NAME: PRATIBA.S DATE:19.08.25 ROLL NO.:241901080

# EXERCISE 3

**UDP CLIENT – SERVER 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 an IP and port using bind ().
3. Receive message using recv ().
4. Send response using send ().
5. Close connection.

CLIENT:

1. Create a socket using socket.socket().
2. Connect to the server using sendto().
3. Receive response using recvfrom ().
4. Close connection.

**CODE:**

SERVER:

import socket

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

server\_socket.bind(('localhost', 55555))

print("UDP Server is running and waiting for messages...")

while True:

# Receive message and address of client

data, addr = server\_socket.recvfrom(1024)

print("Message received from client:", data.decode())

# Send reply back to client

server\_socket.sendto(data, addr)

print("Echoed message back to client.")

choice = input("Do you want to continue (y/n): ")

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

print("Server shutting down.")

break

server\_socket.close()

CLIENT:

import socket

client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

server\_address = ('localhost', 55555)

while True:

msg = input("Enter your message: ")

client\_socket.sendto(msg.encode(), server\_address)

data, \_ = client\_socket.recvfrom(1024)

print("Message received from server:", data.decode())

choice = input("Do you want to send another message (y/n): ")

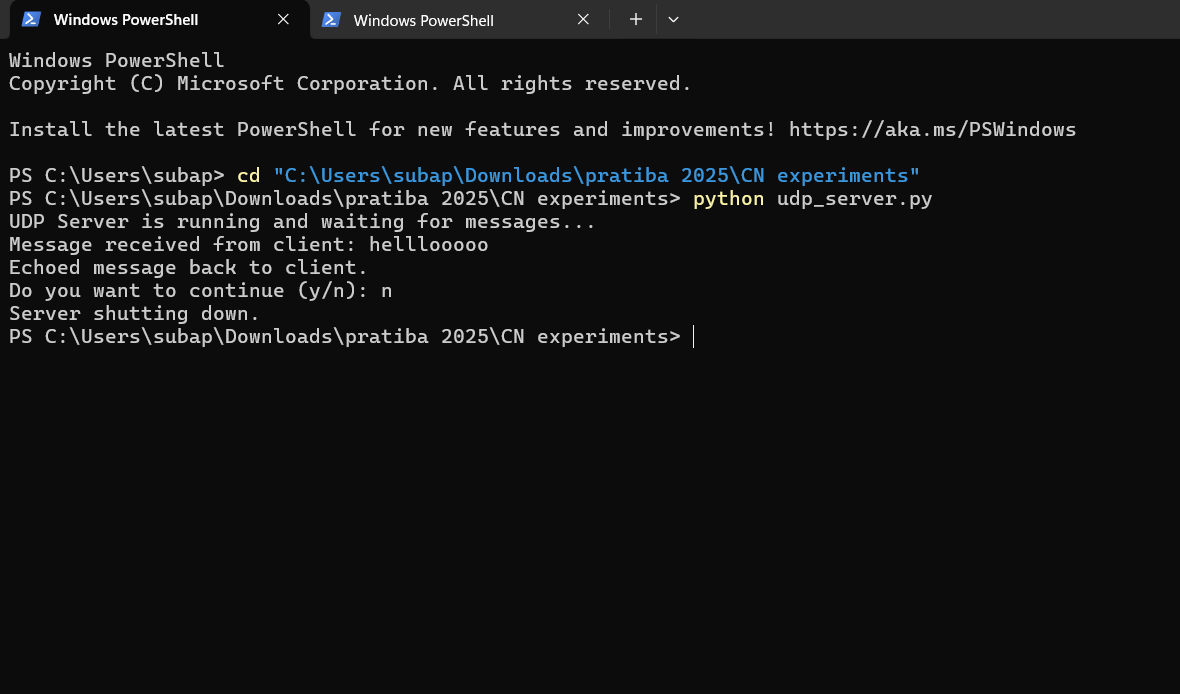
if choice.lower() == 'n':

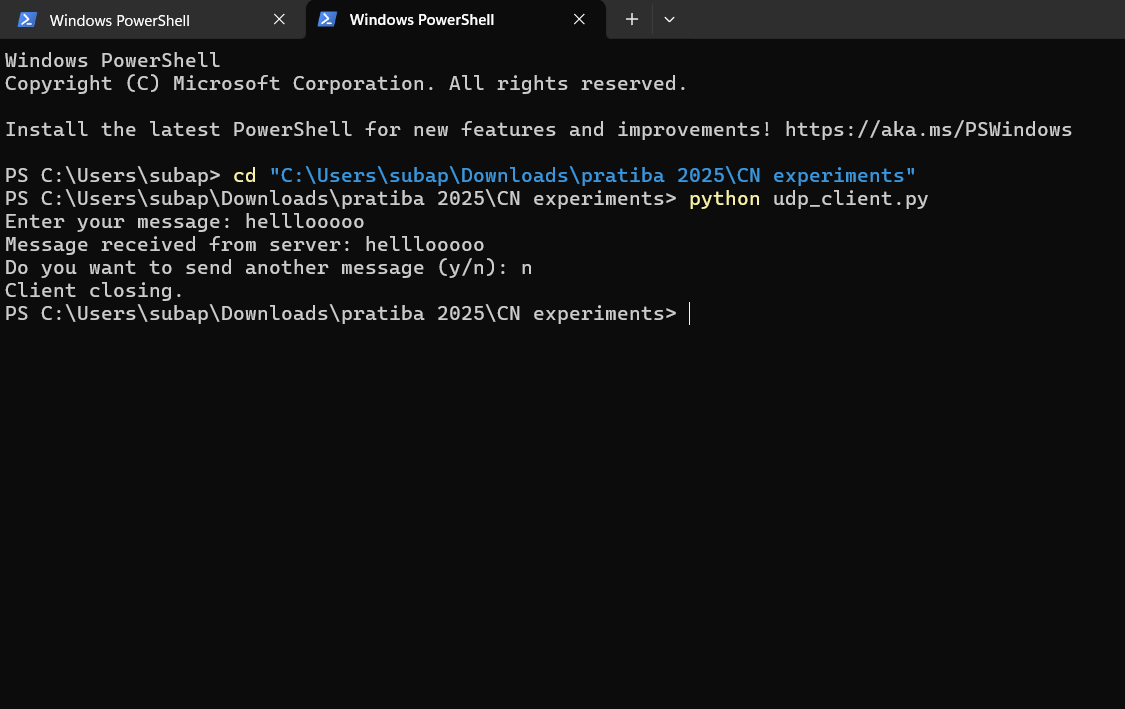
print("Client closing.")

break

client\_socket.close()

**OUTPUT:**

****

****

**RESULT:**

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