PROJECT REPORT

"CHATROOM"

SubmittedBy:
PUNEET SHARMA 2K18/IT/092
OMKAR SINGH 2K18/IT/079

Under The
Guidance Of:
Prof.Anamika
chauhan

Departmentof
Information Technology
November,2020

DEPARTMENT OF IT
DELHI TECHNOLOGICAL UNIVERSITY
Shahbad Daulatpur, Main Bawana Road, Delhi-110042, India

CONTENTS

ACKNOWLEDGEMENT

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Prof.ANAMIKA CHAUHAN under whom we have carried out the project work. Her incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish to reciprocate in full measure the kindness shown by Mr. JASRAJ MEENA Sir who inspired us with his valuable suggestions in successfully completing the project work.

Finally, we must say that no height is ever achieved without some sacrifices made at some end and it is here where we owe our special debt to our parents and our friends for showing their generous love and care throughout the entire period of time.

NOVEMBER 2020

Place: IT DEPARTMENT,DTU
DELHI, INDIA

PUNEET SHARMA 2K18/IT/092
OMKAR SINGH 2K18/IT/079

INTRODUCTION

Here we have to tried to Design an End to End Networking application in Python using Socket Programming as directed.
Now coming to the details about our application. We have created a application named ps chatroom which is compatible in running in both computer and mobile(currently for android).
Two persons from two different phones or desktop can also get connected to the chatroom.
To start chatting, client should first connect to server

Several network systems are built to communicate with one another and are made available through service-oriented model. In this project, we use the client server model to develop a secured Client-Server chat application.

A client-server chat application consists of a Chat Client and a Chat Server and there exists a two way communication between them.

In general, the server process will start on some computer system; in fact, the server should be executed before the client. Server usually initializes itself, and then goes to wait state or sleep state where it will wait for a client request. After that, a client process can start on either the same machine or on some other machine. Whenever the client wants some service from the server, it will send a request to the server and the server will accept the request and process it. After the server has finished providing its service to the client, the server will again go back to sleep, that is, waiting for the next client request to arrive. This process is repeated as long as the server processes is running. Whenever such request comes, the server can immediately serve the client and again go back to the waiting state for the next request to arrive.

MAIN CONCEPTS USED

- Python
- Socket programing(server-client)
- Kivy framework
- Buildozer

SOFTWARE/TOOLS USED

- Window 10
- Python 3.8
- PyCharm 2020.2.1
- Kivy framework
- Linux in virtualbox
- Buildozer tool

BACKGROUND OF THE PROJECT

Client server model is the standard model which has been accepted by many for developing network applications. In this model, there is a notion of client and notion of server. As the name implies, a server is a process (or a computer in which the process is running) that is offering some services to other entities which are called clients. A client on the other hand is process (which is running) on the same computer or other computer that is requesting the services provided by the server.

A chat application is basically a combination of two applications:

Server application
Client application

Server application runs on the server computer and client application runs on the client computer (or the machine with server). In this chat application, a client can send messages to anyone who is connected to the server.

To establish a server connection, a server needs to be created and attachhed, which is where the server listens for connections.
Server Execution: At server the side, a thread is created which receives numerous clients' requests. It also contains a list in which Client's name and IP addresses are stored. After that, it broadcast the list to all the users who are currently in chatroom and when a client logs out then server deletes that particular client from the list, update the list and then broadcast the list to all available clients.

Client Execution: A client firstly must have to register itself by sending username to the server and should have to start the thread so that system can get the list of all available clients. Then any of two registered clients can communicate with each other.

STATEMENT OF THE PROBLEM

The client-server communication model is used in a wide variety of software applications. Where normally the server side is sufficiently protected and sealed from public access, but client applications running on devices like mobile phones and desktops are considered insecure and exposed to security threats.

The main weakness of client-server chat application is that there is no security provided to messages which is transferred between clients. Any unauthorized client can hack the client account and can change the data. This is the main objective of this project (To develop a secured Client-Server Chat Application).

OBJECTIVES OF THE PROJECT

The main objective of this project is to create a mobile and desktop application using python by which we can connect between server to client or between multiple clients and chat among them

Or we can also say that aim of this project is to develop a reliable and secure network programming (Client-Server chat model) which can perform a multithreaded server client chat application based on phython socket programming using client and server

SCOPE OF THE PROJECT

The project shall consider among other things the following issues:

To provide a better understanding of how network programming in python works.
Develop a reliable end to end network communication for a Client-Server chat application.
Analyses of network programming in python (Multi Client-Server Chat applications) for better understanding of the solutions.
Conduct an experimental result in order to establish the parameter of the problem.
In conclusion, suggest ways the problems can be eliminated and recommends how the problems can be prevented.

