

**Student Marks Management  
A MINI PROJECT REPORT**

**Submitted by:  
Nithish Rao P - 220701188  
Novin Jeno S - 220701190**

**In partial fulfillment for the award of the degree of**

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE**

**RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)**

**THANDALAM  
CHENNAI-602105**



**2024-2025**

## **BONAFIDE CERTIFICATE**

**Certified that this project report “Student Marks Management” is the bonafide work of “Nithish Rao P (220701188),Novin Jeno S (220701190) ”who carried out the project work under my supervision.**

**SIGNATURE**

**Dr.R.SABITHA**

**Professor and II Year Academic Head  
Computer Science and Engineering,  
Rajalakshmi Engineering College  
(Autonomous),  
Thandalam, Chennai - 602 105**

**SIGNATURE**

**Ms.D.KALPANA**

**Assistant Professor (SG),  
ComputerScienceandEngineering,  
Rajalakshmi Engineering College  
(Autonomous),  
Thandalam, Chennai - 602 105**

**Submitted for the Practical Examination held on\_\_\_\_\_**

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **ABSTRACT**

In today's digital age, educational institutions are increasingly leveraging technology to enhance administrative efficiency and improve student outcomes. A key component of this transformation is the development of student and teacher portals, which streamline various academic and administrative processes. This project presents the design and implementation of a simple yet functional Student Portal using Java Swing for the graphical user interface and JDBC (Java Database Connectivity) for database operations. The portal consists of two main modules: the student portal and the teacher portal, each catering to specific user needs.

### **System Design**

The Student Portal is designed with a clear focus on simplicity and functionality. The user interface is developed using Java Swing, ensuring a responsive and interactive experience. The backend operations are handled using JDBC, which facilitates robust database interactions.

### **Student Portal**

The student portal allows students to log in using their roll number. Once authenticated, they can view their personal details, including name, roll number, and course information. Additionally, students can access their academic records, such as grades and attendance. The portal ensures data privacy and security, allowing only authorized access to sensitive information.

### **Teacher Portal**

The teacher portal is designed to manage student records efficiently. Teachers can log in using their credentials and perform various tasks:

- Enter and update marks for different subjects.
- View a consolidated list of student performance.
- Identify and list students who have failed in one or more subjects.

The teacher portal provides a comprehensive overview of class performance, helping educators to take necessary actions to support struggling students.

## **TABLE OF CONTENTS**

### **1. INTRODUCTION**

#### **1.1 INTRODUCTION**

#### **1.2 OBJECTIVES**

#### **1.3 MODULES**

### **2. SURVEY OF TECHNOLOGIES**

#### **2.1 SOFTWARE DESCRIPTION**

#### **2.2 LANGUAGES**

##### **2.2.1 SQL**

##### **2.2.2 Java**

### **3. REQUIREMENTS AND ANALYSIS**

#### **3.1 REQUIREMENT SPECIFICATION**

#### **3.2 HARDWARE AND SOFTWARE REQUIREMENTS**

#### **3.3 ARCHITECTURE DIAGRAM**

#### **3.4 ER DIAGRAM**

#### **3.5 NORMALIZATION**

### **4. PROGRAM CODE**

### **5. RESULTS AND DISCUSSION**

### **6. CONCLUSION**

### **7. REFERENCES**

# 1. Introduction

## 1.1 Introduction

In the contemporary educational landscape, digital solutions play a crucial role in enhancing administrative efficiency and improving student outcomes. This project presents a Student Portal developed using Java Swing for the graphical user interface (GUI) and Java Database Connectivity (JDBC) for backend database operations. The portal is divided into two main sections: the student portal and the teacher portal, each designed to cater to specific user needs. The student portal allows students to access their academic details, while the teacher portal enables teachers to manage student marks and identify students who are failing.

## 1.2 Objectives

The primary objectives of this project are as follows:

1. User-Friendly Interface: Develop an intuitive and responsive interface using Java Swing to ensure a seamless user experience for both students and teachers.
2. Efficient Data Retrieval: Enable students to access their personal and academic information using their roll number.
3. Effective Data Management: Provide teachers with tools to input and update student marks, and to identify and manage students who have failed.
4. Data Security: Implement robust security measures to protect user data and ensure privacy.
5. Enhanced Communication: Foster better communication and transparency between students and teachers regarding academic performance.

## 1.3 Modules

The Student Portal consists of two primary modules: the Student Portal and the Teacher Portal. Each module is designed to serve the specific needs of its users.

### 1.3.1 Student Portal

The student portal allows students to log in using their roll number. Once authenticated, they can view their personal details such as name, roll number, and course information. Additionally, students can access their academic records, including grades and attendance. This module ensures that students have secure

and easy access to their academic information, promoting transparency and self-awareness.

Key Features:

- Login Authentication: Students log in using their roll number.
- Personal Information: Display of student's personal details.
- Academic Records: Access to grades and attendance records.

### 1.3.2 Teacher Portal

The teacher portal provides teachers with the necessary tools to manage student academic records efficiently. Teachers can log in using their credentials and perform various tasks related to student evaluation and performance tracking.

Key Features:

- Login Authentication: Secure login for teachers.
- Marks Entry: Ability to enter and update student marks for different subjects.
- Failure Identification: List and manage students who have failed in one or more subjects.
- Class Performance Overview: Consolidated view of class performance to help identify trends and support students effectively.

## 2. Survey of Technologies

### 2.1 Software Description

The development of the Student Portal leverages several key software technologies to ensure robust functionality and user experience. The primary components used in this project are Java Swing for the user interface and JDBC for database connectivity. Java Swing provides a rich set of GUI components that enable the creation of interactive and user-friendly applications. JDBC is a Java-based API that facilitates seamless interaction with relational databases, allowing for efficient data management and retrieval.

**Java Swing:** A part of the Java Foundation Classes (JFC), Swing is used to build graphical user interfaces for Java applications. It provides a wide range of lightweight components that can be customized to create complex and visually appealing interfaces. Swing is platform-independent, ensuring consistent behavior across different operating systems.

**JDBC (Java Database Connectivity):** JDBC is an API that enables Java applications to interact with relational databases. It supports various database operations such as querying, updating, and managing database records. JDBC is designed to be database-agnostic, meaning it can work with different database management systems (DBMS) without requiring major changes to the code.

