Connecting Worlds: Revolutionising Intranet Communication with Real-Time Java Chat Applet

A PROJECT REPORT

CSA0937-PROGRAMMING IN JAVA AND ANALYSIS

Submitted by

P.Sairishika

(192110339)

In partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING IN

COMPUTER SCIENCE

SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA NAGAR, THANDALAM,

SIMATS, CHENNAI – 602105.

MARCH 2024

BONAFIDE CERTIFICATE

This is to certify that the project report entitled "Attendance Management System" submitted by "N.Vamsi Vardhan Reddy (192110209)", to Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, is a record of bonafide work carried out by him/her under my guidance. The project fulfills the requirements as per the regulations of this institution and in my appraisal meets the required standards for submission.

Dr.M.Buvanesvari

Professor
Department of Knowledge
Engineering,
Saveetha School of Engineering
SIMATS, Chennai – 602 105

Internal examiner **External Examiner**

TABLE OF CONTENTS

ABSTRACT

In the ever-evolving landscape of corporate communication, the Intranet serves as a vital hub for facilitating collaboration and disseminating information within organizations. However, traditional Intranet platforms often struggle to meet the demands of modern workplaces characterized by dispersed teams and the need for instant communication.

This abstract presents a pioneering solution: the Real-Time Java Chat Applet, designed to revolutionize Intranet communication by providing a seamless and dynamic platform for interaction. Leveraging the power of Java technology, this chat applet transcends the limitations of conventional Intranet messaging systems, offering real-time communication capabilities that enhance productivity and foster a sense of interconnectedness among employees.

Key features of the Real-Time Java Chat Applet include its lightweight design, compatibility with diverse operating systems, and robust security protocols. Moreover, its intuitive user interface ensures ease of use for employees across various departments and skill levels.

By deploying the Real-Time Java Chat Applet within their Intranet infrastructure, organizations can experience a paradigm shift in internal communication, empowering teams to collaborate efficiently, exchange ideas instantaneously, and bridge geographical boundaries. This abstract outlines the transformative potential of this innovative solution in shaping the future of corporate communication within the digital workplace.

INTRODUCTION

The aim of the Intranet Communication with Real-Time Java Chat Applet is to allow the communication between organization private work

- Aim: Enhance internal communication and collaboration within the organization's intranet environment
- Develop Java applet for Intranet Chatting system
- Real-time communication within organization's intranet network
- User-friendly interface for sending and receiving messages
- Features: Private messaging, group chat, user status indicators
- Secure login with user credentials
- View list of online users

DESCRIPTION

The background of an Attendance Management System project involves an understanding of the existing challenges and inefficiencies in manual attendance tracking processes, prompting the need for an automated solution. Common elements in the background of an AMS project include.

Traditional attendance tracking relies on manual methods, such as paper-based sign-in sheets, which are time-consuming, error-prone, and susceptible to inaccuracies.Manual attendance systems are vulnerable to errors, including unintentional mistakes in data entry and fraudulent practices like proxy attendance, leading to inaccurate records.

Managing attendance manually consumes significant administrative resources, particularly in large organizations, resulting in increased operational costs and potential delays in decision-making.Traditional methods lack the capability for real-time monitoring, making it challenging for administrators to promptly address attendance-related issues or make informed decisions based on current data. With sensitive attendance data being handled manually, concerns about data privacy and security become pertinent, necessitating a more secure and controlled approach.The advancements in biometrics, and cloud computing technologies offer opportunities to revolutionize attendance tracking, providing more accurate, efficient, and secure alternatives.As organizations expand, manual attendance tracking becomes increasingly impractical. An automated system is needed to handle the growing workforce and diverse attendance management requirements.Events such as global health crises may highlight the need for contactless attendance systems, promoting the adoption of technology-driven solutions to ensure a safe and hygienic work environment.

