

Institute of Computer Technology
B. Tech Computer Science and Engineering
Sub: CN

Name – Sujal Suthar

Enrollment no - 23162581026

Batch – 53

Practical - 7

Aim: To implement Socket Programming

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.

Expected Output:

```
-----  
SERVER  
-----  
SERVER is listening..  
Connection accepted from ('192.168.1.6', 53792)  
CLIENT [2024-10-06 21:13:08]: Hi I am Client 1  
ENTER TEXT: I am Server  
CLIENT [2024-10-06 21:13:32]: My task is done  
ENTER TEXT: quit  
CLIENT [2024-10-06 21:13:36]:  
-----  
CLIENT  
-----  
Hello there! msg from SERVER  
ENTER TEXT: Hi I am Client 1  
SERVER [2024-10-06 21:13:17]: I am Server  
ENTER TEXT: My task is done  
SERVER [2024-10-06 21:13:36]: quit
```

Server Code:

```
import socket
from datetime import datetime

print("-" * 40)
print("SERVER")
print("-" * 40)

server_socket = socket.socket()
host = "" # Listen on all available interfaces
port = 5000

server_socket.bind((host, port))
server_socket.listen(1)
print("SERVER is listening..")

conn, address = server_socket.accept()
print(f"Connection accepted from {address}")

while True:
    # Receive message from client
    msg = conn.recv(1024).decode()
    if not msg:
        break

    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"CLIENT [{timestamp}]: {msg}")

    # Get server's response
    print("ENTER TEXT: ", end="")
    response = input()

    # Send response to client
    conn.send(response.encode())

    if response.lower() == 'quit':
        break

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

Client Code:

```
import socket
from datetime import datetime

print("-" * 40)
print("CLIENT")
print("-" * 40)

client_socket = socket.socket()
host = 'localhost' # or the server's IP address
port = 5000

print("Hello there! msg from SERVER")

client_socket.connect((host, port))

while True:
    print("ENTER TEXT: ", end="")
    message = input()

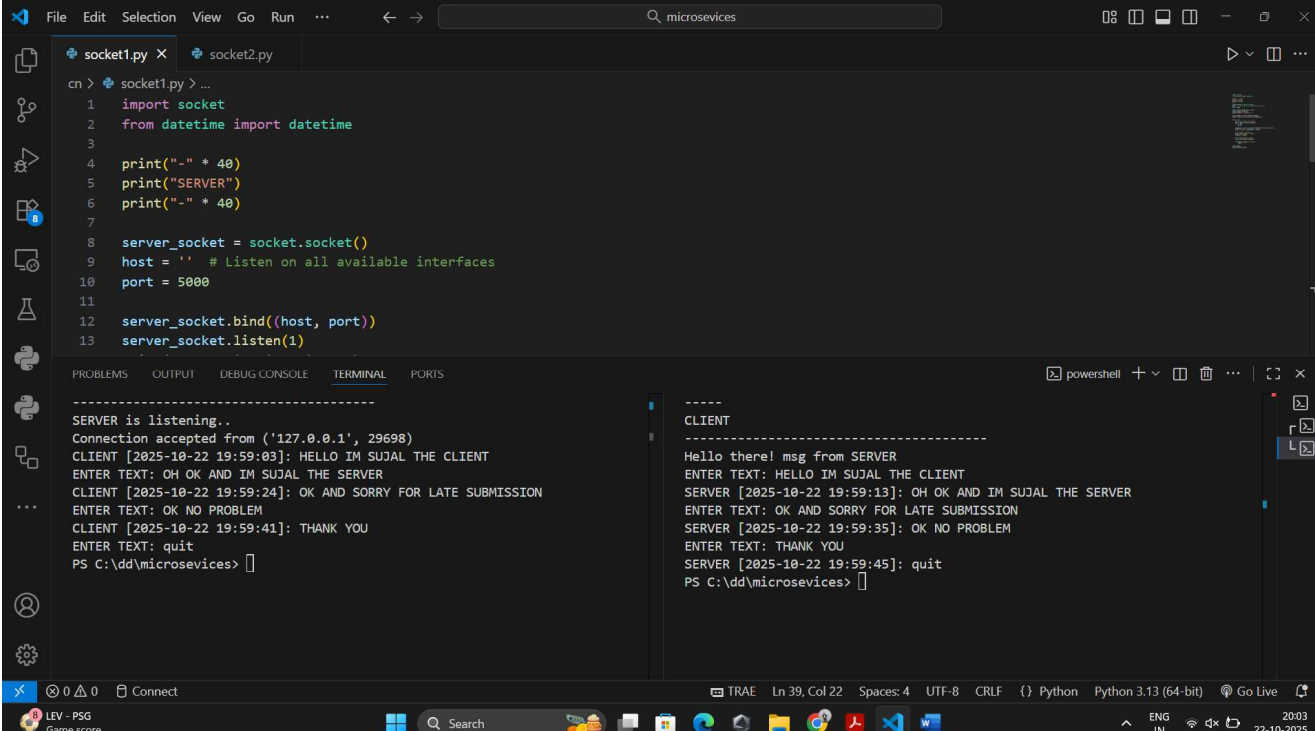
    # Send message to server
    client_socket.send(message.encode())

    # Get response from server
    response = client_socket.recv(1024).decode()
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"SERVER [{timestamp}]: {response}")

    if response.lower() == 'quit':
        break

client_socket.close()
```

Output:



The screenshot shows a Visual Studio Code editor with two Python files, `socket1.py` and `socket2.py`. The `socket1.py` file contains the server code, and the `socket2.py` file contains the client code. The terminal window at the bottom shows the output of the server and client interaction.

```
cn > socket1.py > ...
1 import socket
2 from datetime import datetime
3
4 print("-" * 40)
5 print("SERVER")
6 print("-" * 40)
7
8 server_socket = socket.socket()
9 host = '' # Listen on all available interfaces
10 port = 5000
11
12 server_socket.bind((host, port))
13 server_socket.listen(1)

-----
SERVER is listening..
Connection accepted from ('127.0.0.1', 29698)
CLIENT [2025-10-22 19:59:03]: HELLO IM SUJAL THE CLIENT
ENTER TEXT: OH OK AND IM SUJAL THE SERVER
CLIENT [2025-10-22 19:59:24]: OK AND SORRY FOR LATE SUBMISSION
ENTER TEXT: OK NO PROBLEM
CLIENT [2025-10-22 19:59:41]: THANK YOU
ENTER TEXT: quit
PS C:\dd\microservices>

-----
CLIENT
Hello there! msg from SERVER
ENTER TEXT: HELLO IM SUJAL THE CLIENT
SERVER [2025-10-22 19:59:13]: OH OK AND IM SUJAL THE SERVER
ENTER TEXT: OK AND SORRY FOR LATE SUBMISSION
SERVER [2025-10-22 19:59:35]: OK NO PROBLEM
ENTER TEXT: THANK YOU
SERVER [2025-10-22 19:59:45]: quit
PS C:\dd\microservices>
```

- **Conclusion:** This exercise demonstrated how to establish and use TCP socket connections for two-way communication between a client and server.