### 2.2 Languages

#### 2.2.1 SQL

Structured Query Language (SQL) is used to interact with the relational database that stores the data for the Student Portal. SQL is a standardized language for managing and manipulating databases. It allows for the execution of various operations such as data insertion, updates, deletions, and queries. In the context of this project, SQL is used to create and manage the database schema, and to perform CRUD (Create, Read, Update, Delete) operations.

### Key Features of SQL:

Data Definition Language (DDL): Commands such as CREATE, ALTER, and DROP are used to define and modify the database structure.

Data Manipulation Language (DML): Commands such as SELECT, INSERT, UPDATE, and DELETE are used to manipulate the data within the database.

Data Control Language (DCL): Commands such as GRANT and REVOKE are used to control access to data in the database.

Transaction Control Language (TCL): Commands such as COMMIT and ROLLBACK are used to manage transactions within the database.

### 2.2.2 Java

Java is a versatile and widely-used programming language that forms the backbone of this project. It is known for its platform independence, object-oriented features, and robustness. Java is used to develop both the frontend (using Java Swing) and the backend (using JDBC for database interactions) of the Student Portal.

#### Key Features of Java:

Platform Independence: Java programs are compiled into bytecode, which can run on any device equipped with the Java Virtual Machine (JVM), ensuring cross-platform compatibility.

Object-Oriented Programming (OOP): Java's OOP principles promote code modularity, reusability, and scalability. This is particularly beneficial in managing complex systems like the Student Portal.

Rich API: Java provides a comprehensive standard library that includes a wide range of tools for networking, data structures, and graphical user interfaces, simplifying the development process.

Security: Java includes a robust security model that protects applications from malicious attacks, which is crucial for handling sensitive academic data.

#### Conclusion

The technologies and languages chosen for this project, including Java Swing, JDBC, SQL, and Java, provide a solid foundation for developing a robust and user-friendly Student Portal. These tools ensure that the system is efficient, secure, and capable of meeting the needs of both students and teachers. The successful implementation of this project demonstrates the effectiveness of these technologies in creating practical solutions for educational administration.



### 3. Requirements and Analysis

#### 3.1 Requirement Specification

The requirement specification for the Student Portal involves identifying and documenting the functionalities that the system must support. These requirements are categorized into functional and non-functional requirements.

Functional Requirements:

Student Portal:

Login: Students must be able to log in using their roll number.

View Details: Students must be able to view their personal details(name,roll number,course).

View Academic Records: Students must be able to access their grades and attendance records.

Teacher Portal:

Login: Teachers must be able to log in using their credentials.

Enter Marks: Teachers must be able to enter and update student marks.

View Failures: Teachers must be able to view a list of students who have failed in one or more subjects.

View Class Performance: Teachers must have access to a consolidated view of class performance.

Non-Functional Requirements:

Security: The system must ensure secure access to student and teacher data. Only authenticated users should be able to access their respective portals.

Usability: The interface must be intuitive and user-friendly, making it easy for users to navigate and perform their tasks.

Performance: The system should be responsive, with minimal load times for retrieving and displaying data.

Scalability: The system should be able to handle an increasing number of users and data without significant performance degradation.

Maintainability: The system should be easy to maintain and update, with well-documented code and modular design.

### 3.2 Hardware and Software Requirements

#### Hardware Requirements:

**Server:** A server to host the database and handle requests. Specifications include a multi-core processor, at least 8GB of RAM, and sufficient storage (e.g., 500GB SSD).

**Client Machines:** Any standard computer or laptop that can run Java applications. Specifications include a dual-core processor, 4GB of RAM, and basic storage.

#### Software Requirements:

**Operating System:** Windows, Linux, or macOS.

**Java Development Kit (JDK):** Version 8 or higher.

**Integrated Development Environment (IDE):** IntelliJ IDEA, Eclipse, or NetBeans.

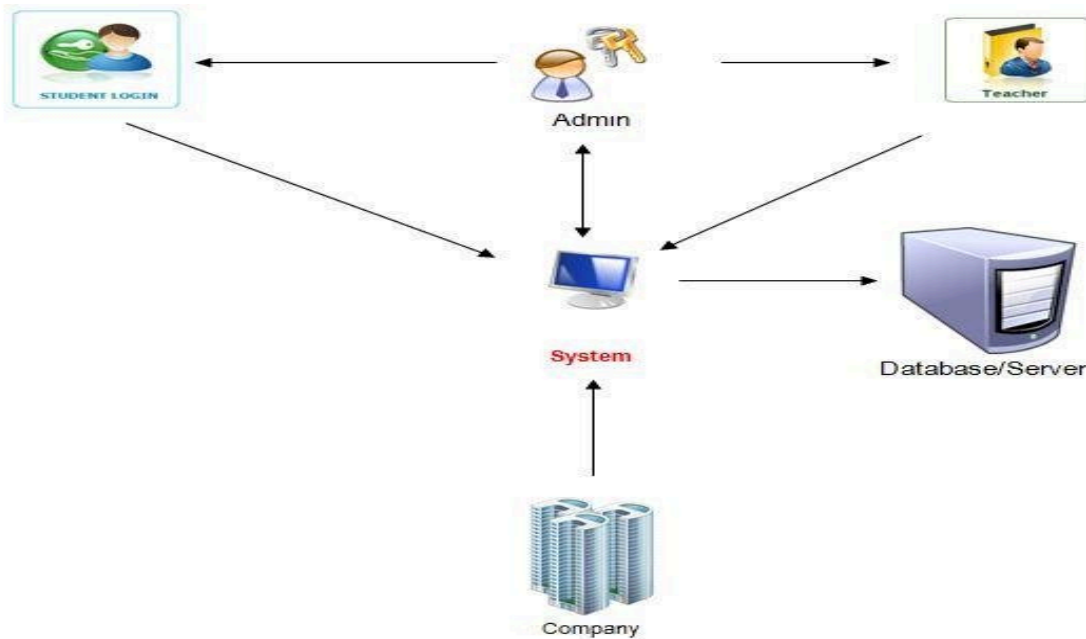
**Database Management System (DBMS):** MySQL, PostgreSQL, or any other compatible relational database.

