JAVA PROJECT REPORT

(Project Term January-May 2023)

APPLICATION FOR INSTANTANEOUS TEXT-BASED CONVERSATION USING CORE JAVA

Submitted by

Raja Jadon **Registration Number :12109953**

Course Code: CSE310

Under the Guidance of

Dr. Ranjith Kumar

School of Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY

DECLARATION

We hereby declare that the project work entitled "APPLICATION FOR INSTANTANEOUS TEXT-BASED CONVERSATION USING CORE JAVA" is an authentic record of our own work carried out as requirements of Capstone Project for the award of B.Tech degree in Computer Science Engineering from Lovely Professional University, Phagwara, under the guidance of Dr. Ranjith Kumar, during January to May 2023. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Name of Student : Raja Jadon
Registration Number: 12109953

Report Contents

APPLICATION FOR INSTANTANEOUS TEXT-BASED CONVERSATION USING CORE JAVA

Introduction

A chat application using JFrames is a graphical user interface (GUI) based program that allows users to communicate with each other in real-time. This type of application typically consists of two main components: a client-side application and a server-side application.

The client-side application is responsible for handling user interactions and sending messages to the server, which in turn distributes the messages to other connected clients. The server-side application is responsible for managing the connections between clients and facilitating the transmission of messages between them.

In a JFrames-based chat application, the user interface is built using Java Swing, a powerful GUI toolkit that provides a wide range of components for building complex UIs. JFrames, a class in Java Swing, provides a window-based container in which other Swing components can be added.

By utilizing JFrames, developers can create a highly interactive and user-friendly chat application that can run on any platform that supports Java. With the ability to customize the look and feel of the UI, developers can design a chat application that suits the needs of their users and enhances the overall user experience.

Profile of the Problem

Our application aims to facilitate instant text-based conversation using Core Java. We recognize that there is a growing need for real-time communication in various industries, such as customer service, online education, and remote collaboration. However, many existing chat applications suffer from slow loading times, poor user experience, and security issues.

Our solution addresses these problems by leveraging Core Java to provide a lightweight and secure chat application that delivers fast, reliable, and user-friendly communication. Our application supports features such as group chats, private messaging, etc

Overall, my project involves a comprehensive analysis of the technical, functional, and user experience requirements of the application, as well as testing and validation to ensure that it meets these requirements. I'm excited about the potential of this application and believe that it could have a significant impact on real-time communication in various industries

Existing System

We provide an overview of the existing text-based communication applications and their limitations. Our aim is to identify the gaps in the existing systems and explain how our application will address these limitations to provide a better user experience and improved functionality.

There are several popular text-based communication applications such as WhatsApp, Messenger, Slack, and Skype, which are widely used for personal and professional communication. These applications are easy to use, accessible, and offer various features for messaging, file sharing, and voice and video calls.

However, these existing systems have several limitations that can impact the user experience and functionality. For instance, there are concerns related to security and privacy, such as data breaches and unauthorized access. Customization options are limited in these applications, and there is a lack of features for specific use cases. Technical limitations such as limited scalability and slow performance can also be a challenge, especially when the volume of messaging is high. In addition, the user experience of these applications can be inconsistent, and there are industry-specific limitations that can impact their usefulness in various sectors, such as customer service, education, and remote collaboration.

Despite these limitations, the existing text-based communication applications have set a high standard for instant messaging, and they remain widely used for personal and professional communication. However, our application aims to

address the limitations of the existing systems by providing a lightweight, reliable, and secure platform that meets the needs of specific industries and use cases.

We aim to offer a more comprehensive and customizable platform that allows users to tailor their messaging experience to their specific needs. Additionally, our application will prioritize security and privacy, and we will incorporate advanced measures to ensure that user data is secure.

Furthermore, our application will focus on providing a seamless user experience with a user-friendly interface that is easy to navigate. Our goal is to create an application that provides a better overall experience than the existing systems, ultimately helping our users to communicate more effectively and efficiently.

Overall, by identifying the limitations of the existing systems, we can highlight the need for our application and demonstrate how it will provide a better solution to the limitations of the existing systems.

Problem Analysis

In this section, we will analyze the problems and challenges that our project, aims to solve. We will provide a detailed analysis of the current communication landscape and identify the specific problems or gaps in the existing systems that our application will address.

The problem that our application aims to solve is the limitations of the existing text-based communication applications, such as WhatsApp, Messenger, Slack, and Skype. These applications have several limitations that can impact the user experience and functionality, including concerns related to security and privacy, customization options, and technical limitations such as slow performance and limited scalability.

