# WEBCHAT APP

Project report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

**Computer Science and Engineering/Information Technology**

By

ARYAN PAL SINGH (161236)

Under the supervision of

Dr. Hari Singh Rawat

to

Department of Computer Science & Engineering and Information Technology

**Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh**

# CERTIFICATE

## Candidate's Declaration

I hereby declare that the work presented in this report entitled **"WebChat App"** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from May 2020 to June 2020 under the supervision of Dr. Hari Singh **,** Assistant Professor (SG) , Department of Computer Science & Engineering and Information Technology.
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Aryan Pal Singh
Roll No - 161236

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Hari Singh
Assistant Professor (SG)
Department of Computer Science
& Engineering and IT
Dated:

## ACKNOWLEDGEMENT

I would like to express my gratitude to my mentor Dr.Hari Singh **who** guided my throughout the phase of making project **(WebChat App).** He helped me in the thorough understanding of fundamentals of project, understanding the basic functionalities, implementing the functionalities and making my project more responsive and much efficient. I would also like to extend my heartiest thanks to all members of ETA, Mysore for there support and guidance throughout my training period.

Secondly I would also like to thank my friends and all my mentors who helped me in completing and finalising project in given time.

# TABLE OF CONTENTS

# LIST OF FIGURES

# ABSTRACT

In this project, one can do live chat with others. For this we use server-client architecture. In this, we used to have a server and with this server different clients can connect. Now, in order to establish connection with server with clients we use the concept of socket programming. Now let me throw some light on socket programming.

Since for a machine to connect with other machine there is need of socket. So with the help of sockets server, can also connect to different clients and thus data transfer can take place in very efficient way.

In this server program will run on server and during the time server program is running, client will also run its program from respective computer and matching the IP address and port number of server connection will take place and data transmission can occur. Similarly, various clients can connect to server through server's port number. Thus, client will send request to server for connection and server will respond to request and thus live chat will take place on server between various clients connected to server.

# CHAPTER - 1

# INTRODUCTION

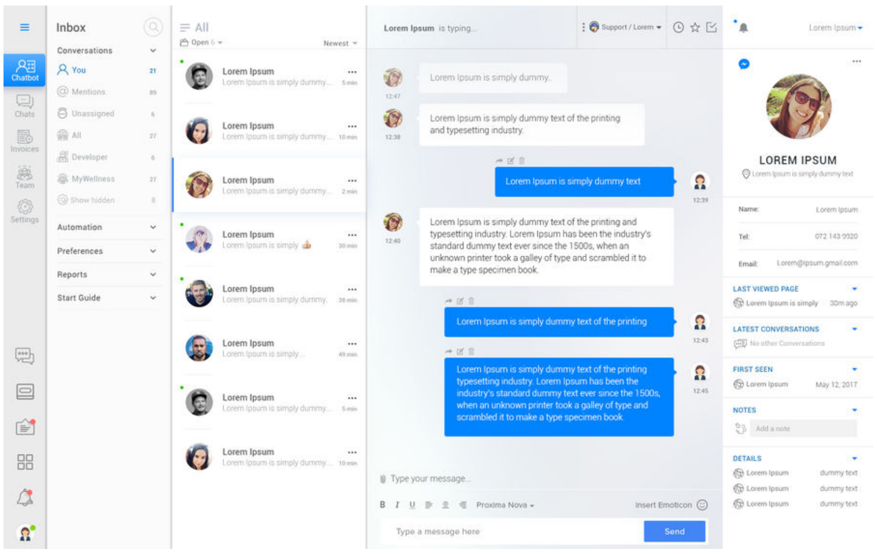
## 1.1 Introduction

This project is about how different users can communicate in real time with each. With real time it can said to communicate live with each other on a single platform. So to make this thing possible we have a communication platform for chatting which is client server architecture.In this application, we will have various clients who can join a private or public server and can communicate with other clients using that server. In this world we have different platforms which uses this concept like youtube. In youtube, when someone do live streaming you will see various people are chatting side by side of video in real time. So here there is a server of youtube and we have various users as clients who are connected to that server and it is server-client architecture through which it was possible that all users were able to have a communication with other clients in present time on a single platform.

Now in this chat app to communicate through machines of individuals and server we have a thing in each machine known as sockets. These are only sockets through which it was possible for different machines to establish connection with each other. This establishment of different machines is possible only due to socket programming. Due to socket programming it is possible for machines to have end to end communication and send data from one device to other. Now we will see about sockets in detail in system development section.

There is one more application where this web chat is brought in great use.

We have a very popular application known as Instagram where we use the concept of server-client architecture. When any user gets live on Instagram then he that user acts like a server and all its followers are clients who can chat with him live and all other followers can also see those messages.



Lorem Ipsum is simply dummy..

Lorem Ipsum is simply dummy text of the printing and typesetting industry.

Lorem Ipsum is simply dummy text

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

Lorem Ipsum is simply dummy text of the printing

Lorem Ipsum is simply dummy text of the printing typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

Figure 1.

Above is an to demonstrate the use of Web Chat. In this example, you can see that we have Web Chat application in which we have different users who are doing real time chat on a common platform which is its server. Each user's machine would be having its socket and server would be having its server through which connection took place between them and they were able to chat with each other.

In this application, we would be using the java and socket programming.
For sending data from machine M1 to machine M2, Output-stream is used and to receive data from machine M2 to M1 Input-stream is used. In socket programming client should know the IP address of server and port number of application in order to do distribution of data.

## 1.2 Problem Statement

In todays world everyone in there life is busy in race of achieving goals and for that they need to run a lot from here to there. Also these days, if any big meeting is to be done or any big gatherings are needed to be held then a person will think many times for attending it considering the factors of traffic jam on his way, facilities for having night stays there. Recently whole world is suffering from viruses like corona where it is impossible to have social gatherings. So to overcome these problem we need a platform where people can virtually attended meetings and do talking to each other in live. So web chat app can be a very good and efficient solution to this problem. Using socket programming and client-server architecture it is very easy to do remote chatting for users from there homes.

But practically it is not possible that everyone has a computer and he can do web chatting. So it is also platform independent. As these days a cheaper smartphone is available with  sockets as is capable of doing virtual live chats. So no matter how far you are the only thing required is any type of connection with server and one can use the benefits of web chat.

Now major question is how to achieve this communication. For this we will use client-server architecture approach. In this different clients located at different locations can connect to a common server by knowing the address of server and server will distribute the data of one server with other users. Now working of this chat app will be seen in implementation section.

## 1.3 Objective

The main objective or we can say that fundamental objective of Web Chat application is to provide chatting service to all users connected to a particular server. So that they can have easy, fast and real time communication with each other. Whenever a new user want to join the server, his/her request is accepted and then server respond with it and thus user gets connected and become a part of this live chat. Whatever data is transferred by a user to server is distributed to all users connected to that particular server.

## 1.4 Methodology

Main aim of Web Chat Application is to connect various users to a common server.

The main method used in this is server-client architecture. In this we will have a server program which will be executed initially and then client program will be executed. Then client will send request to the server by knowing the address of server and on reflection server will give response to that specific client and connection will be established between them. This is done due to the presence of sockets in each machine.

Then after setting up connection data will be sent out by a machine using output stream buffer to the server.

Server will accept data using input stream buffer and to distribute this data to other clients server will use its output stream buffer. Similarly other clients can also follow same method to do live web chatting.

## 1.5 Organisation of report

- Chapter 1 gives the basic introduction about project and basic fundamentals that will be used in implementing the project.

- Chapter 2 gives us the literature survey of Web Chat Application. In this we will have brief of journals, research papers, internet source of application.

- Chapter 3 is most important chapter in which we will discuss about the implementation of project including explanation, algorithms and source code of the project.

- Chapter 4 is the model used for analysing the performance of web char application. Outputs of the project at various stages and comparisons between different outputs.

- Chapter 5 is the end conclusion of the project and what further can be done in future to this project so that it becomes more efficient.

