

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.

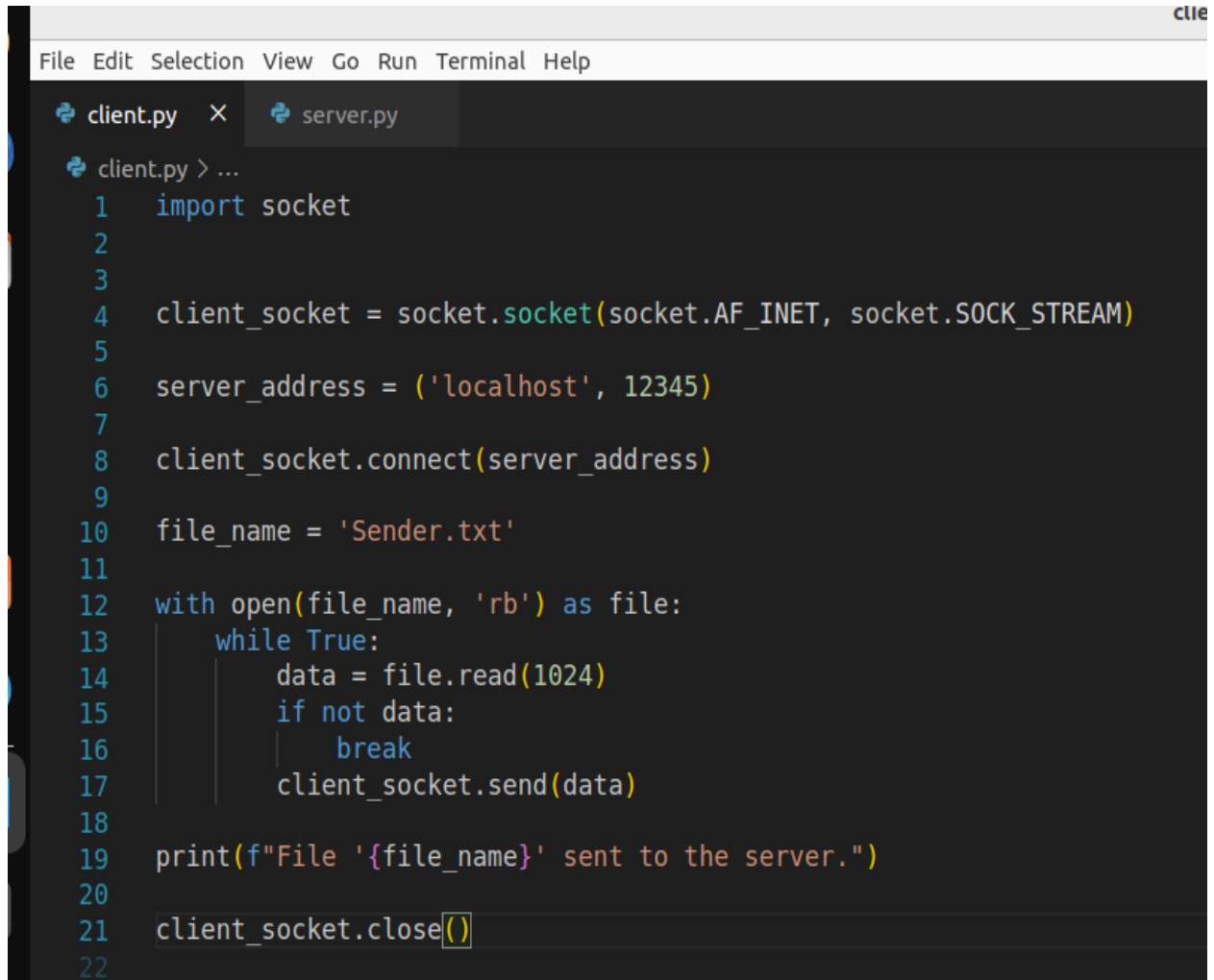
Socket programming is started by importing the socket library and making a simple socket.

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

A client is a requester of this service.