**Java Swing:** For developing the graphical user interface.

**JDBC (Java Database Connectivity):** For database interactions.

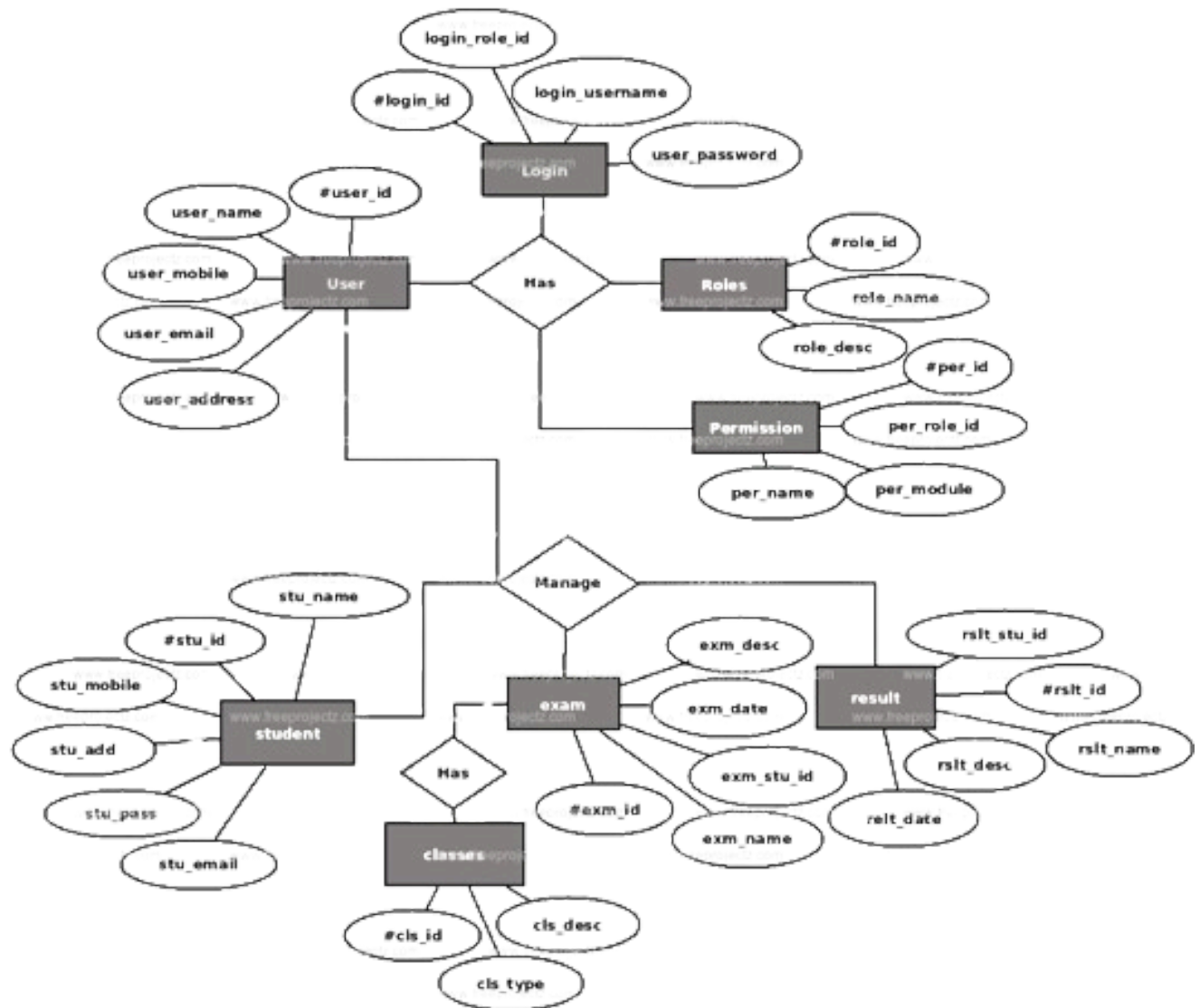
### 3.3 Architecture Diagram

The architecture of the Student Portal follows a client-server model. The client-side application is developed using Java Swing, while the server-side consists of a database managed through JDBC.



### 3.4 ER Diagram

The Entity-Relationship (ER) diagram represents the database schema for the Student Portal. It includes entities such as Students, Teachers, Subjects, and Marks, along with their relationships.



### 3.5) Normalization:

Normalization is a crucial step in database design to reduce redundancy and dependency. In the context of a student marks management system, normalization helps organize data efficiently, ensuring data integrity and making it easier to query and manipulate.

#### 1. First Normal Form (1NF):

- Ensure that each attribute holds only atomic (indivisible) values.
- For example, the "Name" attribute in the Students entity should not contain multiple names separated by commas.

#### 2. Second Normal Form (2NF):

- Ensure that all attributes are fully functionally dependent on the primary key.
- For example, if we have a composite primary key (Student\_ID, Subject\_ID) in the Marks entity, Marks\_Obtained should be dependent on both Student\_ID and Subject\_ID.

#### 3. Third Normal Form (3NF):

- Eliminate transitive dependencies where an attribute depends on another non-key attribute.
- For example, if we have an attribute "Age" derived from "Date\_of\_Birth" in the Students entity, we should remove it to eliminate transitive dependency.