SOFTWARE REQUIREMENTS

- **Server**
- **4 GB RAM or more**
- **Depending on the size of the database, 50 GB or more**
- **Client (for web-based systems):**
- **Modern web browser with JavaScript enabled**
- **Software Requirements:**
- **Linux (e.g., Ubuntu, CentOS) or Windows Server**

- **Windows, macOS, Linux**
- **Database Management System:**
- **MySQL, PostgreSQL, MongoDB, or other databases based on system architecture**
- **Web Server**
- **Apache, Nginx**
- **Depends on the technology stack (e.g., PHP, Python, Java, .NET)**
- **Frameworks (if applicable):**
- **Laravel, Django, Spring, .NET Core, etc.**
- **Internet connection (for cloud-based systems or remote access)**
- **Local area network (LAN) for on-premises systems**
- **Firewalls and security measures to protect data**
- **User authentication and authorization**
- **Regular security updates and patches**

EXISTING

- **A comprehensive school management system that includes attendance tracking for students.**
- **Provides features for managing student information, grades, and communication.**
- **A cloud-based student information system with attendance tracking capabilities.**
- **Offers features for grading, communication, and collaboration.**
- **A mobile and web-based solution for schools and colleges.**
- **Includes attendance management, timetable creation, and communication tools.**
- **An open-source Learning Management System (LMS) that may include attendance tracking features.**
- **Designed for educational content delivery and collaboration.**
- **A web-based platform for school management.**
- **Offers attendance tracking, grade management, and communication tools.**

- **A cloud-based school management system with attendance tracking capabilities**
- **A school management system with attendance tracking, timetable creation, and academic management features.**
- **Offers a cloud-based solution for educational institutions.**

PROPOSED :

- **Attendance is carried out in handwritten registers.**
- **It requires more human effort and is prone to errors such as misplaced sheets and illegible handwriting.**
- **It can be a tedious job to maintain the record.**
- **The system requires more human effort.**
- **It streamlines the recording of attendance by automating the process[3].**
- **It eradicates the need for manual paperwork.**
- **It offers real-time insights into student attendance trends[3].**
- **Administrators, teachers, and parents have digital access to attendance reports and student attendance records.**
- **It reduces the time and effort required for attendance-taking while ensuring accuracy.**
- **The proposed system covers the problem of using the manual system such as an unsecured record of attendance, lots of paper works, and inaccuracy of the data inputs.**
- **This proposed system will benefit the school for its development and improvement by using a new technology and improved system for the day-to-day process of attendance.**