DEFINITION OF TERMS

Socket programming: is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.

Client: A client is a system that accesses or desires for a service made accessible by a server.

Server: A server is a system (hardware or software) program running to provide the service requests of other system programs.

Network: This refers to a system where computers are linked to share software, data, hardware and resources for the benefit of users.

IP: This refers to Internet Protocol; it is the reasonable network address of device on a network. It is notational called dotted-decimal (for instance: 128.1.0.1).

KIVY FRAMEWORK

- Kivy is an opensource multi-platform GUI development library for Python and can run on iOS, Android, Windows and Linux. It can be used to develop applications that make use of innovative, multi-touch UI. kivy enables us to build an app once and use it across all devices, making the code reusable and deployable, allowing for quick and easy interaction design and rapid prototyping.

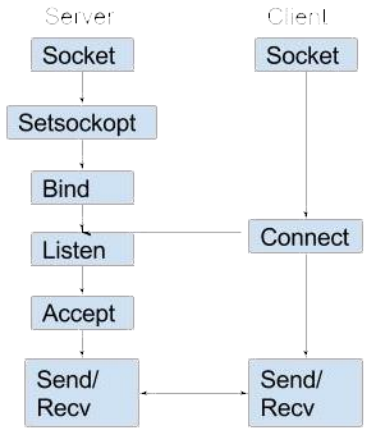
This easy to use framework contains all the elements for building an application such as:

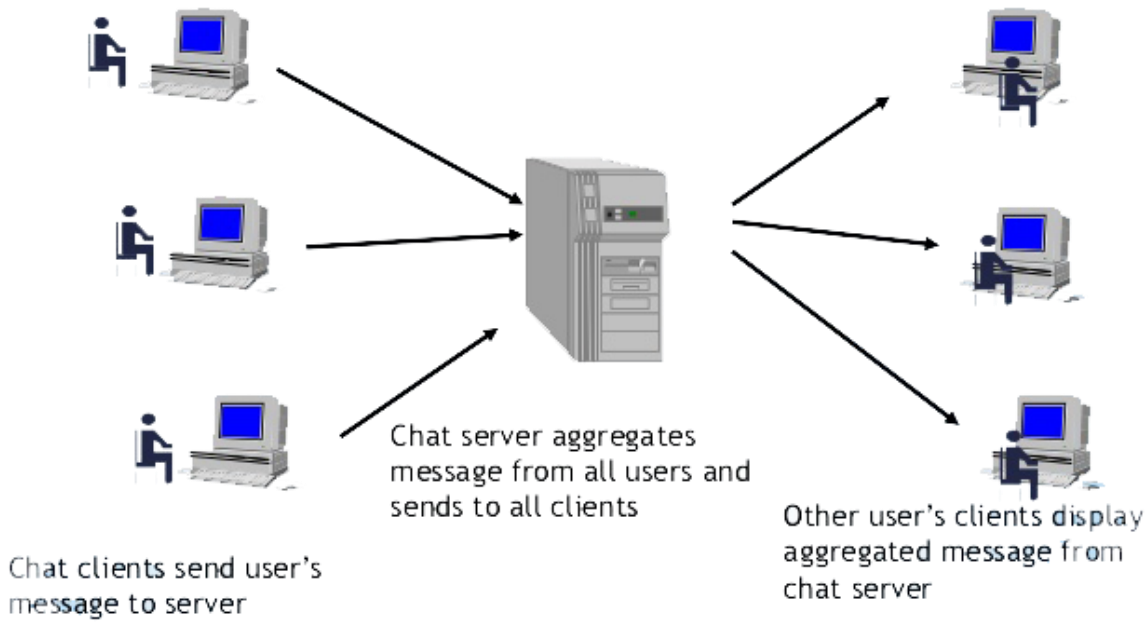
- Extensive input support for input devices such as mouse, keyboard, TUIO, and OS-specific multi-touch events
- A graphic library using only OpenGL ES 2
- A wide range of widgets built with multi-touch support
- Using this we have add graphical interface, buttons, events, changing screen, scheduling tasks etc. in our application.
- This way we are able to build a desktop chat app.