# CHAPTER 2

# LITERATURE SURVEY

## 2.1 Chatting Application with Real Time Translation [Sushma Shinkar, Nikhil Chaudhari, Priyanka Pagare] - Bachelor of Engineering, Computer Engineering Department, Sanghavi college of Engineering , Maharashtra, India

These days social networking is very common thing which is performed by people. Social networking is the thing which not only deals with text or sentences but also deals with pictures which we have achieved to operate the picture for performing face detection and finding the expressions. In an organisation, colleagues working there can send and get reply of messages instantly in very less time without having face to face conversation, and in the mean time the report of work can be shared/sent instantly during the chat session. With this application it is possible to have virtual conference without getting all the people together in a meeting room physically present. Doing instant messages for company communication is more efficient than making phone calls or doing emails. Various clients can do chatting concurrently. Dependent on a conference call or doing electronic mail message for meeting with colleagues is time consuming, but with this application everybody can join and have a discussion on various topics in very less time. If you really want to have fast communication then this application is far better than e-mails. With the support of fast messages one can send a message and get response of message in very few seconds.

This chat program is built in combination of two or more concepts by using Ionic Framework 3.6.0 which makes use of Angular 4.3.* and in order to store data we use database we have mongodb as Firebase and in backend programming we are making use of node.js Express.

Now, the architecture constitute of client and server module which include the given below steps:

1. Initial step is to execute server program on server machine.
2. After that client send request to server through its device and on that basis server give response to him/her and connection start taking place.
3. When the client-server connection is successfully established, the server then broadcast a list to of its connected users to its each connected client.
4. Client has the right to view all its active users and thus can make communication with them.
5. Server establishes a separate connection for each of its connected client, for which server produces a personal thread for each client connection. Now this thread is responsible to send/receive data from and to clients.
6. Whenever a client creates and sends a message to other client, this message initially is transferred to the server.
7. Then the server transmits the following message to the desired receiver of user.
8. Now, when the receiver client receives the message, receiver can read it.
9. In reflection receiver can send back reply but again follow the process of same process as mentioned above.
10. This chat application brings the use of concept of socket programming and the concept of multithreading. There will be different threads in it. One thread is for running server program and a different thread to control each client that wants to establish connection.

## 2.2 Implementation of Socket programming and RMI Using Stimulating Environment [Rounak Sinha, Hemant Kumar Srivastava, Sumita Gupta]

A network basically constitute combination of nodes and the medium (way) which helps in connecting them. The following nodes are in connection to each other due to the reason of transmission of data, which can take place in many forms like media, messages, function invocation and so-on. To do data transmission we basically have two modes those are Synchronous transfer and Asynchronous transfer. In this big world where distribution of data plays a very important role for communication in many organisations, it becomes very essential to make use of network intelligently for much effective transmission with less and controllable traffic. The following research paper basically deals with two such kind of techniques which can be brought into selection as per the need of any company. These two techniques include Socket Programming and Remote Method Invocation which is popularly known as RMI. Socket programming  is one of the efficient and fundamental technologies brought in use on the internet. If we talk about socket programming being implemented at hardware level, then the communication taking place between various hosts will occur much rapidly than it is today. In a machine, socket programming can be implemented or executed at both hardware and software level. In the following research paper both Socket programming and (Remote Method Invocation) RMI have been discussed and explained in great detail and comparative analysis is done between them on the basis of various parameters.

Remote Method Invocation is a basic way for producing and deploying various distributed object applications in java programming language. There are two packages known as java.rmi and java.rmi.server which contain the interfaces and classes that basically defines the functionality of JAVA.RMI systems. It is basically a method to call objects from remote location of a class.
The socket class contains various constructors:
1. public Socket(String host, int port) throws UnknownHostException, IOException.

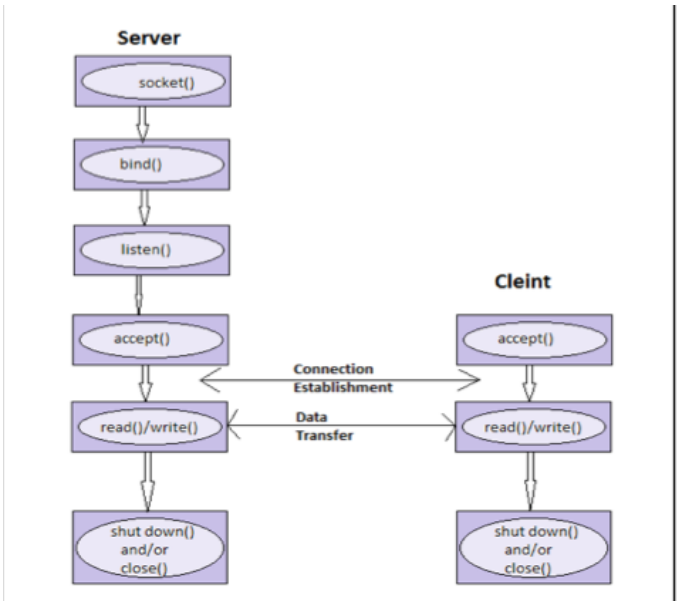


Figure 2.1

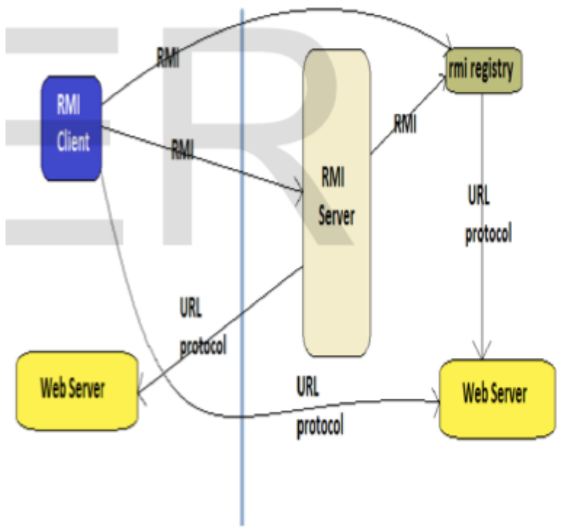


Figure 2.2

2. public Socket(InetAddress host, int port) throws IOException.

3. public Socket(String host, int port, InetAddress localAddress, int localPort) throws IOException.

## **2.3 Java Socket Programming** [ Java Point ]

Basically java socket programming is brought in use for communicating between different applications executing on various Java Runtime Environment.

Java Socket programming can have two types of mediums to communicate. Communications can be connection-oriented or can be connection-less.

If we talk about connection-oriented sockets then for those we have classes like Socket and ServerSocket while, for connection-less socket programming we have Datagram Socket and Datagram packet classes.
The client who are doing socket programming should know two things about server:

1. Client should know the IP Address of Server
2. And Most importantly it should know the port number of server.

Now here we are preparing to establish one-way client and server communication. Taking about this application, client/user transmits a message to the server, then server reads the message using read-line and then prints the message. Here basically we are making use of two classes that includes ServerSocket and Socket. The Socket class is brought in use to communicate between server and client. By using this class, we can do reading and writing of messages. In server program we have server-socket class. We have a method called accept() in ServerSocket class which blocks the console till the time the client is connected. When the client is successfully connected, it brings back the instance of Socket towards side of server.

Basically a socket is an endpoint for communications between two devices. To create a socket we have socket class.

Various essential methods in socket class which are use includes getinputStream(), getOutputStream(). In server socket class we have socket accept(), synchronised close().

## 2.4 Enhanced Education Chat Application Based on Interested Keyword with Username and Password Authentication Security [Er. Kavindra Singh, Er. Rajendra Singh]

These two engineers carried out large amount of Research on this chatting application. So, it is very essential to know and understand that how much research has been achieved in chatting application. The detailed explanation of research is given below :-

Avinas Bamane thought and gave suggestion of Enhanced Chat Application. In the following research paper they added a new feature in chatting application which is a paint tool box. With these paint tool box now client is able to develop 2D diagrams such as circle, triangle, pentagon, rhombus, square and son-on and then can transfer these figures to the client with whom he is chatting. If we talk about earlier research techniques then there is no that type of method in which client could draw/make there own diagrams that's why writer did research in such areas for extending chatting application.