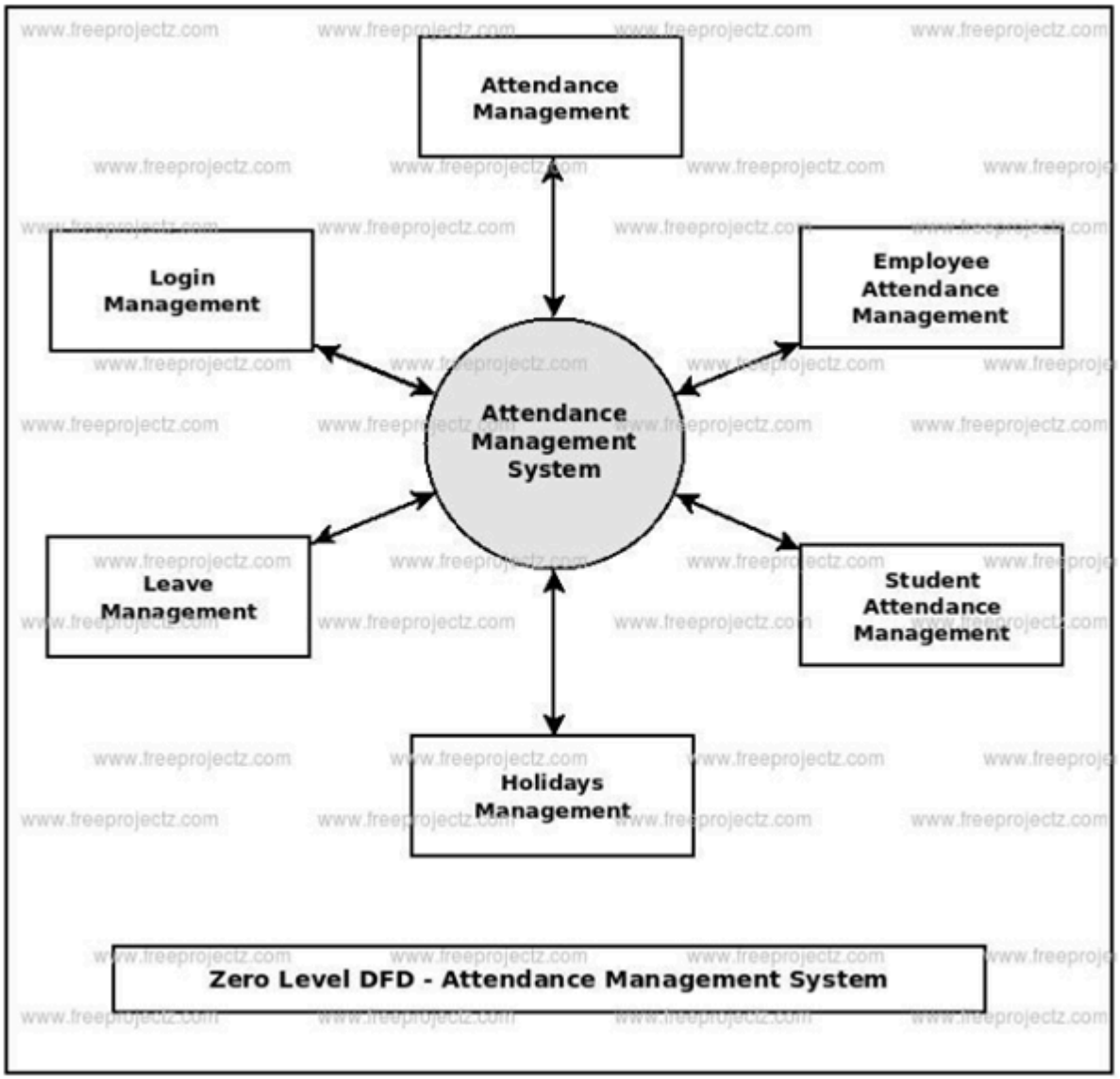
TECHNOLOGY USED

An attendance management system implemented solely in Java utilizes a robust technology stack for seamless tracking and organization of attendance data. The backend logic can be crafted with Java, employing frameworks like Spring Boot for efficient development and management of business logic.

For data storage and retrieval, a Java application might integrate with relational databases such as MySQL or PostgreSQL, or even leverage non-relational databases like MongoDB, depending on specific requirements. The user interface, developed using JavaServer Faces (JSF) or other Java-based UI frameworks, ensures a responsive and user-friendly experience. Security features are implemented using Java's built-in capabilities and additional libraries if needed.

The system can leverage Java's multithreading capabilities for concurrent processing, enhancing performance. Integration with other systems or services can be achieved through Java APIs or web services. Mobile applications may be developed using Java for Android platforms. Overall, a Java-centric approach ensures a robust and scalable architecture for an attendance management system.

