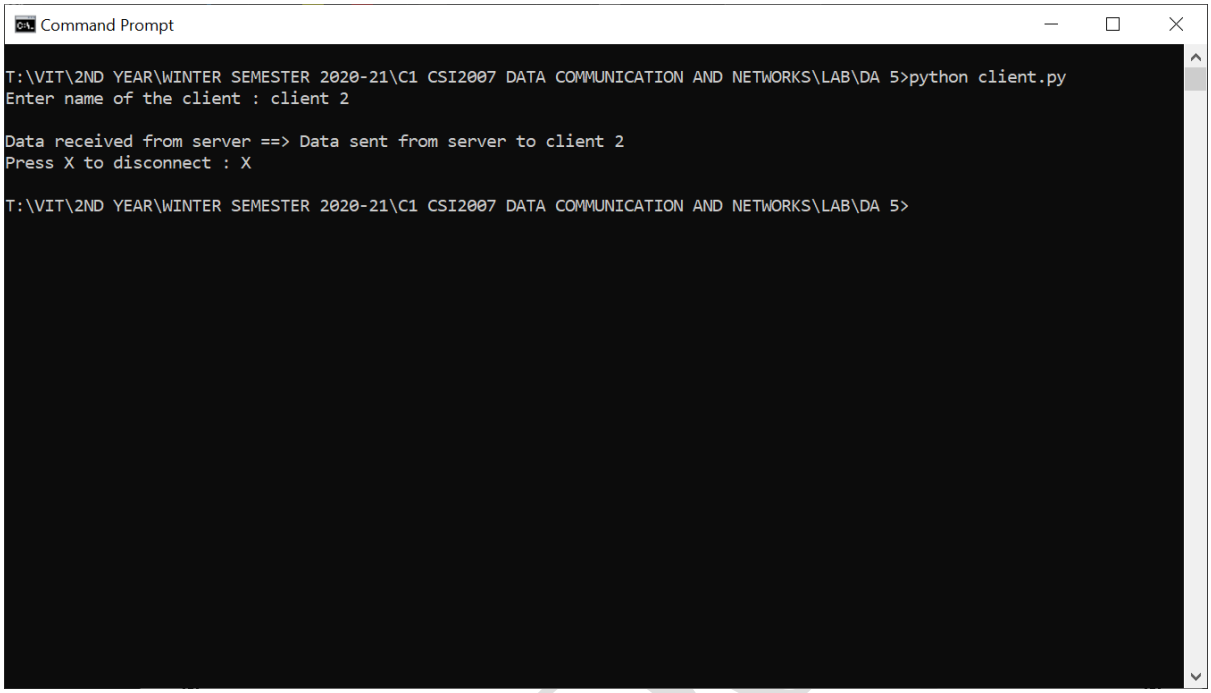
Now “client 2” gets connected.



