SCHOOL OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

B.Tech in COMPUTER SCIENCE AND ENGINEERING (Specialization in Artificial Intelligence and Machine Learning)

CS433P Programming Paradigms
CIA -3 Component 2

MINI-PROJECT REPORT

STUDENT MANAGEMENT SYSTEM

By

ANGELA RODRIGUES(2262027)

ATHUL NAMBIAR(2262041)

BENISON BINOY(2262044)

Subject In-Charge

Dr. ARUNA S K

Department of Computer Science and Engineering

School of Engineering and Technology,

CHRIST (Deemed to Be University),

Kumbalagodu, Bangalore - 560 074.

March 2024

Declaration

We, hereby declare that the Project titled "STUDENT MANAGEMENT SYSTEM" is a record of original project work undertaken by us for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/ Information Technology**. We have completed this study for the course CS433P Programming Paradigms.

We also declare that this project report has not been submitted for the award of any degree, diploma, associate ship, fellowship, for other class internal examination presentation or other title anywhere else. It has not been shared with anyone for any internal presentation purpose.

Place: School of Engineering and Technology, CHRIST (Deemed to be University), Bengaluru

Date:11/3/24

Name:	ANGELA RODRIGUES	Reg No: 2262027	Signature
	ATHUL NAMBIAR	2262041	
	BENISON BINOY	2262044	

INDEX

WORKLOAD DIVISION

S.No	Name	RegNo	WorkDone
1	ANGELA RODRIGUES	2262027	STUDENT DISPLAY PAGE, SQL CONNECTIVITY, DOCUMENTATION.
2	ATHUL NAMBIAR	2262041	LOGIN PAGE,GUI.
3	BENISON BINOY	2262044	STUDENT INSERTION DETAILS, SQL CONNECTIVITY.

INTRODUCTION

The Student Management System (SMS) is a comprehensive Java Swing Project designed to streamline and manage student-related tasks within an educational institution.This application integrates seamlessly with a MySQL database, providing a robust solution for storing, retrieving, and managing student information.

Importance Of Student Management System

The Student Management System (SMS) holds significant importance in educational institutions, offering a range of benefits that contribute to efficient administration, improved communication, and enhanced learning experiences.

SMS automates routine administrative tasks such as student enrolment, performance tracking, and grade management, reducing the burden on administrative staff and minimizing errors. Faculty and staff can quickly retrieve relevant information, facilitating better communication and personalized support for students.

The integration of design elements and responsiveness contributes to the system's overall effectiveness in educational institutions. The immediate feedback mechanisms and secure input handling further enhance the reliability and user satisfaction of the SMS.

Importance of GUI:

Graphical User Interfaces (GUIs) play a crucial role in the Student Management System (SMS) by serving as the user's primary interaction point with the software. The significance of GUIs in SMS lies in their ability to provide an intuitive and visually appealing platform for students, teachers, and administrators to navigate, manage, and access essential educational information. Here are key points highlighting the importance of GUIs in the Student Management System:

User-Friendly Interaction: GUIs in SMS offer user-friendly interfaces with intuitive navigation, making it easy for users to access various features such as class schedules, grades, and attendance records. Accessible Design: Well-designed GUI elements ensure that users, including students, teachers, and administrative staff, can interact with the system efficiently, regardless of their technical expertise.

Efficient Information Retrieval: GUIs provide centralized dashboards presenting key information, such as academic records, upcoming events, and announcements. This facilitates quick and efficient retrieval of essential data for users.

Problem Statetment

The existing Student Management System (SMS) plays a crucial role in managing student information, but there are several challenges and inefficiencies that hinder its optimal functionality. Manual entry, limited data accessibility, and a lack of advanced features contribute to suboptimal student administration. To address these challenges, there is a need for an enhanced Student Management System that automates administrative tasks, provides comprehensive data insights, and offers an improved user experience for both administrators and students.

Objective

The central goal of the Student Management System (SMS) is to establish a unified platform for comprehensive management of student-related functions in educational institutions. Key objectives of the project include:

1. Automating Student Record Management**:** Implementing a system for efficient storage, retrieval, and updating of student records, ensuring accuracy and accessibility.
2. Facilitating User-Friendly Interface: Designing a Graphical User Interface (GUI) that simplifies navigation, making it easy for administrators to manage student data and generate reports.
3. Real-time Information Handling: Integrating features for real-time data updates, allowing quick access to student information, grades, and other relevant details.
4. Enhancing User Experience: Providing a seamless and intuitive experience for administrators through a visually appealing GUI, facilitating streamlined interactions with the system.
5. Improving Administrative Efficiency: Streamlining administrative tasks, including student enrolment, attendance tracking, and academic performance assessment, to enhance overall efficiency in educational processes.

6. By achieving these objectives, the Student Management System aims to revolutionize the management of student-related information, ensuring a more efficient, accurate, and user-friendly experience for educational institutions and administrators.

DESIGN

To tackle the complexities in student management and meet our outlined objectives, we present a comprehensive Student Management System (SMS) designed with a thoughtful combination of top-down and bottom-up methodologies. Our approach emphasizes modular design, intuitive interfaces, and seamless integration.

Methodology:

Our hybrid methodology blends top-down and bottom-up strategies. The top-down aspect defines the overall system architecture, identifies major modules, and establishes high-level functionality. Simultaneously, the bottom-up approach involves designing and implementing individual components, classes, and methods to fulfill specific requirements.

Methods and Classes:

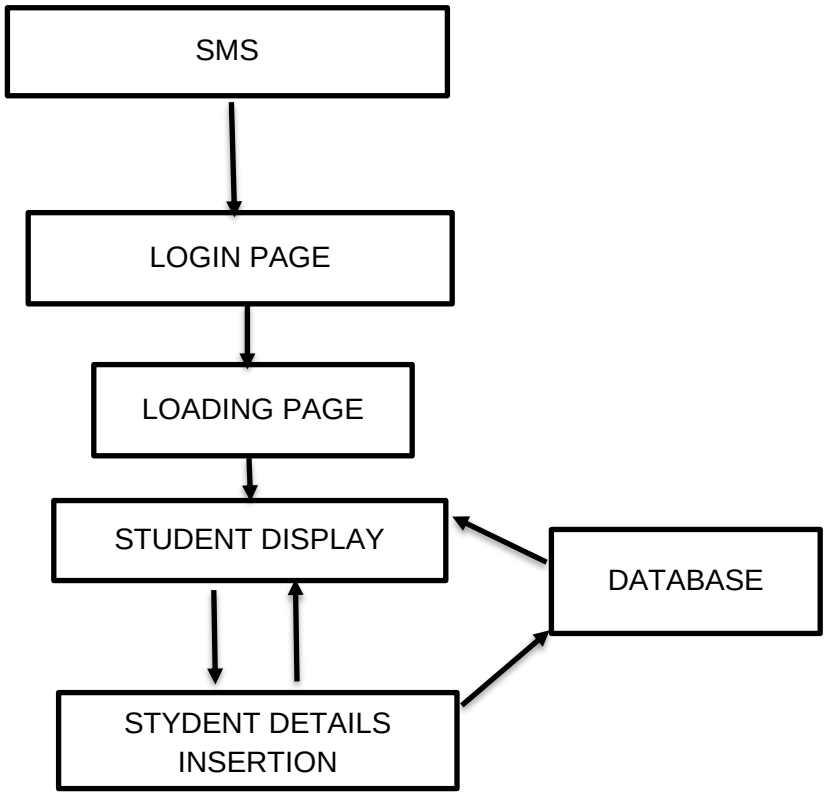
1. User Interface Classes:
    - Login Page: Allows user authentication for system access.
    - Registration Page: Facilitates new user registrations.
    - Dashboard: Provides a centralized overview of student details, attendance, and academic performance.
2. Data Management Classes:
    - Database Handler: Manages database connections and executes queries.
    - Data Models: Define classes representing entities like students, courses, and performance.
    - Data Access Objects (DAOs): Handle CRUD operations for database interactions.

