INSTITUTE OF COMPUTER TECHNOLOGY

B. TECH COMPUTER SCIENCE AND ENGINEERING

Subject: Computer Networks[CN]

Name : Archita Gahoi

Enrollment_No. : 23162171002

SEM : 5

Class : A

Batch : 52 (CS)

Practical 7

Aim: To implement Socket Programing

Scenario:

An organization named Albert Enterprise has established two departments for better performance of the company, as each department will be having some specific set of tasks to perform. So, this will reduce the time and increase the efficiency of the work. As both the departments are dependent on each other, they need to communicate more frequently. To solve the problem, the IT department has suggested the option to create a chat application using socket programming which will work only in the office premises. So, help the IT professionals to create the chat application.

Make sure that the application has the below mentioned features:

1) Department 1 will be set as the SERVER while department 2 will be set as a CLIENT device.

2) The message received by CLIENT or SERVER must be displayed with time stamp.

3) If any of the device irrespective of CLIENT or SERVER has sent the message that the “quit”, then connection should be closed on both the ends.

4) There is no restriction on the protocol selection, you can use UDP or TCP. Justify the reason for selection of the specific protocol.

Server Code:

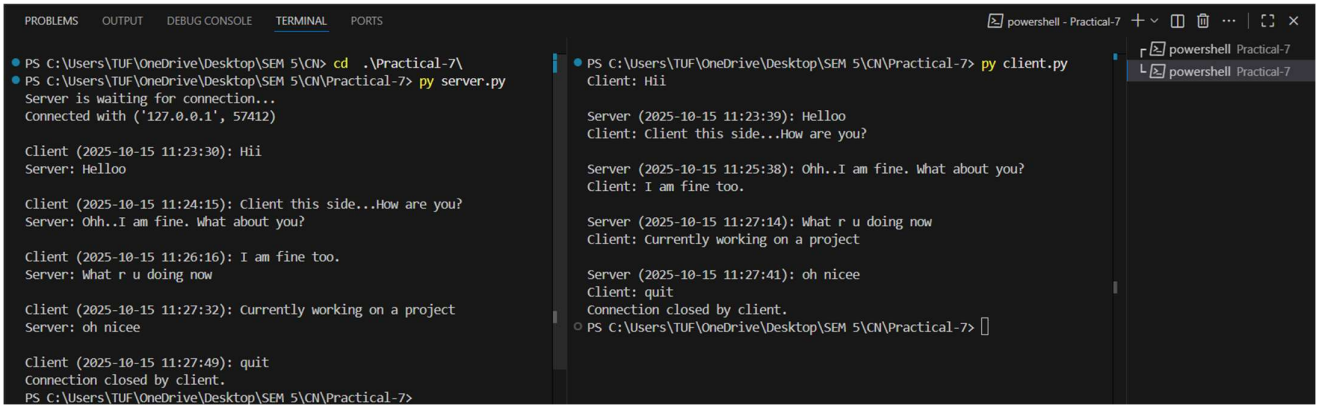
```
import socket
import datetime

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('localhost', 12345))
server_socket.listen(1)

print("Server is waiting for connection...")
conn, addr = server_socket.accept()
print(f"Connected with {addr}")

while True:
    msg = conn.recv(1024).decode()
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"\nClient ({timestamp}): {msg}")
    if msg.lower() == "quit":
        print("Connection closed by client.")
        break

    send_msg = input("Server: ")
    conn.send(send_msg.encode())
    if send_msg.lower() == "quit":
        print("Connection closed by server.")
        break

conn.close()
server_socket.close()
```

Client Code:

```
import socket
import datetime

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('localhost', 12345))

while True:
    send_msg = input("Client: ")
    client_socket.send(send_msg.encode())
    if send_msg.lower() == "quit":
        print("Connection closed by client.")
        break

    msg = client_socket.recv(1024).decode()
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"\nServer ({timestamp}): {msg}")
    if msg.lower() == "quit":
        print("Connection closed by server.")
        break

client_socket.close()
```

Output:

Server | **Client**



Conclusion:

In this practical, a two-way chat application was successfully implemented using TCP socket programming in Python. The system allowed real-time message exchange between a server (Department 1) and a client (Department 2**)** within the same network. The use of timestamps improved message tracking, and the “quit” command enabled safe disconnection from both sides. The implementation demonstrated the working of TCP sockets and reliable communication in a local network environment.