**LAB**

**Name: Vikas Tomar Register Number:21MEI10027**

**Team Number: 09 Exp. No: 09**

**Title:** Consider two processes client and server communicates across a network. The client sends a message to the server in the request and the server responds with the same message. Write a Socket program for the above mentioned scenario.

**Objective:** Sockets programming allows us to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine, and they easily allow access to centralized data. Socket application program interfaces (APIs) are the network standard for TCP/IP.

**Requirements:** Four pieces of information are needed to create a TCP socket:

* The local system's IP address.
* The TCP port number the local application is using.
* The remote system's IP address.
* The TCP port number to which the remote application is responding.

**Theory / Procedure:**

**The client-server model**

The client-server model is one of the most used communication paradigms in networked systems. Clients normally communicates with one server at a time. From a server’s perspective, at any point in time, it is not unusual for a server to be communicating with multiple clients. Client need to know of the existence of and the address of the server, but the server does not need to know the address of (or even the existence of) the client prior to the connection being established

Client and servers communicate by means of multiple layers of network protocols. In this program we will focus on the TCP/IP protocol suite.

**the steps for establishing a TCP socket on the client side are the following**:

* Create a socket using the socket() function;
* Connect the socket to the address of the server using the connect() function;
* Send and receive data by means of the read() and write() functions.

**The steps involved in establishing a TCP socket on the server side are as follows:**

* Create a socket with the socket() function;
* Bind the socket to an address using the bind() function;
* Listen for connections with the listen() function;
* Accept a connection with the accept() function system call. This call typically blocks until a client connects with the server.
* Send and receive data by means of send() and receive().

**Source Code:**

**# Code for Server.**

import socket

s=socket.socket()

print("socket created successfully ")

s.bind(('localhost',54321))

s.listen(3)

print("waiting for the connections")

while True:

    c,addr= s.accept()

    name=c.recv(1024).decode()

    print("connected with",addr,name)

    c.send(bytes('welcome to vikas tomar page','utf-8'))

    c.send(bytes('bye','utf-8'))

    end=c.recv(1024).decode()

    print("the request stop message message recieved from client :",end)

    c.close()

**# Code for client.**

import socket

c= socket.socket()

c.connect(('localhost',54321))

name=input(str("enter your message :"))

c.send(bytes(name,"utf-8"))

print(c.recv(1024).decode())