3. Business Logic Classes:
    - Student Manager: Implements logic for student enrollment, attendance tracking, and academic performance assessment.
    - Course Manager: Manages course-related operations, including curriculum updates and grading.
    - User Manager: Handles user authentication, registration, and profile management.

Connection:

Classes and modules connect through well-defined interfaces and communication channels. User interface classes interact with business logic classes to handle user actions and process requests. Data management classes interact with both user interface and business logic classes to retrieve and store data from/to the database. The system architecture ensures loose coupling between components, fostering scalability, flexibility, and ease of maintenance.

Diagrammatic Representation



IMPLEMENTATION

```
import P2.App;

/**
 *
 * Project Title:   Student Management System
 *
 * @author: Athul Nambiar, Benison Binoy, Angela Rodrigues
 *
 * Objective: To execute and display the successfull working of a fullstack Java Swing Project
 * integrated with a database to store and read the data collected.
 *
 *
 * Database Server Used: MySQL Community Server 8.0
 * Databse Name: SMS
 * Table Name: Student_Details
 *
 * Tested Verified Successfull on: JDK 21.0.4
 *
 *
 * @version 1.0
 * @since 10th March 2024
 */

public class SMS {

    public static void main(String[] args)  {
        App Page2 = new App();
        Page2.OpenHome();
```

```
    }
}

package P2;

import javax.swing.*;
import javax.swing.border.Border;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import P3.LoadingPage;

/**
 * @author Athul Nambior    2262041
 *
 * Details: Login Page of the Student Management System a.k.a SMS.
 * In the event of a succesfull Login the user is presented with
 * a Login Successful pop-up window and a Loading scren on clicking ok.
 * If the Password or ID seem to be incorect the user is presented with
 * a pop-up window indicating that the ologin details are incorrect.
 *
 */

public class App implements ActionListener{

    JButton b1 = new JButton("Log In");
    JPanel outerPanel = new JPanel();
    JPanel innerPanel = new JPanel();
    JPanel loginPanel = new JPanel(); // Panel for login components
    JPanel imagePanel = new JPanel(); // Panel for the image
```

```
    JFrame frame2 = new JFrame();
    JTextField uid = new JTextField();
    JPasswordField pw = new JPasswordField();
    JLabel title = new JLabel("Login to your Account!");
    JLabel imageLabel;

    public App() {
        frame2.setVisible(true);
        frame2.setExtendedState(JFrame.MAXIMIZED_BOTH); // Set frame2 to full screen
        frame2.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Outer Panel with BorderLayout
        outerPanel.setLayout(new BorderLayout());
        outerPanel.setBackground(Color.WHITE); // White background color

        // Inner Panel with BorderLayout
        innerPanel.setLayout(new BorderLayout());
        innerPanel.setBackground(new Color(225, 225, 225)); // #cecece background color
        innerPanel.setBorder(BorderFactory.createEmptyBorder(50, 400, 100, 400)); // Adjusted
insets for better spacing
        innerPanel.setPreferredSize(new Dimension(1200, 500));

        // Add title to the top of the login panel with a gap
        title.setFont(new Font("Arial", Font.BOLD, 24));
        title.setForeground(Color.BLACK);
        title.setHorizontalAlignment(JLabel.CENTER);
        loginPanel.add(title, new GridBagConstraints(0, 0, 1, 1, 0.0, 0.2,
                GridBagConstraints.CENTER, GridBagConstraints.BOTH, new Insets(20, 0, 0, 0), 0,
0));

        // Create a panel for login components
```

```
loginPanel.setLayout(new GridBagLayout());
loginPanel.setBackground(Color.WHITE);
imagePanel.setBackground(Color.WHITE);

GridBagConstraints gbc = new GridBagConstraints();
gbc.gridx = 0; // Set the components to the left column
gbc.gridy = 1; // Start below the title with a gap
gbc.anchor = GridBagConstraints.CENTER; // Center the components
gbc.insets = new Insets(5, 5, 5, 5); // Adjusted insets for better spacing

// UID Text
JLabel s = new JLabel("User ID: ");
s.setForeground(Color.black);
s.setFont(new Font("Arial", Font.PLAIN, 16));
loginPanel.add(s, gbc);

// UID TextBox
gbc.gridy++;
uid.setPreferredSize(new Dimension(200, 50)); // width
uid.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
uid.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
loginPanel.add(uid, gbc);

// Pw Box
gbc.gridy++;
JLabel p = new JLabel("Password: ");
p.setFont(new Font("Arial", Font.PLAIN, 16));
p.setForeground(Color.black);
loginPanel.add(p, gbc);

gbc.gridy++;
```

```
        pw.setPreferredSize(new Dimension(200, 50)); // width
        pw.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
        pw.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
        loginPanel.add(pw, gbc);

        // Log in button
        gbc.gridy++;
        b1.setFont(new Font("Arial", Font.PLAIN, 16));
        b1.setPreferredSize(new Dimension(200, 50)); // width
        b1.setBackground(new Color(0x1c77ff)); // Set background color to #1c77ff
        b1.setForeground(Color.WHITE); // Set text color to white
        b1.setBorderPainted(false);
        b1.setBorder(new RoundedBorder(15, Color.WHITE));
        b1.setFocusPainted(false);
        b1.setBorder(new RoundedBorder(15, new Color(0x1c77ff))); // Set border color to #1c77ff
        b1.setOpaque(true);
        b1.addActionListener(this);


        loginPanel.add(b1, gbc);

        // Add the login panel to the center of the inner panel
        innerPanel.add(loginPanel, BorderLayout.WEST); // Move the login panel to the left

        // Panel for the image on the right side
        imagePanel.setLayout(new BorderLayout());
        // Load and set the image with a specific size
        ImageIcon originalImageIcon = new ImageIcon("lib\\ManChair.png");
        Image originalImage = originalImageIcon.getImage();
        Image scaledImage = originalImage.getScaledInstance(400, 400,
Image.SCALE_SMOOTH);
```

```
        ImageIcon scaledImageIcon = new ImageIcon(scaledImage);
        imageLabel = new JLabel(scaledImageIcon);
        imagePanel.add(imageLabel, BorderLayout.CENTER);

        // Add the image panel to the right of the inner panel
        innerPanel.add(imagePanel, BorderLayout.CENTER);

        // Add the inner panel to the outer panel
        outerPanel.add(innerPanel, BorderLayout.CENTER);

        // Add the outer panel to the frame2
        frame2.add(outerPanel);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle button actions
        if (e.getSource() == b1) {
            String usernameInput = uid.getText();
            char[] passwordInput = pw.getPassword();
            String passwordString = new String(passwordInput);

            // Replace "test" and "123" with your actual username and password
            String correctUsername = "test";
            String correctPassword = "123";

            if (usernameInput.equals(correctUsername) && passwordString.equals(correctPassword))
{
                JOptionPane.showMessageDialog(frame2, "Login Successful", "Success",
JOptionPane.INFORMATION_MESSAGE);
                frame2.dispose();
```

```
                frame2.setVisible(false);
                LoadingPage Page3=new LoadingPage();
                Page3.OpenLoading();
            } else {
                JOptionPane.showMessageDialog(frame2, "Invalid Username or Password", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    public void OpenHome() {
        SwingUtilities.invokeLater(() -> new App());
    }

    class RoundedBorder implements Border {
        private int radius;
        private Color borderColor;

        public RoundedBorder(int radius, Color borderColor) {
            this.radius = radius;
            this.borderColor = borderColor;
        }

        @Override
        public Insets getBorderInsets(Component c) {
            return new Insets(this.radius + 1, this.radius + 1, this.radius + 2, this.radius);
        }

        @Override
        public boolean isBorderOpaque() {
            return true;
```

```
        }

        @Override
        public void paintBorder(Component c, Graphics g, int x, int y, int width, int height) {
            Graphics2D g2d = (Graphics2D) g;
            g2d.setColor(borderColor);
            g2d.drawRoundRect(x, y, width - 1, height - 1, radius, radius);
        }
    }
}
package P3;

import java.awt.*;
import javax.swing.*;

import P4.DisplayWindow;



/**
 * @author Benison Binoy    2262044
 *
 * Details: Loading Page while traversing between Login Page and Main Menu of
 * the Student Management System a.k.a SMS.
 * The Page inserts a 2 second delay in the event of a successfull login from the Login Page.
 * After the 2 second delay the program automatically redirects the window
 * to the Homepage closing the current windows including
 * the Login Page and Loading Screen.
 *
 */

public class LoadingPage extends JPanel {
```

```
private Image image;

public LoadingPage() {
    // Load the image using a MediaTracker for proper loading
    Toolkit t = Toolkit.getDefaultToolkit();
    image = t.getImage("lib/loading.gif");
    try {
        MediaTracker tracker = new MediaTracker(this);
        tracker.addImage(image, 0);
        tracker.waitForAll();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

@Override
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    g.setColor(Color.WHITE);
    g.fillRect(0, 0, getWidth(), getHeight());

    // Calculate scaled image dimensions (50% of original)
    int scaledWidth = image.getWidth(this) / 2;
    int scaledHeight = image.getHeight(this) / 2;

    // Center the image within the panel
    int xOffset = (getWidth() - scaledWidth) / 2;
    int yOffset = (getHeight() - scaledHeight) / 2;

    g.drawImage(image, xOffset, yOffset, scaledWidth, scaledHeight, this);
```

```
    }

    public void OpenLoading() {

        SwingUtilities.invokeLater(() -> {
            JFrame f = new JFrame("Loading...");
            LoadingPage m = new LoadingPage();
            f.add(m);
            f.setSize(400, 300); // Set window size (adjust as needed)
            f.setLocationRelativeTo(null);
            f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            f.setVisible(true);

            new java.util.Timer().schedule(
                    new java.util.TimerTask() {
                        @Override
                        public void run() {
                            DisplayWindow Page4 = new DisplayWindow();
                            Page4.OpenDisplayWindow();
                            f.setVisible(false);
                            f.dispose();
                        }
                    },
                    2000);
        });
    }}
package P4;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.table.DefaultTableModel;

import P6.InsertionWindow;



/**
 * @author Angela Rodrigues 2262027
