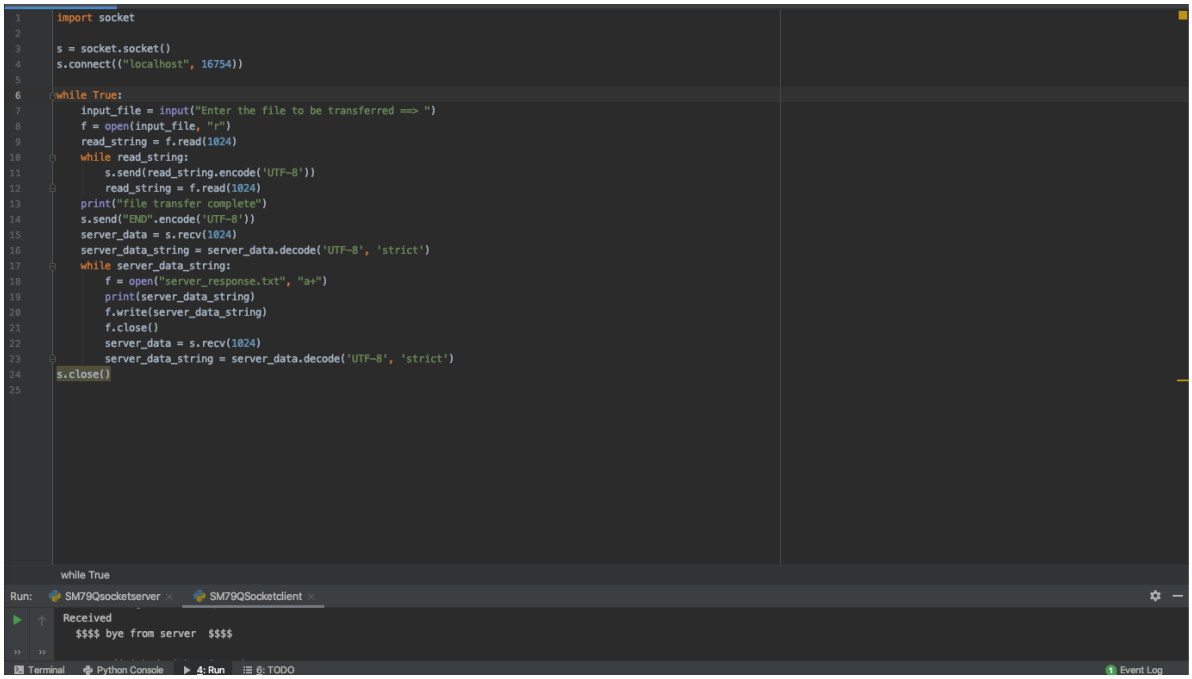
NETWORK ARCHITECTURE-1

Socket programming

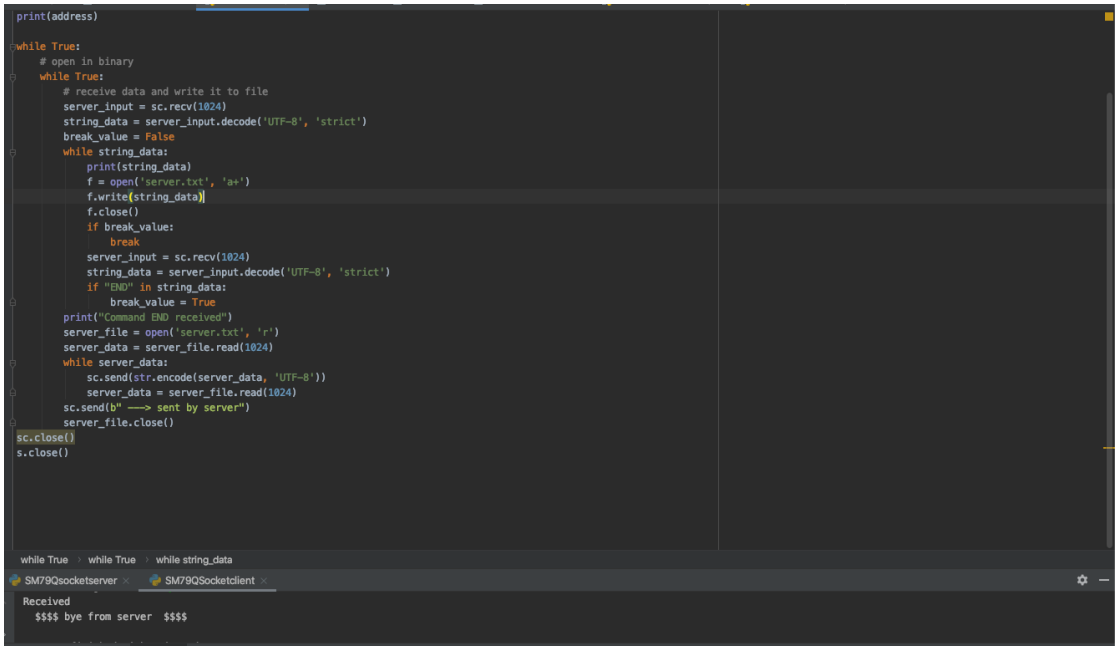