#### 1. Students:

Student_ID	Name	Date_of_Birth	Address
101	John Doe	1999-05-20	123 Main St
102	Jane Smith	2000-03-15	456 Elm St

#### 2. Subjects:

Subject_ID	Subject_Name
1	Math
2	Science

#### 3. Marks:

Student_ID	Subject_ID	Marks_Obtained
101	1	85
101	2	75
102	1	90
102	2	80

## 4)PROGRAM CODE

### App.java:

```
package package1;
import javax.swing.*;
public class App {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Student Records Manager");
        filehandler sw = new filehandler();
        teacherlogin tl = new teacherlogin();
        studentlogin sl = new studentlogin();
        sw.printfunction();
        frame.setSize(1000, 1000);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);
        JLabel l1 = new JLabel("Select known Languages");
        l1.setBounds(100, 0, 160, 80);
        JButton student_button = new JButton("Student");
        student_button.setBounds(100, 50, 100, 30);
        JButton teacher_button = new JButton("Teacher");
        teacher_button.setBounds(100, 80, 100, 30);
        teacher_button.addActionListener(e -> {
            frame.setVisible(false);
            tl.teacherlogin_frame();
        });
        student_button.addActionListener(e -> {
            sl.studentlogin_frame();
        });
        frame.add(l1);
        frame.add(student_button);
        frame.add(teacher_button);
        frame.setVisible(true);
    }
}
```

## EnterMarks.java

```
package package1;
import java.sql.;
import javax.swing.;
public class entermarks {
    public void entermarks_frame(){
        JFrame entermarks_frame = new JFrame("Enter Marks Manager");
        JLabel StudentID = new JLabel("Student ID:");
        JLabel Mark1 = new JLabel("Subject 1:");
        JLabel Mark2 = new JLabel("Subject 2:");
        JLabel Mark3 = new JLabel("Subject 3:");
        JLabel Mark4 = new JLabel("Subject 4:");
        JLabel Mark5 = new JLabel("Subject 5:");
        JTextField studentIDTextField = new JTextField();
        JTextField mark1JTextField = new JTextField();
        JTextField mark2JTextField = new JTextField();
        JTextField mark3JTextField = new JTextField();
        JTextField mark4JTextField = new JTextField();
        JTextField mark5JTextField = new JTextField();
        JButton loginButton = new JButton("Enter Mark");
        StudentID.setBounds(50, 70, 80, 30);
        studentIDTextField.setBounds(130,70,80,30);
        Mark1.setBounds(50, 110, 80, 30);
        Mark2.setBounds(50, 150, 80, 30);
        Mark3.setBounds(50, 190, 80, 30);
        Mark4.setBounds(50, 230, 80, 30);
        Mark5.setBounds(50, 270, 80, 30);
        studentIDTextField.setBounds(130, 70, 150, 30);
        mark1JTextField.setBounds(130, 110, 150, 30);
        mark2JTextField.setBounds(130, 150, 150, 30);
        mark3JTextField.setBounds(130, 190, 150, 30);
        mark4JTextField.setBounds(130, 230, 150, 30);
        mark5JTextField.setBounds(130, 270, 150, 30);
        loginButton.setBounds(130,310,150,30);
        loginButton.addActionListener(e -> {
            try {
                Class.forName("com.mysql.cj.jdbc.Driver");
                String url = "jdbc:mysql://127.0.0.1:3306/student_login";
                String user = "root";
                String password1 = "";
                PreparedStatement preparedStatement = null;
                System.out.println("before connection");
                Connection connection = DriverManager.getConnection(url, user, password1)
```

```

        System.out.println("after connection");
        if (connection != null) {
            String sql = "INSERT INTO studentmark (StudentID, Subject1Marks, Subject2Marks,
Subject3Marks, Subject4Marks, Subject5Marks) VALUES (?, ?, ?, ?, ?, ?)";
            int stuid = Integer.parseInt(studentIDTextField.getText());
            int m1 = Integer.parseInt(mark1JTextField.getText());
            int m2 = Integer.parseInt(mark2JTextField.getText());
            int m3 = Integer.parseInt(mark3JTextField.getText());
            int m4 = Integer.parseInt(mark4JTextField.getText());
            int m5 = Integer.parseInt(mark5JTextField.getText());
            preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setInt(1, stuid);
            preparedStatement.setInt(2, m1);
            preparedStatement.setInt(3, m2);
            preparedStatement.setInt(4, m3);
            preparedStatement.setInt(5, m4);
            preparedStatement.setInt(6, m5);
            int rowsInserted = preparedStatement.executeUpdate();
            if (rowsInserted > 0) {
                JOptionPane.showMessageDialog(entermarks_frame, "Updated");
            } else {JOptionPane.showMessageDialog(entermarks_frame, "Some error occurred");}
            connection.close();} catch (ClassNotFoundException | SQLException ex) {
                System.err.println("Error connecting to the database: " + ex.getMessage());
            }
        }
    });

    entermarks_frame.add(StudentID);
    entermarks_frame.add(studentIDTextField);
    entermarks_frame.add(Mark1);
    entermarks_frame.add(Mark2);
    entermarks_frame.add(Mark3);
    entermarks_frame.add(Mark4);
    entermarks_frame.add(Mark5);
    entermarks_frame.add(mark1JTextField);
    entermarks_frame.add(mark2JTextField);
    entermarks_frame.add(mark3JTextField);
    entermarks_frame.add(mark4JTextField);
    entermarks_frame.add(mark5JTextField);
    entermarks_frame.add(loginButton);
    entermarks_frame.setSize(1000, 1000);
    entermarks_frame.setLayout(null);
    entermarks_frame.setVisible(true);
    entermarks_frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
}