Moreover, the existing systems lack features for specific use cases, such as customer service, education, and remote collaboration. In these industries, communication is crucial, and existing systems may not fully meet the unique needs and requirements of these sectors.

Our application aims to address these limitations and provide a better solution for users who require a more comprehensive and customizable platform. Specifically, we will prioritize security and privacy by incorporating advanced measures to ensure that user data is secure. We will offer a more seamless user experience with a user-friendly interface that is easy to navigate, allowing users to tailor their messaging experience to their specific needs.

Furthermore, our application will focus on providing a platform that can handle high volumes of messaging with fast performance, ensuring that users can communicate quickly and efficiently. We will incorporate features that are tailored to specific industries, such as customer service chatbots and virtual classroom tools for education.

Overall, our application aims to provide a better solution to the limitations of the existing text-based communication applications by offering a more comprehensive, customizable, and industry-specific platform. By analyzing the problems and challenges of the current communication landscape, we have identified the need for our application and the specific areas where it will provide value for our users.

Software Requirement Analysis

In this section, we will outline the software requirements for our project, this analysis will help us to define the functionality, features, and performance requirements for the application.

Functional Requirements:

- The application should provide users with the ability to send and receive instant messages in real-time.

- The application should allow users to create groups for messaging and provide the ability to manage group members.

- The application should offer a search functionality that allows users to search for messages, groups, and contacts.

- The application should allow users to customize their messaging experience with features such as emoji, gifs, and message reactions.

- The application should support file sharing, including the ability to send and receive documents, images, and videos.

- The application should include industry-specific features, such as chatbots for customer service and virtual classroom tools for education.

Non-Functional Requirements:

- The application should prioritize security and privacy, including measures such as encryption and two-factor authentication.

- The application should have a user-friendly interface that is easy to navigate and understand.

- The application should be lightweight, fast, and scalable, supporting a high volume of messaging without performance issues.

- The application should support multiple platforms, including desktop and mobile devices.

Performance Requirements:

- The application should be able to handle a large number of users and messages without significant delays or performance issues.

- The application should support fast message delivery with low latency.

- The application should be reliable, with minimal downtime or service interruptions.

- The application should support high availability, ensuring that users can access the service at any time.

Overall, the software requirements for our application reflect the need to provide users with a secure, reliable, and customizable messaging platform that can meet the needs of specific industries and use cases. By outlining these requirements, we can ensure that our application meets the expectations and needs of our users and provides a better solution to the limitations of existing text-based communication applications.