Meesala Sandeep

16299481

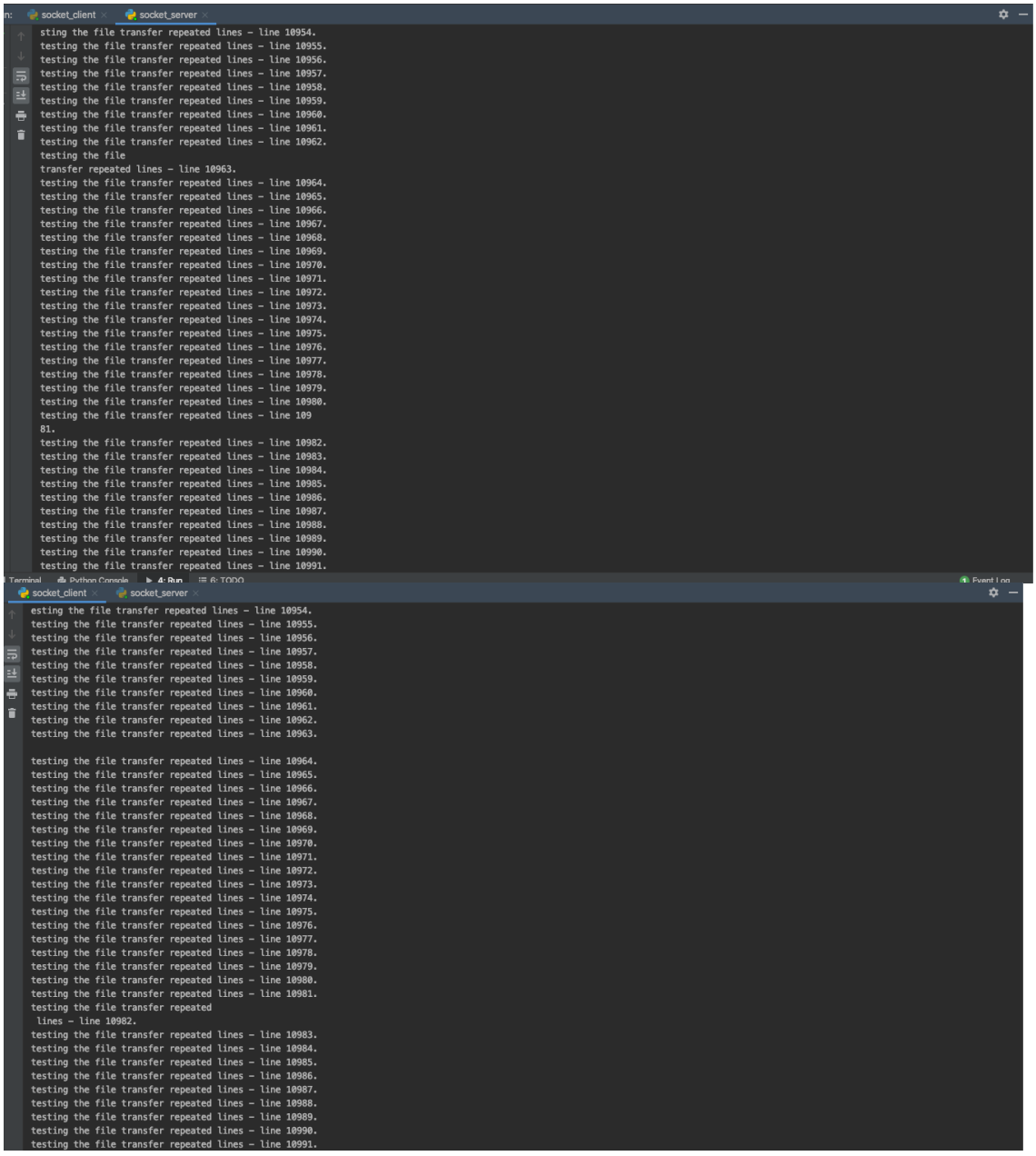
1.
Client Code Snippet:



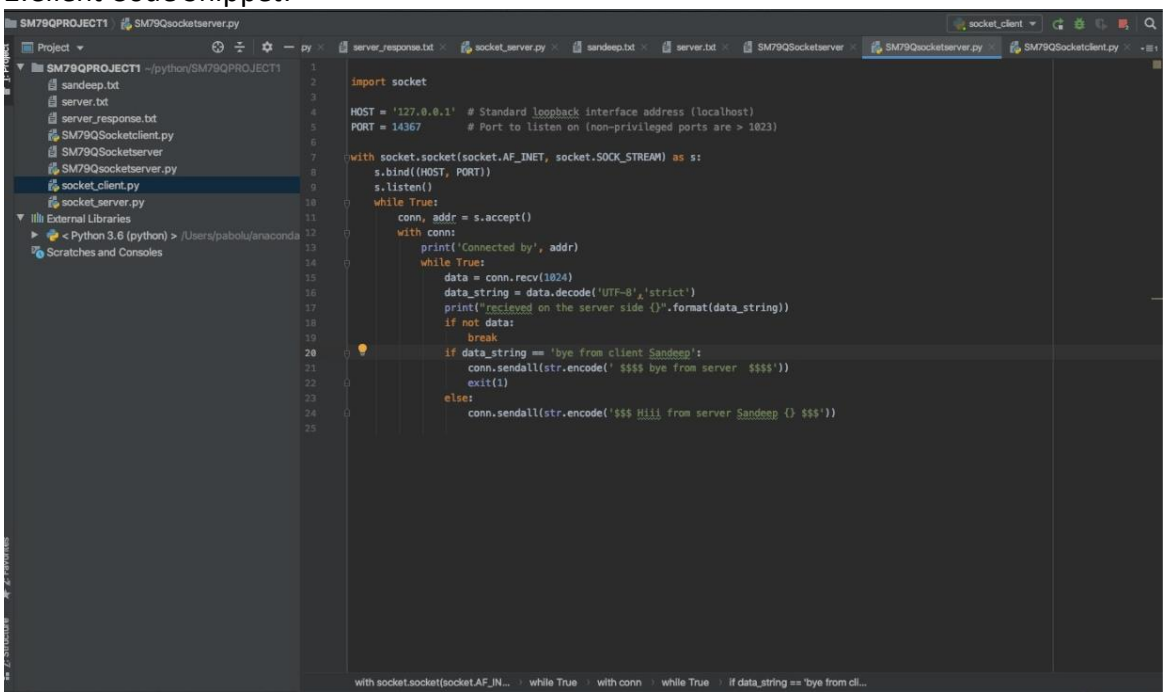
1.Server Code Snippet:


print(address)
while True:
# open in binary
while True:
# receive data and write it to file
server_input = sc.recv(1024)
string_data = server_input.decode('UTF-8', 'strict')
break_value = False
while string_data:
print(string_data)
f = open('server.txt', 'a+')
f.write(string_data)
f.close()
if break_value:
break
server_input = sc.recv(1024)
string_data = server_input.decode('UTF-8', 'strict')
if "END" in string_data:
break_value = True
print("Command END received")
server_file = open('server.txt', 'r')
server_data = server_file.read(1024)
while server_data:
sc.send(str.encode(server_data, 'UTF-8'))
server_data = server_file.read(1024)
sc.send(b" ----> sent by server")
server_file.close()
sc.close()
s.close()
while True
while True
while string_data
SM79Qsocketserver
SM79QSocketclient
Received
$$$$ bye from server $$$$

Results:


socket_client
socket_server
sting the file transfer repeated lines - line 10954.
testing the file transfer repeated lines - line 10955.
testing the file transfer repeated lines - line 10956.
testing the file transfer repeated lines - line 10957.
testing the file transfer repeated lines - line 10958.
testing the file transfer repeated lines - line 10959.
testing the file transfer repeated lines - line 10960.
testing the file transfer repeated lines - line 10961.
testing the file transfer repeated lines - line 10962.
testing the file
transfer repeated lines - line 10963.
testing the file transfer repeated lines - line 10964.
testing the file transfer repeated lines - line 10965.
testing the file transfer repeated lines - line 10966.
testing the file transfer repeated lines - line 10967.
testing the file transfer repeated lines - line 10968.
testing the file transfer repeated lines - line 10969.
testing the file transfer repeated lines - line 10970.
testing the file transfer repeated lines - line 10971.
testing the file transfer repeated lines - line 10972.
testing the file transfer repeated lines - line 10973.
testing the file transfer repeated lines - line 10974.
testing the file transfer repeated lines - line 10975.
testing the file transfer repeated lines - line 10976.
testing the file transfer repeated lines - line 10977.
testing the file transfer repeated lines - line 10978.
testing the file transfer repeated lines - line 10979.
testing the file transfer repeated lines - line 10980.
testing the file transfer repeated lines - line 109
81.
testing the file transfer repeated lines - line 10982.
testing the file transfer repeated lines - line 10983.
testing the file transfer repeated lines - line 10984.
testing the file transfer repeated lines - line 10985.
testing the file transfer repeated lines - line 10986.
testing the file transfer repeated lines - line 10987.
testing the file transfer repeated lines - line 10988.
testing the file transfer repeated lines - line 10989.
testing the file transfer repeated lines - line 10990.
testing the file transfer repeated lines - line 10991.
Terminal
Python Console
4: Run
6: TODO
Event Log
socket_client
socket_server
esting the file transfer repeated lines - line 10954.
testing the file transfer repeated lines - line 10955.
testing the file transfer repeated lines - line 10956.
testing the file transfer repeated lines - line 10957.
testing the file transfer repeated lines - line 10958.
testing the file transfer repeated lines - line 10959.
testing the file transfer repeated lines - line 10960.
testing the file transfer repeated lines - line 10961.
testing the file transfer repeated lines - line 10962.
testing the file transfer repeated lines - line 10963.
testing the file transfer repeated lines - line 10964.
testing the file transfer repeated lines - line 10965.
testing the file transfer repeated lines - line 10966.
testing the file transfer repeated lines - line 10967.
testing the file transfer repeated lines - line 10968.
testing the file transfer repeated lines - line 10969.
testing the file transfer repeated lines - line 10970.
testing the file transfer repeated lines - line 10971.
testing the file transfer repeated lines - line 10972.
testing the file transfer repeated lines - line 10973.
testing the file transfer repeated lines - line 10974.
testing the file transfer repeated lines - line 10975.
testing the file transfer repeated lines - line 10976.
testing the file transfer repeated lines - line 10977.
testing the file transfer repeated lines - line 10978.
testing the file transfer repeated lines - line 10979.
testing the file transfer repeated lines - line 10980.
testing the file transfer repeated lines - line 10981.
testing the file transfer repeated
lines - line 10982.
testing the file transfer repeated lines - line 10983.
testing the file transfer repeated lines - line 10984.
testing the file transfer repeated lines - line 10985.
testing the file transfer repeated lines - line 10986.
testing the file transfer repeated lines - line 10987.
testing the file transfer repeated lines - line 10988.
testing the file transfer repeated lines - line 10989.
testing the file transfer repeated lines - line 10990.
testing the file transfer repeated lines - line 10991.