```

## Failure.java

```
package package1;
import java.awt.BorderLayout;
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.;

public class studentlogin {
    public void studentlogin_frame() {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                createAndShowGUI(); }});}
    private static void createAndShowGUI() {
        JFrame frame = new JFrame("Student Information Fetcher");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 400);
        JPanel panel = new JPanel();
        JLabel studentIdLabel = new JLabel("Enter Student ID:");
        JTextField studentIdField = new JTextField(15);
        JButton fetchButton = new JButton("Fetch");
        panel.add(studentIdLabel);
        panel.add(studentIdField);
        panel.add(fetchButton);
        JLabel resultLabel = new JLabel();
        JScrollPane scrollPane = new JScrollPane(resultLabel);
        scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
        frame.add(panel, BorderLayout.NORTH);
        frame.add(scrollPane, BorderLayout.CENTER);
        fetchButton.addActionListener(new ActionListener() {@Override
public void actionPerformed(ActionEvent e) {
            String studentId = studentIdField.getText().trim();
            if (!studentId.isEmpty()) {
                fetchStudentInfo(studentId, resultLabel);
            } else {resultLabel.setText("Please enter a valid Student ID.");}}
        });frame.setVisible(true);}
    private static void fetchStudentInfo(String studentId, JLabel resultLabel) {
        String url = "jdbc:mysql://localhost:3306/student_login";
        String username = "root";
```



```

String password = "";
String studentQuery = "SELECT StudentID, Name, Email, Class, Section FROM student WHERE
StudentID = ? ";
String marksQuery = "SELECT Subject1Marks, Subject2Marks, Subject3Marks, Subject4Marks,
Subject5Marks FROM studentmark WHERE StudentID = ?";
try (Connection conn = DriverManager.getConnection(url, username, password);
    PreparedStatement studentStmt = conn.prepareStatement(studentQuery);
    PreparedStatement marksStmt = conn.prepareStatement(marksQuery)) {
    studentStmt.setString(1, studentId);
    marksStmt.setString(1, studentId);
    ResultSet studentRs = studentStmt.executeQuery();
    ResultSet marksRs = marksStmt.executeQuery();
    if (studentRs.next() && marksRs.next()) {
        int subject1Marks = marksRs.getInt("Subject1Marks");
        int subject2Marks = marksRs.getInt("Subject2Marks");
        int subject3Marks = marksRs.getInt("Subject3Marks");
        int subject4Marks = marksRs.getInt("Subject4Marks");
        int subject5Marks = marksRs.getInt("Subject5Marks");
        double avgMarks = (subject1Marks + subject2Marks + subject3Marks + subject4Marks +
subject5Marks) / 5.0;
        StringBuilder result = new StringBuilder("<html>");
        result.append("Student ID: ").append(studentRs.getString("StudentID")).append("<br>");
        result.append("Name: ").append(studentRs.getString("Name")).append("<br>");
        result.append("Email: ").append(studentRs.getString("Email")).append("<br>");
        result.append("Class: ").append(studentRs.getString("Class")).append("<br>");
        result.append("Section: ").append(studentRs.getString("Section")).append("<br>");
        result.append("Subject 1 Marks: ").append(subject1Marks).append("<br>");
        result.append("Subject 2 Marks: ").append(subject2Marks).append("<br>");
        result.append("Subject 3 Marks: ").append(subject3Marks).append("<br>");
        result.append("Subject 4 Marks: ").append(subject4Marks).append("<br>");
        result.append("Subject 5 Marks: ").append(subject5Marks).append("<br>");
        result.append("Average Marks: ").append(avgMarks).append("<br>");
        result.append("</html>");
        resultLabel.setText(result.toString());
    } else {
        resultLabel.setText("No student found with the given ID."); }
} catch (SQLException e) {
    e.printStackTrace();
    resultLabel.setText("Error fetching data.");
}
}
}

```

## teacherlogin.java

```
package package1;
import javax.swing.;
import javax.swing.table.DefaultTableModel;
import java.sql.;
import java.util.Vector;

public class teacherlogin {
    public void teacherlogin_frame() {
        teacherportal tp = new teacherportal();
        JFrame teacherlogin_frame = new JFrame("Teacher Login Manager");
        JLabel userLabel = new JLabel("Username:");
        JLabel passwordLabel = new JLabel("Password:");
        JLabel messageLabel = new JLabel("");
        JTextField userTextField = new JTextField();
        JPasswordField passwordField = new JPasswordField();
        JButton loginButton = new JButton("Login");
        userLabel.setBounds(50, 70, 80, 30);
        passwordLabel.setBounds(50, 110, 80, 30);
        userTextField.setBounds(130, 70, 150, 30);
        passwordField.setBounds(130, 110, 150, 30);
        loginButton.setBounds(130, 150, 80, 30);
        messageLabel.setBounds(130, 190, 200, 30);
        messageLabel.setBounds(130, 220, 400, 30);
        loginButton.addActionListener(e -> {
            try {
                Class.forName("com.mysql.cj.jdbc.Driver");
                String url = "jdbc:mysql://127.0.0.1:3306/student_login";
                String user = "root";
                String password1 = "";
                System.out.println("before connection");
                Connection connection = DriverManager.getConnection(url, user, password1);
                System.out.println("after connection");
                if (connection != null) {
                    System.out.println("Connected to the database.");
                    String username = userTextField.getText();
                    String pass = new String(passwordField.getPassword());
                    String check_teacher = "SELECT FROM teacherportal WHERE Email = " + username + "
AND Department = " + pass + """;
                    Statement st = connection.createStatement();
                    ResultSet rs = st.executeQuery(check_teacher);
                    if (rs != null && rs.next())
                        tp.teacherportal_frame();
                }
            }
        });
    }
}
```

```

        else {
            JOptionPane.showMessageDialog(teacherlogin_frame, "Wrong Password or Username");
            connection.close();
        } catch (ClassNotFoundException | SQLException ex) {
            System.err.println("Error connecting to the database: " + ex.getMessage());
        }
    }
    teacherlogin_frame.add(userLabel);
    teacherlogin_frame.add(passwordLabel);
    teacherlogin_frame.add(userTextField);
    teacherlogin_frame.add(passwordField);
    teacherlogin_frame.add(loginButton);
    teacherlogin_frame.add(messageLabel);
    teacherlogin_frame.setSize(400, 300);
    teacherlogin_frame.setLayout(null);
    teacherlogin_frame.setVisible(true);
    teacherlogin_frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

public static DefaultTableModel buildTableModel(ResultSet rs)
throws SQLException {

    ResultSetMetaData metaData = rs.getMetaData();

    Vector<String> columnNames = new Vector<String>();
    int columnCount = metaData.getColumnCount();
    for (int column = 1; column <= columnCount; column++) {
        columnNames.add(metaData.getColumnName(column));
    }

    // data of the table
    Vector<Vector<Object>> data = new Vector<Vector<Object>>();
    while (rs.next()) {
        Vector<Object> vector = new Vector<Object>();
        for (int columnIndex = 1; columnIndex <= columnCount; columnIndex++) {
            vector.add(rs.getObject(columnIndex));
        }
        data.add(vector);
    }
    return new DefaultTableModel(data, columnNames);
}
}

