A Project Report

on

TOPIC

Client -server Model

By:

Priya Jain
Rachit Bundela
Sachin Hedge

ABSTRACT

This project is the detailed overview in developing a client server-based application using socket programming in a distributed computing environment. we developed a client -server based application for Login and signup of the user. CLI, socket programming, python modules, exceptions are also considered in the development stage. The communications between client server application processes using socket mechanism were mainly analyzed. The main objective of this project is to demonstrate the principles and concepts behind the socket programming as well as libraries available in python.

INTRODUCTION

In client-server model, each computer terminal or process on the network could be a client or a server. A “client” is a program or a computer terminal that allows users to access and view its interfaces. Every client communicates and connects with each through server. In an overall concept, client serves as the one who initiates the communication wherein the server is the one who waits passively to respond to the client’s request.

On the other hand, a socket is an abstraction which allows program to send and receive data. The most essential characteristic of socket programming that drive programmers to design a program is due to its transparency. It simply means that, whether a socket program is designed and written in Python language, it still has the ability to communicate with other socket program which is designed and written in other programming languages such as C or C++.

DESCRIPTION

A-Client Server Communication:

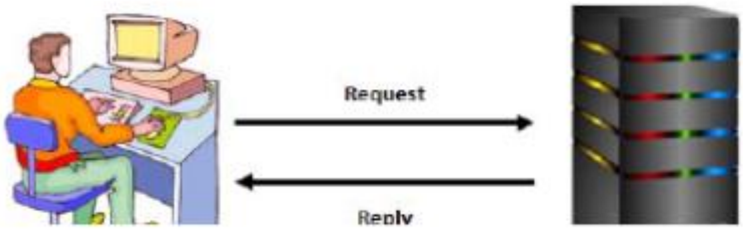


Fig:1.1

Fig 1.1 shown above describes the use of client server model based on its request – reply protocol. In this model, the client directly sends his request to the server asking for an appropriate service, and later server does the work and returns immediately the data or error code as a response back to the client. The server offers various services to the clients based on its client's request.

B. Sockets and Socket Programming: In distributed computing environment, client server application (consist of client program and server program) can be designed using socket programming. Socket is a two-way communication link between client and server programs that are running in a network environment. It has an effective communication mechanism between two computers. In Python programming, socket module are used to represent connections between the client and the server. These sockets connect two programs and implement two sides of the program (client and server).

The client server program allows user to run the client software to create a query. The client connects and sends query to the identified server through the use of socket (client side). The server once received the request from the client analyzes the query and sends the result directly to the client.

C. Communication Protocol: There are two communication protocols that are commonly used in socket programming, the datagram communication and stream communication. The datagram communication protocol is popularly known as UDP (user datagram protocol) is considered a connectionless protocol. This means that every time the client sends datagram, it also requires them to send the local socket descriptor as well as the receiving socket’s address. However, the stream communication protocol, popularly known as TCP (transfer control protocol) is a connection-oriented protocol. This means that in order to establish communication over this type of protocol, a connection must be established first between two identified sockets. One socket listens for a connection request (main responsibility of the server) and the other socket asks for a connection (main responsibility of the client) in order to establish connections. Once they have already established the connection, it can be used to transmit data in both directions.

Proposed Method

The application was designed using socket programming in Python supported by TCP datagram. The client server-based application has many functions. The two programs (client and server) were simulated, demonstrated and analyzed.

A-Application Concept:

This research study used a specific scenario in designing the client server application in order to demonstrate the proper implementation of socket programming. The design and implement a client server-based application to enter the user detail of signup and login. This application implements communication between a client and a server. Further, the application involves multiple client and one server; with the following requirements:

a. The client sends enquiries to the server;

b. The server executes the commands and displays whatever the server sends back to the client;

c. The application must be able to handle multi-users access on the client side;

e. The server manages all the requests coming from the client and maintains user's account information in the SQL database; and

f. The application requires log-in and password for security purposes.

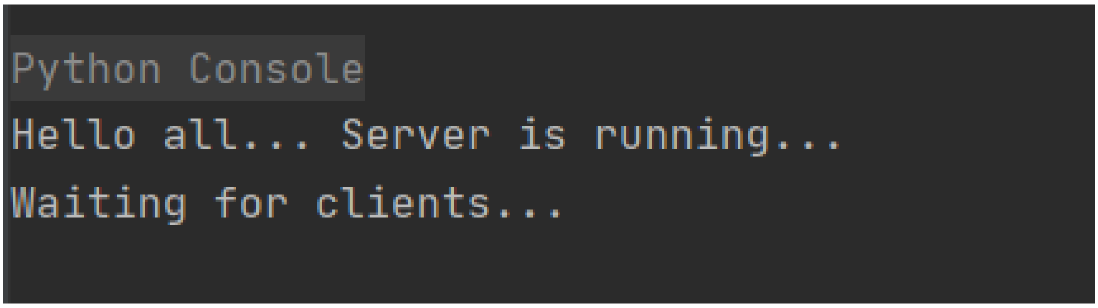
g-After establishing connection user will have the choice to Register or Login.

h-If the user register then the entry will be saved to the database.

i-If the user enters for the login, then various operation can be performed by the user accordingly.

