Assignment 8

Implementation of TCP/UDP Socket Programming

NAME: Shirish Manoj Bobde

Reg. No.: 812

Roll No.: ECE/21152

Problem Statement

Implement a client-server program using TCP/UDP sockets in Python for handling multiple clients on the server with multithreading. Your program should allow multiple clients to connect to the server simultaneously and exchange messages. Each client connection should be handled in a separate thread.

Code:

**Client**

import socket

def start\_client():

    host = "127.0.0.1"

    port = 8888

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

    client\_socket.connect((host, port))

    while True:

        message = input("Enter message to send (type 'quit' to close connection): ")

        if message == "quit":

            break

        client\_socket.send(message.encode("utf-8"))

        response = client\_socket.recv(1024)

        print(f"Server response: {response.decode('utf-8')}")

        server\_message = client\_socket.recv(1024)

        print(f"Server message: {server\_message.decode('utf-8')}")

    client\_socket.close()

if \_\_name\_\_ == "\_\_main\_\_":

    start\_client()

**Server**

import socket

import threading

def handle\_client(client\_socket, client\_address):

    print(f"Accepted connection from {client\_address}")

    while True:

        data = client\_socket.recv(1024)

        if not data:

            break

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

        print(f"Received message from {client\_address}: {message}")

        response = f"You sent: {message}"

        client\_socket.send(response.encode("utf-8"))

        server\_input = input(f"Enter message to client {client\_address}: ")

        client\_socket.send(server\_input.encode("utf-8"))

    print(f"Connection from {client\_address} closed")

    client\_socket.close()

def start\_server():

    host = "127.0.0.1"

    port = 8888

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

    server\_socket.bind((host, port))

    server\_socket.listen(5)

    print(f"Server listening on {host}:{port}")

    while True:

        client\_socket, client\_address = server\_socket.accept()

        client\_thread = threading.Thread(target=handle\_client, args=(client\_socket, client\_address))

        client\_thread.start()

if \_\_name\_\_ == "\_\_main\_\_":

    start\_server()

Output