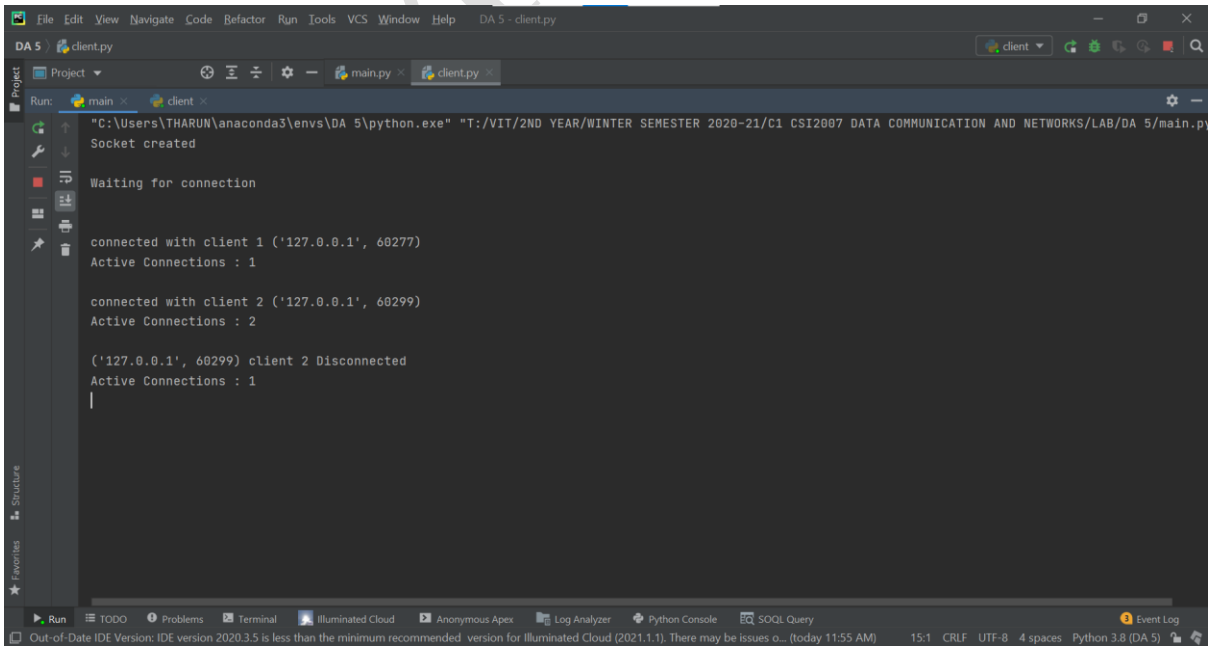
And the server output is



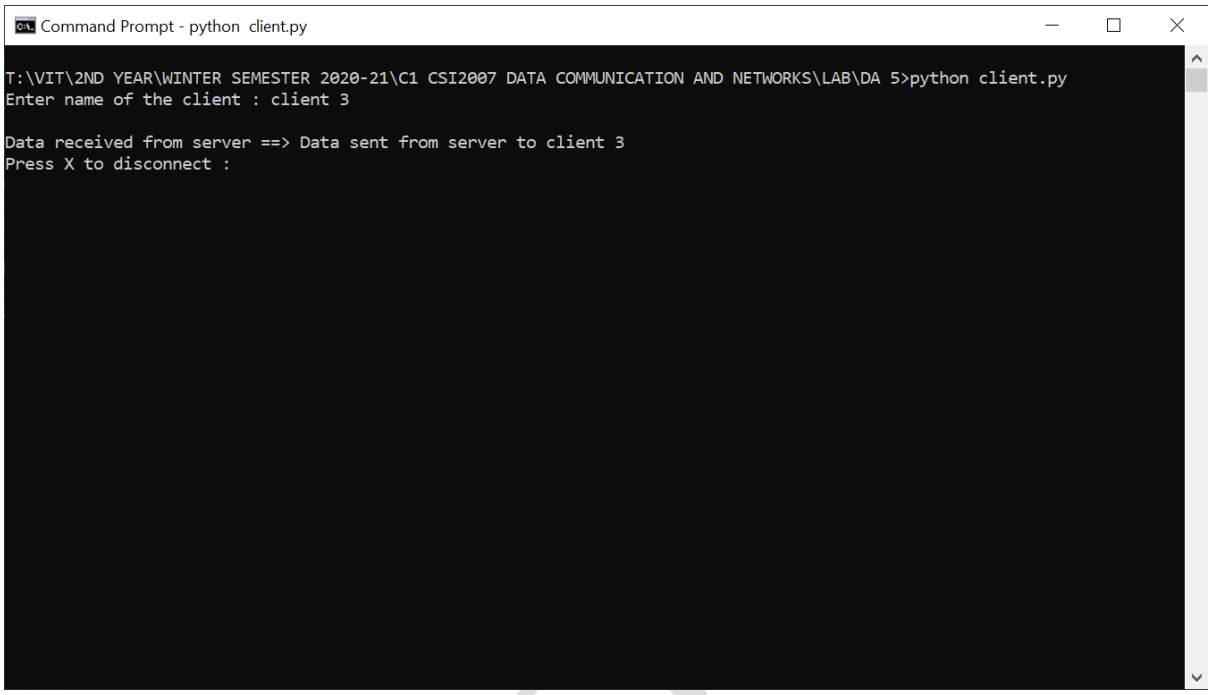
Now “client 2” wishes to get disconnected and presses ‘X’ in its console. The continuation of its output is shown below