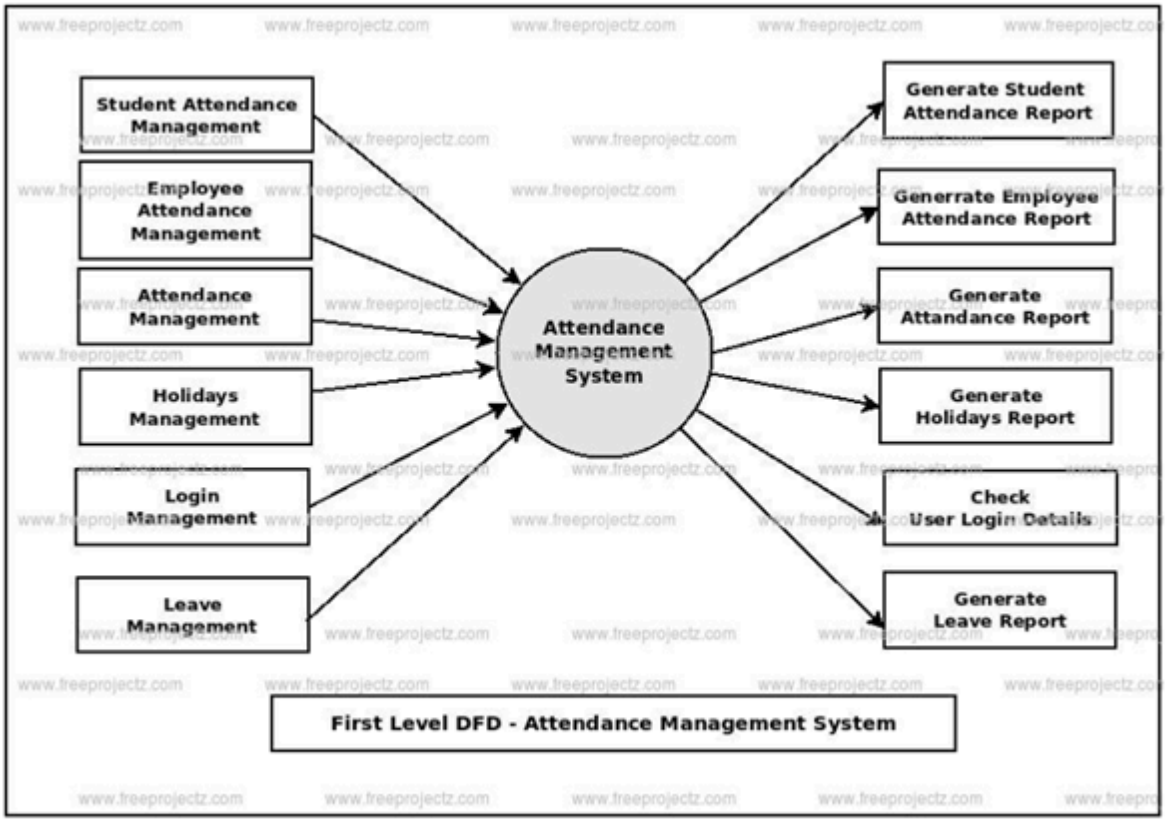
USE CASE DIAGRAMS



Attendance Management System shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an extemal agent, and which together provide all of the functionality of the Attendance Management System system as a whole. It also identifies internal data stores of Authentication, Login, Leave, Holidays, Student Attendance that must be present in order for the Attendance system to do its job, and shows the flow of data between the various parts of Attendance, Student Attendance, Login, Authentication, Leave of the

system. DFD Level 1 provides a more detailed breakout of pieces of the 1st level DFD. You will highlight the main functionalities of Attendance.

DATA FLOW DIAGRAM OF STUDENT



Admin logins to the system and manage all the functionalities of Attendance Management System

Admin can add, edit, delete and view the records of Attendance, Student Attendance, Leave.

Admin can manage all the details of Employee Attendance, Holidays, Login

Admin can also generate reports of Attendance, Employee Attendance, Student Attendance, Holidays, Leave, Login

Admin can search the details of Employee Attendance, Leave, Login

Admin can apply different level of filters on report of Attendance, Holidays, Leave

Admin can tracks the detailed information of Employee Attendance, Student Attendance, Holidays.

