Project Report

On

"THE CHATTING APPLICATION"

Submitted to:

Panjab University, Chandigarh

In partial fulfillment of the requirement for degree of

Bachelor of Computer Application (B.C.A)

(Session – 2020-2021)

Under the Supervision of:

Dr. HarmunishTaneja

Asst. Prof.

Deptt. of Computer Sc. & IT

Submitted by:

Jasbir Singh and Sushil

B.C.A 6th Semester

Roll No: **18046454** and

18048685

DAV COLLEGE, SECTOR 10, CHANDIGARH.

(Certificate from Guide)

DAV COLLEGE, SECTOR 10, CHANDIGARH.

CERTIFICATE

This is to certify that Mr/ Ms. **JASBIR SINGH BAZZAD** and **SUSHIL** Class Roll Nos. **18046454** and **18048685** a bonafide student of B.C.A. 6th Sem being run by DAV College, Chandigarh of Batch 2020-2021 has completed their project entitled as **"THE CHATTING APPLICATION"** under my supervision & Guidance. It further certified that the work done in this project are results of candidate's own effort.

I wish him/her all success in life.

Date:

Dr. HarmunishTaneja

Asstt. Prof.

Deptt. of Comp. Sc. & IT

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to our Project Teacher Dr. HarmunishTaneja for his guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards our H.O.D Dr. Meenakshi Bhardwaj for her kind co-operation and encouragement which helped us in completion of this project.

I would like to express my special gratitude and thanks to our Principle sir Dr. P.K Sharma for giving me such attention and time.

I would like to express my gratitude towards my parents for helping me and it's because of them i was able to complete this project.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

~Jasbir and Sushil

By Sushil and Jasbir

TABLE OF CONTENTS

- Message (Online Menu aspect)

➢ Logout Function (Online Menu aspect)

4. Planning and Scheduling

➢ *Project Phase Schedule*

5. System Development

➢ *Coding Details*

➢ *Screenshots*

6. Milestones and Testing

➢ *Testing Plans*

- Types and Steps of testing
- Test report details
- Hours Distribution
- Gathering Sources
- Testing during/after development

➢ *Software and Hardware Requirements*

7. Final Product

➢ *Look of final project*

8. System Evaluation

➢ *ER Diagrams*

9. Conclusion's

➢ *Conclusion*

➢ *Limitation*

➢ *Future Scope*

1. Introduction

Chatting application is a desktop based application.

This client server chat application is based on java swing and used socket package. It's simple and easy and require only core java knowledge. We have developed this program and tried to give it simpler and more elegant look.

This application/program is a good example of using java.io, java.net package to create a chat application. A beginner of java language, who is familiar with this package can able, be beneficiate.

Chatting is a method of using technology to bring people and ideas "together" despiteof the geographical barriers. The technology has been available for years but the acceptance it was quit recent. Our project is an example of a multiple client chat server.

It is made up of two applications the client application, which runs on the user's PC and server application, which runs on any PC on the same network. To start chatting client should get connected to server. We will focus on TCP and UDP socket connections which are a fundamental part of socket programming.

This project is based on Online Chatting Application which allows two user's to communicate with each other simultaneously. This application can be a grateful way for communication. This application provides Video Call and Voice Call options which is a marvelous way to stay connected with your loved ones.

Other than that it also provides a three-dot option which helps to go to settings of application.

Application consists of two programs:

- SERVER and CLIENT

- SERVER:

The server module of the application waits for the client to connect to it. Then if connection is granted a client interacts communicates and connects to the server, it can mutually communicate with the server. The duty of the server is to let clients exchange the messages.

❖ CLIENT:

The client module is the one that utilizer sends requests to the server. Utilizer utilizes the client as the means to connect to the server. Once he establishes the connection, he can communicate to the connected server.

➢ OBJECTIVES

The aim of this project is to express how we can implement a simple chat application between a server and a client? The application is a desktop based application and is implemented using Swing and awt. The project is developed in Java SE language executed on a single stand-alone java across a network using loop back address concept.

- ✓ Enable User Communication
- ✓ Dynamic in nature
- ✓ Two-way communication
- ✓ Enhances Security/Privacy of user

PURPOSE,AND SCOPE

➢ PURPOSE

The purpose of the chat application is to allow users be able to do the chat with each other, like a normal chat application. The users will be able to chat with each other, most likely only from user to user, no group chatting will be developed, unless there is time to do so.

The chat application will be written in java, but due to the lack of experience in java while developing the application practicing technique with java and working on it as much as possible will help improve some java skills and be more ready to develop the application.

For the scope of the project, the project will be tested as the program is being developed.

A database for the users registered will be developed and tested, a menu will be developed and tested, a Client/Server Interface will be developed and tested, and GUI'S will be developed and teste, for the users benefit.

When the chat application is near completion, more testing will be done in order to make it less buggy or more user friendly.

➢ Target Audience

The target audience is any person who wants to use a live chat application with enhanced security.

➢ SCOPE

With the knowledge we have gained by developing this application, we are confident that in future we can make this application more effectively.

1.Problem of Security: There are always loop holes in most of the communicating application due to which hackers can easily access our data. But this chatting application uses *http, TCP/IP model* and many of web uses udp so this chat application is more secured as compared to others.

2.Problem of Data Loss: Most commonly faced problem is storage, so by using this chat application user can easily create backup of chats i.e users can save chats.

3.Problem of Timeline: users will be able to see the time stamp on the sent and received messages both.

4.Limitation of Live Chat: this application provides live chat feature which enhances the security of users privacy.

5. The fact that the software uses an internal network setup within the organization makes it very secure from outside attacks.

2. Product Overview

The functionality of the chat application is to give the ability to chat with whoever is online on the application. The users and stakeholders will be a small group for now.

The use case will be what is available to the user, and the functional/nonfunctional requirements will be covered, as well as the milestones of the chat application.

➢ User's and Stakeholders

This section will deal with users and stakeholders. The users will be using the chat application and the stakeholders will develop, maintain, and test the chat application.

Myself

I will be developing, maintaining, and testing the chat application through its phases of development.

Users

The users will be anyone who has the chat application and registers for it.

Main Menu

When the client runs the chat application, the client will see the main menu, which will welcome them, At the main menu, the client will have the choice to register for the chat application, login to the chat application, or exit it.

Register/Login/Logout

The user must register in order to login, the user must login in order to send messages to those who are online, and must be able to logout if the user wants to logout.

Message

When the client wants to message the user, the client clicks on the name, the user can send a message to another online user. Some character limitations for typing might be put into place, only if it is necessary.

Typing/Online

When client or server starts to type something other user will be able to see "typing" status on its screen. And if users are not typing anything then it will show "online" on screen.

Chat history

When the client wants to see the chat history, the user will be able to do so. The user willbe able to clear the history if the user wants to. It will be stored in form of .txt file.

➢ Use Cases

These are the use cases for the client of the chat application. The server has access to all of these cases as well.

- Main menu
- Voice call
- Video call
- Back button
- Send button
- Three-dot's

NOTE: ALL EXPLAINED IN SOFTWARE INTERFACE.

3. Functional Requirements

➢ User Interface

The user interface required to be developed for the system should be user friendly and attractive.There are two sets of Java APIs for graphics programming:

AWT (Abstract Windowing Toolkit) and Swing.

- AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components.

- Swing API, a much more comprehensive set of graphics libraries that enhances the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC consists of Swing, Java2D, Accessibility, Internationalization, and Pluggable Look-and-Feel Support APIs. JFC was an add-on to JDK 1.1 but has been integrated into core Java since JDK 1.2.

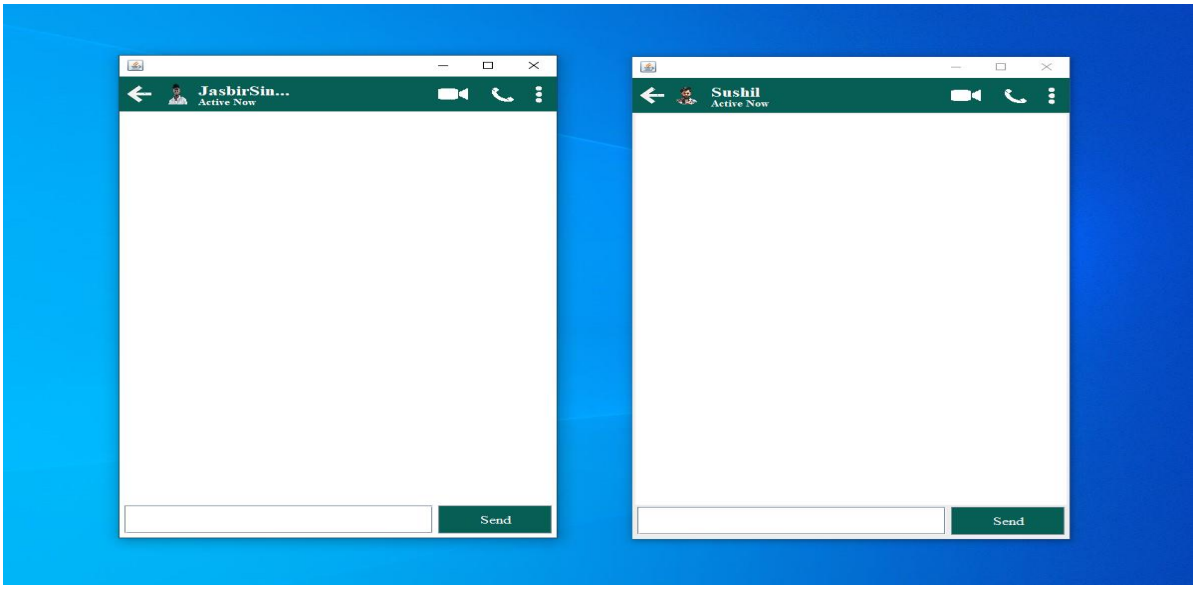
➢ Software Interface

Programming language: Java and Socket Programming

This is how "The Chat Application" software will look like once it's being runned.