 *
 * Details: Main Home Page of the Student Management System a.k.a SMS
 * The Page allow the user to view the student's details all at once or individually
 * by using the buttons allocated at the top of the windoe.
 * On pressing the Insert button the window is closed and The InertionWindow is Opened.
 *
 */

public class DisplayWindow extends JFrame{
    private JTextField regnoField;
    private JTable resultTable;

    public DisplayWindow() {
        setTitle("Student Details Display");
        setSize(800, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
```

```
JPanel topPanel = new JPanel(new GridBagLayout());
topPanel.setBackground(Color.white);

JLabel headingLabel = new JLabel("STUDENT DISPLAY");
headingLabel.setFont(new Font("Arial", Font.BOLD, 24));

GridBagConstraints gbc = new GridBagConstraints();
gbc.gridx = 0;
gbc.gridy = 0;
gbc.insets = new Insets(20, 0, 20, 0);
topPanel.add(headingLabel, gbc);

JPanel inputPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
inputPanel.setBackground(Color.white);

regnoField = new JTextField(8);
JButton searchButton = new JButton("Search by REGNO");
JButton displayAllButton = new JButton("Display All Students");
JButton insertButton = new JButton("Insert");

inputPanel.add(new JLabel("Enter REGNO: "));
inputPanel.add(regnoField);
inputPanel.add(searchButton);
inputPanel.add(displayAllButton);
inputPanel.add(insertButton);

searchButton.setBackground(new Color(30, 144, 255));
searchButton.setForeground(Color.WHITE);
displayAllButton.setBackground(new Color(50, 205, 50));
displayAllButton.setForeground(Color.WHITE);
insertButton.setBackground(new Color(255, 165, 0));
```

```
insertButton.setForeground(Color.WHITE);

gbc.gridy = 1;
topPanel.add(inputPanel, gbc);

add(topPanel, BorderLayout.NORTH);

resultTable = new JTable();
JScrollPane scrollPane = new JScrollPane(resultTable);
add(scrollPane, BorderLayout.CENTER);

searchButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        searchStudentByRegNo();
    }
});

displayAllButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        displayAllStudents();
    }
});
insertButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        InsertionWindow Page6 = new InsertionWindow();
        dispose(); // Close the DisplayWindow
        Page6.OpenInsertion();
    }
```

```
    });

    setLocationRelativeTo(null);
    // setVisible(true); // Commented out to make the frame initially invisible
}

private void searchStudentByRegNo() {
    int regNo = Integer.parseInt(regnoField.getText());
    displayResults("SELECT * FROM Student_Details WHERE REGNO = " + regNo);
}

private void displayAllStudents() {
    displayResults("SELECT * FROM Student_Details");
}

private void displayResults(String sqlQuery) {
    String url = "jdbc:mysql://localhost:3306/sms";
    String username = "root";
    String password = "KARS@123";

    try (Connection connection = DriverManager.getConnection(url, username, password)) {
        try (PreparedStatement preparedStatement = connection.prepareStatement(sqlQuery)) {
            ResultSet resultSet = preparedStatement.executeQuery();
            DefaultTableModel model = new DefaultTableModel();

            model.addColumn("SNO");
            model.addColumn("REGNO");
            model.addColumn("NAME");
            model.addColumn("PP");
            model.addColumn("OS");
            model.addColumn("PQT");
```

```
            model.addColumn("PERCENTAGE");

            while (resultSet.next()) {
                Object[] row = {
                        resultSet.getInt("SNO"),
                        resultSet.getInt("REGNO"),
                        resultSet.getString("NAME"),
                        resultSet.getInt("PP"),
                        resultSet.getInt("OS"),
                        resultSet.getInt("PQT"),
                        resultSet.getDouble("PERCENTAGE")
                };
                model.addRow(row);
            }

            resultTable.setModel(model);
        }
    } catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Database error: " + ex.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);
    }
  }

  public void OpenDisplayWindow(){
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            DisplayWindow displayWindow = new DisplayWindow();
            displayWindow.setExtendedState(JFrame.MAXIMIZED_BOTH);
            displayWindow.setVisible(true);
```

```
            }
        });
    }
    public void CloseDisplayWindow(){
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                DisplayWindow displayWindow = new DisplayWindow();
                displayWindow.setVisible(false);
                displayWindow.dispose();
            }
        });
    }

}
package P6;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.Border;

import P4.DisplayWindow;

/**
 * @author Benison Binoy    2262044
 *
 * Details: Data Insertion Page of the Student Management System a.k.a SMS
 * This window offers the user with the capability to insert Student details such as Name, SNo ,
 * RegNo and Marks of 3 Subjects to the MySQL Database.
```

```
 * In case the user wants to go back to the previeous window the Back button allows the user
 * to go back to the previous window which offers the Display capabilites.
 *
 */

public class InsertionWindow implements ActionListener {
    static String nm;
    static int sno, id, m1, m2, m3;
    JFrame frame6 = new JFrame();
    JPanel op = new JPanel();
    JPanel ip = new JPanel();
    JPanel p = new JPanel();
    JLabel title = new JLabel("Student Details:");
    JLabel jLabel3 = new JLabel();
    JTextField sNoTxt = new JTextField();
    JTextField regNoTextField = new JTextField();
    JTextField nameTextField = new JTextField();
    JTextField ppTextField = new JTextField();
    JTextField osTextField = new JTextField();
    JTextField pqtTextField = new JTextField();

    JButton b1 = new JButton("Back");
    JButton b2 = new JButton("Apply");

    public void Insertion() {

        frame6.setVisible(true);
        frame6.setExtendedState(JFrame.MAXIMIZED_BOTH); // Set frame2 to full screen
        frame6.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Outer Panel with BorderLayout
```

```
        op.setLayout(new BorderLayout());
        op.setBackground(Color.WHITE); // White background color

        // Inner Panel with BorderLayout
        ip.setLayout(new BorderLayout());
        ip.setBackground(new Color(225, 225, 225)); // #cecece background color
        ip.setBorder(BorderFactory.createEmptyBorder(50, 400, 100, 400)); // Adjusted insets for
better spacing
        ip.setPreferredSize(new Dimension(1200, 500));

        // Add title to the top of the login panel with a gap
        title.setFont(new Font("Arial", Font.BOLD, 24));
        title.setForeground(Color.BLACK);
        title.setHorizontalAlignment(JLabel.CENTER);
        p.add(title, new GridBagConstraints(0, 0, 1, 1, 0.0, 0.2,
                GridBagConstraints.CENTER, GridBagConstraints.BOTH, new Insets(20, 0, 0, 0), 0,
0));

        // Create a panel for login components
        p.setLayout(new GridBagLayout());
        p.setBackground(Color.WHITE);

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.gridx = 0; // Set the components to the left column
        gbc.gridy = 1; // Start below the title with a gap
        gbc.anchor = GridBagConstraints.WEST; // Align components to the left
        gbc.insets = new Insets(5, 5, 5, 5); // Adjusted insets for better spacing

        // Label "S No:"
        JLabel s = new JLabel("S No: ");
        s.setForeground(Color.BLACK);
```

```
s.setFont(new Font("Arial", Font.PLAIN, 16));
p.add(s, gbc);

// SNo TextBox
gbc.gridx++;
gbc.gridwidth = 2; // Span across two columns
gbc.fill = GridBagConstraints.HORIZONTAL; // Fill horizontally
sNoTxt.setPreferredSize(new Dimension(200, 50)); // width
sNoTxt.setFont(new Font("Arial", Font.PLAIN, 12)); // Sfont size
sNoTxt.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
p.add(sNoTxt, gbc);

// Reset gridwidth to 1 for subsequent components
gbc.gridwidth = 1;

// Label "Reg No:"
JLabel regNoLabel = new JLabel("Reg No: ");
regNoLabel.setForeground(Color.BLACK);
regNoLabel.setFont(new Font("Arial", Font.PLAIN, 16));
gbc.gridx = 0; // Reset to the first column
gbc.gridy++; // Move to the next row
p.add(regNoLabel, gbc);

// Reg No TextBox
gbc.gridx++; // Move to the next column
gbc.gridwidth = 2; // Span across two columns
regNoTextField.setPreferredSize(new Dimension(200, 50)); // width
regNoTextField.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
regNoTextField.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
p.add(regNoTextField, gbc);
```

```
// Label "Name:"
JLabel nameLabel = new JLabel("Name: ");
nameLabel.setForeground(Color.BLACK);
nameLabel.setFont(new Font("Arial", Font.PLAIN, 16));
gbc.gridx = 0; // Reset to the first column
gbc.gridy++; // Move to the next row
p.add(nameLabel, gbc);

// Name TextBox
gbc.gridx++; // Move to the next column
gbc.gridwidth = 2; // Span across two columns
nameTextField.setPreferredSize(new Dimension(200, 50)); // width
nameTextField.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
nameTextField.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
p.add(nameTextField, gbc);

// Label "PP:"
JLabel ppLabel = new JLabel("PP: ");
ppLabel.setForeground(Color.BLACK);
ppLabel.setFont(new Font("Arial", Font.PLAIN, 16));
gbc.gridx = 0; // Reset to the first column
gbc.gridy++; // Move to the next row
p.add(ppLabel, gbc);

// PP TextBox
gbc.gridx++; // Move to the next column
gbc.gridwidth = 2; // Span across two columns
ppTextField.setPreferredSize(new Dimension(200, 50)); // width
ppTextField.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
ppTextField.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
p.add(ppTextField, gbc);
```

```
// Label "OS:"
JLabel osLabel = new JLabel("OS: ");
osLabel.setForeground(Color.BLACK);
osLabel.setFont(new Font("Arial", Font.PLAIN, 16));
gbc.gridx = 0; // Reset to the first column
gbc.gridy++; // Move to the next row
p.add(osLabel, gbc);

// OS TextBox
gbc.gridx++; // Move to the next column
gbc.gridwidth = 2; // Span across two columns
osTextField.setPreferredSize(new Dimension(200, 50)); // width
osTextField.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
osTextField.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
p.add(osTextField, gbc);

// Label "PQT:"
JLabel pqtLabel = new JLabel("PQT: ");
pqtLabel.setForeground(Color.BLACK);
pqtLabel.setFont(new Font("Arial", Font.PLAIN, 16));
gbc.gridx = 0; // Reset to the first column
gbc.gridy++; // Move to the next row
p.add(pqtLabel, gbc);

// PQT TextBox
gbc.gridx++; // Move to the next column
gbc.gridwidth = 2; // Span across two columns
pqtTextField.setPreferredSize(new Dimension(200, 50)); // width
pqtTextField.setFont(new Font("Arial", Font.PLAIN, 12)); // font size
pqtTextField.setBorder(new RoundedBorder(15, Color.BLACK)); // Black border color
```

