Name: Siddhi Parekh
Reg No. 221071047
Batch C
SY Comps

Experiment 7

AIM:
Write an application to implement client-server programming.

THEORY:

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.

They are the real backbones behind web browsing. In simpler terms, there is a server and a client.
Socket programming is started by importing the socket library and making a simple socket.

A server is a software that waits for client requests and serves or processes them accordingly.

A client is a requester of this service.

A client programs a request for some resources to the server and the server responds to that request. Socket is the endpoint of a bidirectional communications channel between server and client. Sockets may communicate within a process, between processes on the same machine, or between processes on different machines. For any communication with a remote program, we have to connect through a socket port.

Client code:

```python
import socket


client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

server_address = ('localhost', 12345)

client_socket.connect(server_address)

file_name = 'Sender.txt'

with open(file_name, 'rb') as file:
    while True:
        data = file.read(1024)
        if not data:
            break
        client_socket.send(data)

print(f"File '{file_name}' sent to the server.")

client_socket.close()
```

Server Code:

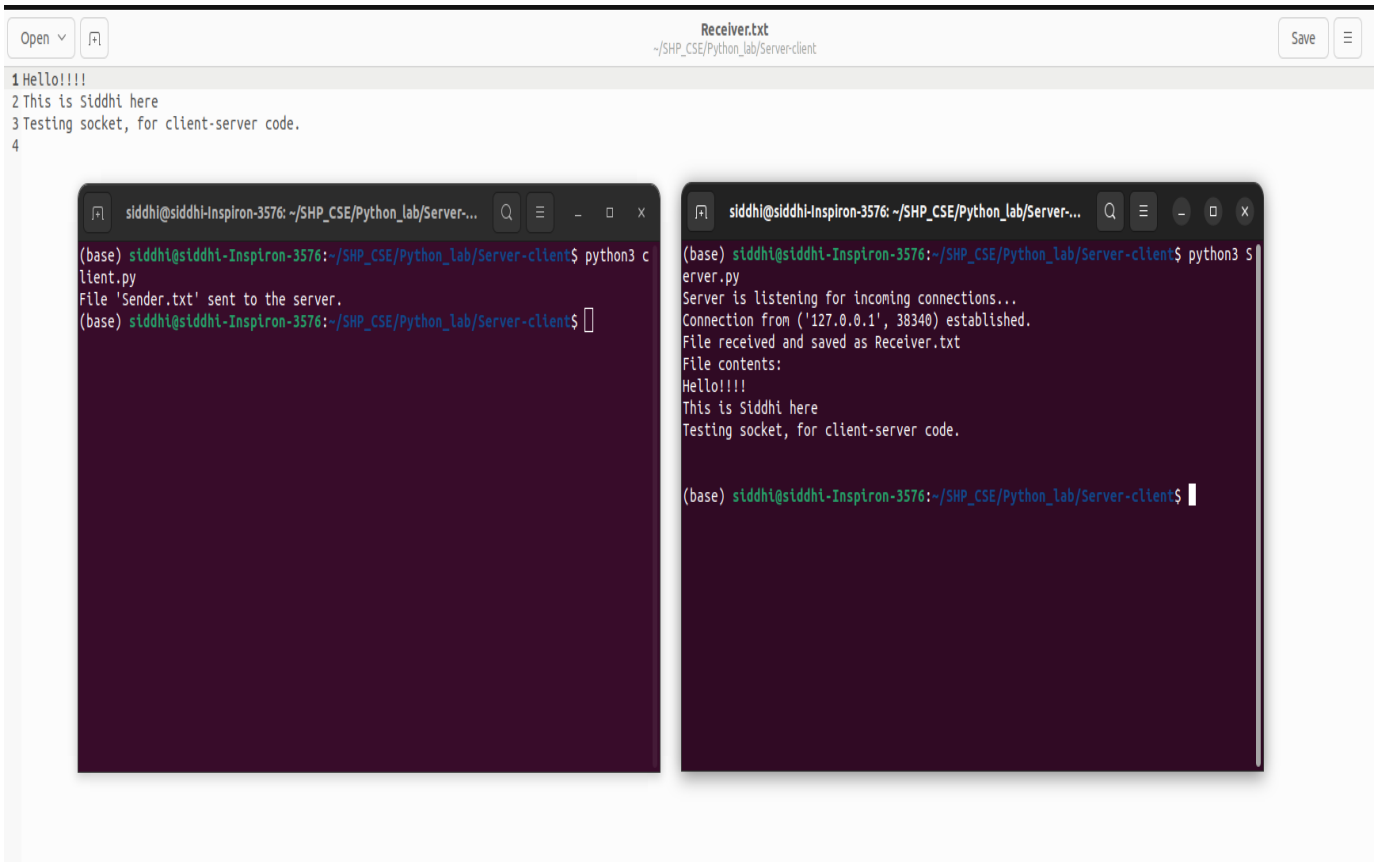
File Edit Selection View Go Run Terminal Help

client.py server.py

server.py > ...

```python
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

server_address = ('localhost', 12345)

server_socket.bind(server_address)

server_socket.listen(1)
print("Server is listening for incoming connections...")

client_socket, client_address = server_socket.accept()
print(f"Connection from {client_address} established.")

file_name = 'Receiver.txt'
with open(file_name, 'wb') as file:
    while True:
        data = client_socket.recv(1024)
        if not data:
            break
        file.write(data)

print(f"File received and saved as {file_name}")

with open(file_name, 'r') as file:
    file_contents = file.read()
    print("File contents:")
    print(file_contents)

client_socket.close()
server_socket.close()
```

Working:



Conclusion:

Thus, with the help of this experiment, we have learnt, how the client, server individually works, and how the message sent by the client is received by the server.