```

## Teacherportal.java

```
package package1;

import javax.swing.*;
public class teacherportal {
    public void teacherportal_frame()
    {
        entermarks em = new entermarks();
        viewmarks vm = new viewmarks();
        failure f = new failure();
        JFrame teacherportal_frame = new JFrame("Teacherportal Manager");
        teacherportal_frame.setSize(1000, 1000);
        teacherportal_frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        teacherportal_frame.setLayout(null);
        JLabel actiontextJLabel = new JLabel("Options Available for teacher");
        actiontextJLabel.setBounds(100, 0, 260, 80);
        JButton enter_mark = new JButton("Enter Mark");
        enter_mark.setBounds(100, 50, 300, 30);
        JButton failurButton = new JButton("View Failures");
        JButton viewmarks = new JButton("View All the Student's Marks");
        failurButton.setBounds(100, 80, 300, 30);
        viewmarks.setBounds(100, 100, 400, 30);

        enter_mark.addActionListener(e -> {
            em.entermarks_frame();
        });
        failurButton.addActionListener(e -> {
            f.failure_frame();
        });
        viewmarks.addActionListener(e -> {
            vm.viewmarks_frame();
        });

        teacherportal_frame.add(actiontextJLabel);
        teacherportal_frame.add(enter_mark);
        teacherportal_frame.add(failurButton);
        teacherportal_frame.add(viewmarks);
        teacherportal_frame.setVisible(true);
    }
}
```

## Viewmarks.java

```
package package1;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.;

import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTable;

public class viewmarks {
    public void viewmarks_frame(){
        try {
            teacherlogin tl = new teacherlogin();
            Class.forName("com.mysql.cj.jdbc.Driver");
            String url = "jdbc:mysql://127.0.0.1:3306/student_login";
            String user = "root";
            String password1 = "";
            System.out.println("before connection");
            Connection connection = DriverManager.getConnection(url, user, password1);
            System.out.println("after connection");
            if (connection != null) {
                System.out.println("Connected to the database.");
                Statement st = connection.createStatement();
                String checkfailure = "SELECT FROM studentmark" ;
                ResultSet cf = st.executeQuery(checkfailure);
                JTable table = new JTable(tl.buildTableModel(cf));
                JOptionPane.showMessageDialog(null, new JScrollPane(table));
                connection.close();
            } else {

            }
        } catch (ClassNotFoundException | SQLException ex) {
            System.err.println("Error connecting to the database: " + ex.getMessage());
        }
    }
}
```

## Sql:

```
-----
-- Host:                127.0.0.1
-- Server version:      10.4.32-MariaDB - mariadb.org binary distribution
-- Server OS:          Win64
-- HeidiSQL Version:    12.6.0.6765
-----

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET NAMES utf8 */;
/*!50503 SET NAMES utf8mb4 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

-- Dumping database structure for student_login
CREATE DATABASE IF NOT EXISTS `student_login` /*!40100 DEFAULT CHARACTER SET
utf8mb4 COLLATE utf8mb4_general_ci */;
USE `student_login`;

-- Dumping structure for table student_login.departments
CREATE TABLE IF NOT EXISTS `departments` (
  `department_id` int(11) NOT NULL,
  `department_name` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`department_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Data exporting was unselected.

-- Dumping structure for table student_login.employees
CREATE TABLE IF NOT EXISTS `employees` (
  `employee_id` int(11) NOT NULL,
  `first_name` varchar(50) DEFAULT NULL,
  `last_name` varchar(50) DEFAULT NULL,
  `department_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`employee_id`),
  KEY `fk_department` (`department_id`),
  CONSTRAINT `fk_department` FOREIGN KEY (`department_id`) REFERENCES `departments`
```

```
(`department_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

-- Data exporting was unselected.

-- Dumping structure for table student\_login.student

```
CREATE TABLE IF NOT EXISTS `student` (  
  `StudentID` int(11) NOT NULL,  
  `Name` varchar(50) DEFAULT NULL,  
  `Email` varchar(100) DEFAULT NULL,  
  `Class` varchar(20) DEFAULT NULL,  
  `Section` varchar(10) DEFAULT NULL,  
  PRIMARY KEY (`StudentID`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

-- Data exporting was unselected.

-- Dumping structure for table student\_login.studentmark

```
CREATE TABLE IF NOT EXISTS `studentmark` (  
  `MarkID` int(11) NOT NULL AUTO_INCREMENT,  
  `StudentID` int(11) DEFAULT NULL,  
  `Subject1Marks` int(11) DEFAULT NULL,  
  `Subject2Marks` int(11) DEFAULT NULL,  
  `Subject3Marks` int(11) DEFAULT NULL,  
  `Subject4Marks` int(11) DEFAULT NULL,  
  `Subject5Marks` int(11) DEFAULT NULL,  
  `AvgMarks` decimal(5,2) DEFAULT NULL,  
  PRIMARY KEY (`MarkID`),  
  KEY `StudentID` (`StudentID`),  
  CONSTRAINT `studentmark_ibfk_1` FOREIGN KEY (`StudentID`) REFERENCES `student`  
  (`StudentID`)  
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_general_ci;
```

-- Data exporting was unselected.

-- Dumping structure for table student\_login.teacherportal

```
CREATE TABLE IF NOT EXISTS `teacherportal` (  
  `TeacherID` int(11) NOT NULL,  
  `Name` varchar(50) DEFAULT NULL,  
  `Email` varchar(100) DEFAULT NULL,  
  `Department` varchar(50) DEFAULT NULL,  
  PRIMARY KEY (`TeacherID`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

```
-- Data exporting was unselected.
```

```
-- Dumping structure for trigger student_login.CalculateAvgMarks
```

```
SET @OLDTMP_SQL_MODE=@@SQL_MODE,  
SQL_MODE='NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION';  
DELIMITER //
```

```
CREATE TRIGGER CalculateAvgMarks
```

```
BEFORE INSERT ON studentmark
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE avg DECIMAL(5, 2);
```

```
    -- Calculate average marks
```

```
    SET avg = (NEW.Subject1Marks + NEW.Subject2Marks + NEW.Subject3Marks +  
NEW.Subject4Marks + NEW.Subject5Marks) / 5;
```

```
    -- Set the calculated average marks value in the new row
```

```
    SET NEW.avgmarks = avg;
```

```
END//
```

```
DELIMITER ;
```

```
SET SQL_MODE=@OLDTMP_SQL_MODE;
```

```
/*!40103 SET TIME_ZONE=IFNULL(@OLD_TIME_ZONE, 'system') /;
```

```
/*!40101 SET SQL_MODE=IFNULL(@OLD_SQL_MODE, "") /;
```

```
/*!40014 SET FOREIGN_KEY_CHECKS=IFNULL(@OLD_FOREIGN_KEY_CHECKS, 1) /;
```

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT /;
```

```
/*!40111 SET SQL_NOTES=IFNULL(@OLD_SQL_NOTES, 1) /;
```



## 5)Results and Discussion:

The Student Portal has been successfully developed and tested, achieving the set objectives. The student portal provides an efficient means for students to access their academic details, while the teacher portal streamlines the process of entering and managing student marks.