Maha Sabri Altemam also made suggestions on this chatting application and introduced their research paper on Voice Chat Application by making use of Socket Programming. In this research paper he made use of socket programming to have a feature of recording their own voice and then sending it to the person who is on other side of communication. In his research paper he made use of chatting which is text based to if anytime voice communication system server became down then we can also have text chatting which can be done.

Nikita Mahajan also did their research paper. This was basically on design of chatting application which is based on bluetooth of android. In this research, design of chatting application by making use of bluetooth of android basically contains messaging with the help of bluetooth between the two smart phones having operating system android.

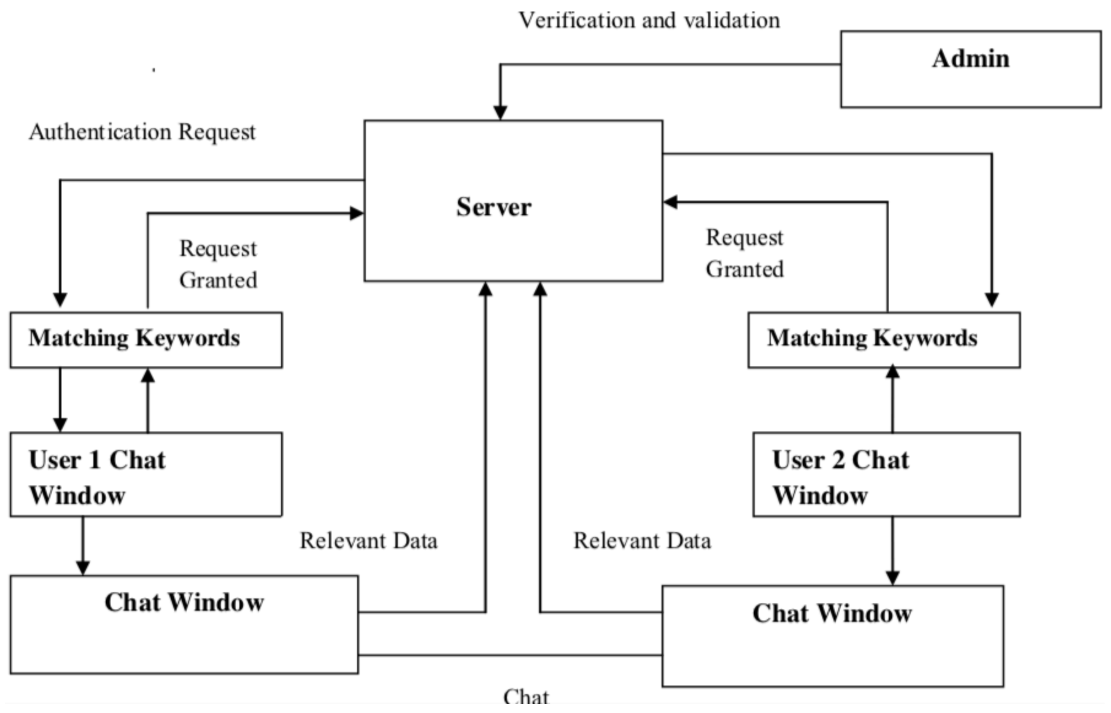


Figure 2.3

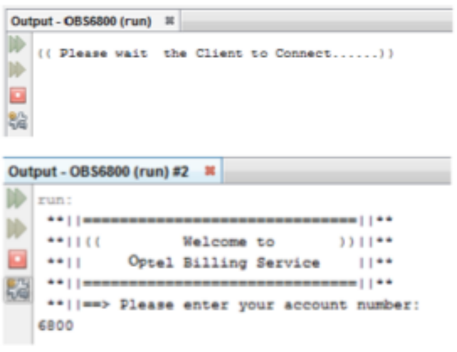
The web chat application is based on client - server architecture within a Local Area Network. This client server model of doing computing is basically a distributed application that do pieces of tasks between the provider of resource or we can say that service called servers and service requesters are called clients. The Server side always would be continuously running service listening to a different - different clients enquiring its service. The servers will maintain database of users. Whenever a client makes login to the application, the Server immediately makes authentication of the user . Once the user is been authenticated the IP address of the client/user will registered to the list of the Server and user transmits that list of online client partners and remaining relevant data to the Client/user. When the client wants to have some chat with any other user, then that user's IP address along with his Port address would be sent to the client and vice versa. So a connection would take place and the two clients would be able to chat with each other.

## 2.5 Design and Implementation of Client-ServerBased Application using Socket Programming in a Distributed computing environment [Rolou Lyn R. Maata, Ronald Cordova, Balaji Sudramurthy, Alrence Halibas ]

In this research we have a detailed discussion on client-server application with the use of socket programming in an environment where we have distributed computing. In this the researchers have programmed a client-server based application known as OpTel Billing System in short known as OBS on platform of java Netbeans and TCP datagram to explain the fundamentals of socket programming and is transmissions in distributed computing. Style of socket programming, design of interface, classes of java and exception handling are also considered part of development stage. Mainly the analysis by researcher was made on communication process between client and server by making use of socket programming. Also analysis and examination is done on classes used in this programming. According to these researchers, basically sockets of devices is a two-way link between client and sever codes which are executing in environment of network. It is a very efficient way of communication process.

Request
Reply

Figure 2.4



```
(( Please wait  the Client to Connect......))
```

```
run:
 **||================================||**
 **||((           Welcome to        ))||**
 **||      Optel Billing Service      ||**
 **||================================||**
 **||==> Please enter your account number:
6800
```

Figure 2.5

# CHAPTER 3

# SYSTEM DEVELOPMENT

## 3.1 Description of the Web Chat Application

In this web chat application, we will have different users who can do chatting with each other in real time. In real time means that they will be chatting live on a platform. We have various applications around us, where this concept is used like in YouTube we have live streaming of lectures by teachers where students can do live chat with teacher remotely from there homes and can chat with teacher regarding there doubts at that time only with the help of such application. Also other online teaching applications also use this. We have seen its use in entertainment apps like Facebook where celebrities go live and people in real time chat with there favourite star.

So behind this web chat application we have concept of server-client architecture where we will have a server and different clients will be there who will request server for connection and server will in reflection give response and various clients similarly can get connected to the server. Now all clients which are connected to the server are able to send messages to server and further server will distribute that message to all other clients. In this way, we will be able to achieve live web chatting.

Now for devices or computers to have connection with each other, there is requirement of socket programming which is required to establish end to end connection. Since each device has its own socket so using these sockets connection can take place.

Now in this application we will have two programs one of server class and one of client class. Firstly, we will run server program and server will be continuously waiting for clients and then we will run client program as many times as much clients we will be requiring. Then clients will get connected and whatever message will be sent by a client will go to server and then will be transmitted to other clients. We will use java language in our project.

In socket programming, if we talk about case of server then it only has server sockets. So there is need of creating object of server which is done in server program where objects of server are created and sockets of server are thus created. Then using these sockets data transmission between devices will take place. Now using these chatting application we can send text as well as other files also. As I have also mentioned in literature surveys where with text a client can also send 2-D images with it and other files like audios and videos can be sent.

The client who want to send data to a sever requires the IP address of the sever machine and also requires the port number of server of that particular application. In server program we will pass the port number as argument while in client class of program, IP address and port number of particular application are passed as arguments.

There are some important methods used in server socket class :-

1. **int getLocalPort() -** This method returns the port number of server on which it is waiting. This method is very essential if in a constructor is passed as port number.

2. **socket() accept throws IOException -** It basically do waiting of incoming clients. The following method is blocked till a client connects to a server on a particular port or time out took place as time-out value is set by method setSoTime().

3. **setSocketTimeot(arg) -** It is used to set the time-out to tell that how much time will be taken by to accept() client till he has been waiting.

4. **void bind(host socket-id, backlog) -** When server socket's accept() function is invoked, returning is not done by method till client connects. When client connects it is responsibility of server socket to create a new socket on unspecified ports and do return of reference to this new socket.