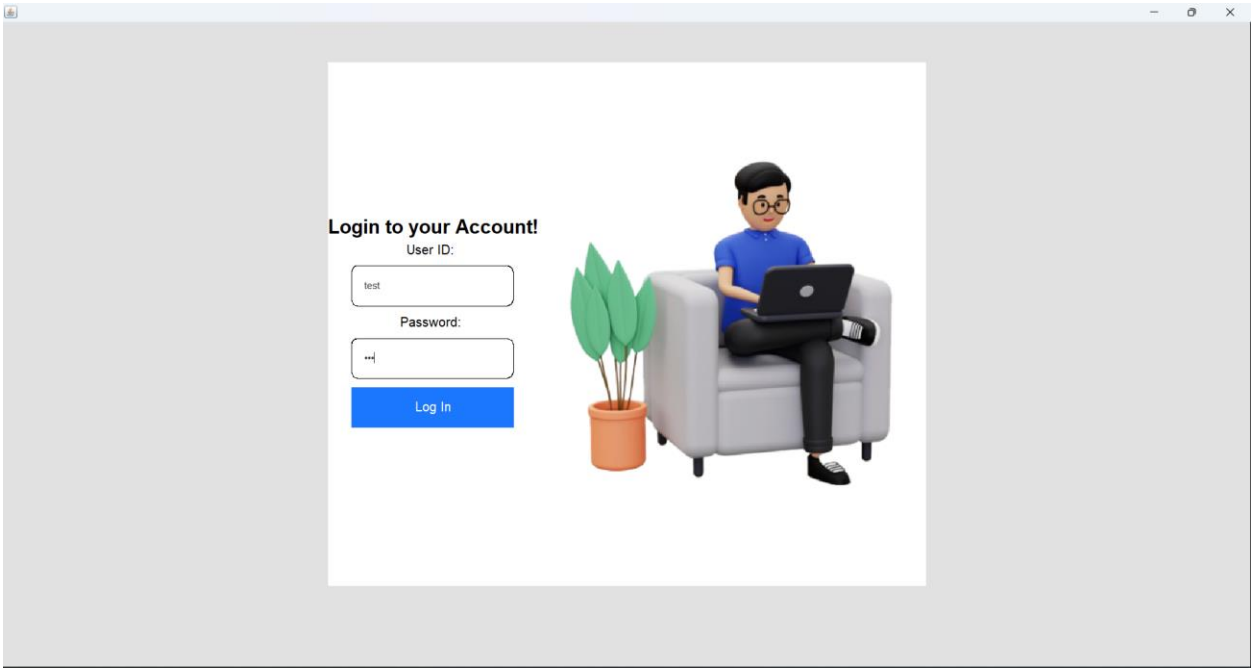
```
p.add(pqtTextField, gbc);

// Add Apply and Cancel buttons just below the PQT label and text field
gbc.gridy++; // Move to the next row
gbc.gridx = 0; // Reset to the first column
gbc.gridwidth = 3; // Span across three columns
gbc.fill = GridBagConstraints.HORIZONTAL; // Fill horizontally

// Add b1 (Back) button
gbc.gridy++; // Move to the next row
gbc.anchor = GridBagConstraints.CENTER; // Align button to the center
gbc.fill = GridBagConstraints.NONE; // Set to no fill
gbc.weightx = 0; // Reset weight
gbc.gridwidth = 1; // Set grid width to 1
gbc.insets = new Insets(20, 5, 5, 5); // Adjusted insets for better spacing
b1.setPreferredSize(new Dimension(200, 50)); // Set button size
b1.setFont(new Font("Arial", Font.PLAIN, 16)); // Set font
b1.setBackground(new Color(0x1c77ff)); // Set background color to #1c77ff
b1.setForeground(Color.WHITE); // Set text color to white
b1.setBorderPainted(false); // Disable border painting
b1.setFocusPainted(false); // Disable focus painting
b1.setOpaque(true); // Set opaque to true for background color
b1.setBorder(new RoundedBorder(15, Color.WHITE)); // Apply rounded border
p.add(b1, gbc); // Add button to panel

// Add b2 (Apply) button
gbc.gridx++; // Move to the next column
gbc.insets = new Insets(20, 5, 5, 5); // Adjusted insets for better spacing
b2.setPreferredSize(new Dimension(200, 50)); // Set button size
b2.setFont(new Font("Arial", Font.PLAIN, 16)); // Set font
b2.setBackground(new Color(0x1c77ff)); // Set background color to #1c77ff
```