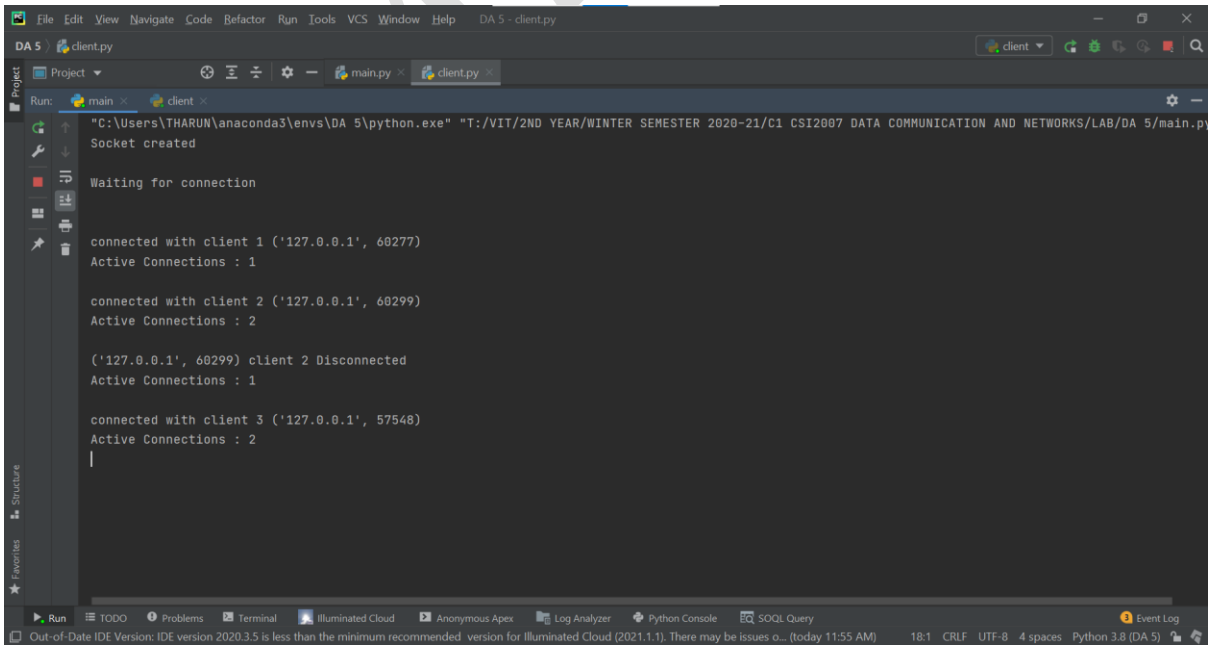
Now the server output console is updated as shown below