Also there are some methods in socket class :-

1. **Socket throws unknown host Exception, IO Exception -** The following method tries to connect server at the particular port. If following constructor do not throw an exception we can say that connection is successful and client is well in connection with the server.

2. **Socket (InetAddress host, int port) -** The following method is very much similar to previous constructor, only difference is that in this host is shown by InetAddress object.
3. **Socket (String Host, int Port, Intetaddress local address, int local Port) -** It connects particular host and port, developing a socket on local host at particular port and address.

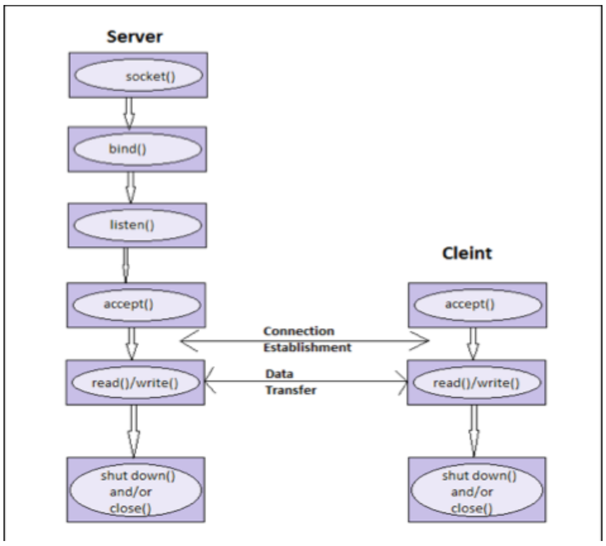


Figure 3.1

## 3.2 Design Diagram

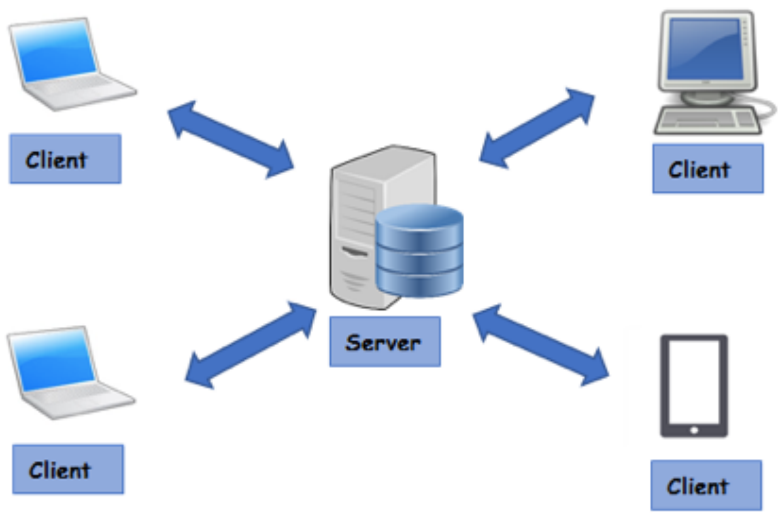


Figure 3.2

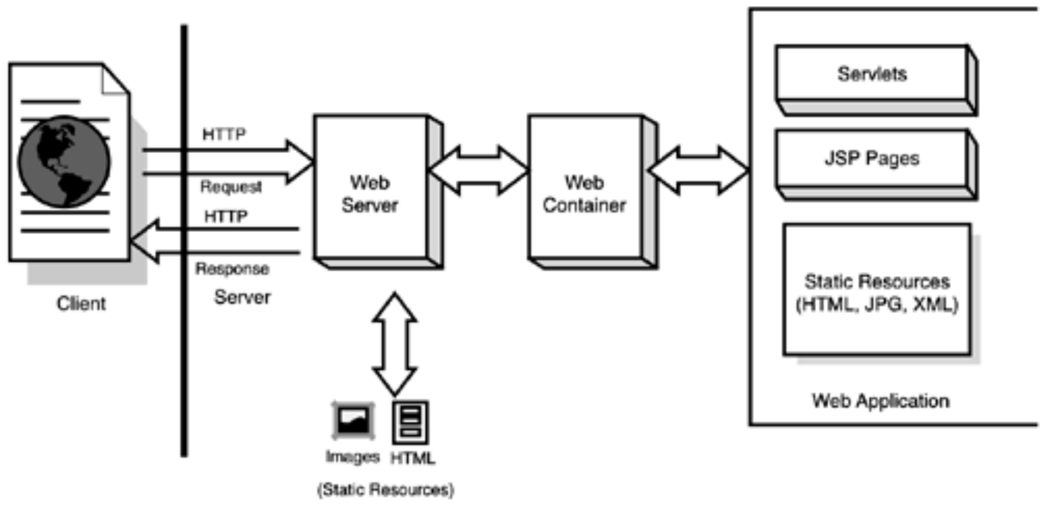


Figure 3.3

## 3.3 Functional And Non-Functional Requirement

### 3.3.1 Functional Requirement

So the functional requirement for web chat application are:

- Netbeans or Eclipse java compiler
- Graphic user interface libraries in order to make application attractive.
- Background colour of application will be black.
- There will be button through which server could start and would shot down.

### 3.3.2 Non-Functional Requirement

Now some non-functional requirement are:

- Client should at starting change password provided it for logins.
- Socket IP address and Port Number are required in order to establish connection with particular server.
- The following application can be used on different operating systems, thus denotes its portability.
- Clients chats within server will be secured and protected from outside world.

## 3.4 Implementation Detail

In implementation of web chat application we will a server program and client program.

### 3.4.1 Source code for server program:

```
package Project;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.util.*;
import java.net.*;

public class Server {
  ArrayList clientOutputStream;

  public class ClientHandler implements Runnable {
    BufferedReader reader;
    Socket sock;

    public ClientHandler(Socket clientSocket) {
      sock = clientSocket;
      try {
        InputStreamReader isr = new InputStreamReader(sock.getInputStream());
        reader = new BufferedReader(isr);
      } catch (IOException e) {
            e.printStackTrace();
        }
    }

    //job
    public void run() {
        String msg;
        System.out.println("server run");
        try {
            while((msg=reader.readLine())!=null){
                System.out.println("read : "+msg);
                tellEveryone(msg);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

}

public static void main(String[] args) {
    Server s = new Server();
    s.gui();
    s.go();
}
//gui start
JFrame frame;
JLabel label;
void gui(){
    frame = new JFrame("Server Window");
```

```
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        label = new JLabel("Don't exit without closing");
        label.setForeground(Color.LIGHT_GRAY);
        JButton close = new JButton("Shut down Server ");
        close.addActionListener(new CloseListener());
        panel.setBackground(Color.BLACK);
        panel.add(label);
        panel.add(close);
        frame.getContentPane().add(panel);
        frame.setSize(400,300);
        frame.setLocation(20,20);
        frame.setResizable(false);
        frame.setVisible(true);
    }
    //
    public class CloseListener implements ActionListener {
        public void actionPerformed(ActionEvent ev){
            try {
                serverSocket.close();
                label.setText("Server Shut down Close app or restart to start server again");
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
ServerSocket serverSocket;
    //
    void go() {
        clientOutputStream = new ArrayList();
        try {
            serverSocket = new ServerSocket(5001);
            while (true) {
                Socket clientSocket = serverSocket.accept();
                PrintWriter writer = new PrintWriter(clientSocket.getOutputStream());
                clientOutputStream.add(writer);
                Thread t = new Thread(new ClientHandler(clientSocket));
                t.start();
                System.out.println("Got a connection");
            }
        } catch (IOException e) {
            label.setText("Can't setup Server");
            e.printStackTrace();
        }
    }
    public void tellEveryone(String msg){
        Iterator it = clientOutputStream.iterator();
        while (it.hasNext()){
            PrintWriter writer = (PrintWriter) it.next();
            writer.println(msg);
            writer.flush();
        }
    }
}
```

**Explanation -** In this server program we have class named as Server. In server class one is ClientHandler which is implementing runnable interface and one is close listener class which is implementing the interface ActionListener. All these classes are build under project package.