### 1. Performance Analysis:

Compare the performance of students across different subjects.Highlight the top-performing students and subjects with the highest and lowest average scores.

### 2. Assessment Analysis:

Evaluate the difficulty level of assessments by analyzing the average marks obtained in each assessment.Identify any assessments with unusually high or low average scores and discuss possible reasons.

## Discussion :

### 1. Factors Influencing Performance:

Analyze factors that may have influenced student performance, such as teaching methods, study habits, socio-economic background, etc.Discuss how these factors may have contributed to the observed results.

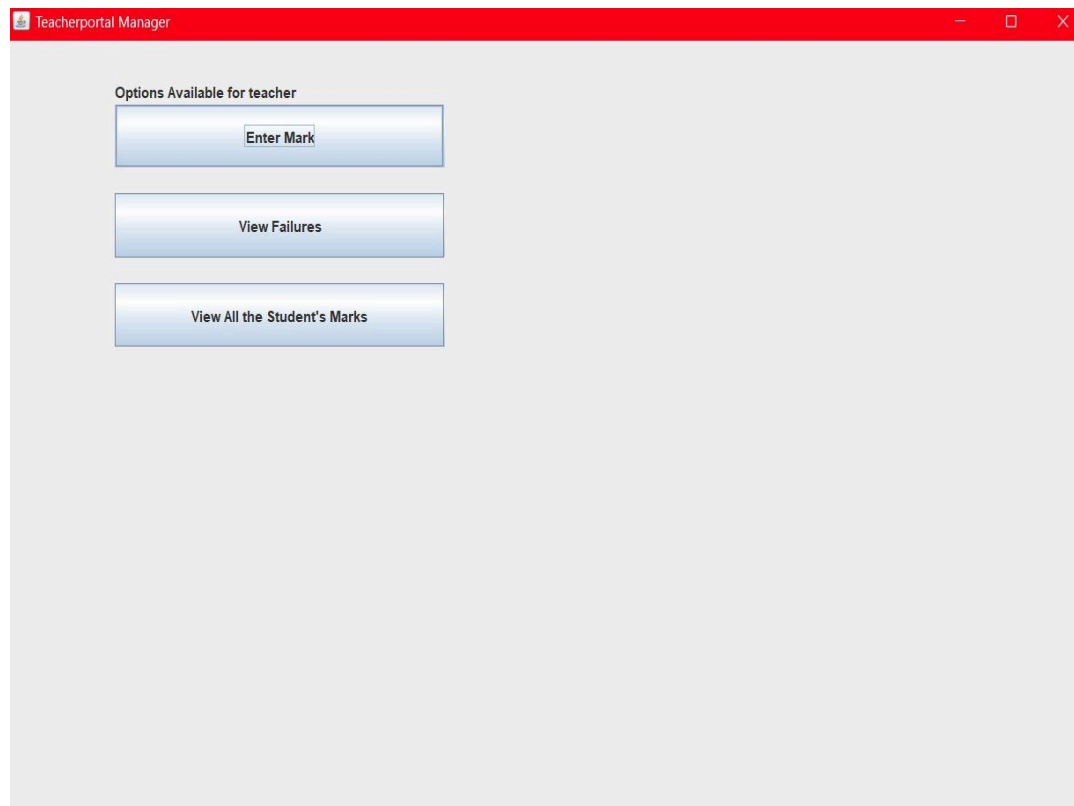
### 2. Recommendations:

Based on the findings, suggest strategies for improving student performance.Propose changes to the curriculum, teaching methods, assessment practices, etc., to enhance learning outcomes.

### 3.Future Research Directions:

Suggest areas for further research to deepen understanding of student performance.Highlight topics that warrant investigation based on the findings of the current study.

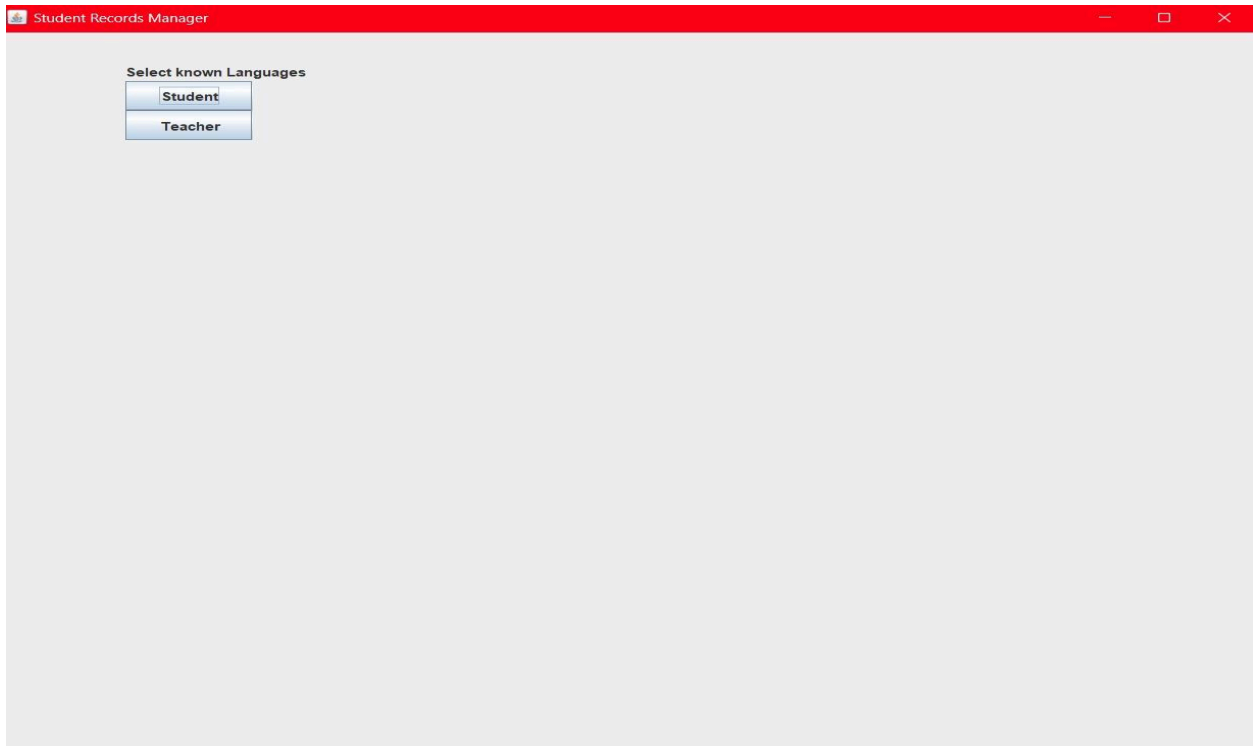
## Teacher Module:



## Student Module:



## Selection Module:



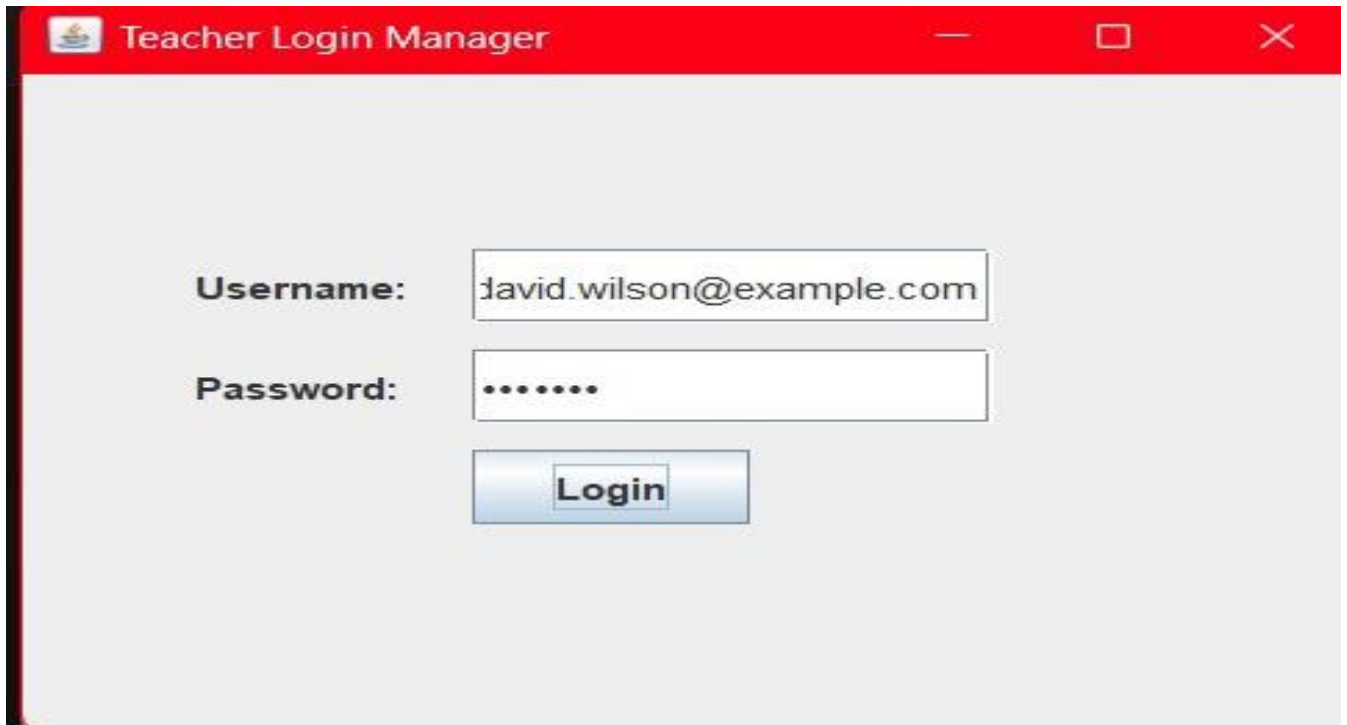
The screenshot shows a window titled "Student Records Manager" with a red title bar. Inside the window, there is a section titled "Select known Languages" with two buttons: "Student" and "Teacher".

Select known Languages

Student

Teacher

## Teacher Login Module:



The screenshot shows a window titled "Teacher Login Manager" with a red title bar. Inside the window, there is a login form with the following fields and buttons:

- Username:** A text input field containing the email address "david.wilson@example.com".
- Password:** A password input field with seven dots representing the masked password.
- Login:** A blue button with the text "Login".

## Student Mark Enter Module:

Enter Marks Manager

Student ID:

Subject 1:

Subject 2:


Subject 3:

Subject 4:

Subject 5:

## View Marks Module:

Message



MarkID	StudentID	Subject...	Subject...	Subject3...	Subject...	Subject5...	AvgMarks
12	2207011...	15	90	78	12	18	42.60
13	2207011...	15	90	78	12	18	42.60
14	2207011...	90	90	90	90	90	90.00
15	2207011...	90	90	90	90	90	90.00
48	2207011...	10	20	30	40	50	30.00

## 6)Conclusion

The Student Portal project illustrates the potential of using modern software technologies to streamline educational processes and enhance the overall learning experience. By creating a platform that addresses the needs of both students and teachers, the project has demonstrated how technology can bridge gaps in communication and data management within educational institutions.

The success of this project lays a strong foundation for future developments. As technology continues to evolve, further enhancements can be made to extend the functionality of the Student Portal, ensuring it remains a valuable tool for educational administration and academic success.

Overall, this project highlights the importance of integrating technology in education, not only to improve administrative efficiency but also to foster a more supportive and transparent educational environment.

## 7)References:

1)<https://stackoverflow.com/questions/13970978/sql-resultset-in-java>

2)<https://stackoverflow.com/questions/419021/how-does-javas-preparedstatement-work#:~:text=If%20you%20are%20calling%20built,SQL%20server%20then%20use%20CallableStatement.>