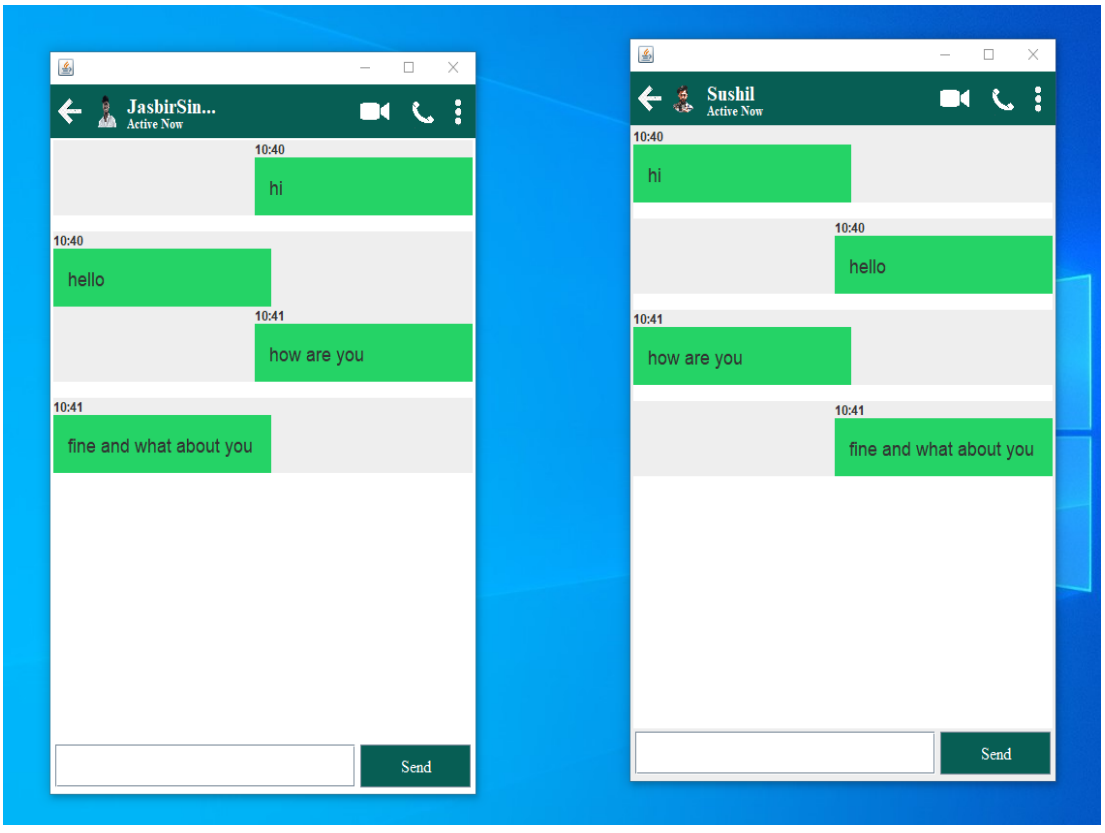
SERVER SIDE

CLIENT SIDE



- **Online Menu function**

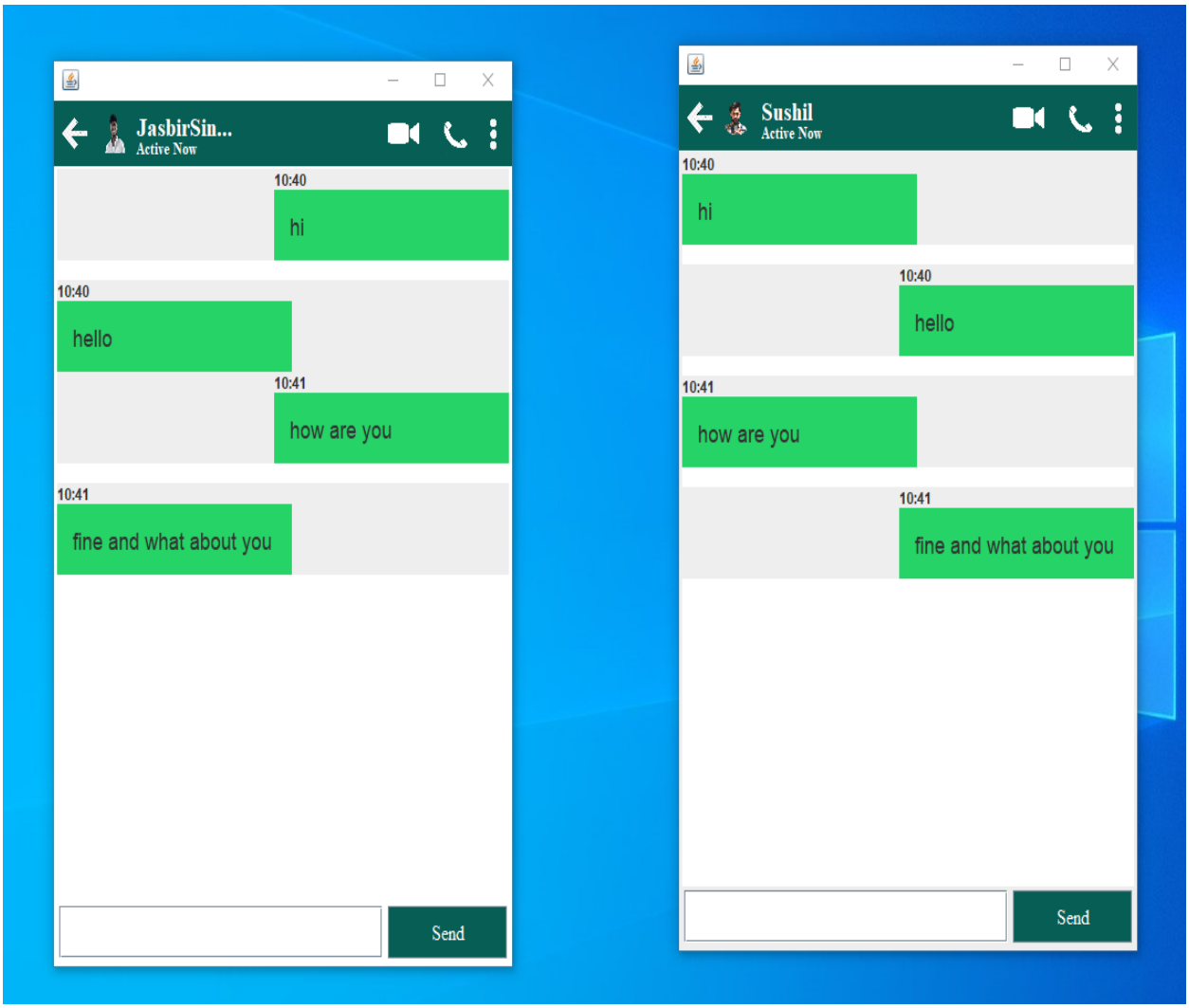
This feature will let the users know the live status of them whether they are online or not.



- **Message (online aspect)**

When both Server and Client are Online they can chat with each other. All the messages sent will appear in form of bubbles and that too with time stamp.

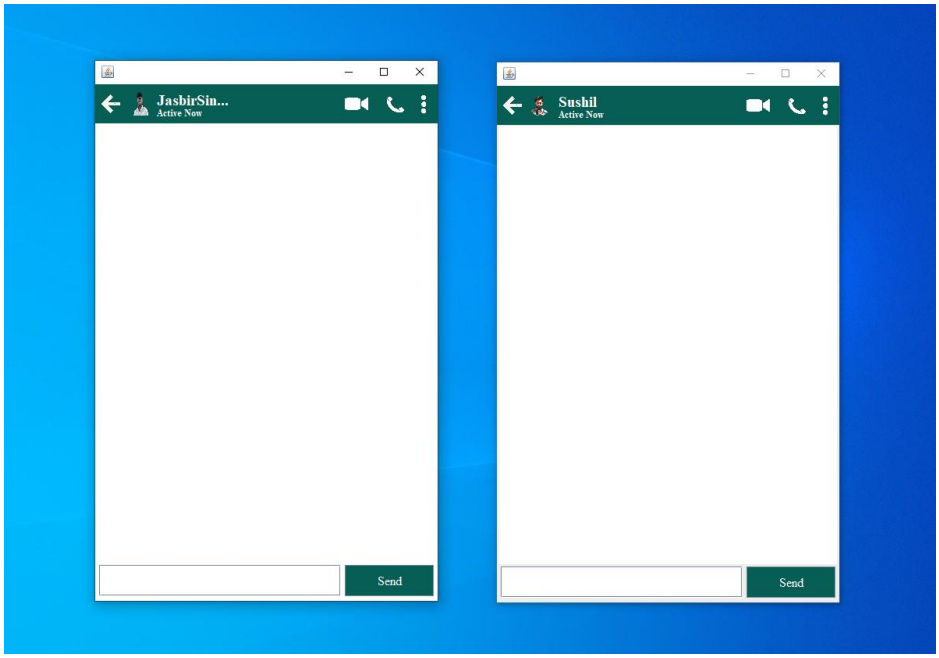
Sender's message / Receivers message will look as represented below.



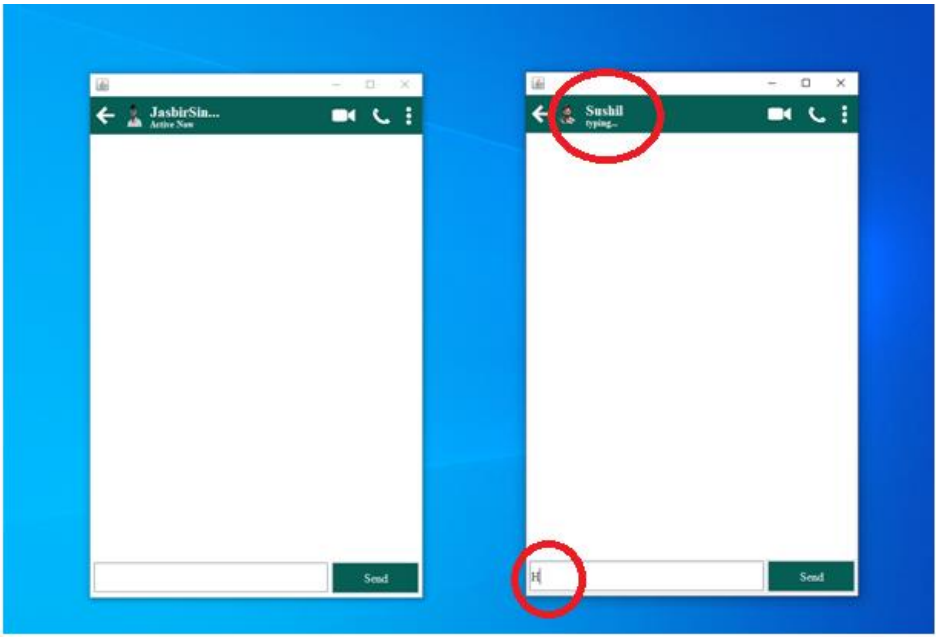
- **typing (online aspect)**

Users will be able to see the live status of their current status. Let's say if user is typing then he will be notified by changing the status from *Active now* to *typing*.

typing will appear for 2 seconds until we stop typing.



BEFORE

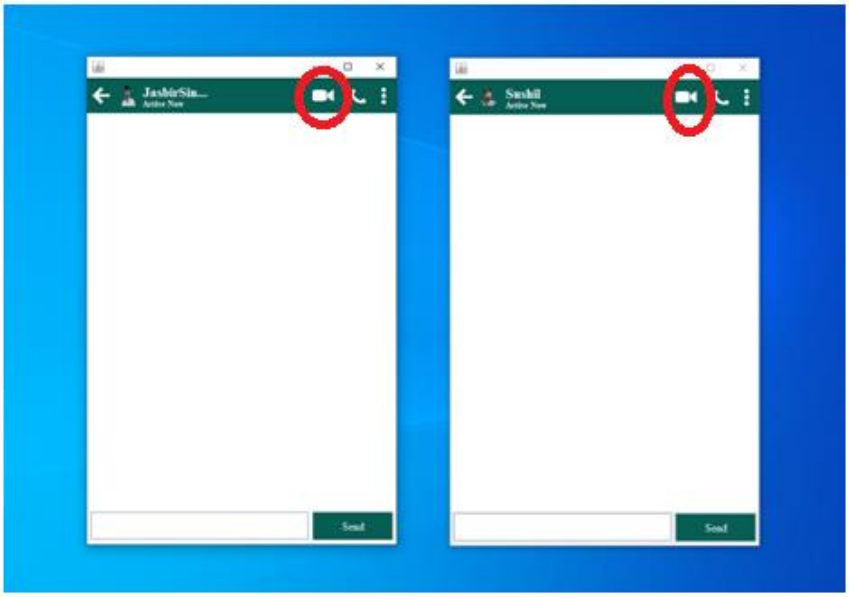


AFTER

- **Voice/Video Call**

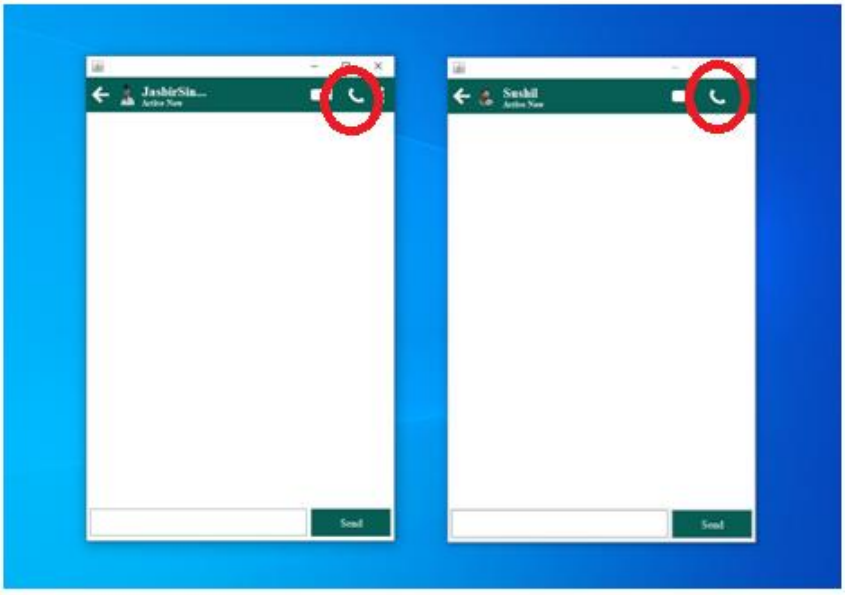
This feature is under development and will be enabled in the future updates because of limited knowledge right now.