We have also imported some pre-defines packages like which we have imported in this class. Like we have imported swing for Graphic User Interface. Action listener for button. Then we have io for input and output of data.

Now in class Server firstly we took ArrayList of clientOutputStream. Then using sockets Server will be waiting for the clients. As client will request server the server with the help of accept method will receive client request and will give him response back. Then in clientHandler class we implements runnable interface. Using gettext() from input stream we took input from keyboard. After that we will set socket. Now when object of socket has been created using object of socket we will have input stream thus then using printwriter() the message will be in it and using flush command the message will be distributed to other client of array list except the client from which message has been received.

Now in order to add some graphics in this application we called gui. In that we set frame name as Server window. We have label in this as warning saying " Don't exit without closing". Then we made a button using JButton with name "Shut Down Server", Then we set size and location of button. Then also did colouring on the black. In Closer Listener, we have action listener of button which is used for whether button has been pressed or not.

All this work is done by keeping in mind of handling exception. So that if any exception takes place then in catch block exception is handled and printStackTrace is used to display the error message.

## 3.4.2 Source code for client program

```
package Project;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.io.*;


public class ClientGui {
  JTextField outgoing;
  JTextArea incoming;
  BufferedReader reader;
  PrintWriter writer;
  Socket sock;
  JLabel label;
  JLabel host;

  void go() {
    JFrame frame = new JFrame("Chat app");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel panel = new JPanel();
    panel.setBackground(Color.BLACK);
    incoming = new JTextArea(15, 30);
    incoming.setLineWrap(true);
    incoming.setWrapStyleWord(true);
    incoming.setEditable(false);
    JScrollPane scroller = new JScrollPane(incoming);
scroller.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_SCROLLBAR_AL
WAYS);

scroller.setHorizontalScrollBarPolicy(ScrollPaneConstants.HORIZONTAL_SCROLLBA
R_NEVER);
    scroller.setBorder(BorderFactory.createMatteBorder(10, 0, 30, 0, Color.BLACK));

    outgoing = new JTextField(25);
    outgoing.setFont(new Font("serif", Font.PLAIN, 16));
    JButton sendButton = new JButton("send");
    SBListener sb = new SBListener();
    sendButton.addActionListener(sb);
    outgoing.addActionListener(sb);
    //label to show status
    label = new JLabel("Staus :");
    label.setBorder(BorderFactory.createMatteBorder(5, 5, 5, 5, Color.BLACK));
    label.setForeground(Color.LIGHT_GRAY);
    panel.add(scroller);
    panel.add(outgoing);
    panel.add(sendButton);
    panel.add(label);
    frame.getContentPane().add(BorderLayout.CENTER, panel);
    setNetworking();
    Thread t = new Thread(new IncomingReader());
    t.start();
    frame.setSize(400, 400);
    frame.setResizable(false);
    frame.setVisible(true);
  }
```

```
//setup networking
void setNetworking() {
   try {
      sock = new Socket("127.0.0.1", 5001);
      //Thread.sleep(10);
      InputStreamReader isr = new InputStreamReader(sock.getInputStream());
      reader = new BufferedReader(isr);
      writer = new PrintWriter(sock.getOutputStream());
      System.out.println("Network Established");
      label.setText("Status : Connected");
   } catch (Exception e) {
      e.printStackTrace();
      System.out.println("Can't connect to the server");
      label.setText("Status : Not Connected");
   }
}

//send Button listener
public class SBListener implements ActionListener {
   public void actionPerformed(ActionEvent ev) {
      writer.println(outgoing.getText());
      writer.flush();
      outgoing.setText("");
      outgoing.requestFocus();
   }
}

public class IncomingReader implements Runnable {
   public void run() {
```

```
            try {
                String msg;
                while ((msg = reader.readLine()) != null) {
                    System.out.println("read:" + msg);
                    incoming.append(msg + "\n");
                    incoming.setCaretPosition(incoming.getDocument().getLength());
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}


package Project;

public class Client {
    public static void main(String[] args){
        ClientGui g= new ClientGui();
        g.go();
    }
}
```

**Explanation -** Now as server program is in running phase, we will now run client program which will send request to server program for connect with the complete address of server in order to have chat with other clients which are also similarly requesting sever and waiting for there response.

Here in this, we have client class. It has main method. Here in main method we will make object of ClientGui, which is another class. Explanation of ClientGui will be done later. Now we have a method go() in ClientGui class which will be called in this class server.

For graphic user interface, we have swing package. So firstly this package will be imported. Then using JFrame we will set frame name as ' Chat App'.
Background colour is set black. Then we had sent a button named as 'Send' with actionListener interface implemented by SBlistener class. This actionListener of send button is used for checking the press of button whether button is pressed or not.

Then we made a method setNetworking() where we wrote code in order to set connection with the server. In this we will have a socket object in which we will pass the IP address and port number of server machine as arguments.
Then the socket object with input stream reader is called in order to get input message from client socket. Then using reader the we will read message.
In writer we store socket's output stream of printer writer. Thus connection gets established.

We majorly need to focus on exception handling in this. So that whenever exception occurs printstacktrace prints it.

### 3.4.3 Method for implementation

The method used behind this web-chat application is server-client architecture.

Client Server architecture is a application which has distributed structure that basically do tasks in pieces in between the resource provider or service we can say service provider known as servers and the one who request service are called clients. Generally often times clients and servers do communication over a computer network on different devices, also it is possible that both client and server may be running on same computer.

A server host generally executes on one or many set of instructions, that often share their details with clients. But a client never share any of its details, it only do request of service from the server. So the clients start communication phases with servers, that results in awaiting of incoming requests.

This architecture is basically a like a producer-consumer architecture where we can say that the server behaves like the producer and the client behaves like a consumer. The server provides a huge and fully computing intensive services to the present clients on there demand.

All such services could include application accessing, storing files, transmission of files, accessing printer and having an access to the server's basic computing powers.

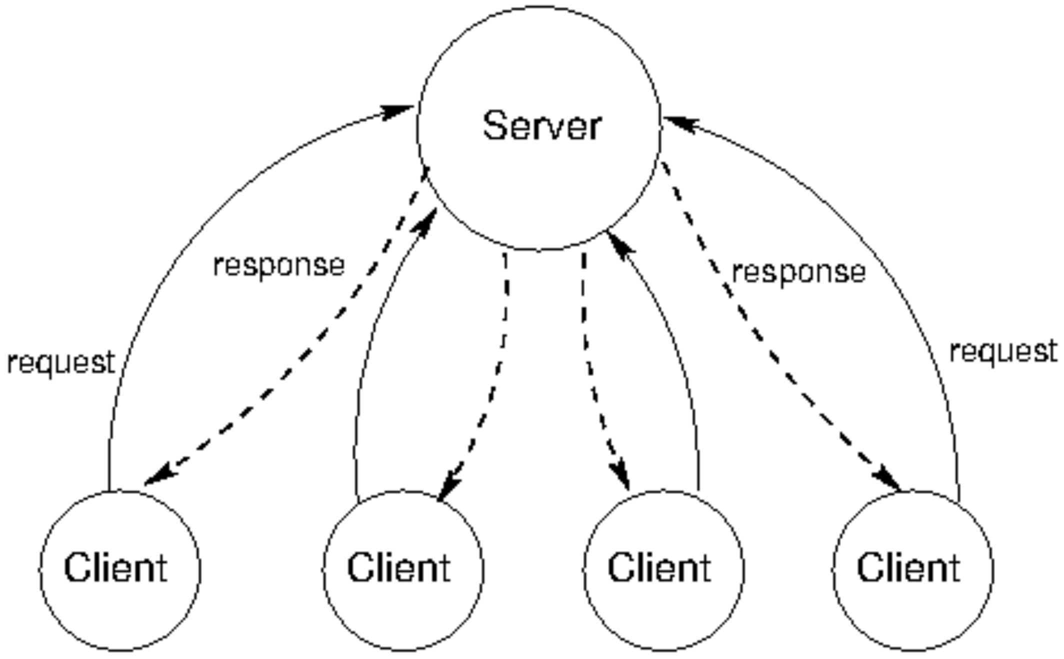


Figure 3.4

Client/server architecture works when the client computer sends a resource or process request to the server over the network connection, which is then processed and delivered to the client. A server computer can manage several clients simultaneously, whereas one client can be connected to several servers at a time, each providing a different set of services. In its simplest form, the internet is also based on client/server architecture where web servers serve many simultaneous users with website data.