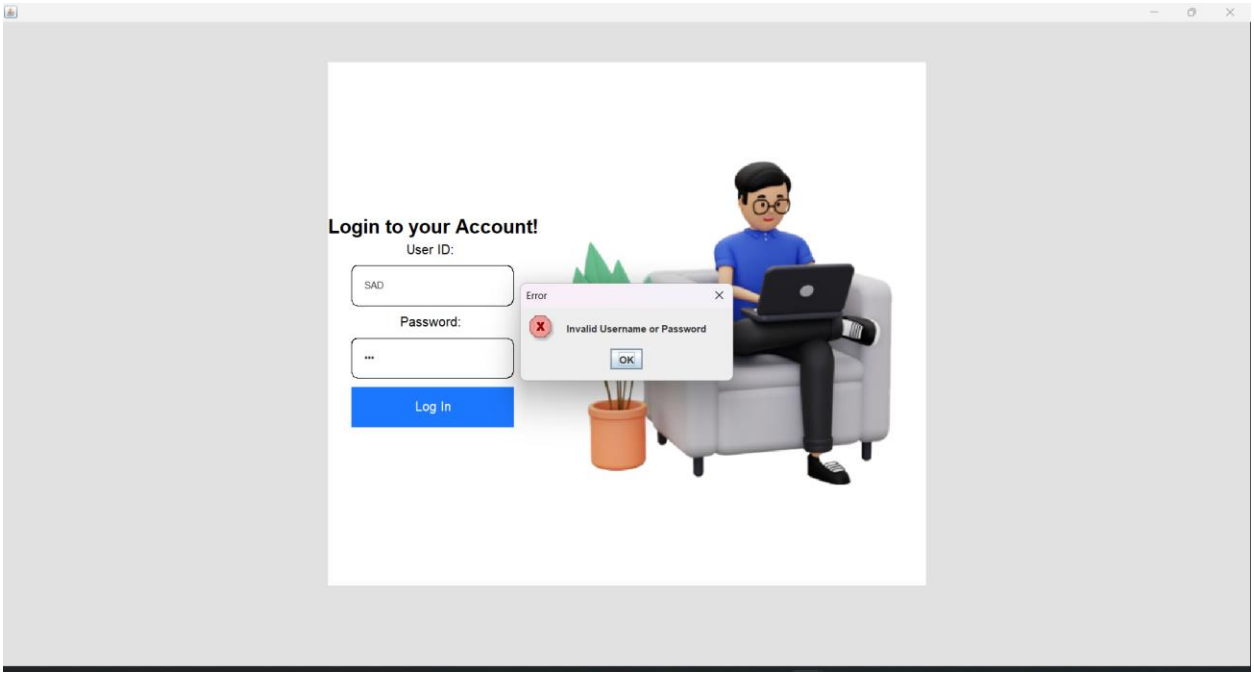
```
    b2.setForeground(Color.WHITE); // Set text color to white
    b2.setBorderPainted(false); // Disable border painting
    b2.setFocusPainted(false); // Disable focus painting
    b2.setOpaque(true); // Set opaque to true for background color
    b2.setBorder(new RoundedBorder(15, Color.WHITE)); // Apply rounded border
    p.add(b2, gbc); // Add button to panel

    frame6.add(op);
    frame6.add(ip);
    frame6.add(p);

    b1.addActionListener(this);
    b2.addActionListener(this);
}

class RoundedBorder implements Border {
    private int radius;
    private Color borderColor;

    public RoundedBorder(int radius, Color borderColor) {
        this.radius = radius;
        this.borderColor = borderColor;
    }

    @Override
    public Insets getBorderInsets(Component c) {
        return new Insets(this.radius + 1, this.radius + 1, this.radius + 2, this.radius);
    }

    @Override
    public boolean isBorderOpaque() {
```