Video calling option for both Serve and Client side.



VIDEO CALL

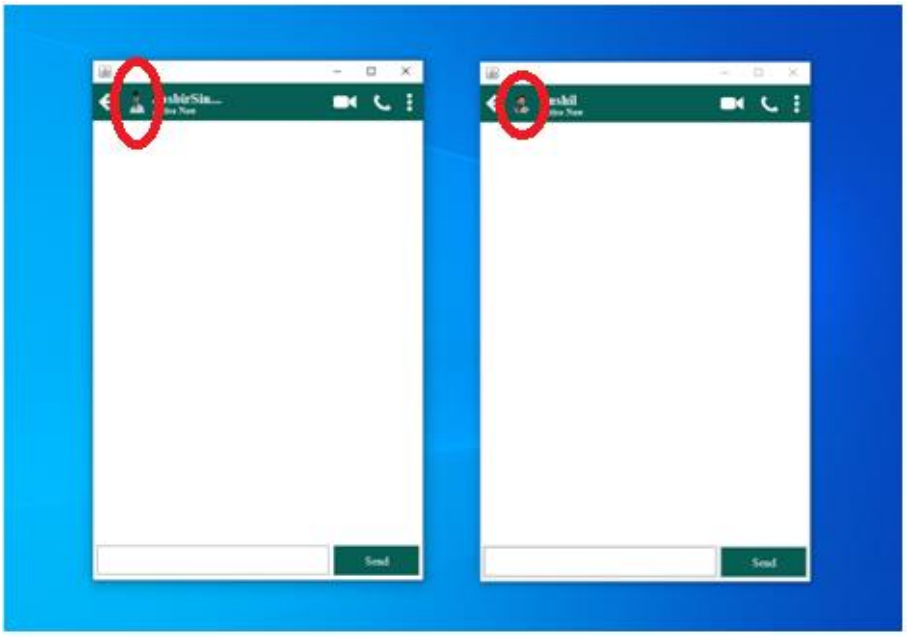
Voice/Audio calling option for both Server and Client side.



AUDIO CALL

- **Profile (DP)**

This section of the chat will help user to set their favorite images as their Profile Image. Down below are the samples of DP of testing users as Server and Client.

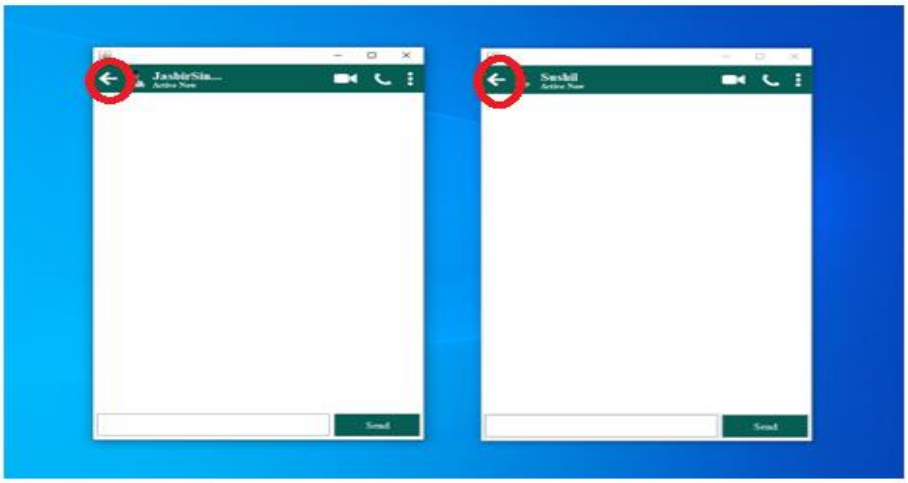


PROFILE PICS

- **Logout function:**

The Back button highlighted below implies the Logout function.

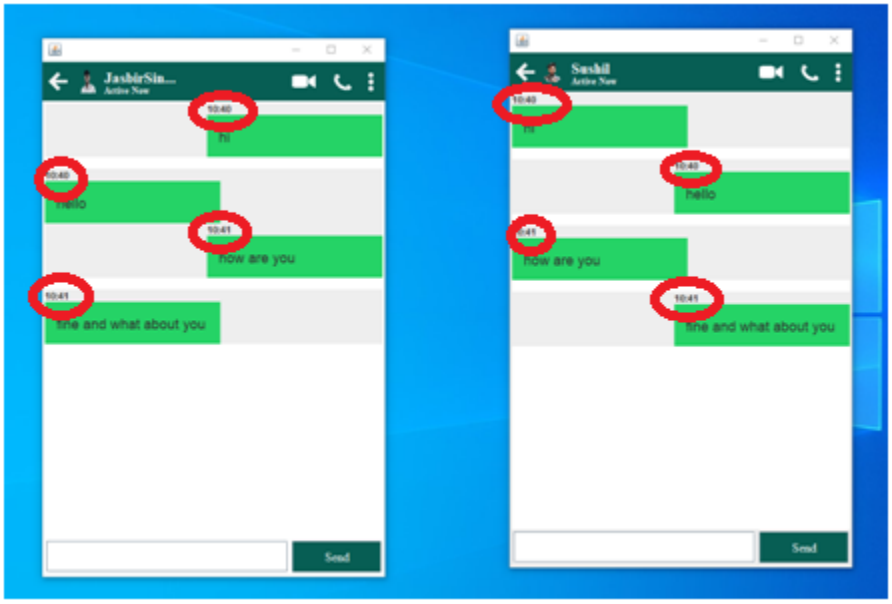
By clicking on the Back button the respective user will be logged out.



LOGOUT

- **Time Stamp:**

The time stamp feature highlighted below will help user to see at what the message was sent and received by other person.



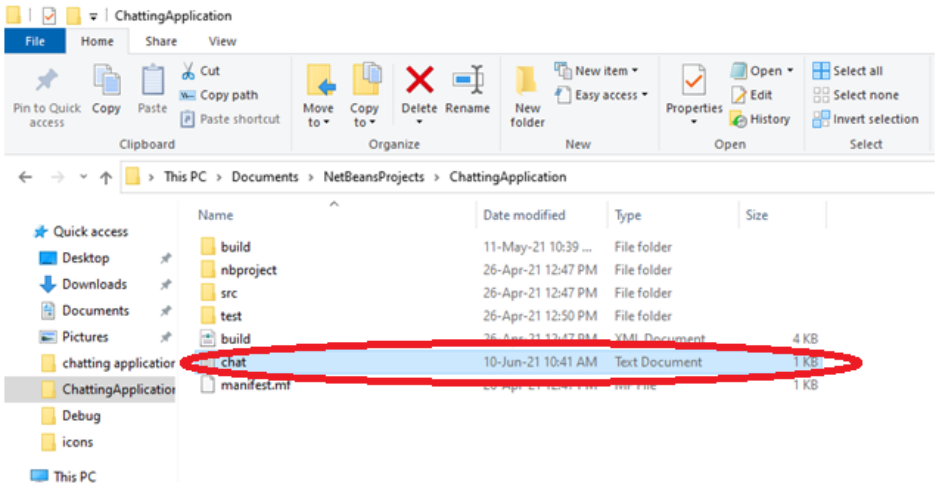
TIME STAMP

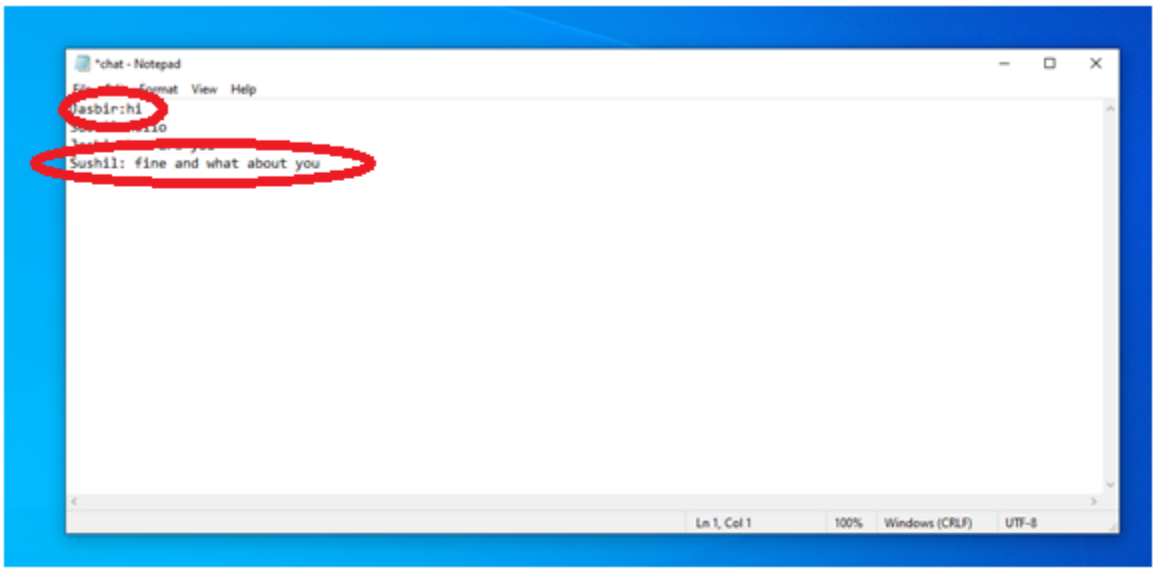
- **Automatic Backup file data:**

The most useful and most unique feature of this application is that it automatically backups the chat data in form of *.txt* file as shown below.