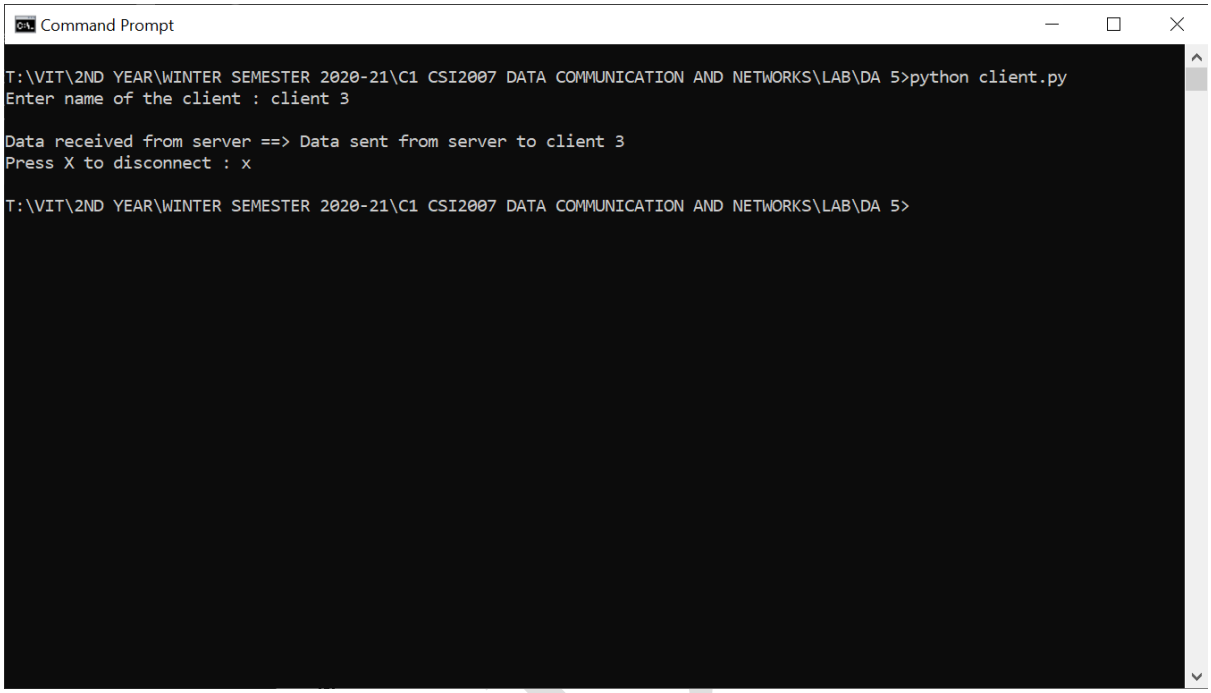
Now “client 3” gets connected to server and the client console output is shown below



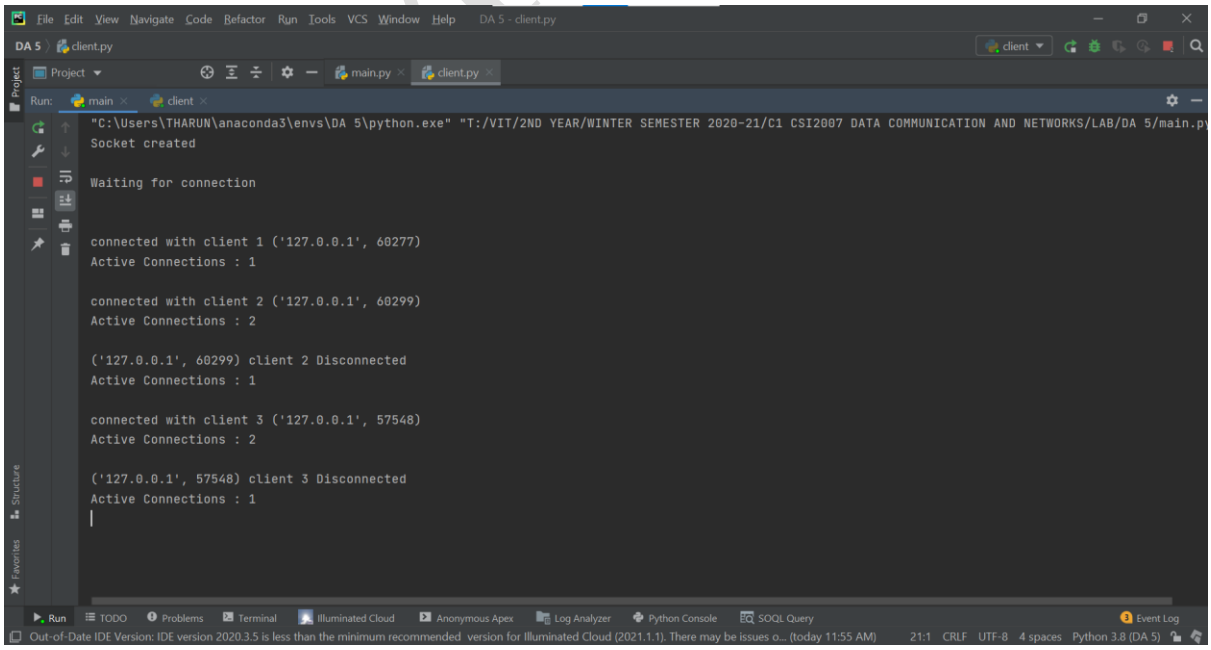
The server output window after “client 3” gets connected