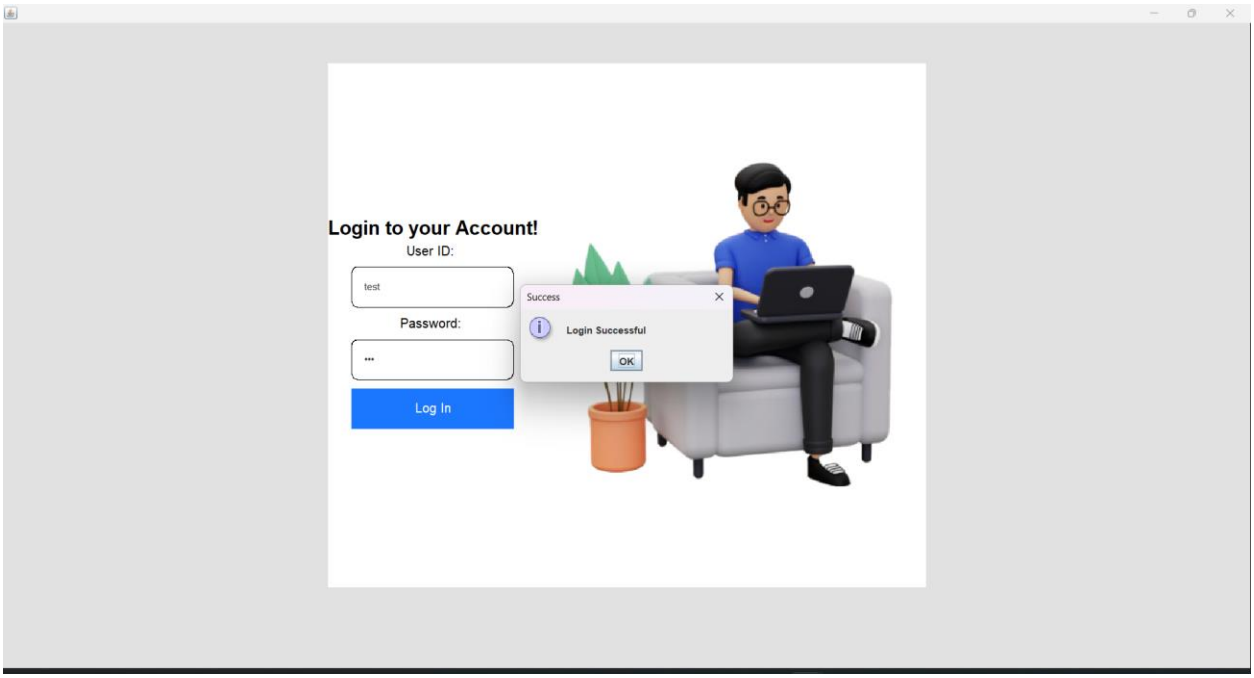
```
        return true;
    }

    @Override
    public void paintBorder(Component c, Graphics g, int x, int y, int width, int height) {
        Graphics2D g2d = (Graphics2D) g;
        g2d.setColor(borderColor);
        g2d.drawRoundRect(x, y, width - 1, height - 1, radius, radius);
    }
}

@SuppressWarnings("static-access")
@Override
public void actionPerformed(ActionEvent e) {
    SQL_q qr = new SQL_q();
    if ((e.getSource() == b2)) {
        nm = nameTextField.getText();
        sno = Integer.parseInt(sNoTxt.getText());
        id = Integer.parseInt(regNoTextField.getText());
        m1 = Integer.parseInt(ppTextField.getText());
        m2 = Integer.parseInt(osTextField.getText());
        m3 = Integer.parseInt(pqtTextField.getText());
        qr.runQuery();
    }
    if (e.getSource() == b1) { // Go back to the previous window
        // Hide and dispose the Insertion window
        frame6.setVisible(false);
        frame6.dispose();
        // Show the Display Window
        DisplayWindow Page4 = new DisplayWindow();
        Page4.OpenDisplayWindow();
```