BUILDOZER

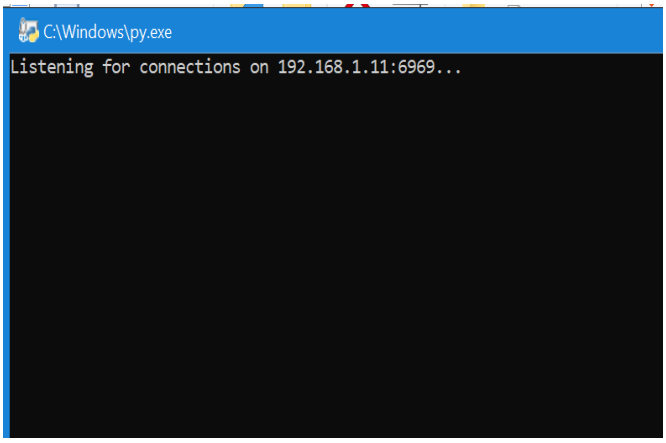
- Buildozer is a tool for creating application packages.
- This tool is not compatible with windows so for this, a virtual machine and then ubuntu 20.0.4 image is installed to use buildozer with its dependencies.
- The goal is to have one "buildozer.spec" file in your app directory, describing your application requirements, permissions and settings such as title, icon, included modules etc. Buildozer will use that spec alongwith our main python app and some minor files to create a package for Android.

Flow chart:

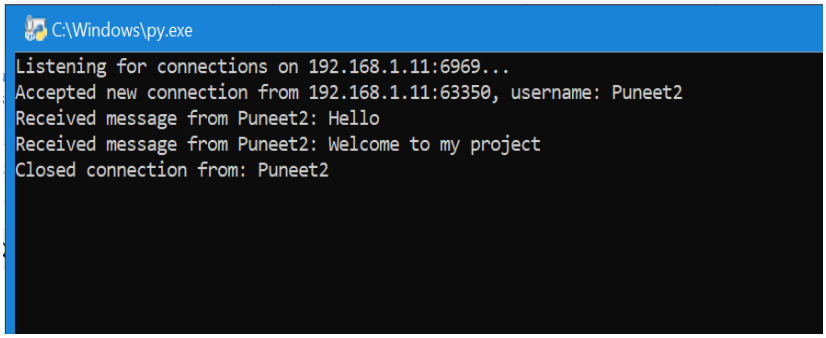




SCREENSHOTS AND WORKING

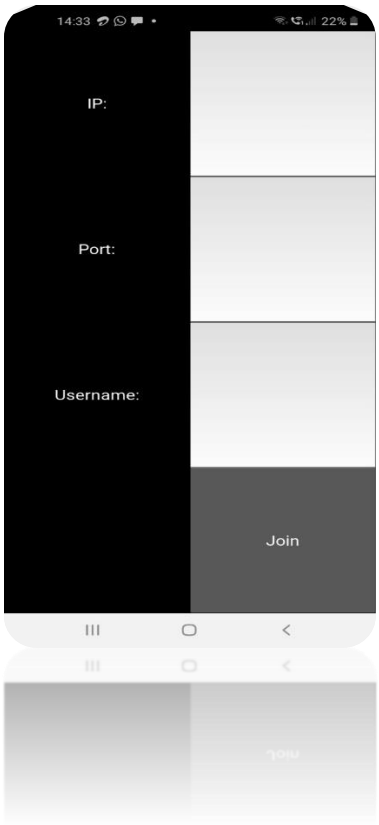


- First of all, the server must be started so that the clients an connect to it.
- We have to set the IP address and any free port number of the host machine.



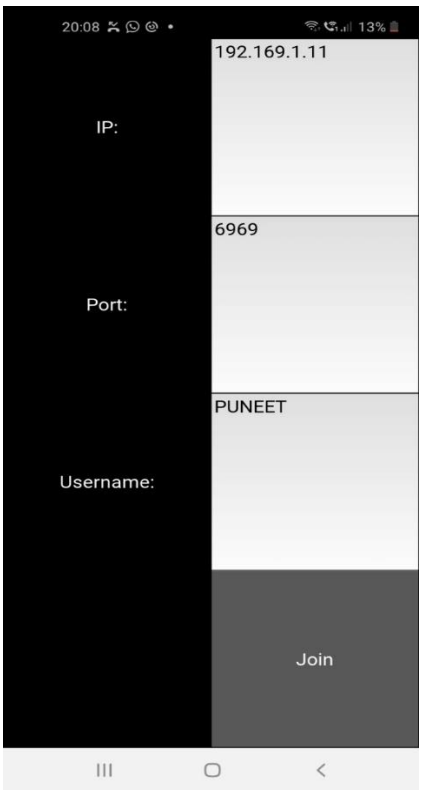
- When a client connect using the IP address of the host, it will display its information.
- Beside this, it can also show received message from all connected clients.