Now “client 3” gets disconnected as the client presses ‘x’ in its console to close the connection.

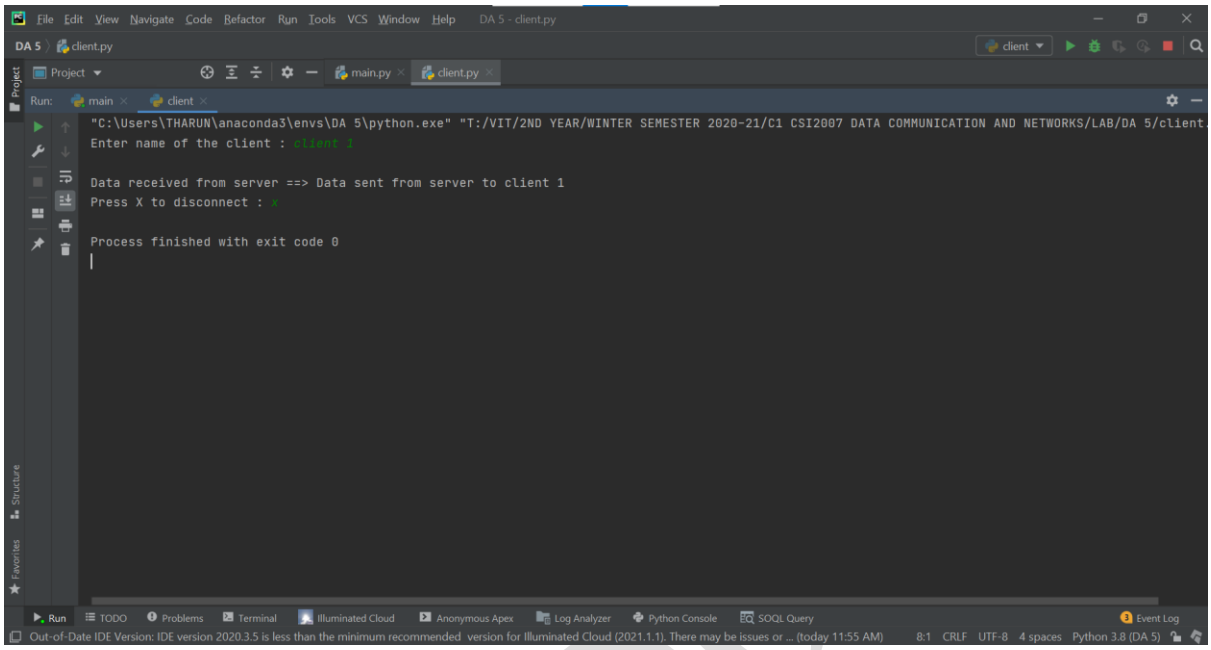


The sever program output



Now there is only 1 connection remaining and that is “client 1”.

So now “client 1” closes its connection and leaves the server



The server output