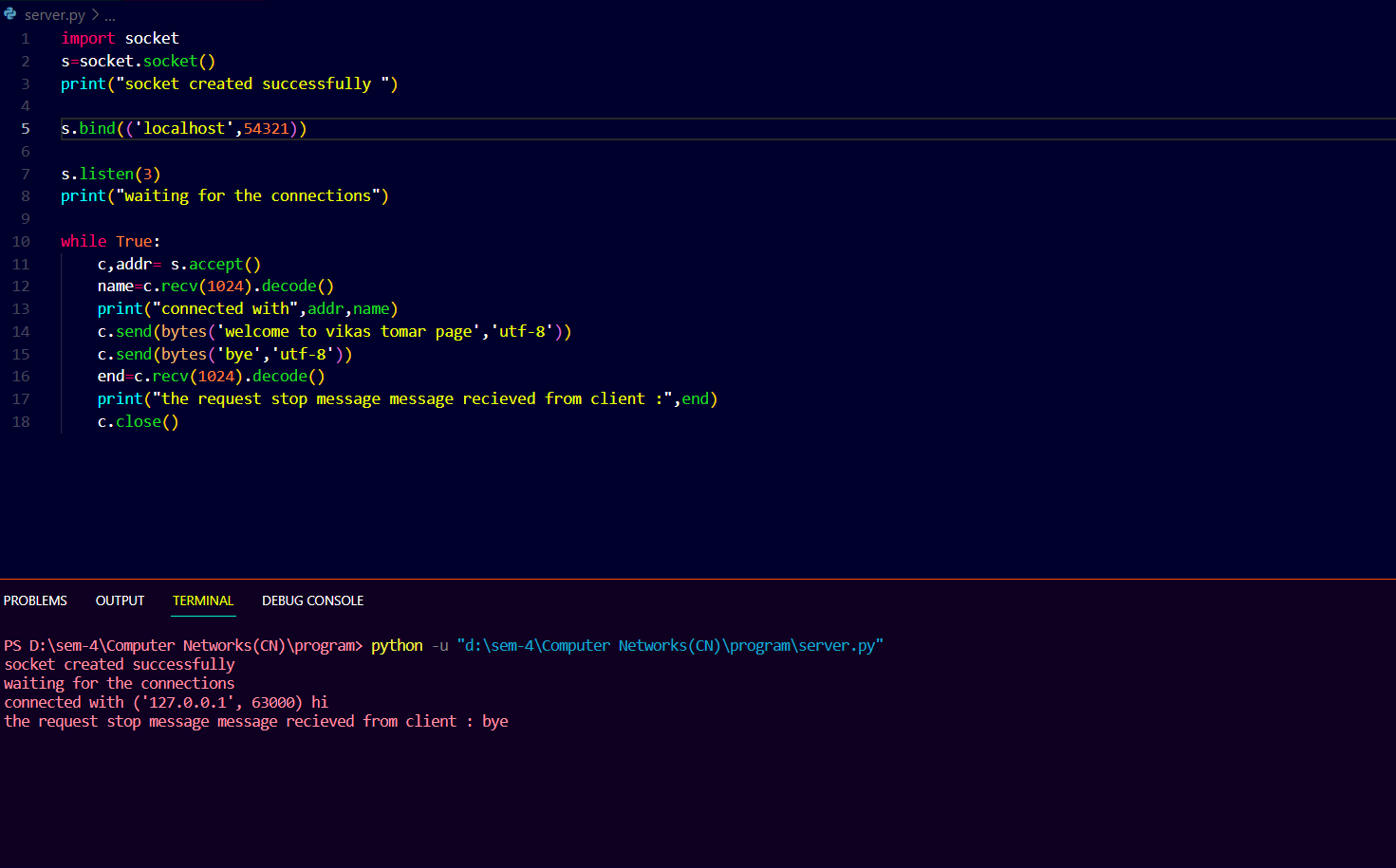
end=input(str("enter your message :"))

print(c.recv(1024).decode())

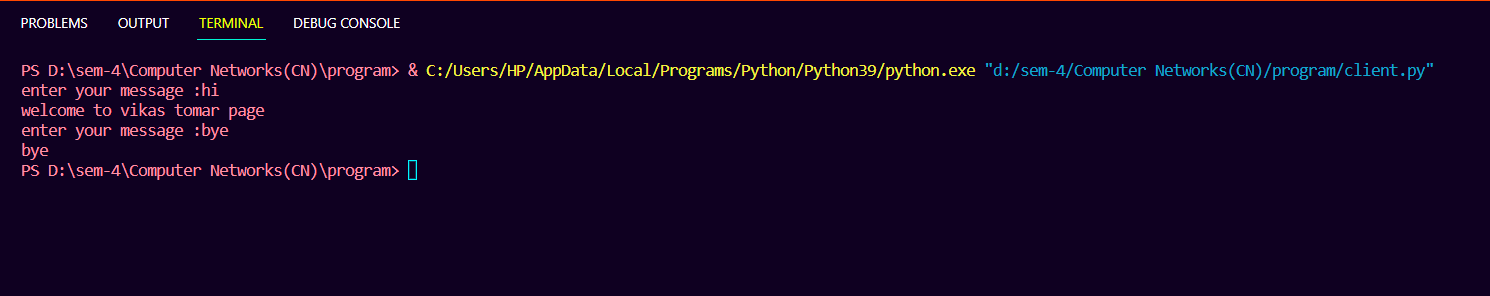
c.send(bytes(end,"utf-8"))

**Output Screen Shots:**

**Server side**

****

**Client side**

****

**Result & Observation**

the result of using this program is we are able to establish a connection between two sides and able to send and receive the message