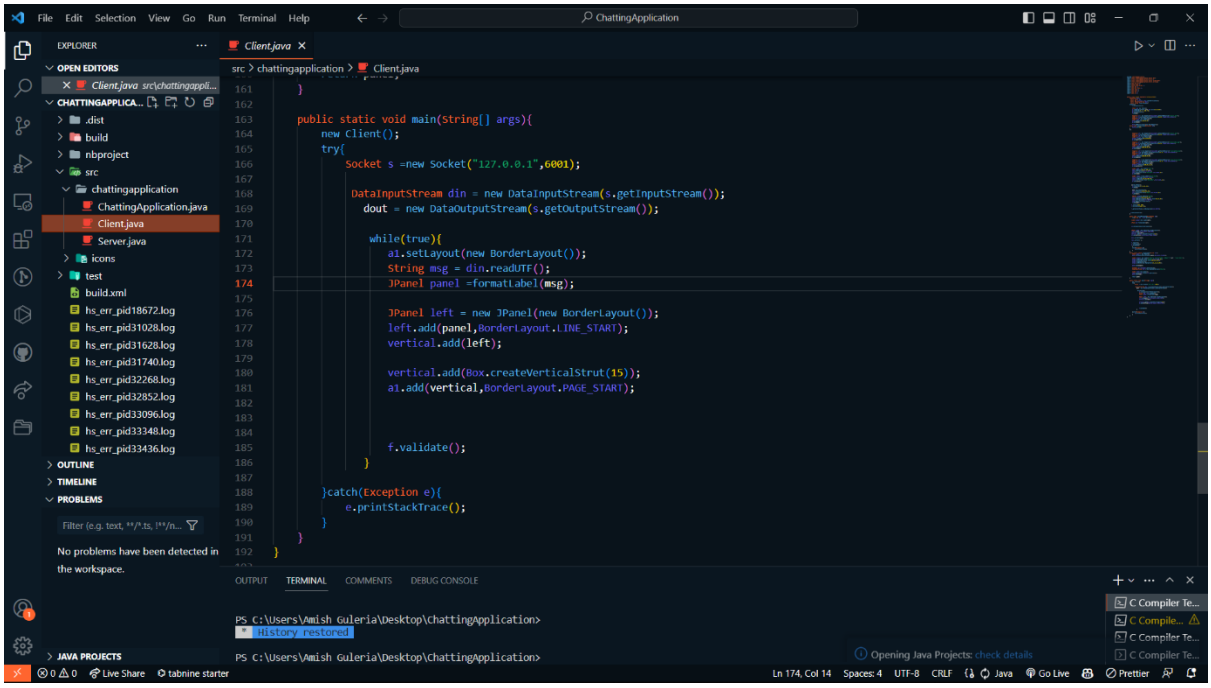
Implementing the Client Side

The client side of a chat application built using JFrames typically involves the graphical user interface (GUI) that users interact with. The GUI is created using the JFrame class, which provides the necessary components for displaying windows, buttons, text fields, and other UI elements. The client side also includes the logic for handling user interactions, such as sending and receiving messages, displaying notifications, and managing the chat history. This logic is typically implemented using event listeners and callbacks, which respond to user actions and update the UI accordingly. Overall, the client side is responsible for providing a visually appealing and user-friendly interface for communicating with other users in the chat application.

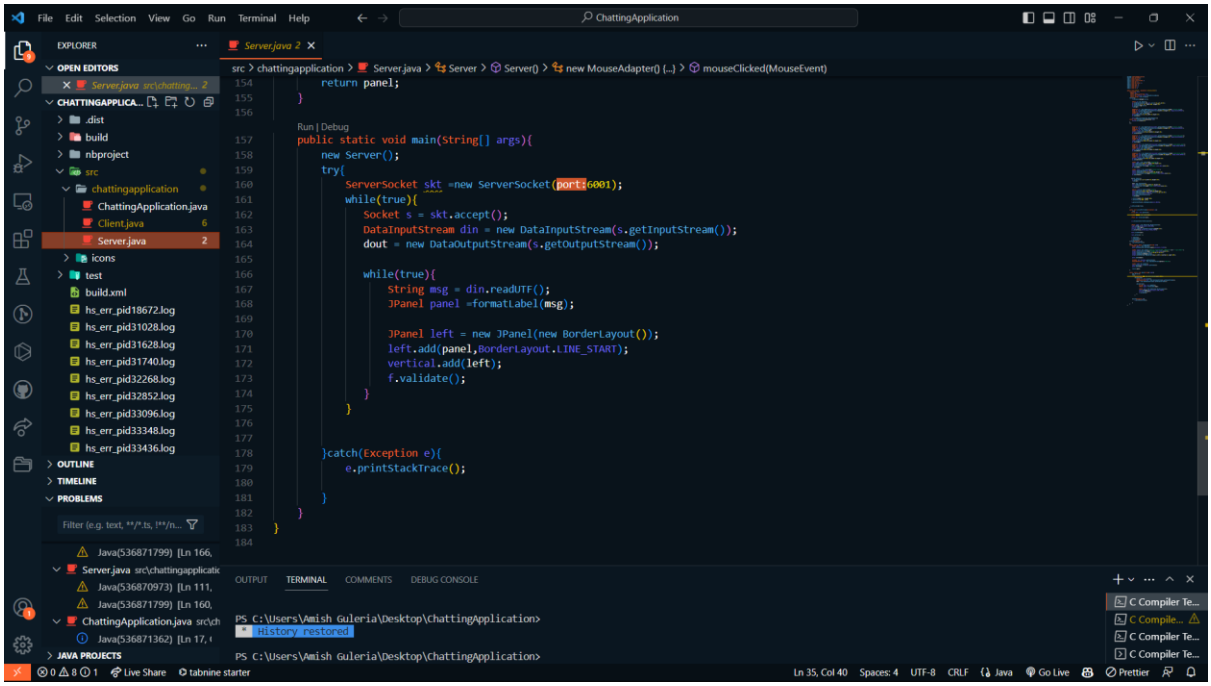
Implementing the Server Side

The server side of this chat application involves handling communication between clients, storing and managing user data, and ensuring the security and reliability of the system. It typically uses socket programming to establish and maintain the connection between clients and the server. The server also handles tasks such as sending and receiving messages, managing user accounts, and monitoring the system for any issues or errors. Additionally, it may use various protocols such as TCP/IP or HTTP to facilitate communication and data transfer between clients and the server.