NOTE :

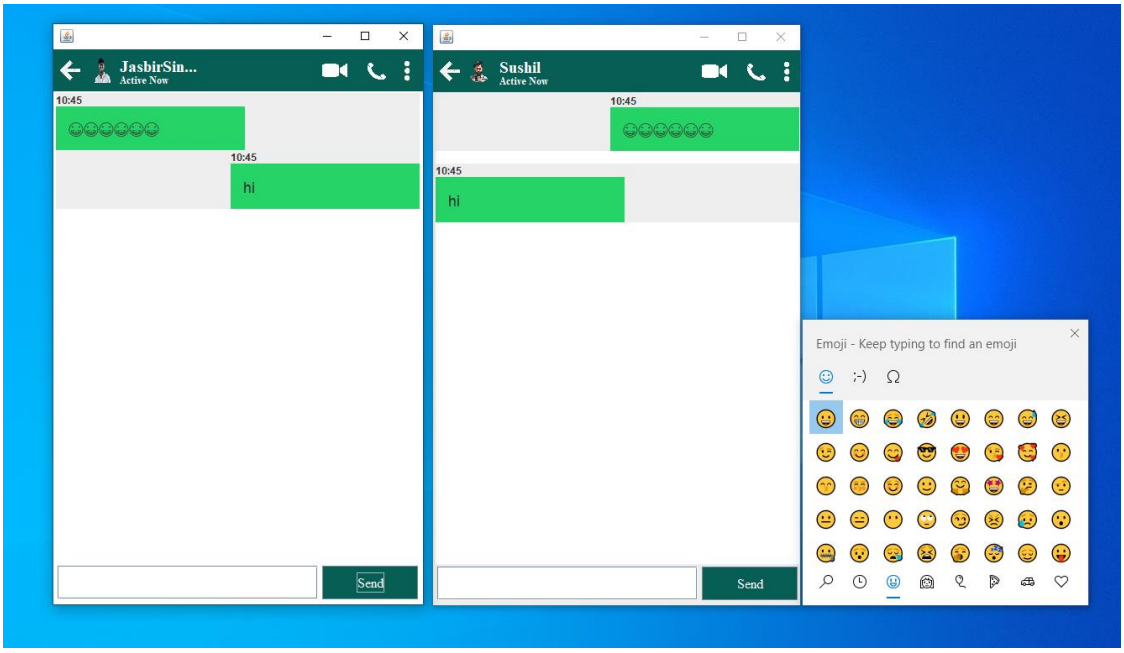
IN BACKUP WE HAVE NAME IN START OF EVERY MESSAGE THAT WHICH USER SENDED THE MESSAGE


ChattingApplication
File
Home
Share
View
Cut
Copy path
Paste shortcut
Pin to Quick access
Copy
Paste
Clipboard
Move to
Copy to
Delete
Rename
Organize
New item
Easy access
New folder
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
This PC › Documents › NetBeansProjects › ChattingApplication
Quick access
Desktop
Downloads
Documents
Pictures
chatting applicatior
ChattingApplicatior
Debug
icons
This PC
Name
Date modified
Type
Size
build
11-May-21 10:39 ...
File folder
nbproject
26-Apr-21 12:47 PM
File folder
src
26-Apr-21 12:47 PM
File folder
test
26-Apr-21 12:50 PM
File folder
build
4 KB
chat
10-Jun-21 10:41 AM
Text Document
1 KB
manifest.mf
1 KB


*chat - Notepad
Format
View
Help
Jasbir:hi
Sushil: fine and what about you
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

- **Inbuilt Emoji's:**

This feature of the chat application enables user to use the in-built function of emoiji's which enhances the chatting application.



4. Planning and Scheduling

- *Project Phase Schedule*

Process name	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9
Collect system requirements									
Prepare report									
Module testing									
Test case									
User training									
Software handover									

5. SYSTEM DEVELPOMENT

➢ Coding Details

The Server Side Coding.

```
package chattingapplication;


import java.awt.Image;

import javax.swing.*;
```

```
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.io.*;
import java.text.SimpleDateFormat;
import javax.swing.border.*;
//for the scrollpane editing
import javax.swing.plaf.*;
import javax.swing.plaf.basic.*;

import java.util.Calendar;
import java.text.DateFormat;

/**
 *
 * @author Sushil and Jasbir
 */
public class Server implements ActionListener {


    JPanel p1;
    JTextField t1;
    JButton b1;
    static JPanel a1;
```

```
 static JFrame f1= new JFrame();

 static Box vertical = Box.createVerticalBox();

 //Socketing
 static ServerSocket skt;
 static Socket s;
 static DataInputStream din;
 static DataOutputStream dout;

 //it is for changing active now to typing...
 Boolean typing;//by default it is false

 Server(){
     f1.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
     p1 = new JPanel();
     p1.setLayout(null);
     p1.setBackground(new Color(7, 94, 84));
     p1.setBounds(0, 0, 450, 50);
     f1.add(p1);

    ImageIcon i1= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/3.png"));
    Image i2=i1.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
    ImageIcon i3=new ImageIcon(i2);
```

```
JLabel l1= new JLabel(i3);
 l1.setBounds(5,10,30,30);
 p1.add(l1);

 //working on the back button to exit the window
 l1.addMouseListener(new MouseAdapter(){
 public void mouseClicked(MouseEvent ae){
 System.exit(0);
 }
 });


    ImageIcon i4= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/jasbir.png"));
 Image i5=i4.getImage().getScaledInstance(35,35 , Image.SCALE_DEFAULT);
 ImageIcon i6=new ImageIcon(i5);
 JLabel l2= new JLabel(i6);
 l2.setBounds(40,10,50,30);
 p1.add(l2);

 //1

    ImageIcon i7= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/video.png"));
 Image i8=i7.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
 ImageIcon i9=new ImageIcon(i8);
```

```
JLabel l5= new JLabel(i9);
  l5.setBounds(320,10,30,30);
  p1.add(l5);
  //2


  ImageIcon i11= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/phone.png"));
JLabel l6= new JLabel(i13);
  l6.setBounds(370,12,30,30);
  p1.add(l6);
  //3


    ImageIcon i14= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/3icon.png"));
JLabel l7= new JLabel(i16);
  l7.setBounds(410,10,22,30);
  p1.add(l7);


  JLabel l3 = new JLabel("JasbirSingh  ");
  l3.setFont( new Font("Serif",Font.BOLD,20) );
```

```
l3.setForeground(Color.WHITE);
l3.setBounds(79,  10, 100, 18);
p1.add(l3);



JLabel l4 = new JLabel("Active  Now");
l4.setFont(  new Font("Serif",Font.BOLD,12)  );
l4.setForeground(Color.WHITE);
l4.setBounds(79,  26, 100, 20);
p1.add(l4);


//timer  used
Timer  t=new Timer(1,  new ActionListener()  {
    @Override
    public  void actionPerformed(ActionEvent  ae) {
      if(!typing){
          l4.setText("Active  Now");
      }
    }
});
t.setInitialDelay(2000);//takes  time  in the millisecond


//object for the textarea
a1=new JPanel();
a1.setBackground(Color.WHITE);
```

```
// a1.setBounds(3, 52,433,560);//this is commented because we make these things in scrollpane now
a1.setFont( new Font("Serif",Font.PLAIN,16) );
//f1.add(a1);//as we make this in scroll pane

//adding scroll bar to panel
JScrollPane sp = new JScrollPane(a1);
sp.setBounds(3, 52,433,560);
sp.setBorder(BorderFactory.createEmptyBorder());//we have to remove the border

//editing of scrollpane(increase,decrease,track)
//so we have to import the abstract class ScrollBarUi and make object of its child class
ScrollBarUI ui= new BasicScrollBarUI(){
protected JButton createDecreaseButton(int orientation){

    JButton button = super.createDecreaseButton(orientation);
    button.setBackground(new Color( 7, 94, 84));
    button.setForeground(Color.WHITE);
    this.thumbColor = new Color( 7, 94, 84);//this is used to change color of the scroll
   // this.trackColor = new Color(7,83,93);//to change the track color
    return button;
}
protected JButton createIncreaseButton(int orientation){
```

```
    JButton button = super.createIncreaseButton(orientation);
    button.setBackground(new Color( 7, 94, 84));
    button.setForeground(Color.WHITE);
   this.thumbColor = new Color( 7, 94, 84);
    return button;
}

};
sp.getVerticalScrollBar().setUI(ui);
f1.add(sp);

//for the textfield to write it into
t1= new JTextField();//object for the textfeild
t1.setBounds(5,615,310,40);
t1.setFont( new Font("Serif",Font.PLAIN,16));
f1.add(t1);

//working on the textfield working to change the active status to typing....
t1.addKeyListener(new KeyAdapter() {
 public void keyPressed(KeyEvent ke){
l4.setText("typing...");

t.stop();
typing=true;
 }
```

```
public void keyReleased(KeyEvent ke){

typing=false;

if(!t.isRunning()){
t.start();//this is used as when the typing is stopped then the timer will be started for 2sec and then the Active now again appears
}
} }
);

//button which play a role for sending the text
b1=new JButton ("Send");
b1.setBounds(320,615,114,40);
b1.setBackground(new Color(7, 94, 84));
b1.setForeground(Color.WHITE);

b1.setFont( new Font("Serif",Font.PLAIN,16));
b1.addActionListener(this);
f1.add(b1);



//CHANGING THE BACKGROUND THEME
f1.getContentPane().setBackground(Color.WHITE);
```

```
        f1.setLayout(null);
       f1.setSize(455,700);
       f1.setLocation(100,100);
  //IS TO REMOVE THE TOP DEFAULT BORDER WHICH CONTAIN THE OPTION OF MINMIZE ETX...
       //setUndecorated(true);
       f1.setVisible(true);




}


public void actionPerformed(ActionEvent ae)
{ try {
     String out = t1.getText();
     sendTextToFile(out);//the textarea message is sending for storing into the file
     JPanel p2 = formatLabel(out);//for giving the output in formated string for bubble


     a1.setLayout(new BorderLayout());


     JPanel right=new JPanel(new BorderLayout());
     right.add(p2,BorderLayout.LINE_END);
     vertical.add(right);
     vertical.add(Box.createVerticalStrut(15));
```

```
    a1.add(vertical,BorderLayout.PAGE_START);


    //a1.add(p2);
    dout.writeUTF(out);
    t1.setText("  ");


   } catch (Exception  e) {


   }


}


public  void  sendTextToFile(String  message)  throws FileNotFoundException  {
   try(FileWriter  f = new FileWriter("chat.txt",true);//we  have to pass the true value  as if we dont pss it it dont allow  us to append in the file..
          PrintWriter  p = new PrintWriter  (new BufferedWriter(f));){//buffered  is used to take every single  character from the sending  message
          p.println("Jasbir:"  + message);



       }catch(Exception  e){
       e.printStackTrace();
       }



}
```

```
//funtion  for the panel buble  layout
    public  static  JPanel formatLabel(String  out){
    JPanel  p3 = new JPanel();
    p3.setLayout(new  BoxLayout(p3,BoxLayout.Y_AXIS  ));

    JLabel l1 = new JLabel("<html><p  style=\"width:150px\">"+out+"</p></html>");
    l1.setFont(new  Font("Comic-Sans",Font.PLAIN,18));
    l1.setBackground(new  Color( 37, 211, 102));
    l1.setOpaque(true);
    l1.setBorder(new  EmptyBorder(15,15,15,15));

    Calendar  cal =Calendar.getInstance();
    SimpleDateFormat  sdf = new SimpleDateFormat("HH:mm");

    JLabel l2 = new JLabel();
    l2.setText(sdf.format(cal.getTime()));

    p3.add(l2);//as  label (l2) is of time-format  it is our wish where we have to print it in the
above label which  print  the sended text or below the text

    p3.add(l1);

    return p3;
    }
```

```
    public static void main (String[] args)
{
new Server().f1.setVisible(true);

String msginput = "";

    try {
      skt = new ServerSocket(6001);
      while(true){
      s = skt.accept();

      din= new DataInputStream(s.getInputStream());
      dout =new DataOutputStream(s.getOutputStream());


      while(true){
         msginput=din.readUTF();
         JPanel p2 = formatLabel(msginput);
         JPanel left= new JPanel(new BorderLayout());
         left.add(p2,BorderLayout.LINE_START);
         vertical.add(left);
         f1.validate();
      }
```

```
        }
    } catch (Exception  e) {
    }

}
}
```