Now for connection establishment we need the method of socket programming. In this socket programming we have end to end connection. It is for connecting nodes with each other. Nodes here represent the sockets. Now lets take case of two nodes. One of the node act as listener which listens at a specific port on its Internet Protocol address while other comes to it in order to have connection establishment with it by know the port number and Internet Protocol address.

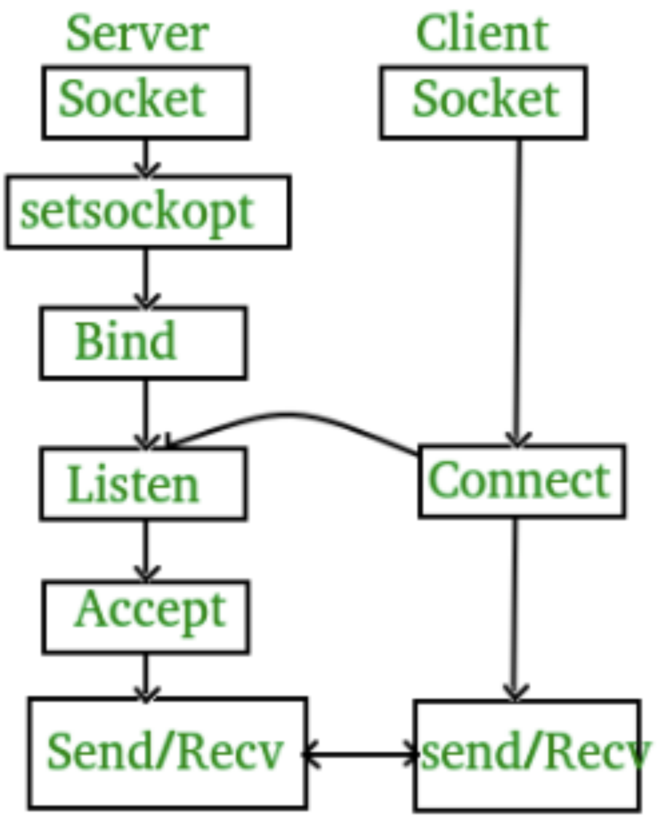


Figure 3.5

### 3.4.4 Swing

Java Swing is a small weight toolkit of Graphical User Interface (GUI) that do addition of a large set of small gadgets. It also adds package through which you can make Graphic User Interface parts for your application of Java and also it is platform independent.

This Swing library is developed at top of toolkit known as Abstract Widget Toolkit (AWT) of java which is an older and is a platform dependent Graphic User Interface toolkit. You can make use the Java Graphic User Interface parts including buttons, text-boxes and so-on by making use of library and there is no need to create these components from starting.

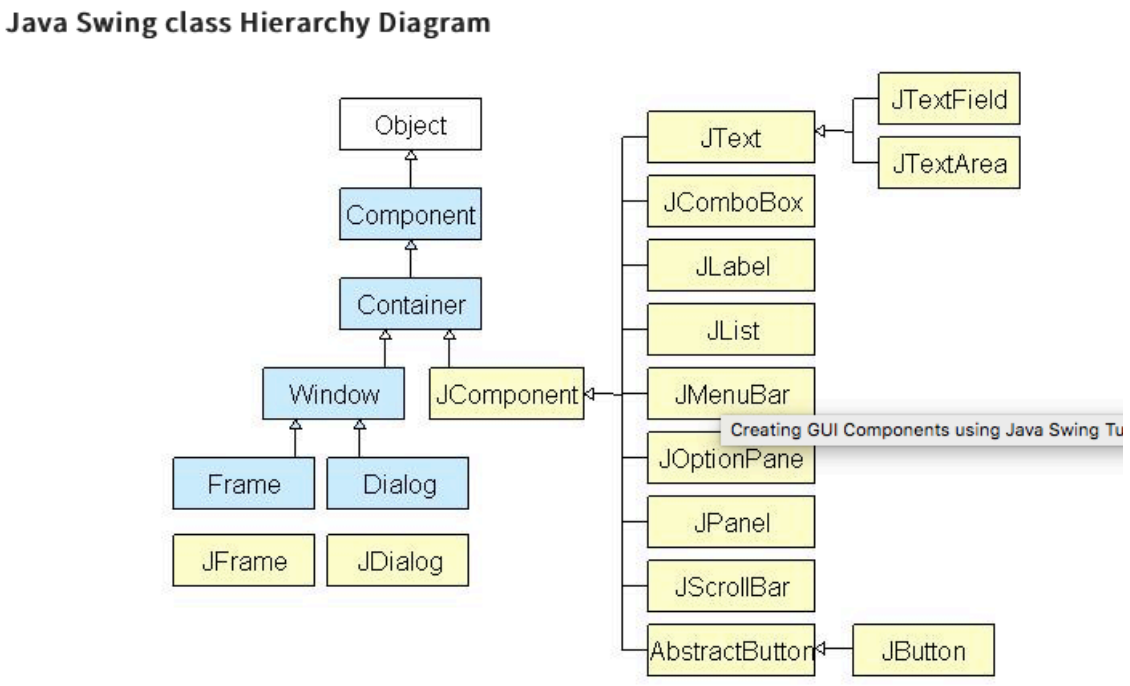


Figure 3.6

Then we have container classes. These are the classes that have components on it. For creating Graphic user interface we need at least one container object.

We have variety of containers :-

1. **Panel -** It is considered as full container and don't contains a window in itself. The main aim of this panel is to arrange components on window.

2. **Frame -** This is a whole functioning window consisting of its titles and its icons.

3. **Dialog -** This can be considered like a popup window which has responsibility to pop out the window whenever a message came and has to be displayed. It is not like a window which is fully functioning.

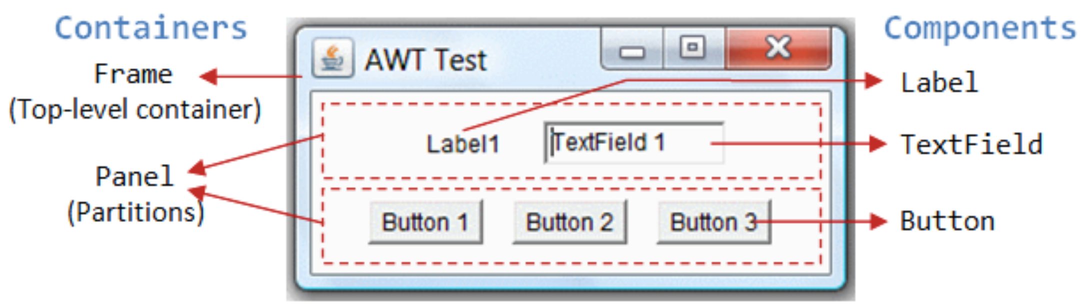


Figure 3.7

Now let us throw some light on Graphic User Interface elements.

Generally, graphic user interface has two types of elements :

1. **Component :** Basically, components are those Graphic User Interface entities which are elementary which includes Labels, Button and TextField.

2. **Container :** There are containers which include frame and panel. These are brought in use to make a hold on components in a particular layout form. Layouts can be Flow-layout or can be Grid-layout A container also has capability to make hold on sub-containers.

Now we talk about figure 3.7, In this there is constitute of three containers. Those include one frame and two panels. The Frame is container which is topmost of an Abstract Widget Toolkit (AWT) program. In figure as we can see that the following frame is constitute of a title bar, which contains a title, an icon and some buttons like minimise, maximise and close button. Also there is an optional bar which is menu bar and there is also display area for content. Basically, a panel is an area in rectangular shape which is used to do grouping of Graphical User Interface components which are related in a particular layout. As you can see in above figure (3.7), the frame which is at top-most level consist of two Panels. Here we have a total of five components:

- There is a Label which is providing description.
- There is a TextField where user can enter data.
- Also three are buttons are there for user so that it can be used to trigger particular programmed actions.