B-Interface Design:

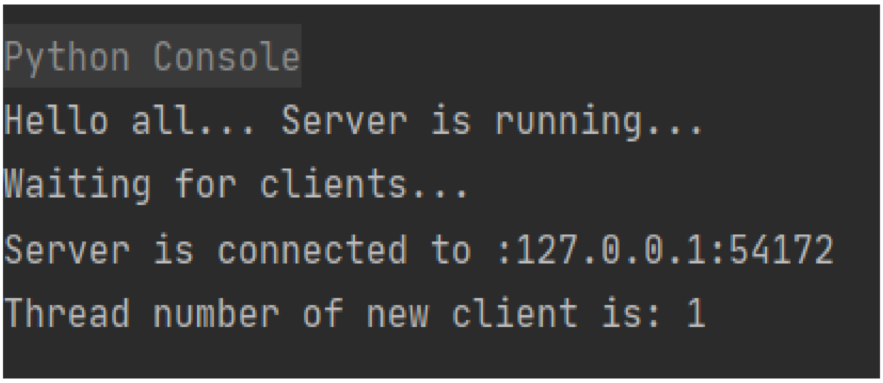
Fig:1.2 illustrates the authentication procedure of the application whereas it requires users to perform authentication access in order for them to access. Established connections with the use of temporary ports assigned. Demonstrate how the two programs is connected through sockets with the use of port. The two programs were successfully connected using the port 1234. The ports of two programs are essential as it



```
Python Console
Hello all... Server is running...
Waiting for clients...
```

Fig-1.2

Fig:1.3 Established connections with the use of temporary ports assigned

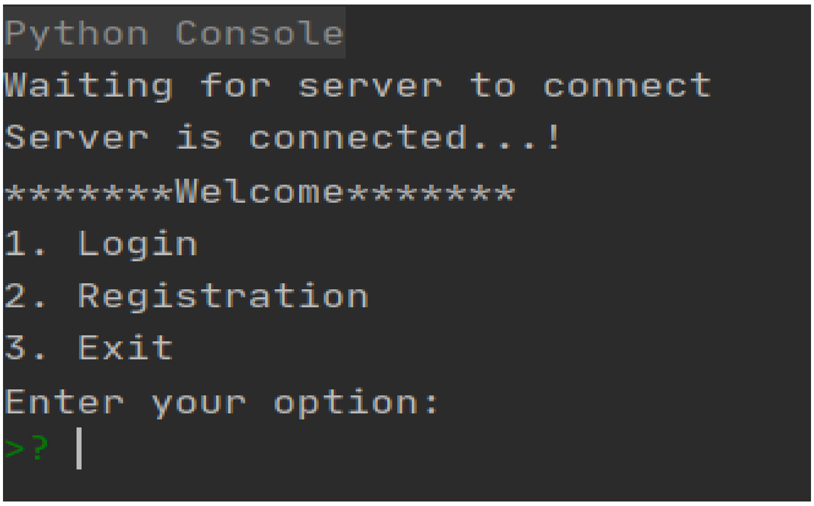


```
Python Console
Hello all... Server is running...
Waiting for clients...
Server is connected to :127.0.0.1:54172
Thread number of new client is: 1
```

Fig:1.3

C. Main User Interface:

Fig:1.4 Main User Interface of the Client-Server application. Demonstrates the main user interface of an application. The interface includes labels that increase the usability and user-friendliness of the application.



```
Python Console
Waiting for server to connect
Server is connected...!
*******Welcome********
1. Login
2. Registration
3. Exit
Enter your option:
>?
```

Fig:1.4

SYSTEM REQUIREMENTS

Hardware Requirements:

Device: Laptop or Desktop computer

Processor: Intel core i5 8th Gen

Ram: 2 GB

Hard Disk: 512GB

Software Requirements:

Platforms: MySQL, PyCharm IDE

Language: Python

Technological requirements:

Python-Socket Programming, Multithreading, Python-database connectivity.

Conclusion and Future Work

This project presents the overview and the detail implementation of the Client-server application using socket programming method. In the development of this study, we have chosen Python programming language as it covers a wide range of functions and classes. In addition, we found out that programming style, concepts, functions, socket programming are easily done in Python programming language. The designed application is used to simulate scenario and able to illustrate the use of socket programming and how it works in the real-world environment. Thus, we would like to recommend that the application can be designed for the user by having Hosting with Heroku to enable multiple clients on different hardware to access the server for future enhancement and further improvement. Other functionalities can also be embedded and integrated in order to design a newly updated version of this application in the future.

References

1-https://codezup.com/socket-server-with-multiple-clients-model-multithreading-python/

2- https://wiki.python.org/moin/UdpCommunication

3-https://www.pubnub.com/blog/socket-programming-in-python-client-server-p2p/

4- https://www.tutorialspoint.com/python/python_networking.htm