Group Chatting using Socket Programming

Abstract—The latest development of the Internet has brought the world into our hands. Everything happens through internet from passing information to purchasing something. Internet made the world as small circle. This project is also based on internet. This paper shows the importance of group chat application in day today life and its impact in technological world. This project is to develop a group chat system based on socket programming concept. The application allows people to transfer messages through the server client model .This online system is developed to interact or chat with one another on the Internet. It is much more reliable and secure than other traditional systems available as it uses a TCP connection. Socket programming and client server concepts were used to develop the group chat application. This application is developed with proper architecture for future enhancement. It can be deployed in all private organizations like Colleges, IT parks, etc.

Index Terms—Socket Programming, TCP, Server-Client

I. Introduction

The chatting application has huge impact on day to day life. There are numerous chatting application available in this world. Each application has different additional features varying from other applications. These application organizations compete with each other and add some competing features during each release. They have reached people much and have an impact on people's life. People find a better application from an available internet application which they feel much reliable and secure. Some of the available charting applications that are available in these days are Whatsapp, Facebook, Instagram, Hike, etc...

The above mentioned applications have billion users all over the world. Those companies are one of the top companies in the world. They have higher revenue per year and have many employees for their organizations developing additional features to compete with other organizations during their each release. These applications have different features and follows different ways to ensure security of their user data.

Today a data theft is the major crime and most people are involved in it. There are many cases being filed these days about personal data loss. So the organizations have to ensure the security from data loss by the third party

Fig. 1.

data crisis. The basic chatting system should involve both sending and receiving processes simultaneously.In this application both sending and receiving messages simultaneously happens through Socket Programming.

Chatting refers to the kind of communication done with the help of the internet which present live transmission of text messages from sender to receiver. Online chatting can be termed as the point-to-point, one sender-to-one receiver, or one sender-to-many receiver. It also features voice, video, and also web conferencing services. Chatting over the internet has made it a lot easier to have a conversation with anyone. People from any corner of the world are now able to contact the person on the other edge.

Chat application or platform that enables users to instant message and connect with each other through their computers or mobile devices. A chat application makes it easy to communicate with people anywhere in the world by sending and receiving messages in real time.

II. Objectives

- To implement a group chat system
- To develop a two way communication system
- To enable easy and fast way of communication between people.
- To make people get connected to others at anytime, from anywhere.
- To have unlimited size to store message data.

III. Design of the program

TCP is used in socket programming as the tranport layer protocol. TCP stands for Transmission Control

Protocol. TCP enables programs to exchange messages over a network. TCP is a connection-oriented program unlike the UDP, which is a connectionless program. TCP is designed to send packets across internet and ensure the successful delivery of message over a network.

The TCP port is a unique number assigned to different applications. Each protocol and address have a port known as a port number. The TCP and UDP (User Datagram Protocol) protocols mainly use the port numbers. A port number is a unique identifier used with an IP address. A port is a 16-bit unsigned integer, and the total number of ports available in the TCP/IP model is 65,535 ports. Therefore, the range of port numbers is 0 to 65535. In the case of TCP, the zero-port number is reserved and cannot be used, whereas, in UDP, the zero port is not available.

A single client can have multiple connections with the same server or multiple servers. The client may be running multiple applications at the same time. When the client tries to access some service, then the IP address is not sufficient to access the service. To access the service from a server, the port number is required. So, the transport layer plays a major role in providing multiple communication between these applications by assigning a port number to the applications.

IV. Architecture

This application has been implemented based on client server model. A client is a program that runs on the local machine requesting service from the server. A client program is a finite program means that the service started by the user and terminates when the service is completed. A server is a program that runs on the remote machine providing services to the clients. When the client requests for a service, then the server opens the door for the incoming requests, but it never initiates the service. A client and server networking model is a model in which computers such as servers provide the network services to the other computers such as clients to perform a user based tasks. This model is known as client server networking model.

A. Server

The server is implemented as a singleton class. The main thread opens a server socket on the local inet address and port 8080, which has been arbitrarily chosen and hard coded. It then waits for clients to connect to it. When a client connects

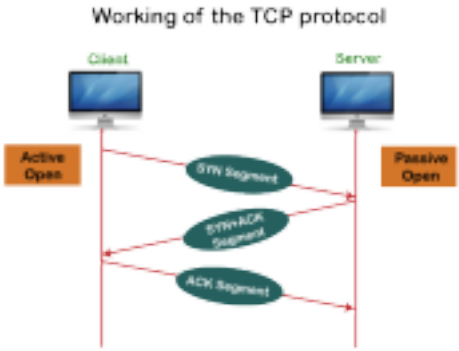

Fig. 2. Working of TCP

to the server, it creates a separate thread as well as a User object dedicated to it. The server maintains a hashmap of User objects associated with the clients connected to it hashed against the user name. The thread dedicated to the client opens a buffered stream to read input messages from the client and a print writer stream to send messages to the client.

The messages going from a client to the server are strings with at least two words separated by a space. The first word is the header, which is the destination username. The messages going from the server to a client are strings with at least two words separated by a space. The first word is the header, which is the source username.

The thread dedicated to the client waits for incoming messages in the input buffered stream, tokenizes the messages into the header and the message string, and checks if the destination user is online by looking into the hash map. If the user is online, it attaches a new header displaying the source user name to the beginning of the message string and copies it to the print writer stream of the User object associated with the destination user. If the destination user isn't online, it sends back error message to the source client.