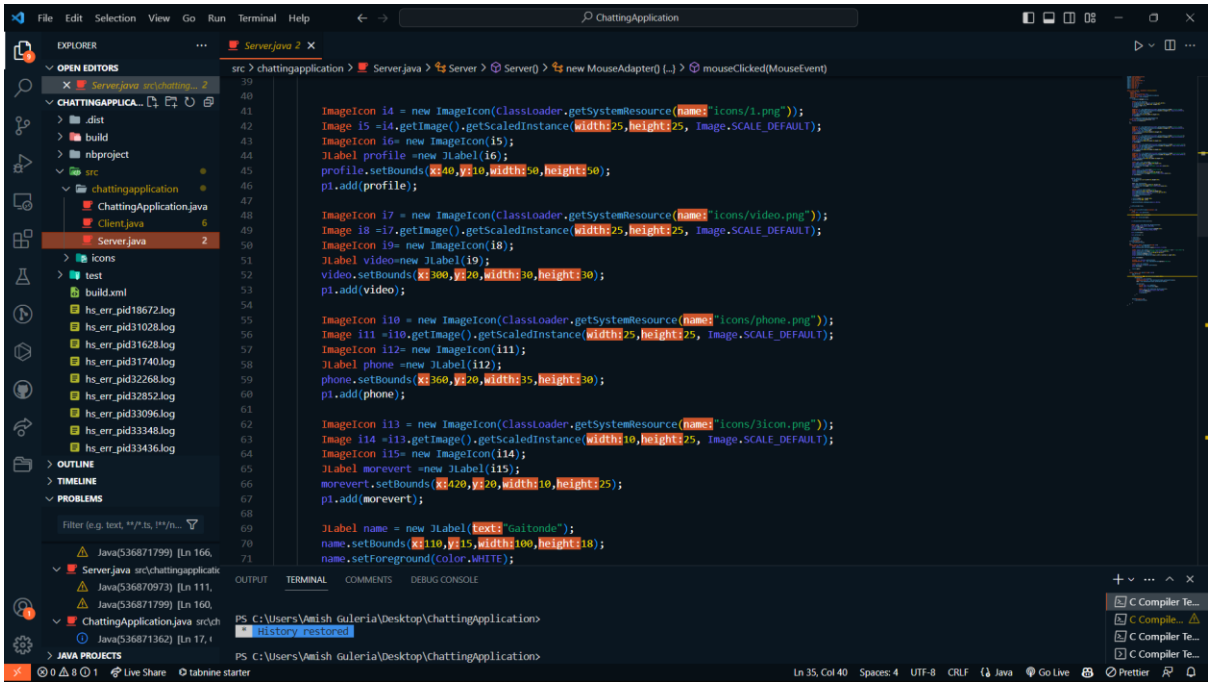
Code Snippets



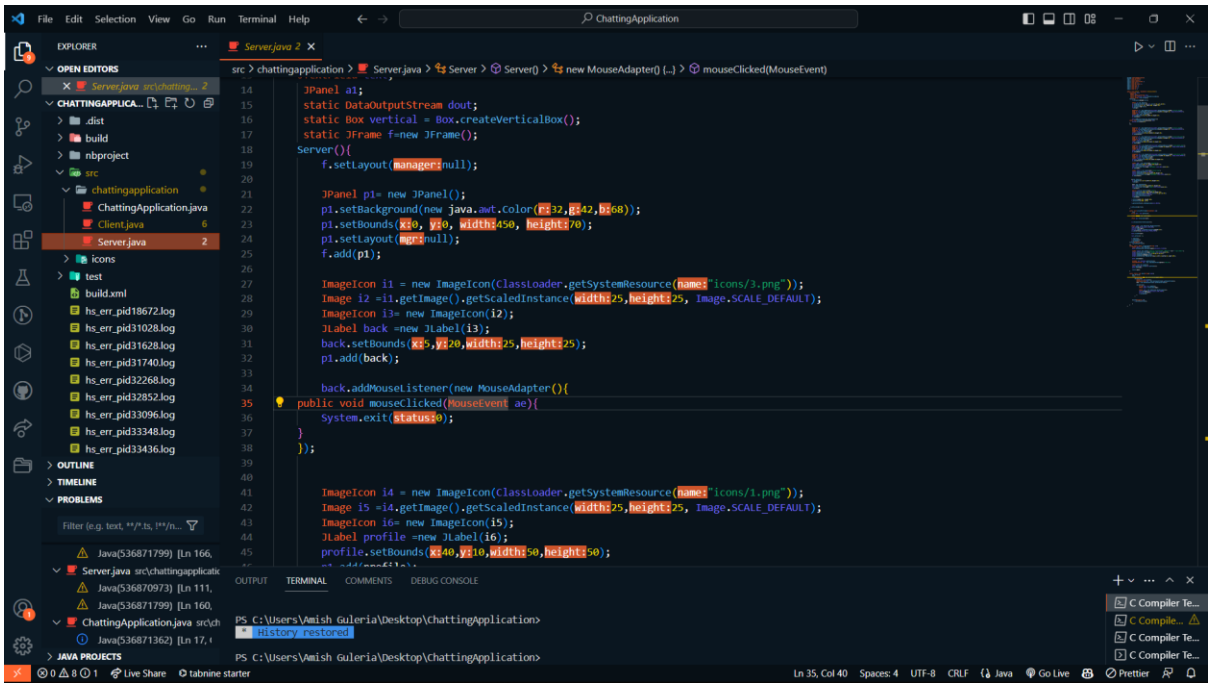
```
}

public static void main(String[] args){
    new Client();
    try{
        Socket s =new Socket("127.0.0.1",6001);

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

          while(true){
            a1.setLayout(new BorderLayout());
            String msg = din.readUTF();