### 3.4.5 Some other important DataStructure and Method used in implementation

- **ArrayList -** The following data structure is brought in use in program in order to have a complete list of programs which are connected to the server.

- **printStackTrace () -** This is used in program in order to handle the exceptions. This is written in catch block so that whenever an exception is thrown from try block then that gets handled and this method displays the error description.

- **ActionListener -** It is an interface in java pre defined packages. So action listener of a button is for whether button is executed or not after making a click on it.

- **hasnext() -** The following method returns true or false. It checks in list whether there is any other client connected in then it returns true else it return false.

- **Inputstreamreader -** It is used to input the data. Whenever client send data to server then server use inputstreamreader to input data and whenever server sends data to client , client uses inputstreamreader.

- **OutputStreamReader -** It is used to transmit data out data. When client want to send data out to server this can be done using this outputstream reader.

- **Request focus -** This method is used with active listener. So that you press any key, active listener of button will press i.e to send message using button.

## 3.5 implementation Issues

There are various implementation issues in this which include:

- Sign In options needs to be enabled. As it is important that particular clients are needed to be connected with server. Like if any client wants to connect to a specific server than he/she should be given credentials. So that whenever client wants to connect username and passwords are should be put on sign up page.

- Need to make Graphic User interface to be more interactive. So that more clients gets attracted towards application.

- There is issue of remembering the port number of server device and Internet Protocol Address.

- You always need to shut down the server when not in use, otherwise that port gets blocked. So it is necessary to shut down server when not in use. If once port gets blocked then you cannot do chat or data transmissions through that port.

- There is no scope of deletion of message once sent. Suppose you sent a message on server and server distribute it to other clients and now sender wants to delete that sent message then it is not possible which also results in an issue.

So, following are the basic issues that are related to implementation of web chat application.

# CHAPTER 4

# PERFORMANCE ANALYSIS

## 4.1 Agile Methodology

Agile Methodology is basically a process in which we do constant iterations of production and testing phase whole time during the software development life cycle and contains more advantages over waterfall model. If we talk about waterfall model then it initially required to be fully designed then it is made forward for doing testing but in agile model we could produce a few of product commit it and then do parallel testing and finally verify it. It is pretty simple to make alters in the program and deployment of product is also quick. I am following agile method in chat application.

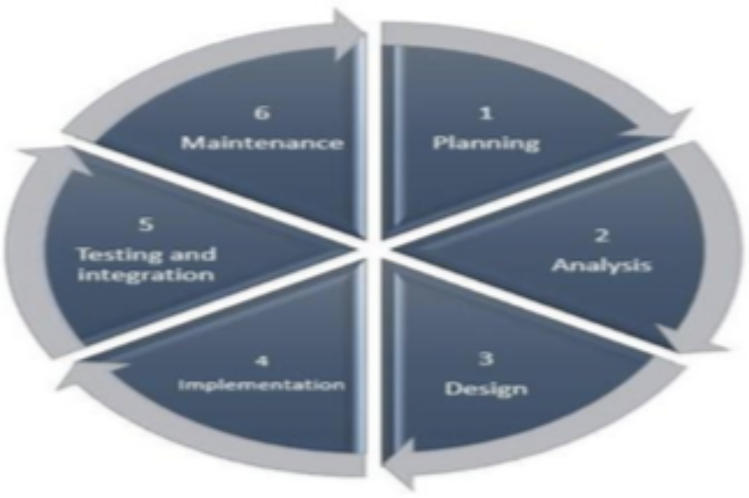


Figure 4.1

## 4.2 Analytics

- In this project, we are not having tools that can help us to monitor or check the performance of project which is very basic requirement. Basically here, whatever tests we are doing this time are not automatic So we should have tools that could exactly check that how much time the server is taking to transmit and make approval of all push requests.

- We require some good logs and better information. It's very complex to find what is exact reason for slowing of the process due to the lack of that kind of knowledge. If we talk about the server then this is very easy, only thing logs required are that they need to be be more fleshed out and thus creation of work takes place.

- There is need that once we close the application then the server should be closed immediately with that. But in this project if we close server without shutting down it, then that particular port on which it was running will get blocked.

- Also there is a need to focus on graphics of application using the Graphic User Interface.

- There is requirement of feature of forwarding the message of client or giving reactions on messages of a particular client.

## 4.3 OUTPUTS

**For server program**

**Output:**

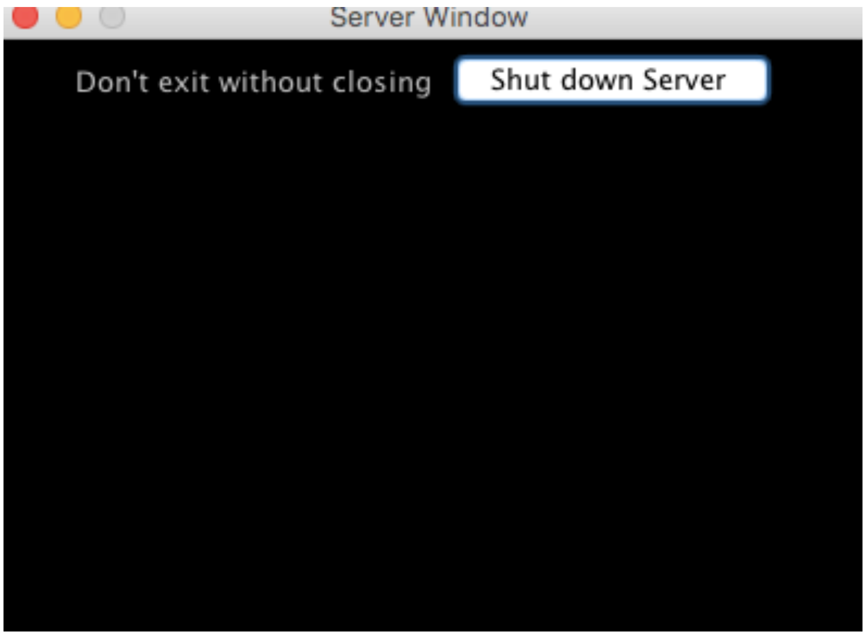
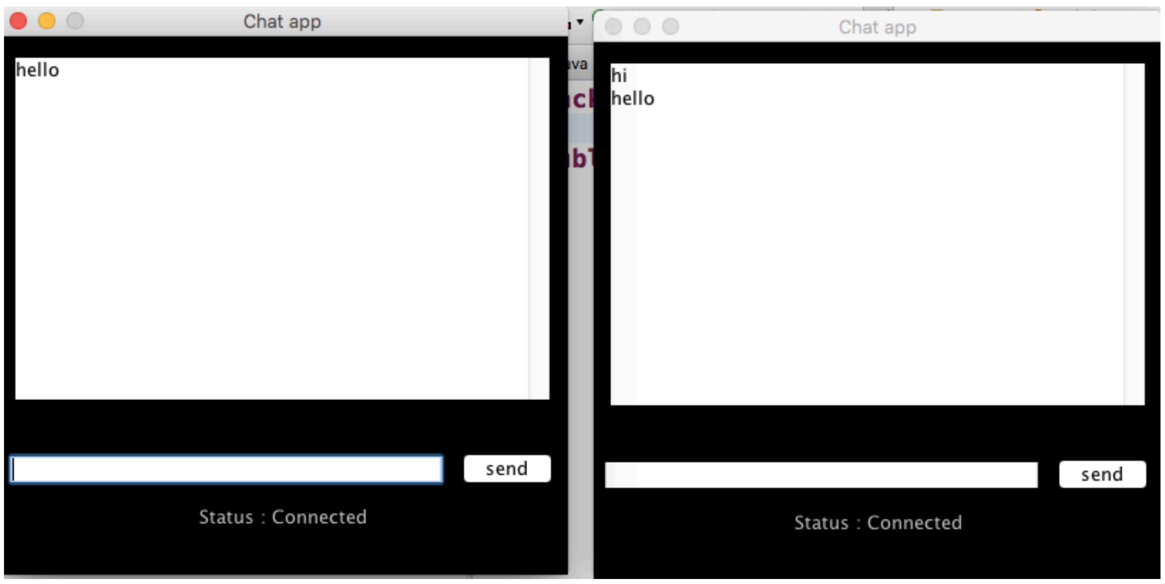


Figure 4.2

Here we have executed server program, where you can see we have a button shut down server and we have a label “Don’t exit without closing”.

