import socket

# Define host and port to listen on

import threading

HOST = 'localhost'

PORT = 12345

# Function to handle client requests

def handle\_client(c\_socket):

while True:

data = c\_socket.recv(1024) # Receive data from client

if not data:

break # No more data from client

# Process data (implement your protocol logic here)

response = process\_data(data)

c\_socket.send(response) # Send response back to the client

c\_socket.close()

# Function to process received data (implement your protocol logic here)

def process\_data(data):

# Implement your logic to parse and respond to the data here

# Example: echo the data back

return data

# Create a socket object

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

# Bind the socket to the specified host and port

s\_socket.bind((HOST, PORT))

# Start listening for incoming connections

s\_socket.listen(5)

print(f"Listening on {HOST}:{PORT}")

while True:

c\_socket, addr = s\_socket.accept() # Accept a new connection

print(f"Accepted connection from {addr[0]}:{addr[1]}")

c\_handler = threading.Thread(target=handle\_client, args=(c\_socket,))

c\_handler.start()

import socket

s\_host = 'localhost'

s\_port = 12345

# Create a socket object

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

# Connect to the server

c\_socket.connect((s\_host, s\_port))

while True:

msg = input("Enter a message to send to the server: ")

c\_socket.send(msg.encode()) # Send the message to the server

data = c\_socket.recv(1024) # Receive the response from the server

print("Received from server:", data.decode())

# Close the socket when done

c\_socket.close()