SOURCE CODE:

```
   import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.*;
import java.io.*;

public class IntranetChatApp extends JFrame implements ActionListener {

   private JTextField usernameField, messageField;
   private JTextArea chatArea;
   private JPasswordField passwordField;
   private JButton loginButton, signUpButton, submitButton;
    private JPanel loginPanel, signUpPanel, messagePanel, statusPanel, groupChatPanel,
onlineUsersPanel;
   private CardLayout cardLayout;
   private String username;

   public IntranetChatApp() {
      setTitle("Intranet Chat");
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      cardLayout = new CardLayout();
      setLayout(cardLayout);
      initLoginPanel();
      initSignUpPanel();
      initMessagePanel();
      initStatusPanel();
      initGroupChatPanel();
      initOnlineUsersPanel();
      add(loginPanel, "login");
      add(signUpPanel, "signup");
      add(messagePanel, "message");
      add(statusPanel, "status");
      add(groupChatPanel, "groupchat");
      add(onlineUsersPanel, "onlineusers");
      setSize(400, 300);
      setLocationRelativeTo(null);
      setVisible(true);
   }

   private void initLoginPanel() {
```

```
    loginPanel = new JPanel(new GridLayout(4, 1));
    JPanel usernamePanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    usernamePanel.add(new JLabel("Username:"));
    usernameField = new JTextField(15);
    usernamePanel.add(usernameField);
    JPanel passwordPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    passwordPanel.add(new JLabel("Password:"));
    passwordField = new JPasswordField(15);
    passwordPanel.add(passwordField);
    JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    loginButton = new JButton("Login");
    loginButton.addActionListener(this);
    signUpButton = new JButton("Sign Up");
    signUpButton.addActionListener(this);
    buttonPanel.add(loginButton);
    buttonPanel.add(signUpButton);
    loginPanel.add(usernamePanel);
    loginPanel.add(passwordPanel);
    loginPanel.add(buttonPanel);
}

private void initSignUpPanel() {
    signUpPanel = new JPanel(new GridLayout(4, 1));
    JPanel usernamePanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    usernamePanel.add(new JLabel("Username:"));
    usernameField = new JTextField(15);
    usernamePanel.add(usernameField);
    JPanel passwordPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    passwordPanel.add(new JLabel("Password:"));
    passwordField = new JPasswordField(15);
    passwordPanel.add(passwordField);
    JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
    submitButton = new JButton("Submit");
    submitButton.addActionListener(this);
    buttonPanel.add(submitButton);
    signUpPanel.add(usernamePanel);
    signUpPanel.add(passwordPanel);
    signUpPanel.add(buttonPanel);
}

private void initMessagePanel() {
    messagePanel = new JPanel(new BorderLayout());

    // Create a panel for buttons on the left
```

```
        JPanel buttonPanel = new JPanel(new GridLayout(3, 1));

        // Add status button
        JButton statusButton = new JButton("Status");
        statusButton.addActionListener(this);
        buttonPanel.add(statusButton);

        // Add group chat button
        JButton groupChatButton = new JButton("Group Chat");
        groupChatButton.addActionListener(this);
        buttonPanel.add(groupChatButton);

        // Add online users button
        JButton onlineUsersButton = new JButton("Online Users");
        onlineUsersButton.addActionListener(this);
        buttonPanel.add(onlineUsersButton);

         messagePanel.add(buttonPanel, BorderLayout.WEST); // Add the button panel on
the left

        // Create a panel for the chat area and input field
        JPanel chatInputPanel = new JPanel(new BorderLayout());

        // Add the chat area
        chatArea = new JTextArea();
        chatArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(chatArea);
        chatInputPanel.add(scrollPane, BorderLayout.CENTER);

        // Add the input field
        messageField = new JTextField();
        JButton sendButton = new JButton("Send");
        sendButton.addActionListener(this);
        JPanel inputButtonPanel = new JPanel(new BorderLayout());
        inputButtonPanel.add(messageField, BorderLayout.CENTER);
        inputButtonPanel.add(sendButton, BorderLayout.EAST);
        chatInputPanel.add(inputButtonPanel, BorderLayout.SOUTH);

         messagePanel.add(chatInputPanel, BorderLayout.CENTER); // Add the chat area
and input field panel
    }

    private void initStatusPanel() {
        statusPanel = new JPanel(new BorderLayout());
```

```
        String[] columnNames = {"Username", "Status"};
            Object[][] data = {{"User 1", "Available"}, {"User 2", "Busy"}, {"User 3",
"Away"}};
        DefaultTableModel model = new DefaultTableModel(data, columnNames);
        JTable statusTable = new JTable(model);
        statusPanel.add(new JScrollPane(statusTable), BorderLayout.CENTER);
        JButton backButton = new JButton("Back");
        backButton.addActionListener(this);
        statusPanel.add(backButton, BorderLayout.SOUTH);
    }

    private void initGroupChatPanel() {
        groupChatPanel = new JPanel(new BorderLayout());

        // Create a panel for buttons on the left
        JPanel buttonPanel = new JPanel(new GridLayout(3, 1));

        // Add user list
        DefaultListModel<String> userListModel = new DefaultListModel<>();
        userListModel.addElement("User 1");
        userListModel.addElement("User 2");
        userListModel.addElement("User 3");
        JList<String> userList = new JList<>(userListModel);
        groupChatPanel.add(new JScrollPane(userList), BorderLayout.CENTER);

        // Add create group chat button
        JButton createGroupChatButton = new JButton("Create Group Chat");
        createGroupChatButton.addActionListener(this);
        buttonPanel.add(createGroupChatButton);

        // Add back button
        JButton backButton = new JButton("Back");
        backButton.addActionListener(this);
        buttonPanel.add(backButton);

         groupChatPanel.add(buttonPanel, BorderLayout.WEST); // Add the button panel
on the left
    }

    private void initOnlineUsersPanel() {
        onlineUsersPanel = new JPanel(new BorderLayout());

        // Create a panel for buttons on the left
        JPanel buttonPanel = new JPanel(new GridLayout(2, 1));
```

```
        // Add online user list
        DefaultListModel<String> onlineUserListModel = new DefaultListModel<>();
        onlineUserListModel.addElement("User 1");
        onlineUserListModel.addElement("User 2");
        onlineUserListModel.addElement("User 3");
        JList<String> onlineUserList = new JList<>(onlineUserListModel);
        onlineUsersPanel.add(new JScrollPane(onlineUserList), BorderLayout.CENTER);

        // Add back button
        JButton backButton = new JButton("Back");
        backButton.addActionListener(this);
        buttonPanel.add(backButton);

         onlineUsersPanel.add(buttonPanel, BorderLayout.WEST); // Add the button panel
on the left
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == loginButton) {
            String username = usernameField.getText();
            char[] password = passwordField.getPassword();
            if (checkCredentials(username, password)) {
                cardLayout.show(getContentPane(), "message");
                this.username = username;
            } else {
                    JOptionPane.showMessageDialog(this, "Credentials not found. Sign up to
login.");
            }
        } else if (e.getSource() == signUpButton) {
            cardLayout.show(getContentPane(), "signup");
        } else if (e.getSource() == submitButton) {
            String username = usernameField.getText();
            char[] password = passwordField.getPassword();
            if (username.isEmpty() || password.length == 0) {
                JOptionPane.showMessageDialog(this, "Username and password cannot be
empty.");
            } else {
                if (saveCredentials(username, password)) {
                        JOptionPane.showMessageDialog(this, "Account created successfully.
Please login.");
                    cardLayout.show(getContentPane(), "login");
                } else {
```

```
                JOptionPane.showMessageDialog(this, "Failed to create account. Please try
again.");
            }
        }
    } else if (e.getActionCommand().equals("Send")) {
        sendMessage(messageField.getText());
    } else if (e.getActionCommand().equals("Status")) {
        cardLayout.show(getContentPane(), "status");
    } else if (e.getActionCommand().equals("Group Chat")) {
        cardLayout.show(getContentPane(), "groupchat");
    } else if (e.getActionCommand().equals("Online Users")) {
        cardLayout.show(getContentPane(), "onlineusers");
    } else if (e.getActionCommand().equals("Back")) {
        cardLayout.show(getContentPane(), "message");
    }
  }

  private boolean checkCredentials(String username, char[] password) {
                        try (BufferedReader reader = new BufferedReader(new
FileReader("credentials.txt"))) {
        String line;
        while ((line = reader.readLine()) != null) {
            String[] parts = line.split(",");
                            if (parts.length == 2 && parts[0].equals(username) &&
parts[1].equals(String.valueOf(password))) {
                return true;
            }
        }
    } catch (IOException ex) {
        ex.printStackTrace();
    }
    return false;
  }

  private boolean saveCredentials(String username, char[] password) {
                        try (BufferedWriter writer = new BufferedWriter(new
FileWriter("credentials.txt", true))) {
        writer.write(username + "," + String.valueOf(password) + "\n");
        return true;
    } catch (IOException ex) {
        ex.printStackTrace();
        return false;
    }
  }
```

```
    private void sendMessage(String message) {
        chatArea.append(username + ": " + message + "\n");
        messageField.setText("");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(IntranetChatApp::new);
    }
}
```