Messages destined for the server(control messages) contain the string server as the header. Control messages are sent to know who is online and to exit. In case of the former, the thread dedicated to the user, gets all the keys from the hash map and returns a string containing the online user names to the user. When exit is requested, the thread closes the streams and the socket associated with the client and exits.

B. Client

The Client is implemented using two threads, one each for incoming and outgoing messages. The main thread opens the socket and connects to the server. It

then opens input buffered stream and print writer for incoming and outgoing messages respectively. It also creates a buffered stream to read from the console. The main thread takes care of the outgoing message, while another thread deals with incoming

Fig. 3. Flow of Client-Server

messages. In order to send a message, the user types the destination user name following by the message string on the console, which is then read into the buffered stream reading from the console. The main thread creates the formatted message by adding the destination user name to the beginning of the message string and writes it to the print writer stream. Control messages have the string server as the header.

The reader thread, waits for incoming messages on the buffered stream. When a message arrives, it tokenizes it into the header and the message string and prints ¡source user¿ says: ¡message string¿ on the console. Responses from the server are displayed as server says: ¡message string¿.

When exit is requested, the client sends exit message to the server, waits for its response, and closes the streams and exits.

C. Working Flow

- A static Server socket is created in beginning which is then bind with host and port
- After server instantiation Socket in particular host, it begins to listen in the particular port. Then the server is made to accept the request from the client through the particular port
- After starting the server, it can accept the requests from clients
- The socket is instantiated in client side to connect to the server
- A new Server Thread using socket is created to accept all the requests from multiple clients
- After accepting the request both read and write operation occurs simultaneously, clients who request the server can communicate with each other and share resources
- After finishing the communication the socket is closed both in the client and server side

V. Limitations

There is no graphical user interface (GUI). So there are no chat windows and the user needs to write destination

Fig. 4. Working Flow

user name before message string. Also the user needs to enquire, who is online, rather than the server automatically updating the list of users logged in a window. There is no encryption of message strings being sent across the network. If the server fails, all of the system fails.

VI. Future Works

Further enhancements would be involved in the area of security, video call, large size transfer and some additional features that are required in the competing world. Other work is involving in implementation of the system in private networks.

VII. Outputs

The Following are the methods to get the required outputs of the group chat application-

- Coded for the client, server, and the string in C++

programming language

- Coded proto, server, and string header files
- Compiled the files
- Initialised the server first
- Ran the required number of clients in multiple terminals
- Named the clients uniquely
- Waited for connection with the server
- Message through various clients
- Exit/terminating the terminals

The following is the required output of the group chat application:

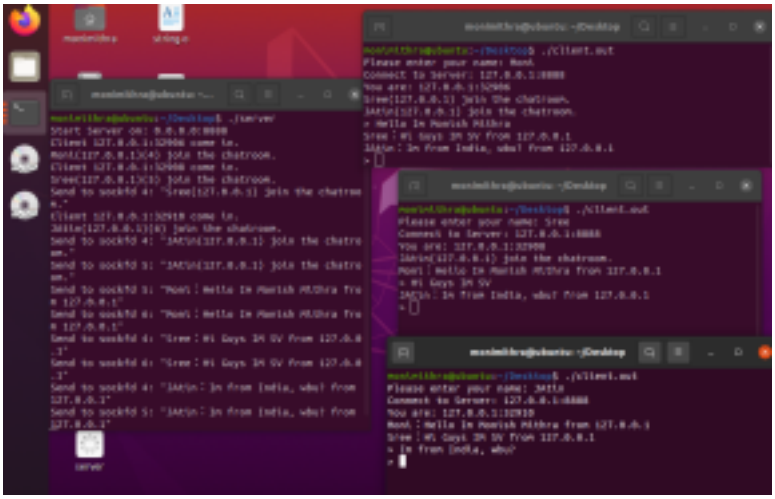

Fig. 5. Output

VIII. Conclusion

The chat application provides a better and flexible system for chatting. It is developed with recent advanced technologies in a way to provide a reliable system. Main advantages of the system are instant messaging, real-world connectivity, adding security, group chat, etc. This application can find better need in the market for most of the organizations aim at having private applications for them. Additional features will also be included in the system based on the public need which includes conference call, video chat. Location share, etc. based on the need.

References

[1] “Multiusers communicating system in client/server mode based on www”,by Shen Ruiming , Computer Engineering , 1998(2):53-58

[2] ”The design of instant communicating system in server side”, by Huan Kai,Tao Hongcai,Journal of Chengdu University of Information Technology,2006(4):535-538

[3] . ”Implementation of enterprise instant communicating system based on application layer with java programming”,by Lin Jianbing, Zou Jinan, vol.27, no. 6,pp. 56-61,July,2015.

[4] “The data visual development and application based on three layers structure ”,by Li Zuohong,Luo Zhijia, Microcomputer Information,2006(21):182-185

[5] The Java Tutorials, “Lesson: All about Sockets”.

[6] ”Implementation of enterprise instant communicating system based on application layer with java programming”,by Lin Jianbing, Zou Jinan, vol.27, no. 6,pp. 56-61,July,2015.

[7] “Private information retrieval,” by B. Chor, E. Kushilevitz, O. Goldreich, and M. SudanJ. ACM, vol. 45, no. 6, pp. 965–981, 1998.

[8] https://www.javatpoint.com/socket-programming