B.Naren Kartic
241901065

EXPT: 2 TCP CLIENT-SERVER COMMUNICATION USING SOCKET PROGRAMMING IN PYTHON

Aim:

To understand and apply socket programming in Python by creating communication between a client and a server using the TCP/IP protocol.

Algorithm:

Server:

1. Import the socket library.
2. Create a socket using `socket.socket()`.
3. Attach the socket to a specific host and port using `bind()`.
4. Wait for incoming connections using `listen()`.
5. Accept a client connection using `accept()`.
6. Receive a message using `recv()`.
7. Send a reply back using `send()`.
8. Close the connection using `close()`.

Client:

1. Import the socket library.
2. Create a socket using `socket.socket()`.
3. Connect to the server using `connect((host, port))`.
4. Send a message using `send()`.
5. Receive a reply using `recv()`.
6. Close the socket using `close()`.

Code:

SERVER:

```
import socket
sockfd=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print('Socket Created')
sockfd.bind(('localhost',55555))
sockfd.listen(3)
print('Waiting for connections')
while True:
    clientfd,addr=sockfd.accept()
    receivedMsg=clientfd.recv(1024).decode()
    print("Connected with ",addr)
    print("Message Received from Client: ",receivedMsg)
```

```
    clientfd.send(bytes(receivedMsg,'utf-8'))
    print("Message reply sent to Client!")
    print("Do you want to continue(type y or n):")
    choice=input()
    if choice=='n':
        break
```

CLIENT:

```
import socket
clientfd=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientfd.connect(('localhost',55555))
name=input("Enter your message:")
clientfd.send(bytes(name,'utf-8'))
print("Message Received from Server: ",clientfd.recv(1024).decode())
```

***Output*: *Server*:**

```
PS H:\bala> python .\TCP_s.py
Socket Created
Waiting for connections
Connected with  ('127.0.0.1', 50708)
Message Received from Client:  hi
Message reply sent to Client!
Do you want to continue(type y or n):
n
```

***Client*:**

```
PS H:\bala> python .\TCP_c.py
Enter your message:hi
Message Received from Server:  hi
```

Result:

The Python socket program ran successfully. A TCP link was created between the client and server, allowing both sides to send and receive messages properly.