The Cleint Side Coding.

```
package chattingapplication;

import  static chattingapplication.Server.a1;
import  java.awt.Image;
import  javax.swing.*;
import  java.awt.*;
import  java.awt.event.*;
import  java.net.*;
import  java.io.*;
import  java.text.SimpleDateFormat;
import  javax.swing.border.*;
import  java.util.Calendar;
import  java.text.DateFormat;
import  javax.swing.plaf.ScrollBarUI;
import  javax.swing.plaf.basic.BasicScrollBarUI;
```

```
/**
 *
 * @author Sushil and Jasbir
 */
public class Client implements ActionListener {

    JPanel p1;
    JTextField t1;
    JButton b1;
      static JPanel a1;

    static JFrame f1= new JFrame();

   static Box vertical = Box.createVerticalBox();


    //socket only is made up because there is only server
    static Socket s;
     static DataInputStream din;
   static DataOutputStream dout;

   //it is for changing active now to typing...
        Boolean typing;//by default it is false
```

```
    Cleint(){
        p1 = new JPanel();
        p1.setLayout(null);
        p1.setBackground(new Color(7, 94, 84));
        p1.setBounds(0, 0, 450, 50);
        f1.add(p1);

      ImageIcon i1= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/3.png"));
      Image i2=i1.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
      ImageIcon i3=new ImageIcon(i2);
      JLabel l1= new JLabel(i3);
       l1.setBounds(5,10,30,30);
       p1.add(l1);

       //working on the back button to exit the window
       l1.addMouseListener(new MouseAdapter(){
       public void mouseClicked(MouseEvent ae){
       System.exit(0);
       }
       });

          ImageIcon i4= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/sushil.png"));
```

```
Image i5=i4.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
ImageIcon i6=new ImageIcon(i5);
JLabel l2= new JLabel(i6);
 l2.setBounds(40,10,30,30);
 p1.add(l2);

 //1

    ImageIcon i7= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/video.png"));
Image i8=i7.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
ImageIcon i9=new ImageIcon(i8);
JLabel l5= new JLabel(i9);
 l5.setBounds(320,10,30,30);
 p1.add(l5);
 //2

 ImageIcon i11= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/phone.png"));
Image i12=i11.getImage().getScaledInstance(30,30 , Image.SCALE_DEFAULT);
ImageIcon i13=new ImageIcon(i12);
JLabel l6= new JLabel(i13);
 l6.setBounds(370,12,30,30);
 p1.add(l6);
 //3
```

```
        ImageIcon i14= new
ImageIcon(ClassLoader.getSystemResource("chattingapplication/icons/3icon.png"));
    Image i15=i14.getImage().getScaledInstance(10,25  , Image.SCALE_DEFAULT);
    ImageIcon i16=new ImageIcon(i15);
    JLabel l7= new JLabel(i16);
     l7.setBounds(410,10,22,30);
     p1.add(l7);



     JLabel l3 = new JLabel("Sushil ");
     l3.setFont(  new Font("Serif",Font.BOLD,20)  );
     l3.setForeground(Color.WHITE);
     l3.setBounds(79,  10, 100, 18);
     p1.add(l3);



     JLabel l4 = new JLabel("Active  Now");
     l4.setFont(  new Font("Serif",Font.BOLD,12)  );
     l4.setForeground(Color.WHITE);
     l4.setBounds(79,  26, 100, 20);
     p1.add(l4);

        //timer  class used
         Timer  t=new Timer(1,  new ActionListener()  {
```

```
    @Override
    public void actionPerformed(ActionEvent ae) {
      if(!typing){
        l4.setText("Active Now");
      }
    }
  });
  t.setInitialDelay(2000);//takes time in the millisecond

  //object for the textarea
  a1=new JPanel();
  a1.setBackground(Color.WHITE);
 // a1.setBounds(3, 52,433,560);//this is commented because we make these things in scrollpane now
  a1.setFont( new Font("Serif",Font.PLAIN,16) );
  //f1.add(a1);//as we make this in scroll pane

  //adding scroll bar to panel
  JScrollPane sp = new JScrollPane(a1);
  sp.setBounds(3, 52,433,560);
  sp.setBorder(BorderFactory.createEmptyBorder());//we have to remove the border

  //editing of scrollpane(increase,decrease,track)
  //so we have to import the abstract class ScrollBarUi and make object of its child class
```

```
ScrollBarUI  ui=  new BasicScrollBarUI(){
protected JButton createDecreaseButton(int  orientation){

    JButton  button  = super.createDecreaseButton(orientation);
    button.setBackground(new  Color( 7, 94, 84));
    button.setForeground(Color.WHITE);
    this.thumbColor  = new Color( 7, 94, 84);//this  is used to change  color of the scroll
   // this.trackColor  = new Color(7,83,93);//to  change  the track color
    return button;
}
protected JButton createIncreaseButton(int  orientation){

    JButton  button  = super.createIncreaseButton(orientation);
    button.setBackground(new  Color( 7, 94, 84));
    button.setForeground(Color.WHITE);
    this.thumbColor  = new Color( 7, 94, 84);
    return button;
}

};
sp.getVerticalScrollBar().setUI(ui);
f1.add(sp);


//for  the textfield  to write  it into
```

```
t1= new JTextField();//object  for the textfeild
t1.setBounds(5,615,310,40);
t1.setFont( new Font("Serif",Font.PLAIN,16));
f1.add(t1);

//working  on the textfield  working  to change  the active  status  to typing....
t1.addKeyListener(new  KeyAdapter() {
 public  void keyPressed(KeyEvent  ke){
l4.setText("typing...");

t.stop();
typing=true;
 }
 public  void keyReleased(KeyEvent  ke){

 typing=false;

 if(!t.isRunning()){
 t.start();//this  is used as when the typing  is stopped then the timer  will  be started  for 2sec and then  the Active  now again  appears
 }
 }  }
);

//button  which  play  a role for sending  the text
```

```
        b1=new JButton ("Send");
        b1.setBounds(320,615,114,40);
        b1.setBackground(new  Color(7, 94, 84));
        b1.setForeground(Color.WHITE);
        b1.setFont( new Font("Serif",Font.PLAIN,16));
       b1.addActionListener(this);
        f1.add(b1);




  //CHANGING THE BACKGROUND THEME
      //getContentPane().setBackground(Color.YELLOW);
      f1.setLayout(null);
      f1.setSize(455,700);
      f1.setLocation(800,100);
  //IS TO REMOVE THE TOP DEFAULT BORDER WHICH CONTAIN THE OPTION OF
MINMIZE ETX...
      //setUndecorated(true);
      f1.setVisible(true);




}


public  void actionPerformed(ActionEvent  ae)
```

```
{  try {
    String  out = t1.getText();
          sendTextToFile(out);
    JPanel p2 = formatLabel(out);//for  giving  the output in formated  string  for bubble

    a1.setLayout(new  BorderLayout());

    JPanel right=new  JPanel(new  BorderLayout());
    right.add(p2,BorderLayout.LINE_END);
    vertical.add(right);
    vertical.add(Box.createVerticalStrut(15));

    a1.add(vertical,BorderLayout.PAGE_START);

    //a1.add(p2);
    dout.writeUTF(out);
    t1.setText("  ");

   } catch (Exception  e) {

   }


}
```

```
public void sendTextToFile(String message) throws FileNotFoundException {
   try(FileWriter f = new FileWriter("chat.txt",true);//file is created if not already else append
        PrintWriter p = new PrintWriter (new BufferedWriter(f));){//buffered is used to take every single character from the sending message
        p.println("Sushil:"+message);


      }catch(Exception e){
      e.printStackTrace();
      }


}

 //funtion for the panel buble layout
      public static JPanel formatLabel(String out){
      JPanel p3 = new JPanel();
      p3.setLayout(new BoxLayout(p3,BoxLayout.Y_AXIS ));

      JLabel l1 = new JLabel("<html><p style=\"width:150px\">"+out+"</p></html>");
      l1.setFont(new Font("Comic-Sans",Font.PLAIN,18));
      l1.setBackground(new Color( 37, 211, 102));
      l1.setOpaque(true);
      l1.setBorder(new EmptyBorder(15,15,15,15));
```

```
        Calendar cal =Calendar.getInstance();
        SimpleDateFormat  sdf = new SimpleDateFormat("HH:mm");

        JLabel l2 = new JLabel();
        l2.setText(sdf.format(cal.getTime()));
        p3.add(l2);



        p3.add(l1);
        return p3;
        }



    public static void main (String[] args)
{
new Cleint().f1.setVisible(true);

    try {
        s= new Socket("127.0.0.1",6001);
        din= new DataInputStream(s.getInputStream());
        dout= new DataOutputStream(s.getOutputStream());

        //reading the msg
```

```
        String msginput="";

        while(true){
      a1.setLayout(new BorderLayout());
       msginput=din.readUTF();
       JPanel p2= formatLabel(msginput);
       JPanel left= new JPanel(new BorderLayout());
       left.add(p2,BorderLayout.LINE_START);

       vertical.add(left);
       vertical.add(Box.createVerticalStrut(15));
       a1.add(vertical,BorderLayout.PAGE_START);
       f1.validate();


        }
     } catch (Exception e) {
     }
}
}
```

6. Milestones and Testing

➢ Testing Plans

System testing is the process of performing a variety of tests on a system to eplore functionality or to identify problems. System testing is usually required before and after a system is put in place. A series of systematic producers are referrd to while testing is being performed. These procedures tell the tester how the system should perform and where common mistakes may be found. Testers usually try to "break the system" by entering data that may cause the system to malfunction or return incorrect information.

So, this could mean two things depending on an SDLC model. The first type of testing is the actual testing by users. This is usually done in models wherein implementation does not go with pre-testing with users. On the other hand, there are also testing that uses professionals in field. This testing is aimed in cleaning the software of all the bugs altogether. For the software that is set for public release the software is first tested by other developers who were not in charge in creating this software. They will weed out the bugs and suggest fixes if every they find one. Once this stage is completed, it is time to test the software not just to developers but to actual users.

• Types and Steps of Testing:

The following steps are important to perform System Testing:

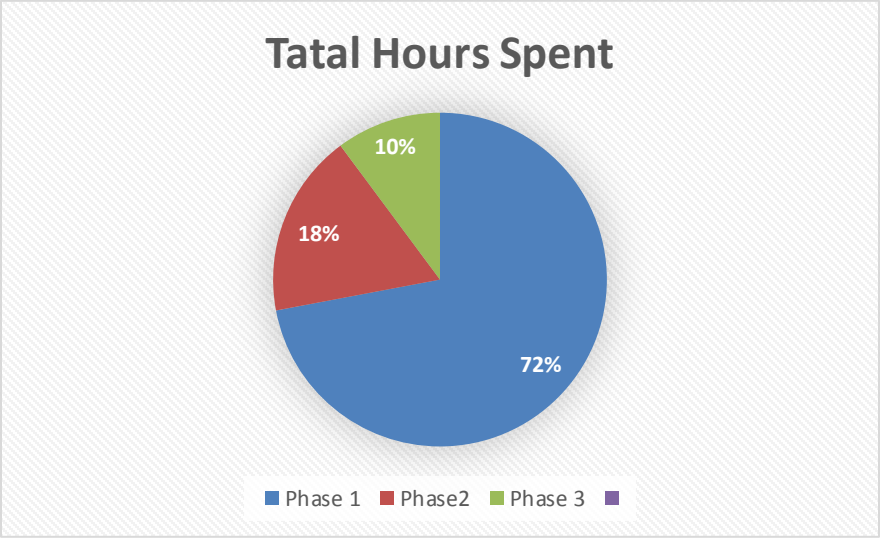
- ✓ Step 1: Create a system Test plan.
- ✓ Step 2: Create test cases.
- ✓ Step 3: Carefully Build Data used as Input for System Testing.
- ✓ Step 4: If applicable create scripts to build environment and to automatic execution of test cases.
- ✓ Step 5: Execute the test cases.
- ✓ Step 6: Fix the bugs if any and re test the code.
- ✓ Step 7: Repeat the test cycle as necessary.