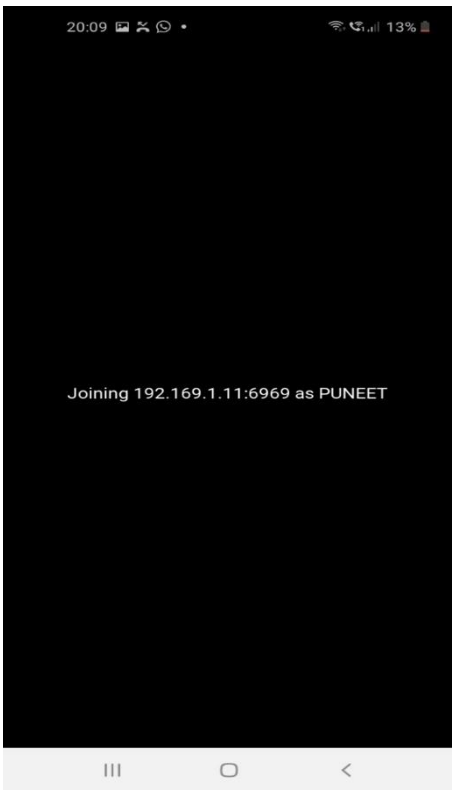
APPLICATION INTERFACE

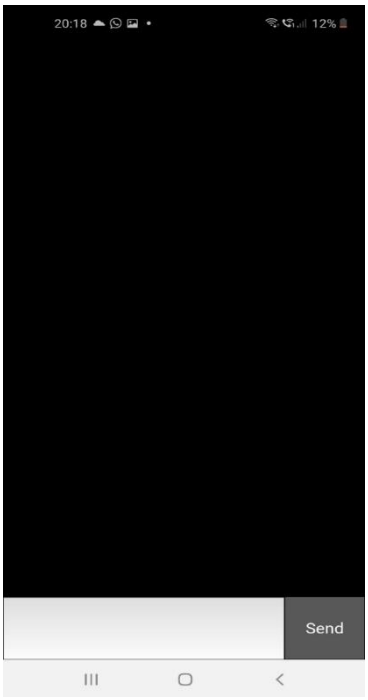


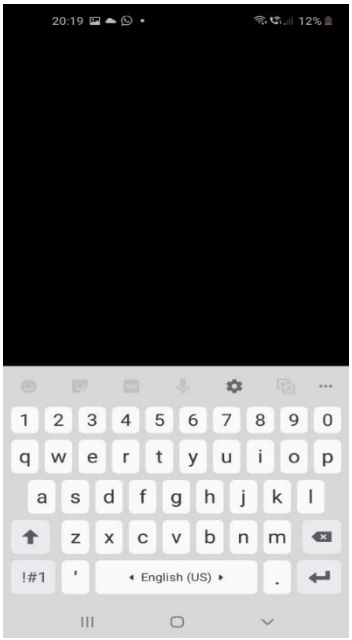
At the first page user need to add the host server's ip address and pot number and the username also.

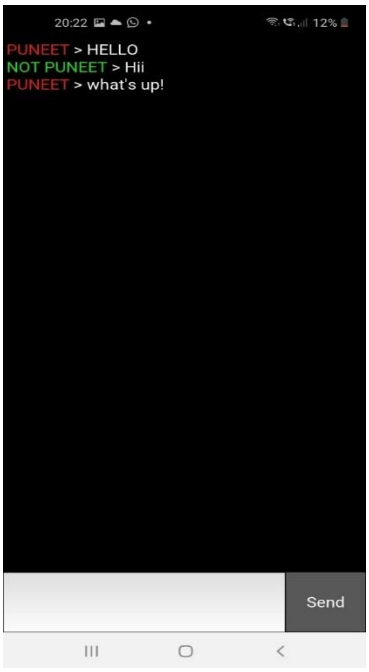
If the client enter correct IP address and port number then he will be able to use chat feature and the server will be notified for this connection.