2.Client Code Snippet:

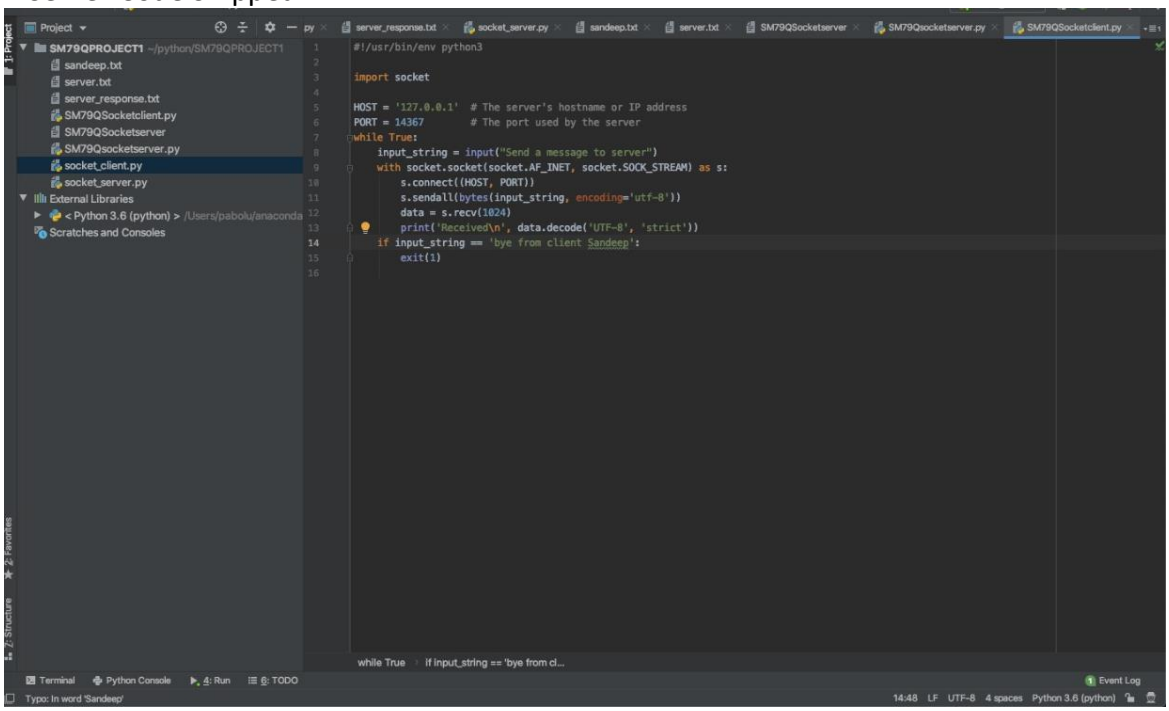


```
import socket

HOST = '127.0.0.1'  # Standard loopback interface address (localhost)
PORT = 14367        # Port to listen on (non-privileged ports are > 1023)

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen()
    while True:
        conn, addr = s.accept()
        with conn:
            print('Connected by', addr)
            while True:
                data = conn.recv(1024)
                data_string = data.decode('UTF-8','strict')
                print("recieved on the server side {}".format(data_string))
                if not data:
                    break
                if data_string == 'bye from client Sandeep':
                    conn.sendall(str.encode(' $$$$ bye from server  $$$$'))
                    exit(1)
                else:
                    conn.sendall(str.encode('$$$ Hiii from server Sandeep {} $$$'))
```