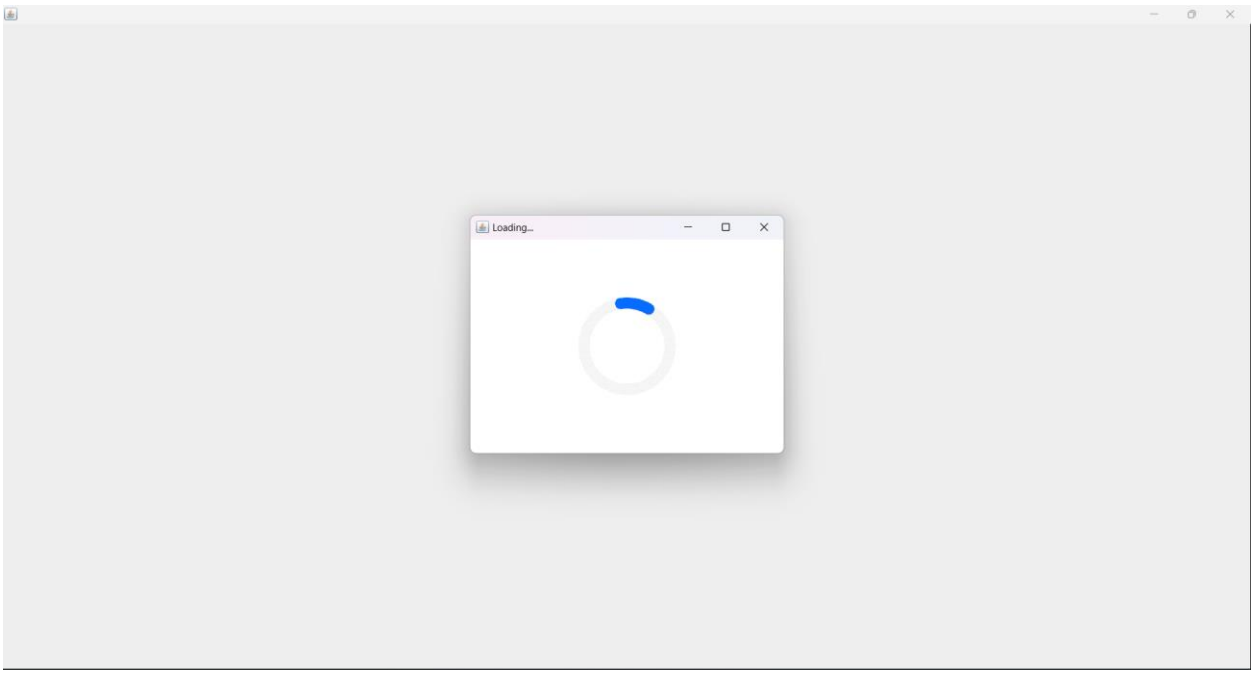
```
        }
    }
    public void OpenInsertion(){
        Insertion();
    }
}
package P6;

import java.sql.*;

public class SQL_q extends InsertionWindow {
    static void runQuery() {
        try {
            Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://localhost/sms",
                    "root", "KARS@123");

            Statement st = connection.createStatement();
            double p = (m1 + m2 + m3) / 3.0;
            String sql = "INSERT INTO Student_Details VALUES (" + sno + ", " + id + ", '" + nm +
"', " + m1 + ", " + m2 + ", " + m3 + ", " + p + ")";

            st.executeUpdate(sql);
            st.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

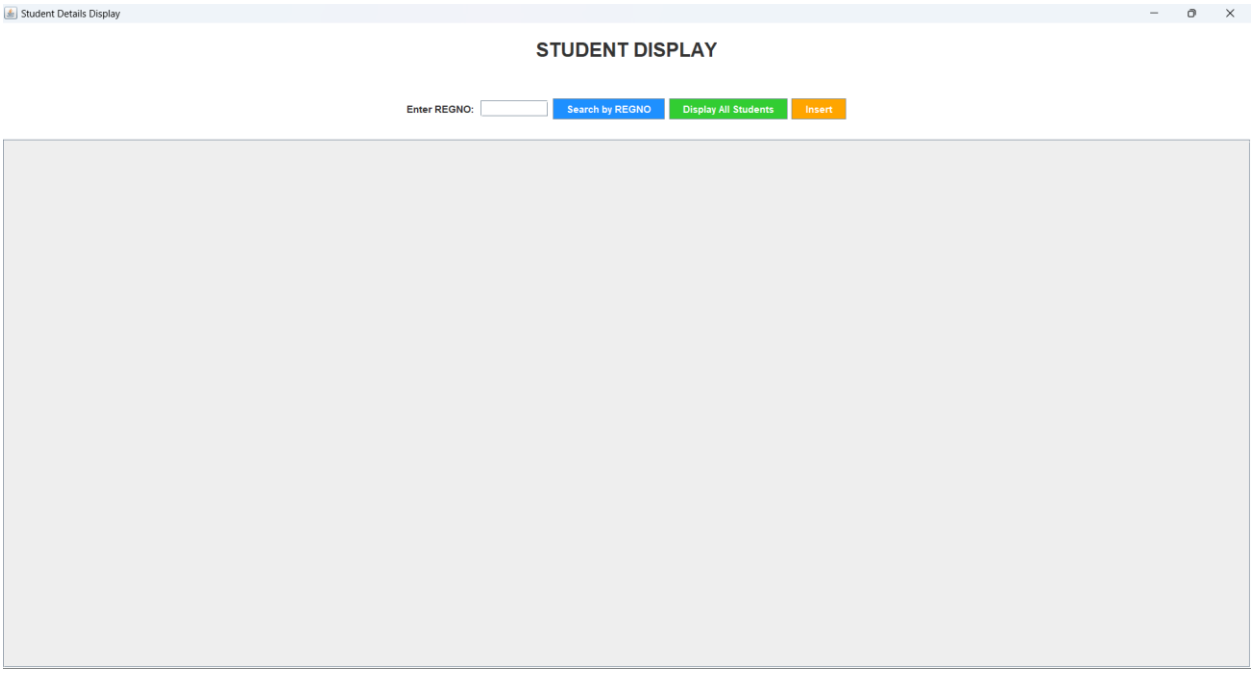
RESULTS

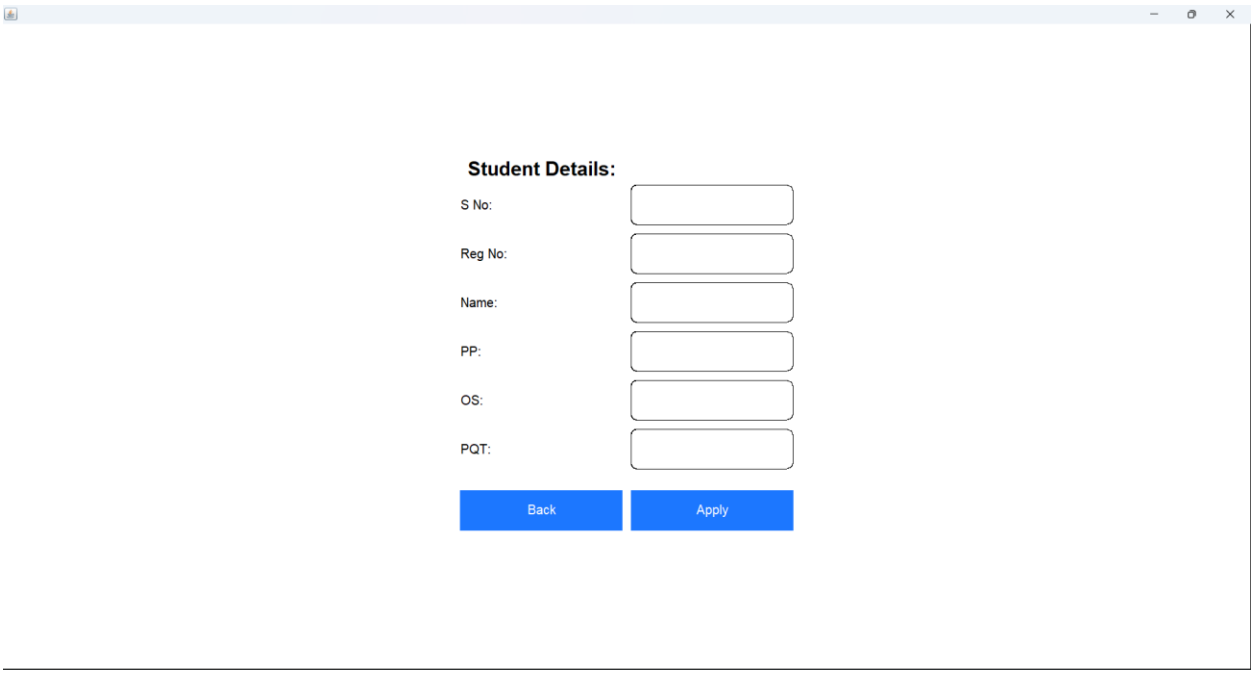

Login to your Account!
User ID:
test
Password:
Log In

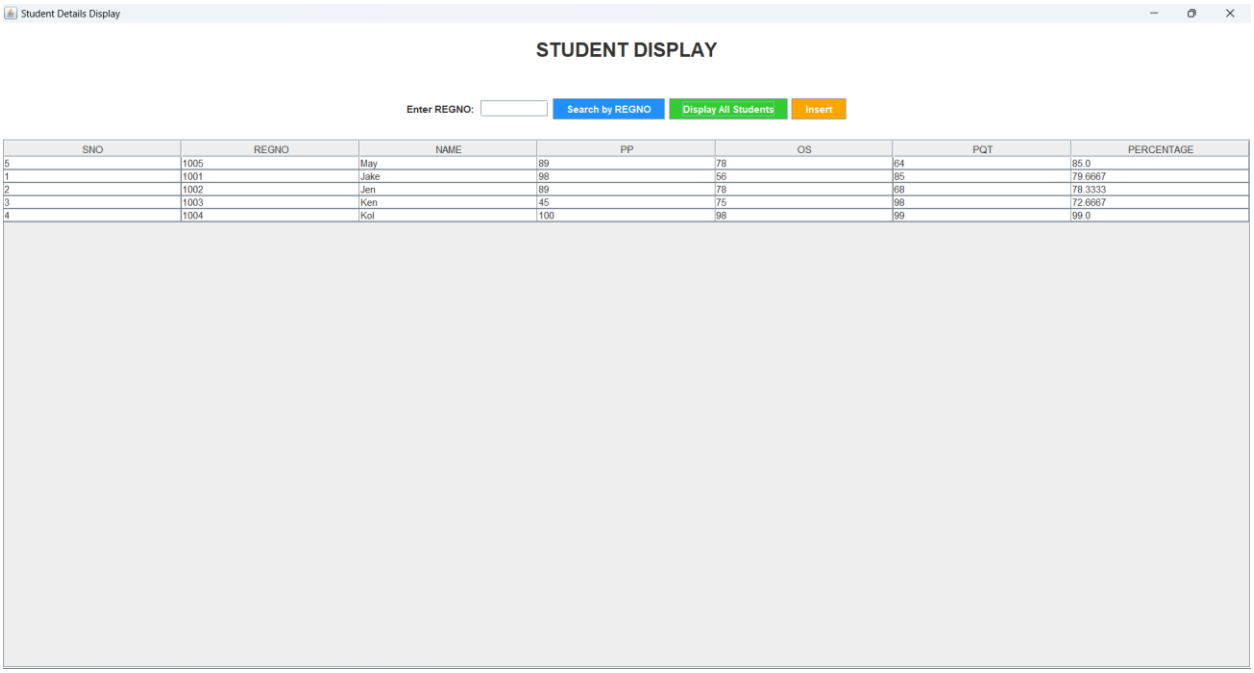

Login to your Account!
User ID:
SAD
Password:
Log In
Error
Invalid Username or Password
OK

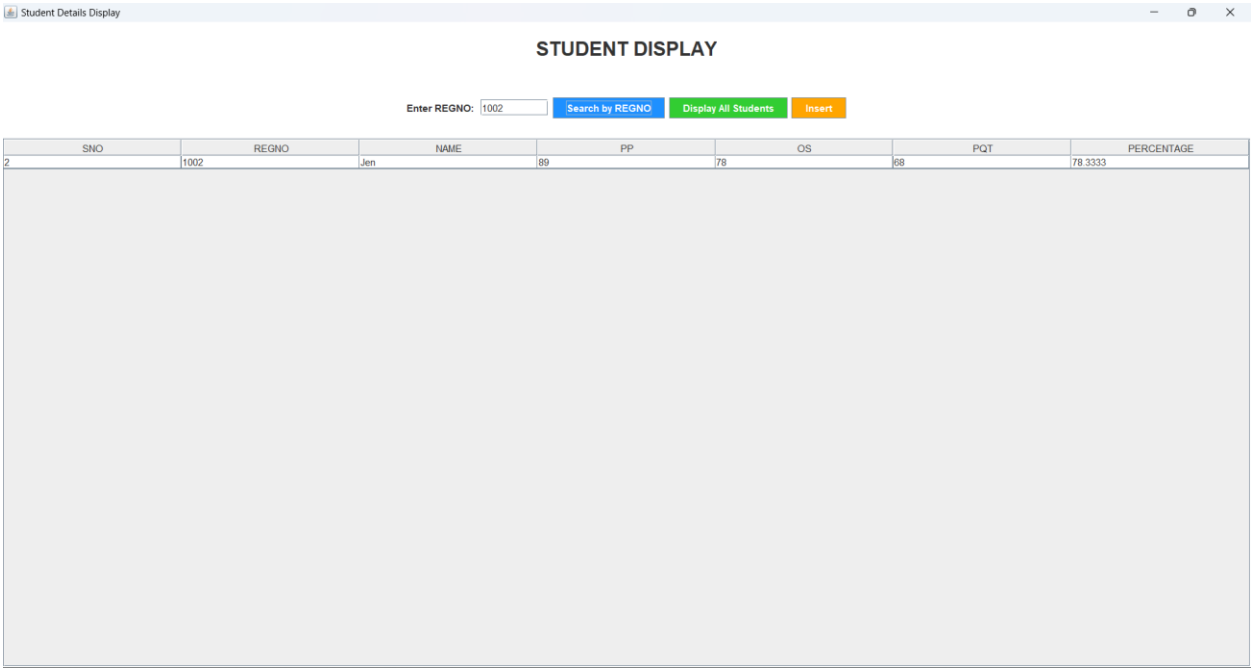

Login to your Account!
User ID:
test
Password:
Log In
Success
Login Successful
OK


Loading...


Student Details Display
STUDENT DISPLAY
Enter REGNO:
Search by REGNO
Display All Students
Insert


Student Details:
S No:
Reg No:
Name:
PP:
OS:
PQT:
Back
Apply


Student Details Display
STUDENT DISPLAY
Enter REGNO:
Search by REGNO
Display All Students
Insert
SNO REGNO NAME PP OS PQT PERCENTAGE
5 1005 May 89 78 64 85.0