- **Test Report Details:**

Every test of the actual-testing or unit-testing is documented in a test report.
Test reports include the following data:

- ✓ Name of the tester.
- ✓ Date and time at which the test was performed.
- ✓ Scope of test.
- ✓ Test environment including operating system and type of test client.
- ✓ Test plan i.e the list of test cases giving method/use case and input parameters used.
- ✓ Test log listening the outcome of test case execution (manual testing or automated testing)
- ✓ Summary of test results.
- ✓ Recommendations for action when errors had been encountered.

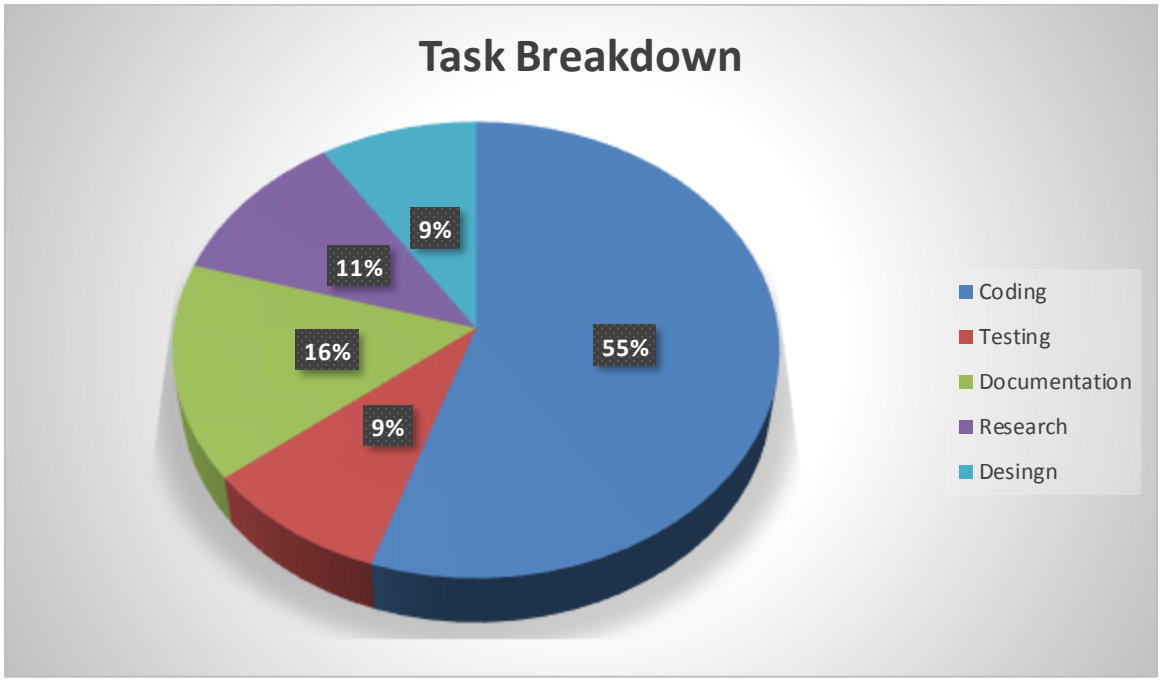
- **Total Hours Spent in Project**



Phase 1: Coding	Phase 2: Testing	Phase 3: Testing

- **Task Breakdown**

For the final completion of this project, tasks are being divided into many parts which can be seen in the below pie chart.



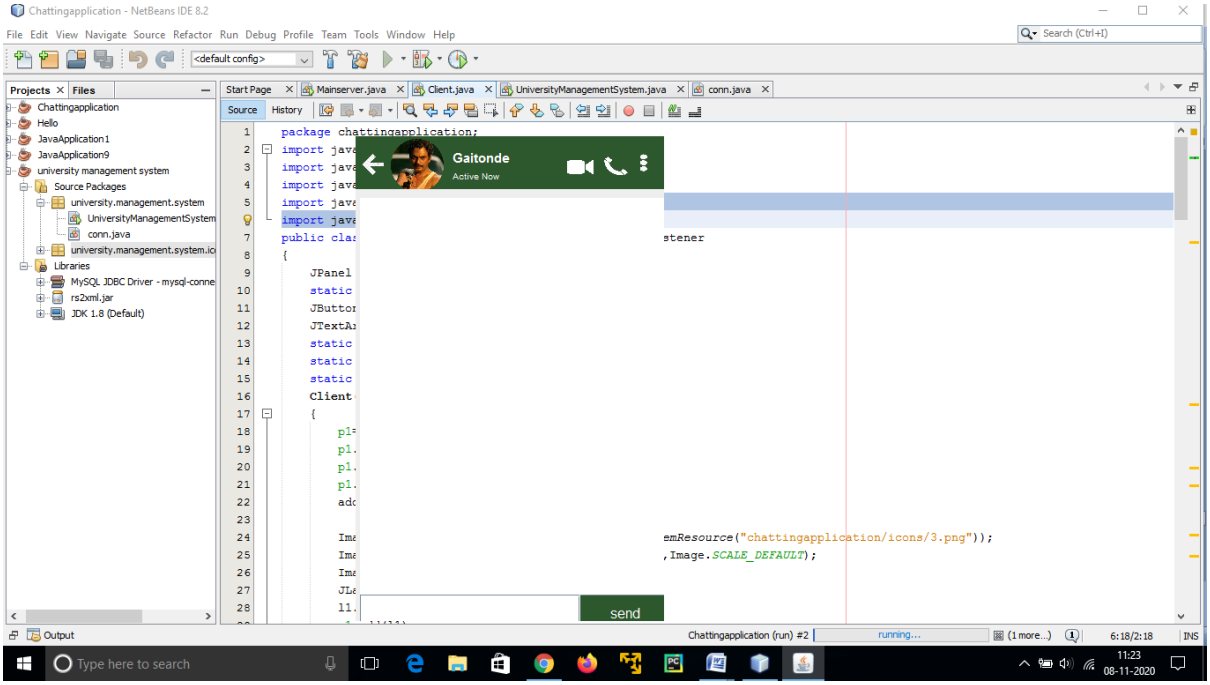
- **Testing during and after development:**

This part of the testing will include

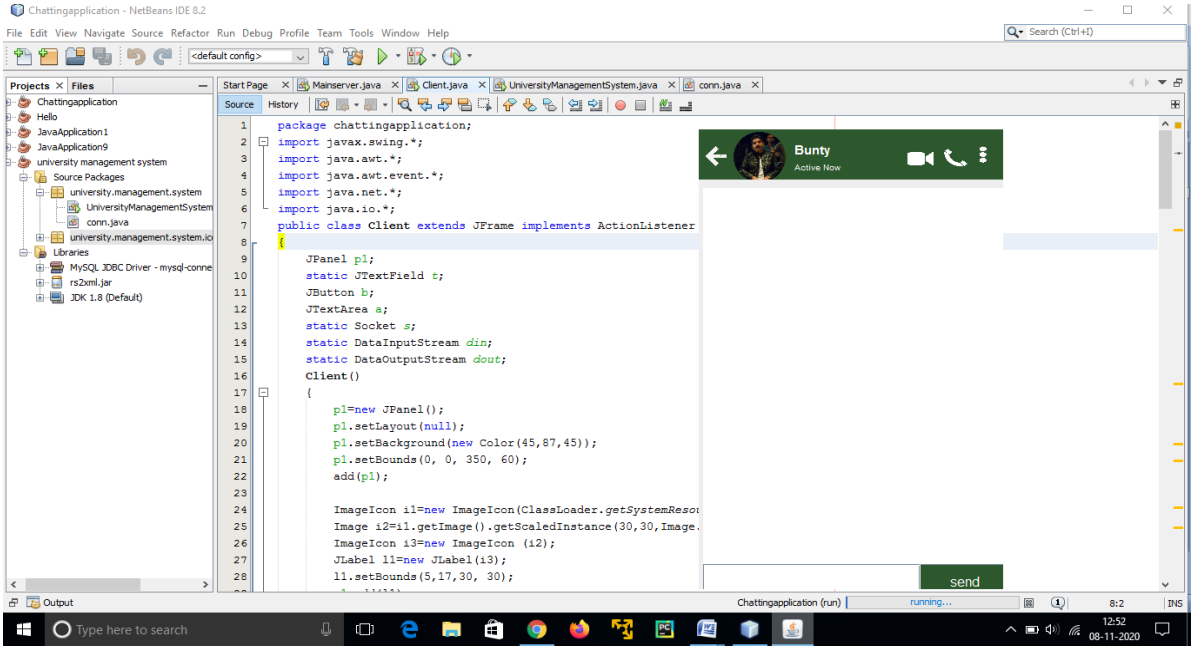
Sample User Name for Server Side and Client Side both.

Here during the testing will the screen will look like represented below as below.

The Server Side during Testing



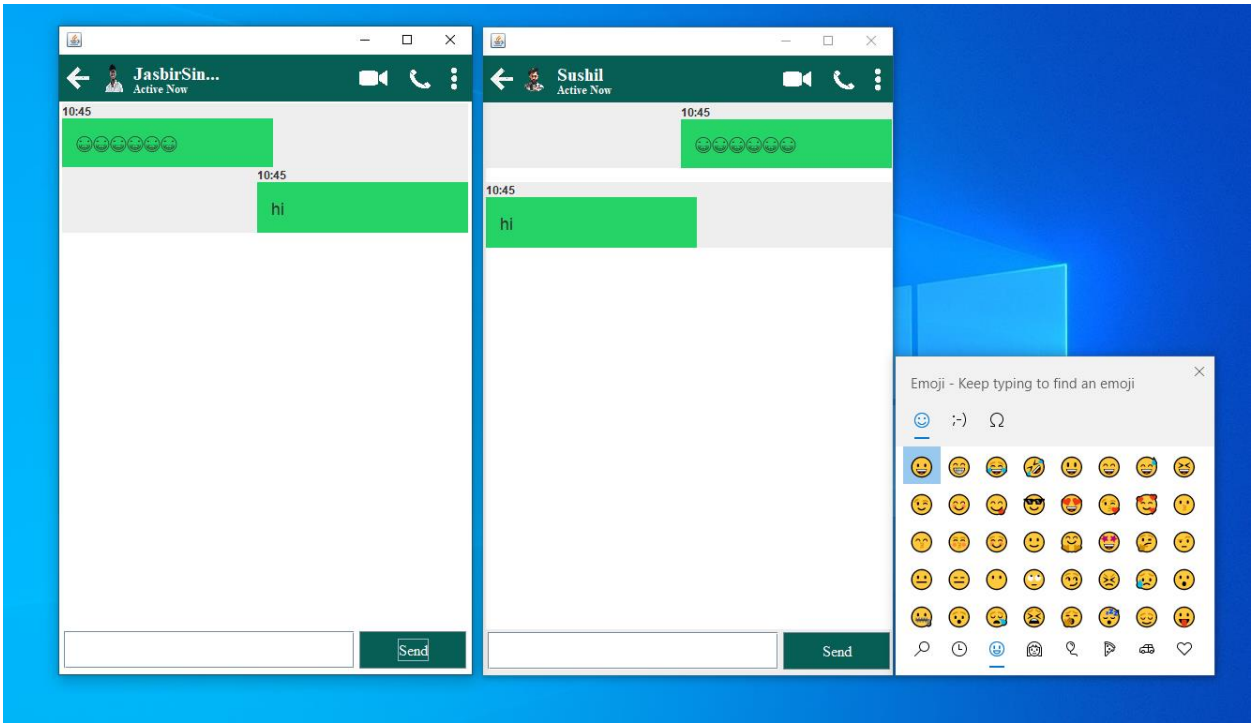
The Client Side during Testing



The Server Side and Client Side After Testing

SERVER SIDE

CLIENT SIDE



➢ Software and Hardware Requirements

Hardware Configuration:

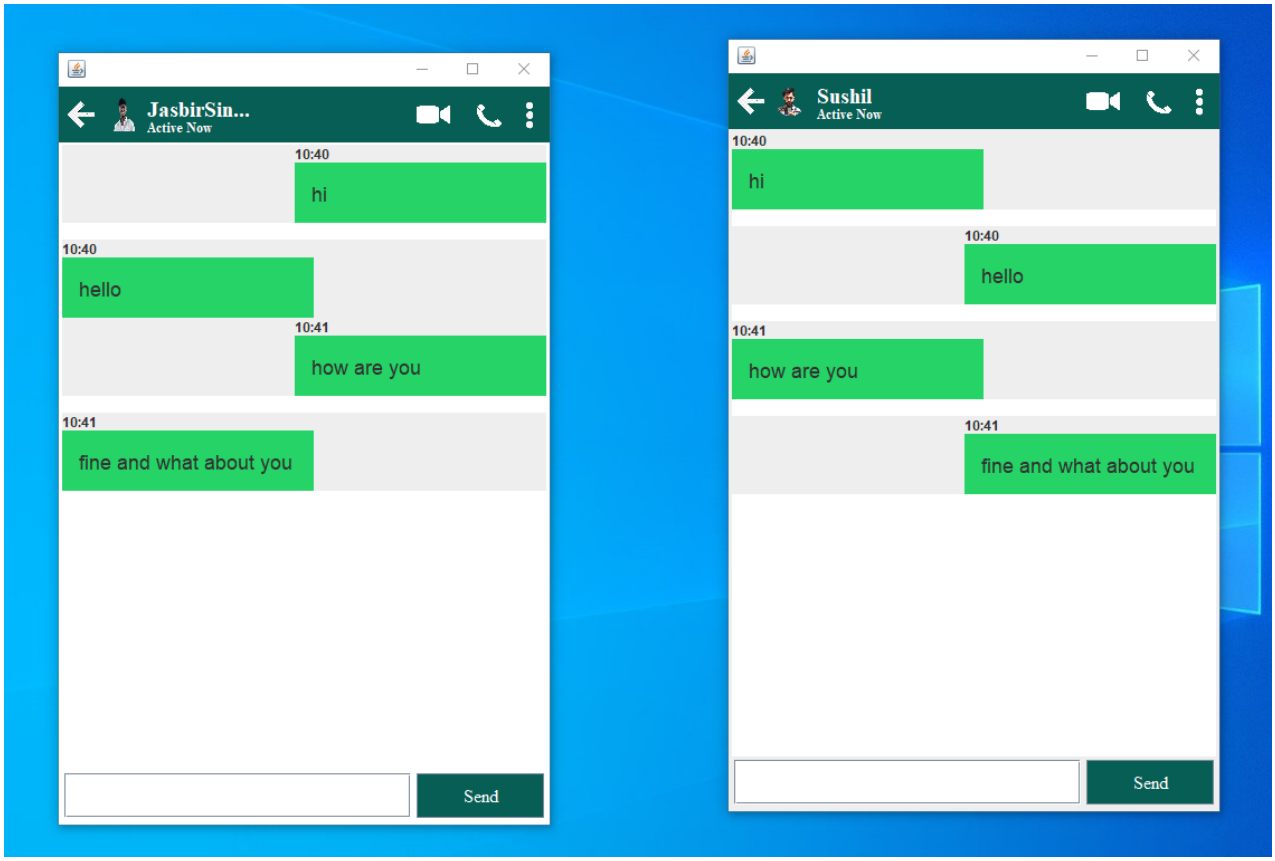
Processor	Intel Core 2 Duo or above
Memory	512 MB RAM or above
Cache Memory	128 KB or above
Hard Disk	10 GB or above

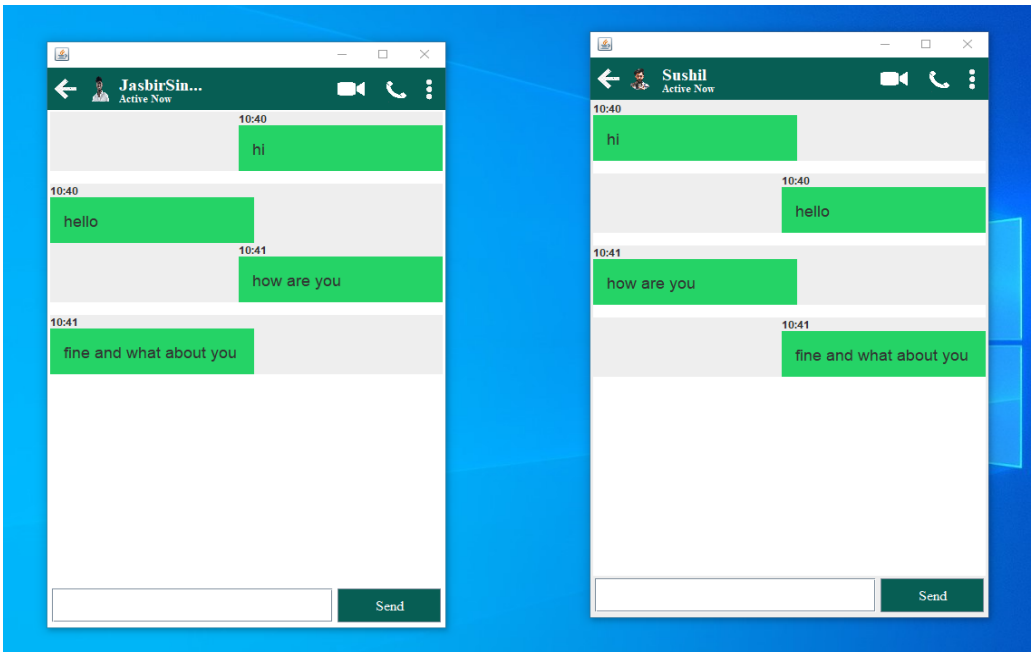
Software Requirements:

Operating System	Windows or any equivalent OS
Coding	Java version 1.8.0.271 or above

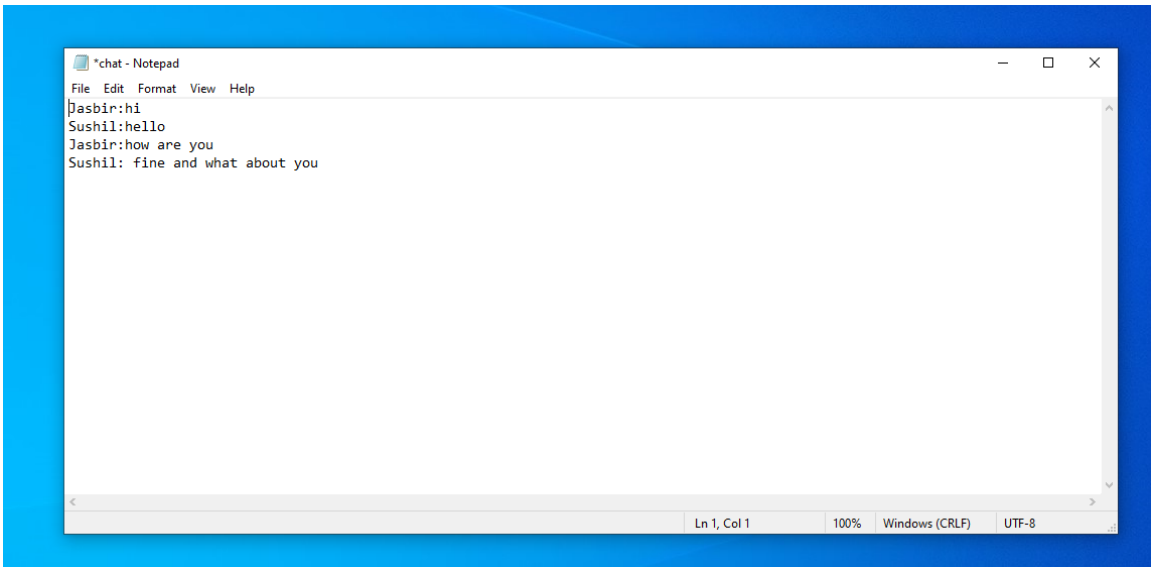
7. FINAL PRODUCT VIEW

Now our final product is ready after with testing and research. Now we can easily avail the chatting feature of the application. Some of the final pictures of the product are represented below.