OUTPUT:

CONCLUSION

The implementation of the proposed Attendance Management System marks a significant leap forward from the limitations of manual attendance tracking. By automating the process, integrating advanced technologies such as biometrics and RFID, and leveraging cloud computing for real-time access, the system addresses the inefficiencies, inaccuracies, and security concerns associated with traditional methods. The enhanced user interfaces, analytics tools, and notification systems contribute to a more streamlined, efficient, and secure attendance management process.

The shift towards automation not only improves the accuracy of attendance records but also provides a foundation for data-driven decision-making. The system's scalability ensures adaptability to organizational growth, making it a sustainable solution for evolving workforce needs. The integration with existing systems facilitates seamless data flow across different organizational departments, promoting consistency and reducing manual efforts.

FUTURE ENHANCEMENT

Explore the integration of Internet of Things (IoT) devices for more comprehensive and context-aware attendance tracking, allowing for additional data points and insights. Implement machine learning algorithms to analysis historical attendance data and predict future attendance trends, enabling proactive management strategies.

Further develop and optimize the mobile application, providing additional features such as geolocation-based attendance marking for field-based employees.

Investigate the use of blockchain technology to enhance the security of attendance records, ensuring tamper-proof and transparent data management. Integrate the Attendance Management System with smart office technologies for a cohesive workplace experience, including automated room access based on attendance.

REFERENCES

extension://fheoggkfdfchfphceeifdbepaooicaho/html/site_status_block_page.html

- **https://code-projects.org/attendance-management-system-in-java-with-source-code/**
- **https://sourcecodehero.com/attendance-management-system-project-in-java-with-source-code/**
- **https://itsourcecode.com/free-projects/java-projects/attendance-management-system-project-in-java-with-source-code/**
- **https://myprojectideas.com/online-attendance-management-system-java-project/**