NAME: SIVARANJANII.M

DATE:19.08.25

ROLL NO.:241901109

EXERCISE 3

UDP CLIENT-SERVER COMMUNICATION USING SOCKET PROGRAMMING IN PYTHON

AIM:

To implement UDP client-server communication using socket programming in Python.

ALGORITHM:

SERVER:

1. Create a socket using socket.socket().
2. Bind the socket to IP and port using bind().
3. Receive message using recvfrom ().
4. Send response using sendto ().
5. Close connection.

CLIENT:

1. Create a socket using socket.socket().
2. Send a message to the server using sendto ().
3. Receive response using recvform ().
4. Close connection.

CODE:

<u>SERVER</u>:

```python
import socket

sockfd = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

print("UDP Socket Created")

sockfd.bind(('localhost', 55555))

print("Waiting for messages on localhost:55555")

while True:

    data, addr = sockfd.recvfrom(1024)

    message = data.decode('utf-8')

    print(f"Received from {addr}: {message}")


    sockfd.sendto(data, addr)

    print(f"Echoed back to {addr}")

    choice = input("Continue receiving? (y/n): ")

    if choice.lower() == 'n':

        break

sockfd.close()

print("Server shutdown")
```

<u>CLIENT</u>:

```python
import socket

clientfd = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

server_addr = ('localhost', 55555)

msg = input("Enter your message: ")

clientfd.sendto(msg.encode('utf-8'), server_addr)

data, _ = clientfd.recvfrom(1024)
```
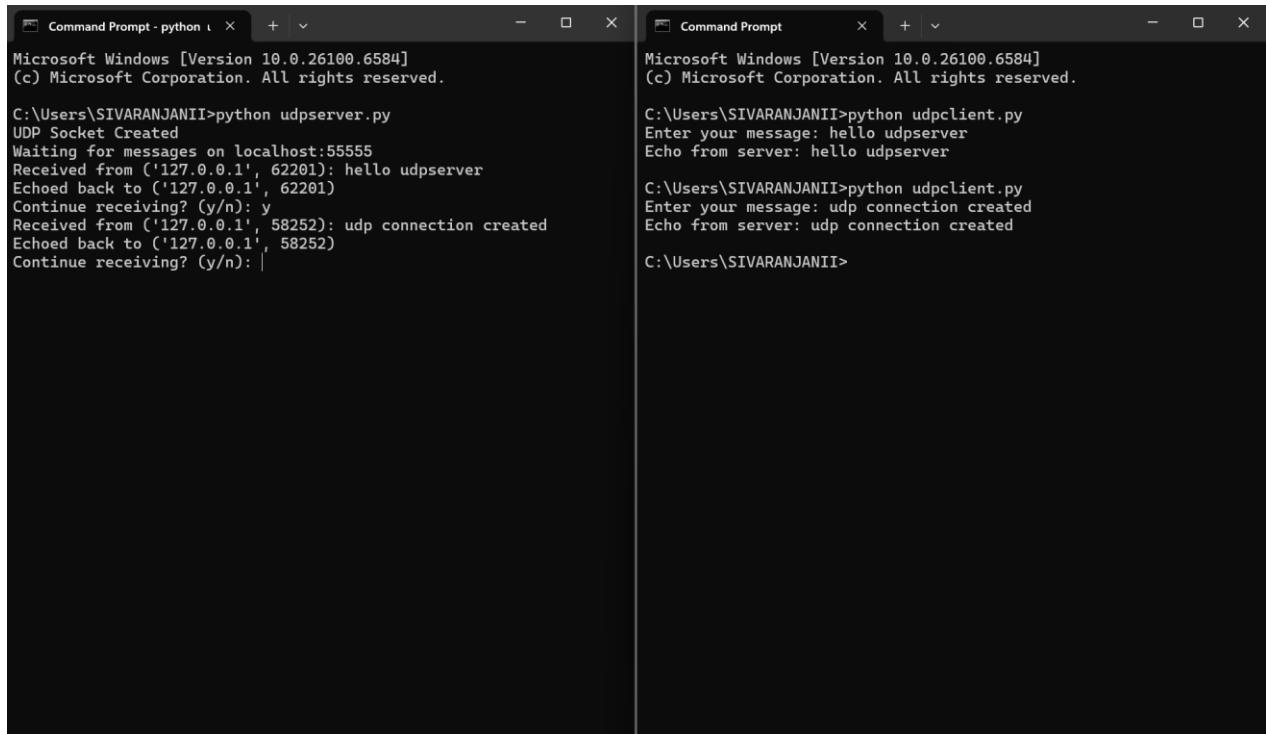
print("Echo from server:", data.decode('utf-8'))

clientfd.close()

OUTPUT:

RESULT:

Thus, UDP client-server communication was successfully implemented using Python.