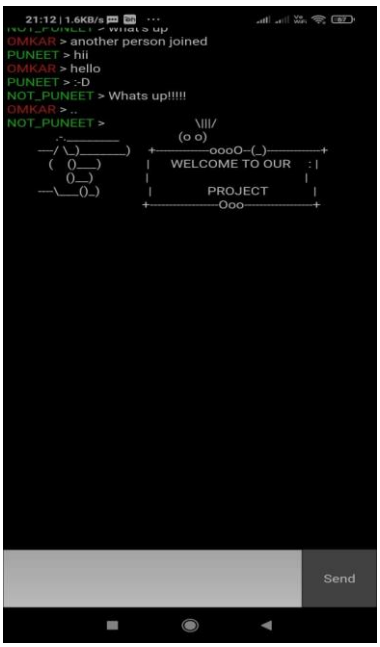


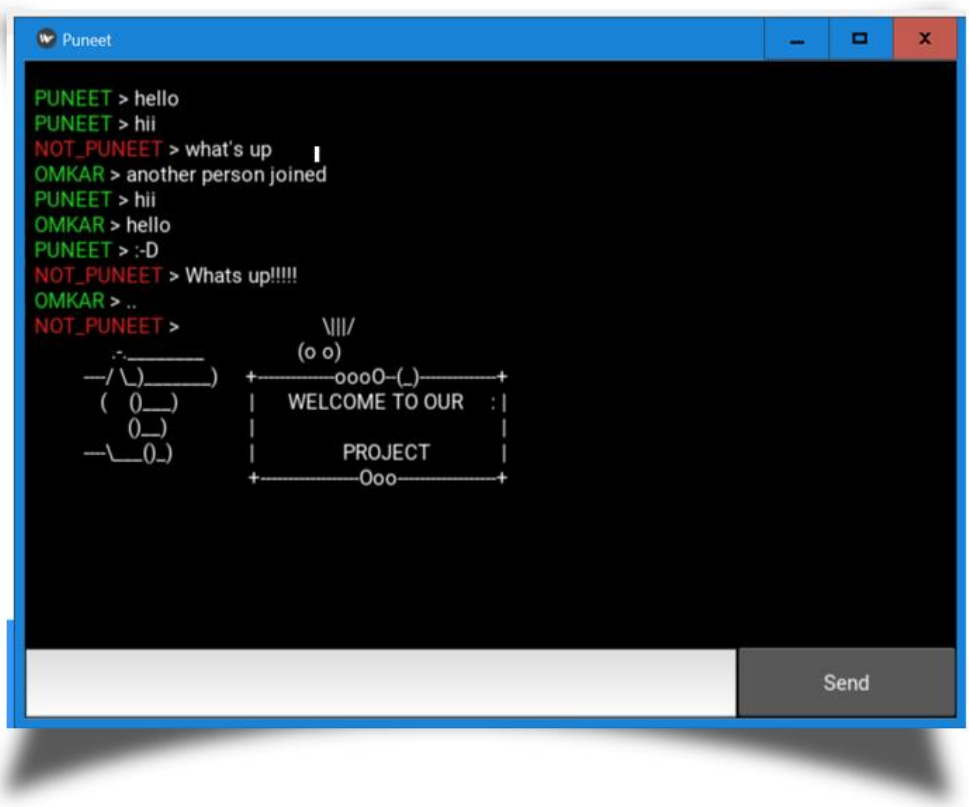



20:18
12%
Send


20:19
12%
1 2 3 4 5 6 7 8 9 0
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
!#1
English (US)


20:22
12%
PUNEET > HELLO
NOT PUNEET > Hii
PUNEET > what's up!
Send


21:12 | 1.6KB/s
OMKAR > another person joined
PUNEET > hii
OMKAR > hello
PUNEET > :-D
NOT_PUNEET > Whats up!!!!!
OMKAR > ..
NOT_PUNEET >
\|||/
(o o)
WELCOME TO OUR
PROJECT
Send



CONCLUSION:

Right now, we are just dealing with the text communication. In future this software may be extended to include features such as:
Voice chat- this will enhance the application to a higher level where the communication will be possible via voice calling as in telephone.
File Transfer- this will enable the user to send files of different formats to others via chat application.