BELOW SHOWN IS THE CHAT HISTORY

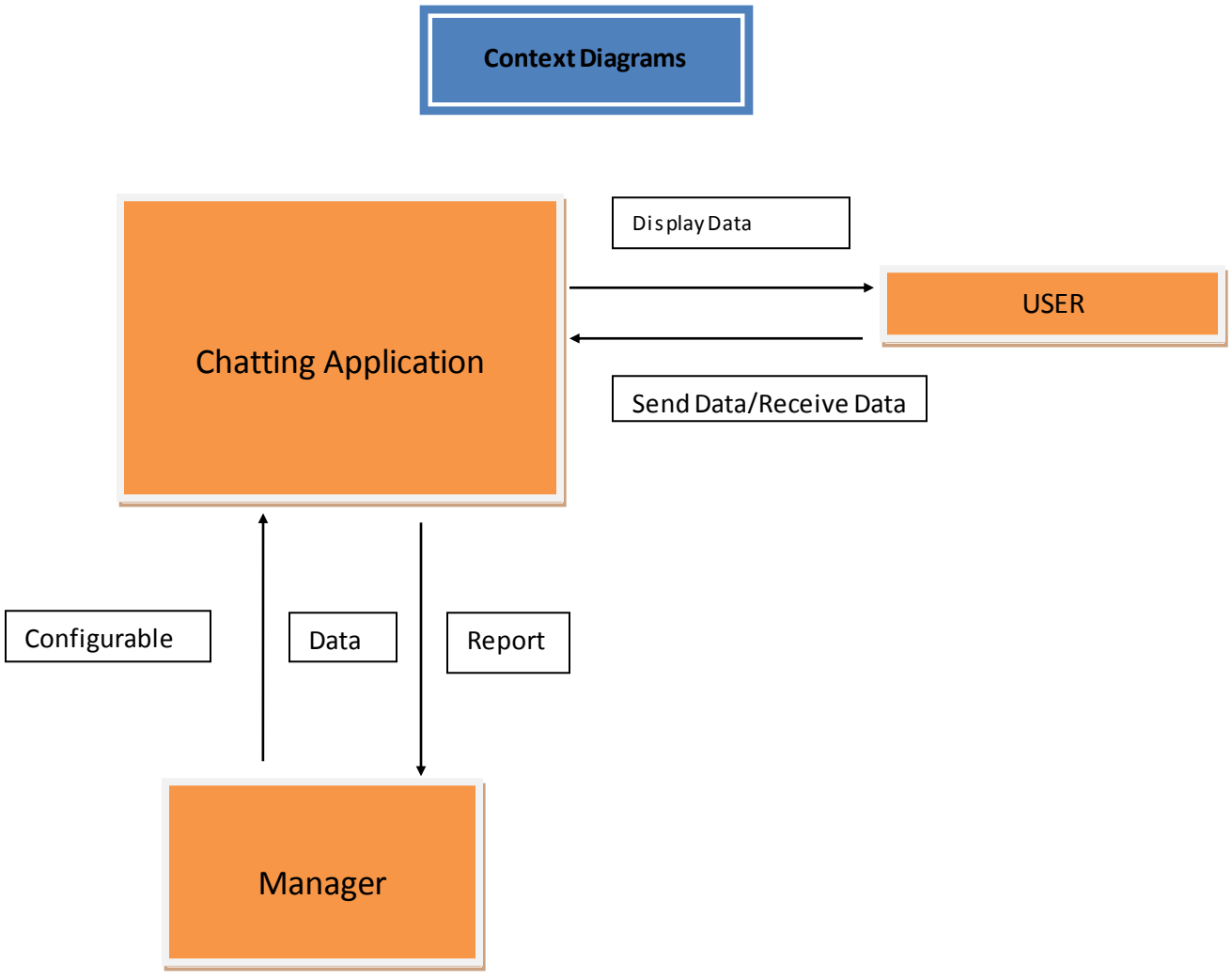


8. System Evaluation

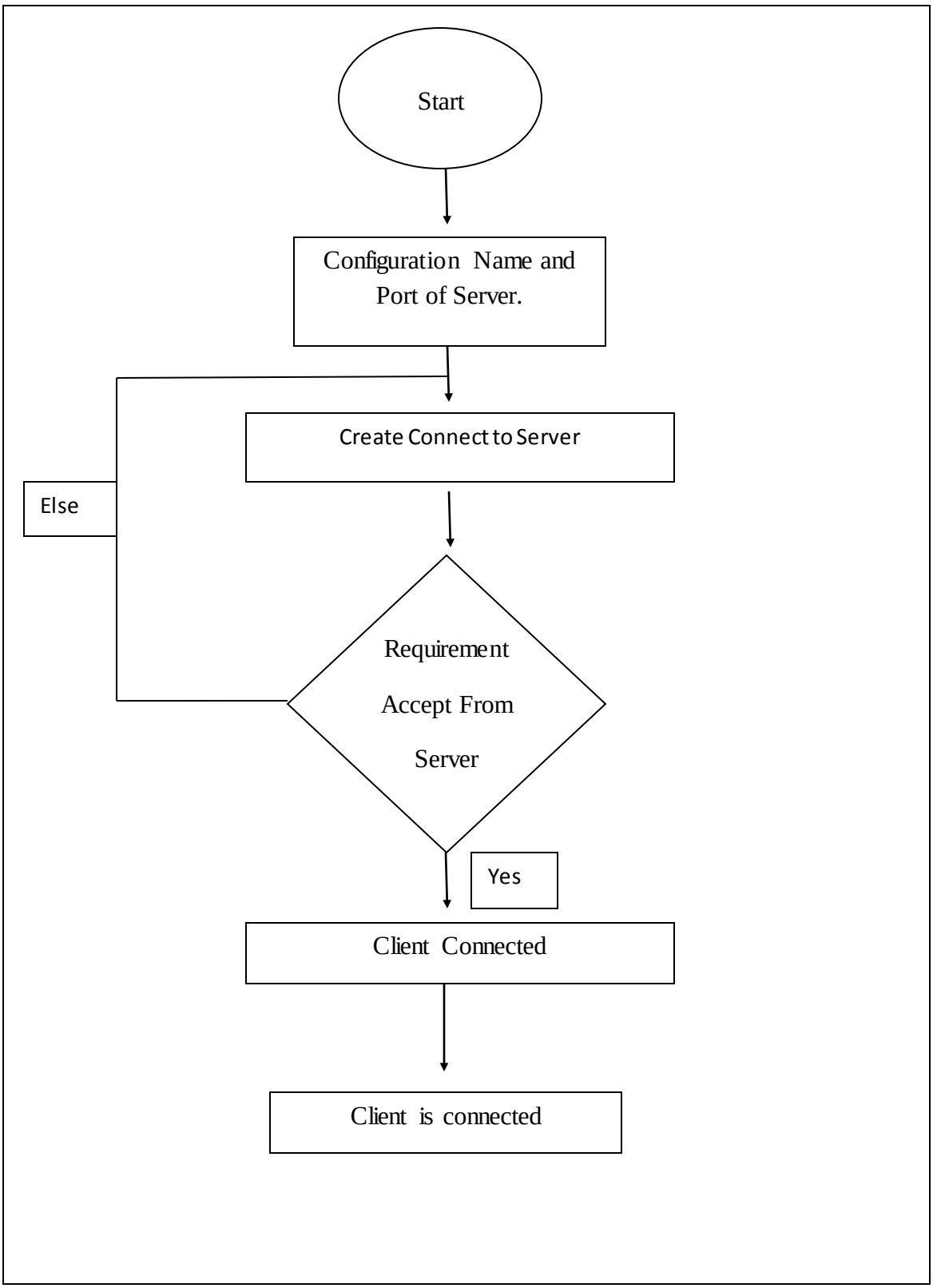
➢ ER Diagrams

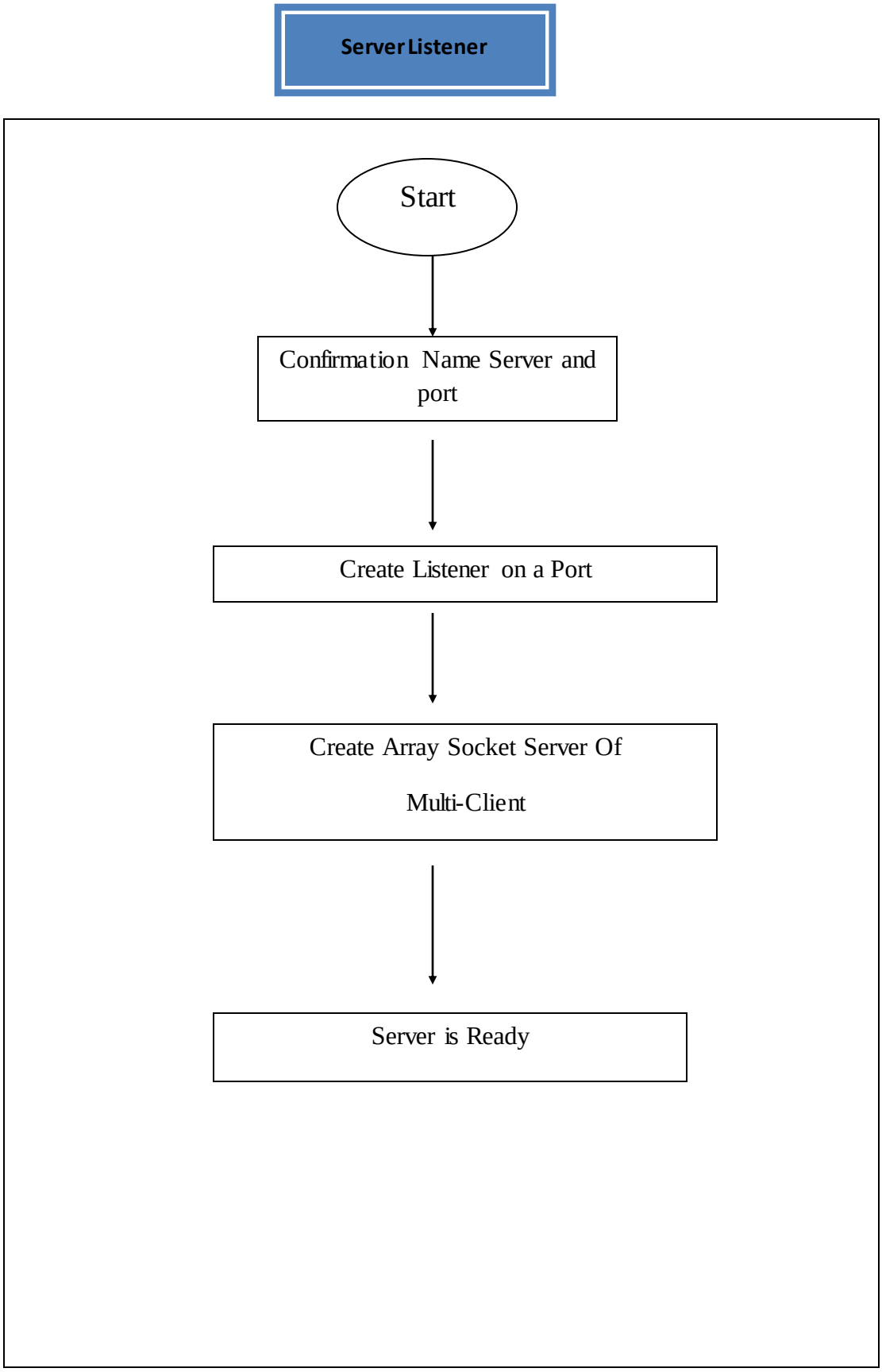
➔ Context Diagram

The following figure shows the free context of the work of the project.



Client Connect




Server Listener
Start
Confirmation Name Server and port
Create Listener on a Port
Create Array Socket Server Of Multi-Client
Server is Ready

9. CONCLUSION

Conclusion:

We Developed network applications in Java by ***using sockets, threads, and Web services.***

These software is portable, efficient, and easily maintainable for large number of clients. Our developed web-based chatting software is unique in its features and more importantly easily customizable. The java.net package provides a powerful and flexible set of classes for implementing network applications. Typically, programs running on client machines make requests to programs on a server Machine. These involve networking services provided by the transport layer. The most widely used transport protocols on the Internet are TCP (Transmission control Protocol) and UDP (User Datagram Protocol).

TCP is a connection-oriented protocol providing a reliable flow of data between two computers. On the other hand, UDP is a simpler message-based connectionless protocol which sends packets of data known as datagrams from one computer to another with no guarantees of arrival.

➢ Limitation

Though, the proposed system has many useful features, it has some limitations.

Also because all the current used software's provides a lot more features for the users but somewhere they also stores the users data which leads to data breaches which leads to us that no application is always perfect. It grows over time. However there are some limitations in our project too which we consider that will be solved in future updates.

- ✓ Some changes are required so it can be used by any user by Registering.
- ✓ If one of the user goes offline then the application creates error in background and the message cannot be delivered.
- ✓ Only and the last thing which will be definitely solved in the upcoming updates is that there is no end-to-end encryption available right now.

➢ Future Scope

There is always a room for improvements in any software package, however good and efficient it may be done. But the most important thing should be flexible to accept further modification. Right now we are just dealing with text communication. In future this software may be extended to include features such as:

- **Files transfer**: this will enable the user to send files of different formats to others via the chat· application.
- **Voice chat**: this will enhance the application to a higher level where communication will be· possible via voice calling as in telephone.
- **Video chat**: this will further enhance the feature of calling into video communication.
- **Inbuilt Emoji's**: this will further enhance the text feature and will make the chatting more exciting for the users.
- **Background**: this feature will further enhance the personalization feeling of users according to which user can change the background of chat.
- **Login/Register**: this feature will be provided in the future which will help all types of users to register themselves along with their authentication on the chat application before use which will make the application more dynamic in nature.
- **Database**: this feature of the application will help users to store their chat on a database.

BIBLOGRAPHY

- INTERNET, YOUTUBE
- BLACK BOOK JAVA, BY DREAMTECH
- AMRIT SAGAR, OUR JAVA TEACHER
- DR. HARMUNISH TANEJA, OUR PROJECT GUIDE

Other Used Links:

- ✓ www.javapoint.com
- ✓ www.w3school.com
- ✓ www.slideshare.net
- ✓ www.youtube.com