1 1001 Jake 98 56 85 79.6667
2 1002 Jen 89 78 68 78.3333
3 1003 Ken 45 75 98 72.6667
4 1004 Kol 100 98 99 99.0


Student Details Display
STUDENT DISPLAY
Enter REGNO: 1002
Search by REGNO
Display All Students
Insert
SNO REGNO NAME PP OS PQT PERCENTAGE
2 1002 Jen 89 78 68 78.3333

CONCLUSION

In conclusion, the Student Management System (SMS) presented here serves s a comprehensive solution for educational institutions, offering efficient administration, improved communication, and enhanced learning experiences. The system includes a graphical user interface (GUI) for easy interaction, allowing users to seamlessly navigate through various functionalities. With features such as student data display, individual search capabilities, and a dedicated insertion window, the SMS streamlines the management of student information. Furthermore, the system incorporates a database backend, ensuring robust data storage and retrieval. The InsertionWindow facilitates the addition of new student details, contributing to the system's completeness. Overall, the SMS plays a crucial role in modern educational environments by promoting organizational efficiency and providing a user-friendly platform for effective student data management .The Student Management project successfully implements all the concepts essential for java programming being Interface, Exception handling ,Java swing/awt, Constructors, Methods, Access Specifiers, SQL/database connectivity and others.