CS 461

Lab Assignment 1

Name: Gandhi Dhruv Vipulkumar

Institute ID: 202151053

Date: 2-9-2024

Q. Implement basic Client-Server in distributed environment

**Server.py**

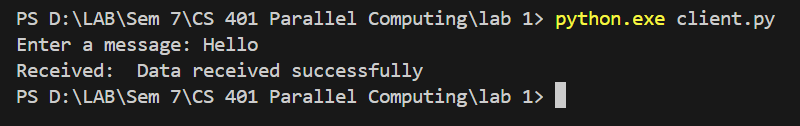
import socket  
  
# Define the host and port for the server to listen on  
# Localhost, meaning the server will only be accessible from this machine  
HOST = '127.0.0.1'  
PORT = 4000  # Port to listen on  
  
# Create a socket object using IPv4 addressing (AF\_INET) and TCP (SOCK\_STREAM)  
server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
  
# Bind the socket to the host and port, making the server ready to accept connections  
server\_socket.bind((HOST, PORT))  
  
# Enable the server to accept connections, with a default backlog of connections  
server\_socket.listen()  
  
print('Server listening on port', PORT)  
  
# Server loop to continuously accept and handle connections  
while True:  
    # Accept a new connection from a client  
    connection, address = server\_socket.accept()  
    print('Connected by', address)  # Display the client's address  
  
    # Receive data from the client (up to 1024 bytes)  
    data = connection.recv(1024)  
  
    # If no data is received, the connection is likely closed  
    if not data:  
        break  
  
    # Print the received data, decoding it from bytes to string  
    print('Received Data: ', data.decode())  
  
    # Send a response back to the client confirming receipt of the data  
    connection.sendall(b'Data received successfully')  
  
    # Close the connection with the current client  
    connection.close()  
  
# Close the server socket when done  
server\_socket.close()

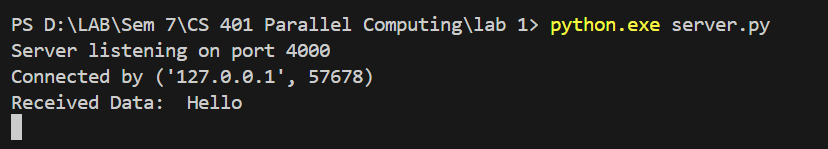
**Client.py**

import socket  
  
# Define the host and port to connect to the server  
# The server's hostname or IP address (localhost in this case)  
HOST = '127.0.0.1'  
PORT = 4000  # The port used by the server  
  
# Create a socket object using IPv4 addressing (AF\_INET) and TCP (SOCK\_STREAM)  
client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
  
# Connect the socket to the specified server and port  
client\_socket.connect((HOST, PORT))  
  
# Prompt the user to enter a message to send to the server  
message = input('Enter a message: ')  
  
# Send the user's message to the server, encoding it into bytes  
client\_socket.sendall(message.encode())  
  
# Wait to receive a response from the server (up to 1024 bytes)  
data = client\_socket.recv(1024)  
  
# Print the server's response, decoding it from bytes to string  
print('Received: ', data.decode())  
  
# Close the connection to the server  
client\_socket.close()

**Testing Phase:**

Sending message: Hello (from client to server)





**Conclusion:** Message received successfully to the server