2.Server Code Snippet:

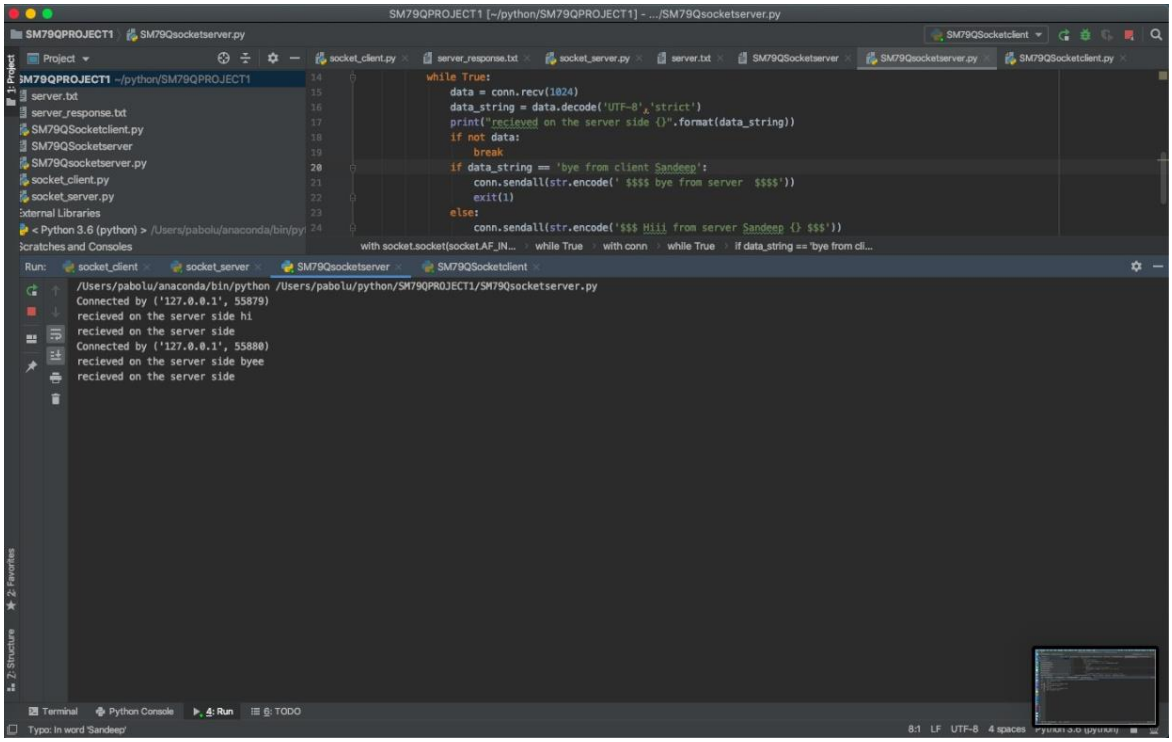


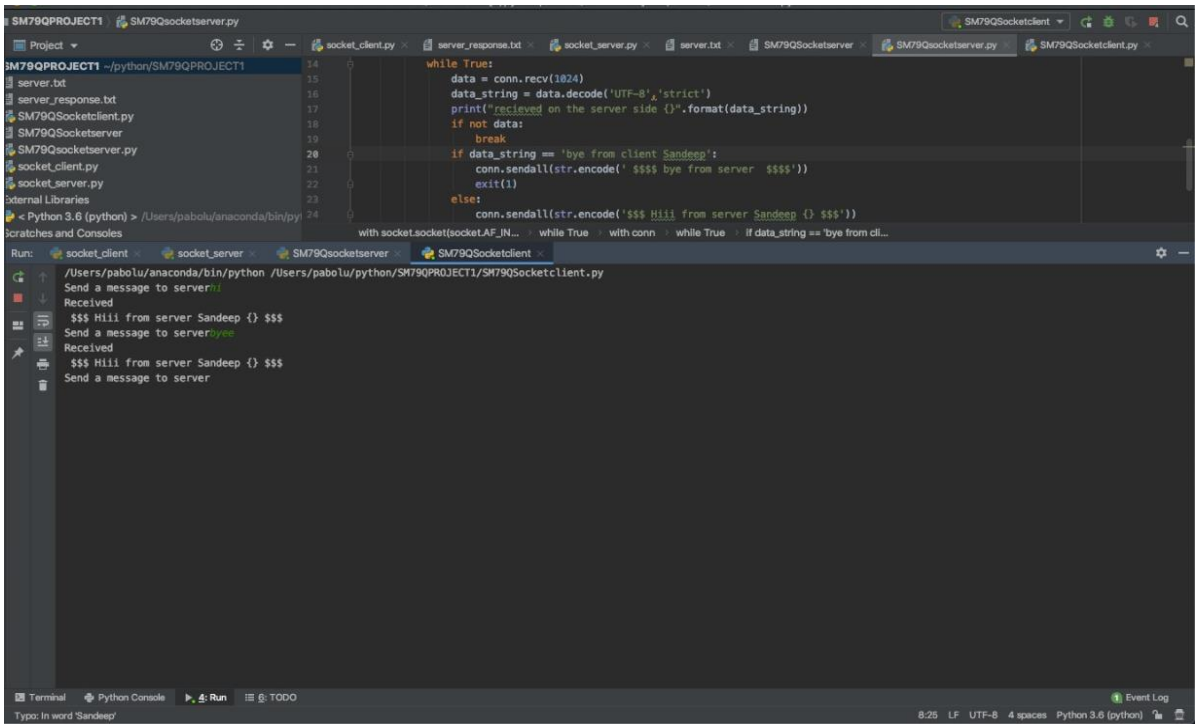
```
#!/usr/bin/env python3

import socket

HOST = '127.0.0.1'  # The server's hostname or IP address
PORT = 14367        # The port used by the server
while True:
    input_string = input("Send a message to server")
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((HOST, PORT))
        s.sendall(bytes(input_string, encoding='utf-8'))
        data = s.recv(1024)
        print('Received\n', data.decode('UTF-8', 'strict'))
    if input_string == 'bye from client Sandeep':
        exit(1)
```

Results:


SM79QPROJECT1 [~/python/SM79QPROJECT1] - .../SM79Qsocketserver.py
while True:
data = conn.recv(1024)
data_string = data.decode('UTF-8','strict')
print("recieved on the server side {}".format(data_string))
if not data:
break
if data_string == 'bye from client Sandeep':
conn.sendall(str.encode(' $$$$ bye from server $$$$'))
exit(1)
else:
conn.sendall(str.encode('$$$ Hiii from server Sandeep {} $$$'))
/Users/pabolu/anaconda/bin/python /Users/pabolu/python/SM79QPROJECT1/SM79Qsocketserver.py
Connected by ('127.0.0.1', 55879)
recieved on the server side hi
recieved on the server side
Connected by ('127.0.0.1', 55880)
recieved on the server side byee
recieved on the server side


while True:
data = conn.recv(1024)
data_string = data.decode('UTF-8','strict')
print("recieved on the server side {}".format(data_string))
if not data:
break
if data_string == 'bye from client Sandeep':
conn.sendall(str.encode(' $$$$ bye from server $$$$'))
exit(1)
else:
conn.sendall(str.encode('$$$ Hiii from server Sandeep {} $$$'))
/Users/pabolu/anaconda/bin/python /Users/pabolu/python/SM79QPROJECT1/SM79Qsocketclient.py
Send a message to serverhi
Received
$$$ Hiii from server Sandeep {} $$$
Send a message to serverbyee
Received
$$$ Hiii from server Sandeep {} $$$
Send a message to server

THANK YOU