**For Client Program**

**Output:**

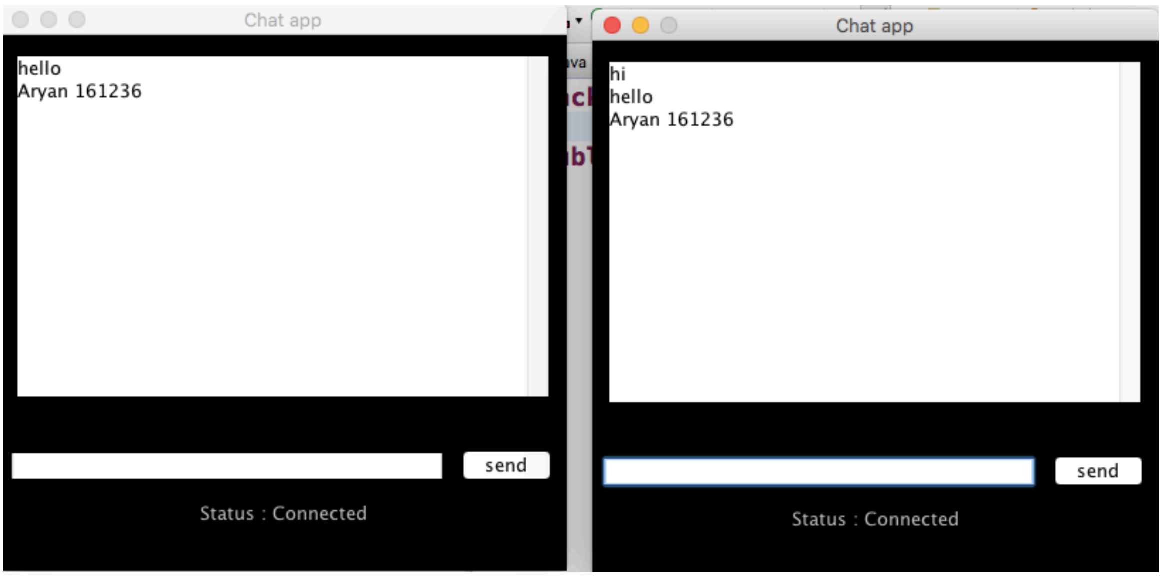


```
hello
```

```
hi
hello
```

Figure 4.3

Here we have executed the client program two times. Now first client typed hello and sent it on server which then is distributed by server to other clients also.



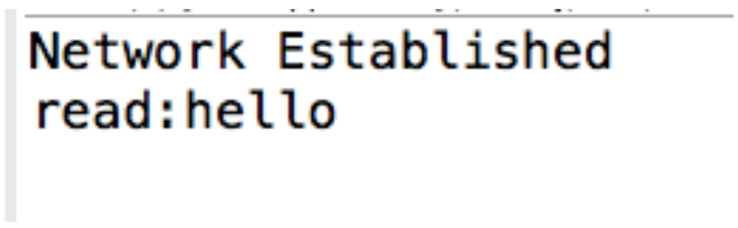
```
hello
Aryan 161236
```

```
Status : Connected
```

```
hi
hello
Aryan 161236
```

```
Status : Connected
```

Figure 4.4



```
Network Established
read:hello
```

Figure 4.5

# CHAPTER 5

# CONCLUSIONS

## 5.1 CONCLUSION

So, this is the last chapter of report where there is whole conclusion of the report. So at last conclusion is that these days communication doesn’t depend on the factor of distance. Now it is not required that to have immediate communication you we be close to them. Remote communication from homes has now become possible with the help of this web chat application.

Also in order to communicate with a group of people, there is no requirement to send a personal message to each and every person. So if you want to send a similar message to all the clients then this thing can be achieved with the help of client server program. The user who want to send message to a group can send it on common server where all other clients are also connected and server will further distribute the particular message to all other clients.

It can be said a much user friendly web application. In this project it is brought in great attention to remove all the errors and bugs from code and do the debugging of program and finally done proper testing of the program. So at the end it can said that it is type of application which is in great demand in the market.

## 5.2 FUTURE SCOPE

If throwing some light on the future of this program, so this project has great scope in future. In future we will include many features to this program.

- Like we can enable username and password method to the code and maintain the database of the clients connecting to the server.

- We can increase the capacity of server, so that more number of clients can make connection with the server.

- In future we will bring the feature of making forward of one users message directly by other user to other.

- Feature of reacting on particular message needs to be enabled.

- Server's requesting and responding speed needs to be increased and made more efficient.

So there are some factors on which in future we can think on applying.

# REFERENCES

1. https://en.wikipedia.org/wiki/Client–server_model
2. https://www.techopedia.com/definition/438/clientserver-architecture
3. https://www.w3schools.in/what-is-client-server-architecture/
4. https://www.geeksforgeeks.org/client-server-model/
5. https://www.irjet.net/archives/V5/i5/IRJET-V5I5886.pdf
6. https://www.ijser.org/researchpaper/Implementation-of-Socket-Programming-and-RMI-Using-Simulating-Environment.pdf
7. https://www.javatpoint.com/socket-programming
8. http://ijarcsse.com/Before_August_2017/docs/papers/Volume_6/6_June2016/V6I6-0112.pdf
9. https://www.geeksforgeeks.org/socket-programming-cc/
10. https://www.tutorialspoint.com/unix_sockets/what_is_socket.htm
11. https://www.javatpoint.com/socket-programming
12. https://en.wikipedia.org/wiki/Graphical_user_interface

Aryan Web Chat

ORIGINALITY REPORT

| 14% | 9% | 2% | 8% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | ijarcsse.com<br>Internet Source | 2% |
| 2 | www.ijser.org<br>Internet Source | 2% |
| 3 | www.techopedia.com<br>Internet Source | 2% |
| 4 | www.tutorialspoint.com<br>Internet Source | 1% |
| 5 | media.neliti.com<br>Internet Source | 1% |
| 6 | Rolou Lyn R. Maata, Ronald Cordova, Balaji Sudramurthy, Alrence Halibas. "Design and Implementation of Client-Server Based Application Using Socket Programming in a Distributed Computing Environment", 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2017<br>Publication | 1% |

# JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

## PLAGIARISM VERIFICATION REPORT

**Date:** 15/07/2020

**Type of Document (Tick):** PhD Thesis | M.Tech Dissertation/ Report | [✓ B.Tech Project Report] | Paper

**Name:** Aryan Pal Singh **Department:** CSE **Enrolment No** 161236

**Contact No.** 8629025294 **E-mail.** apsdau@gmail.com

**Name of the Supervisor:** Dr. Hari Singh

**Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters):** WEB CHAT APPLICATION

## UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

**Complete Thesis/Report Pages Detail:**

- Total No. of Pages =52
- Total No. of Preliminary pages = 7
- Total No. of pages accommodate bibliography/references = 1

[illegible]

**(Signature of Student)**

## FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at 14 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

**(Signature of Guide/Supervisor)** **Signature of HOD**

## FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

| Copy Received on | Excluded | Similarity Index (%) | Generated Plagiarism Report Details (Title, Abstract & Chapters) | |
|---|---|---|---|---|
| | • All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String | | Word Counts | |
| **Report Generated on** | | | Character Counts | |
| | | **Submission ID** | Total Pages Scanned | |
| | | | File Size | |

**Checked by**
**Name & Signature** **Librarian**

……………………………………………………………………

**Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com**