            JPanel panel =formatLabel(msg);

            JPanel left = new JPanel(new BorderLayout());
            left.add(panel,BorderLayout.LINE_START);
            vertical.add(left);

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



            f.validate();
          }

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



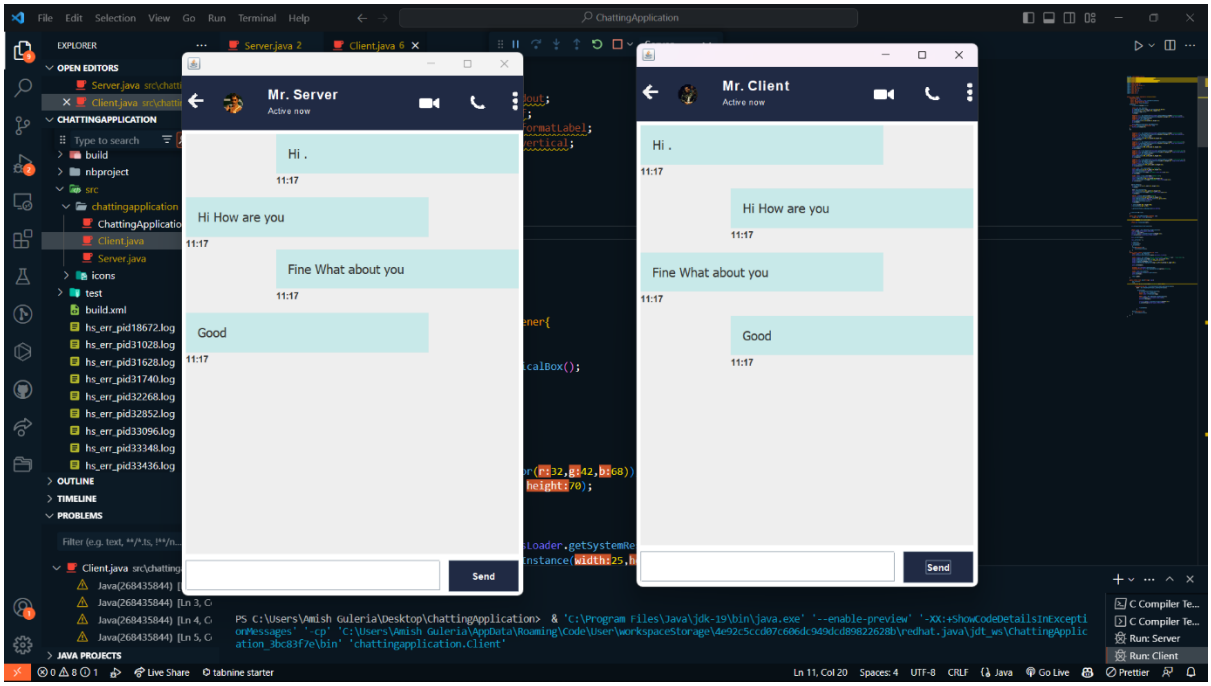
```
        return panel;
    }