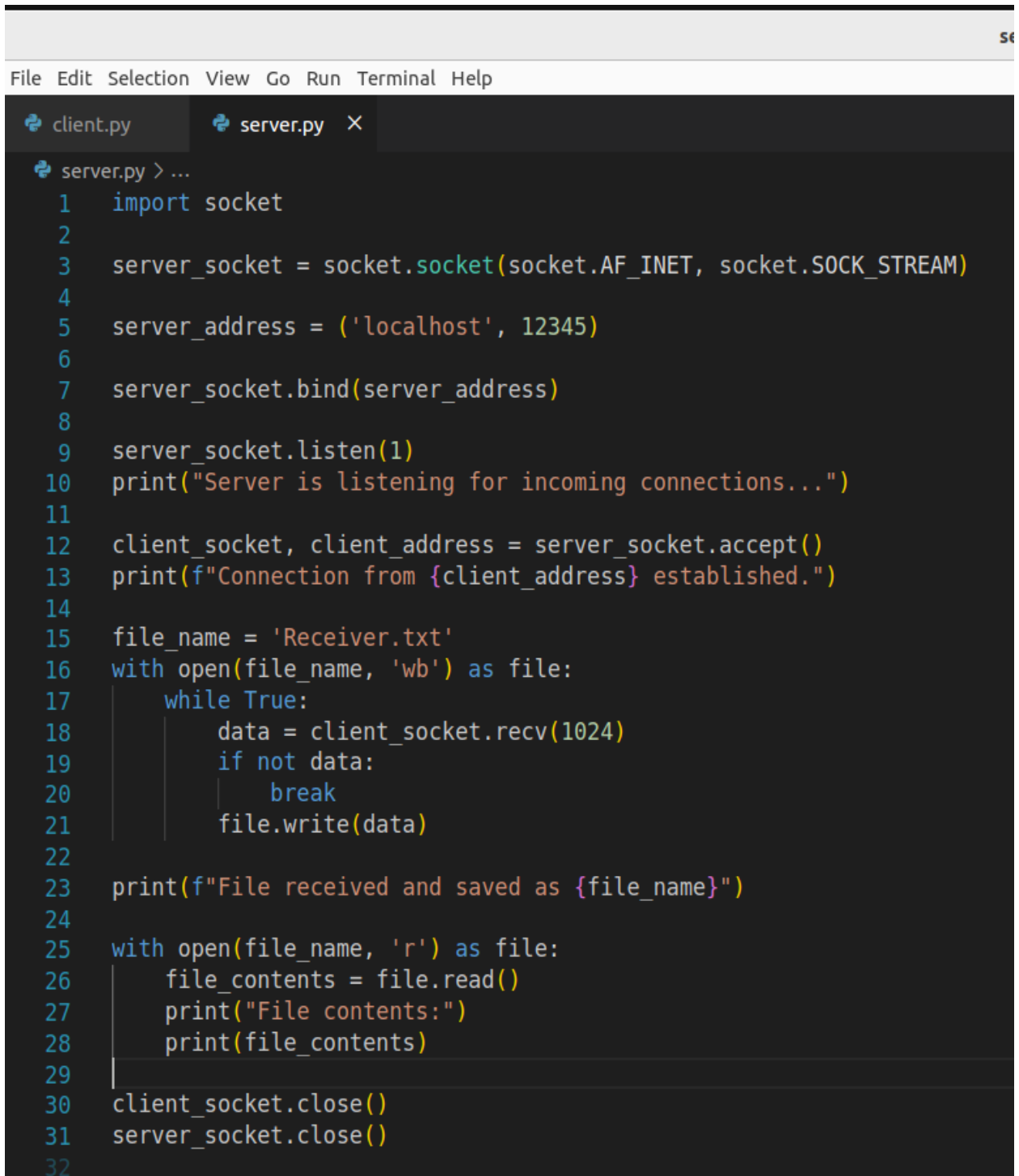
A client programs a request for some resources to the server and the server responds to that request. Socket is the endpoint of a bidirectional communications channel between server and client. Sockets may communicate within a process, between processes on the same machine, or between processes on different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

Client code:

A screenshot of a code editor window. The title bar at the top says "client.py" on the right. Below the title bar is a menu bar with "File", "Edit", "Selection", "View", "Go", "Run", "Terminal", and "Help". Below the menu bar are two tabs: "client.py" (active) and "server.py". The main area shows the code for client.py, with line numbers 1 through 22 on the left. The code is as follows:

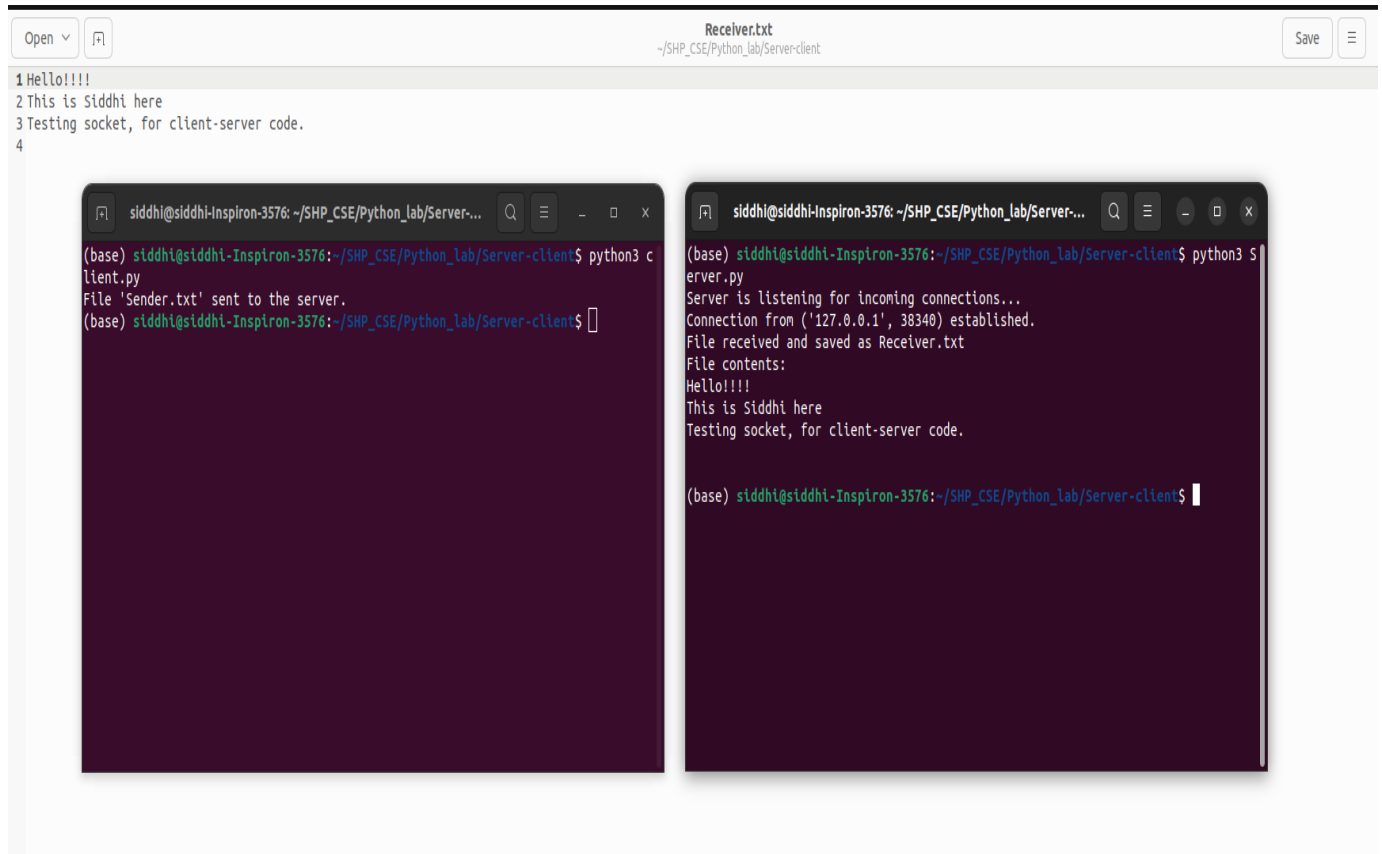
```
1  import socket
2
3
4  client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5
6  server_address = ('localhost', 12345)
7
8  client_socket.connect(server_address)
9
10 file_name = 'Sender.txt'
11
12 with open(file_name, 'rb') as file:
13     while True:
14         data = file.read(1024)
15         if not data:
16             break
17         client_socket.send(data)
18
19 print(f"File '{file_name}' sent to the server.")
20
21 client_socket.close()
22
```

Server Code:



```
server.py > ...
1  import socket
2
3  server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
4
5  server_address = ('localhost', 12345)
6
7  server_socket.bind(server_address)
8
9  server_socket.listen(1)
10 print("Server is listening for incoming connections...")
11
12 client_socket, client_address = server_socket.accept()
13 print(f"Connection from {client_address} established.")
14
15 file_name = 'Receiver.txt'
16 with open(file_name, 'wb') as file:
17     while True:
18         data = client_socket.recv(1024)
19         if not data:
20             break
21         file.write(data)
22
23 print(f"File received and saved as {file_name}")
24
25 with open(file_name, 'r') as file:
26     file_contents = file.read()
27     print("File contents:")
28     print(file_contents)
29
30 client_socket.close()
31 server_socket.close()
32
```

Working:



```
Receiver.txt
~/SHP_CSE/Python_lab/Server-client

1 Hello!!!!
2 This is Siddhi here
3 Testing socket, for client-server code.
4

(base) siddhi@siddhi-Inspiron-3576: ~/SHP_CSE/Python_lab/Server-client$ python3 client.py
File 'Sender.txt' sent to the server.
(base) siddhi@siddhi-Inspiron-3576: ~/SHP_CSE/Python_lab/Server-client$

(base) siddhi@siddhi-Inspiron-3576: ~/SHP_CSE/Python_lab/Server-client$ python3 server.py
Server is listening for incoming connections...
Connection from ('127.0.0.1', 38340) established.
File received and saved as Receiver.txt
File contents:
Hello!!!!
This is Siddhi here
Testing socket, for client-server code.

(base) siddhi@siddhi-Inspiron-3576: ~/SHP_CSE/Python_lab/Server-client$
```

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.