    Run | Debug
    public static void main(String[] args){
        new Server();
        try{
            ServerSocket skt =new ServerSocket(port:6001);
            while(true){
              Socket s = skt.accept();
              DataInputStream din = new DataInputStream(s.getInputStream());
              dout = new DataOutputStream(s.getOutputStream());

              while(true){
                  String msg = din.readUTF();
                  JPanel panel =formatLabel(msg);

                  JPanel left = new JPanel(new BorderLayout());
                  left.add(panel,BorderLayout.LINE_START);
                  vertical.add(left);
                  f.validate();
              }
            }

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



```
        ImageIcon i4 = new ImageIcon(ClassLoader.getSystemResource("icons/1.png"));
        Image i5 =i4.getImage().getScaledInstance(25,25, Image.SCALE_DEFAULT);
        ImageIcon i6= new ImageIcon(i5);
        JLabel profile =new JLabel(i6);
        profile.setBounds(40,10,50,50);
        p1.add(profile);

        ImageIcon i7 = new ImageIcon(ClassLoader.getSystemResource("icons/video.png"));
        Image i8 =i7.getImage().getScaledInstance(25,25, Image.SCALE_DEFAULT);
        ImageIcon i9= new ImageIcon(i8);
        JLabel video=new JLabel(i9);
        video.setBounds(300,20,30,30);
        p1.add(video);

        ImageIcon i10 = new ImageIcon(ClassLoader.getSystemResource("icons/phone.png"));
        Image i11 =i10.getImage().getScaledInstance(25,25, Image.SCALE_DEFAULT);
        ImageIcon i12= new ImageIcon(i11);
        JLabel phone =new JLabel(i12);
        phone.setBounds(360,20,35,30);
        p1.add(phone);

        ImageIcon i13 = new ImageIcon(ClassLoader.getSystemResource("icons/3icon.png"));
        Image i14 =i13.getImage().getScaledInstance(10,25, Image.SCALE_DEFAULT);
        ImageIcon i15= new ImageIcon(i14);
        JLabel morevert =new JLabel(i15);
        morevert.setBounds(420,20,10,25);
        p1.add(morevert);

        JLabel name = new JLabel("Gaitonde");
        name.setBounds(110,15,100,18);
        name.setForeground(Color.WHITE);
```



```
    JPanel a1;
    static DataOutputStream dout;
    static Box vertical = Box.createVerticalBox();
    static JFrame f=new JFrame();
    Server(){
        f.setLayout(manager:null);

        JPanel p1= new JPanel();
        p1.setBackground(new java.awt.Color(r:32,g:42,b:68));
        p1.setBounds(x:0, y:0, width:450, height:70);
        p1.setLayout(mgr:null);
        f.add(p1);

        ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/3.png"));
        Image i2 =i1.getImage().getScaledInstance(width:25,height:25, Image.SCALE_DEFAULT);
        ImageIcon i3= new ImageIcon(i2);
        JLabel back =new JLabel(i3);
        back.setBounds(x:5,y:20,width:25,height:25);
        p1.add(back);

        back.addMouseListener(new MouseAdapter(){
    public void mouseClicked(MouseEvent ae){
        System.exit(status:0);
    }
    });


        ImageIcon i4 = new ImageIcon(ClassLoader.getSystemResource("icons/1.png"));
        Image i5 =i4.getImage().getScaledInstance(width:25,height:25, Image.SCALE_DEFAULT);
        ImageIcon i6= new ImageIcon(i5);
        JLabel profile =new JLabel(i6);
        profile.setBounds(x:40,y:10,width:50,height:50);
```


Mr. Server
Active now
Hi .
11:17
Hi How are you
11:17
Fine What about you
11:17
Good
11:17
Send
Mr. Client
Active now
Hi .
11:17
Hi How are you
11:17
Fine What about you
11:17
Good
11:17
Send

Conclusion

In conclusion, chat applications have revolutionized the way we communicate with each other in the digital age. They allow for instant messaging, which enables us to have real-time conversations with people across the world, regardless of distance or time zone. Chat applications have become an essential part of our daily lives, with millions of people using them every day for personal and business purposes. With the advent of new technologies, chat applications will continue to evolve and offer new and innovative ways for people to connect and communicate with each other.