**Java Mini Project: School management system**

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll no.** | **Sap id** |
| **Nishant Baruah** | **A053** | **70022100516** |
| **Tanmay Sahasrabudhe** | **A031** | **70022100402** |
| **Abhay Rathore** | **A043** | **70022100491** |

**Problem statement –**

Developing a student management system to simplify the administrative tasks and improve communication between students, teachers, and parents.

**Project description –**

The student management system project aims to create a comprehensive online platform that can streamline the administrative tasks of managing student records, academic performance and other related data. The current manual system for managing student records and related data is time-consuming, inefficient, and prone to errors. Hence, a digital solution is needed to improve accuracy, save time, and enhance communication.

The student management system should be accessible to all stakeholders, including students, teachers, parents, and administrative staff. The system should be easy to use and navigate, allowing students to access their records like report card, personal data etc.

The administrator has the access to edit , insert, or update the student data according to various needs time to time.

The development of a student management system project will improve the overall efficiency of managing student records and academic performance while enhancing communication between stakeholders.

**Algorithm used–**

* We have created a login page where administrator will be able to access and logged in.
* The next page of applications contains two buttons where admin can either access the student module section or the report card formation at a time.
* The student module contains the options or buttons by which the admin can access the student data and create, delete, or update the existing data of any student.
* The admin can access the report card section and can create result cards for the students.

**Project modules –**

Project Modules:

1) Admin's Registration

3) Report card Module

4) Students’ Module

**Methods and datasets–**

Login page –

* initComponents(): This method is auto-generated by the NetBeans IDE and initializes all the Swing components.
* jButton1ActionPerformed(): This method is invoked when the "Submit" button is clicked. It retrieves the values of the username and password fields, establishes a connection to a MySQL database, executes a select query on the user\_login table, and compares the retrieved username and password with the entered ones. If the username and password match, it opens the "welcome" page; otherwise, it displays an error message.
* main(): This method is the entry point for the application. It sets the look and feel of the application and creates an instance of the LoginPage class.
* Variables: The code declares several private instance variables that hold references to various Swing components used in the user interface.

Welcome page –

* The jLabel2MouseClicked method is an event handler that is triggered when a user clicks on jLabel2. It creates an instance of the LoginPage class, sets its visibility to true, and disposes the current JFrame object.
* The jButton1ActionPerformed method is another event handler that is triggered when a user clicks on jButton1. It creates an instance of the studentsmodule class, sets its visibility to true, and disposes the current JFrame object.
* The jButton4ActionPerformed method is yet another event handler that is triggered when a user clicks on jButton4. It creates an instance of the reportcard class, sets its visibility to true, and disposes the current JFrame object.
* The main method is the entry point of the application. It creates a new instance of the welcome class and sets its visibility to true by invoking the setVisible method on the object.
* Overall, the methods are used to create and manage graphical user interface components in the application, and handle user events. They use the Swing framework to create and manipulate GUI components, such as JLabel, JButton, and JFrame.

Students module page –

* jButton2ActionPerformed: This method is called when jButton2 is clicked. It creates an instance of the Addstu class, sets its visibility to true, and disposes the current frame.
* jButton3ActionPerformed: This method is called when jButton3 is clicked. It creates an instance of the editstu class, sets its visibility to true, and disposes the current frame.
* jButton1ActionPerformed: This method is called when jButton1 is clicked. It creates an instance of the SDStud class, sets its visibility to true, and disposes the current frame.
* jLabel2MouseClicked: This method is called when jLabel2 is clicked. It creates an instance of the welcome class, sets its visibility to true, and disposes the current frame.
* main: This is the main method that is called when the program starts. It creates an instance of the studentsmodule class, sets its visibility to true, and runs it on the event dispatch thread.

Add and edit student page –

* initComponents(): This method is responsible for initializing all the Swing components that make up the registration form. It sets up the labels, text fields, and buttons that are used to input student information.
* Addstu(): This is the constructor method of the Addstu class. It calls the initComponents() method to create the user interface.
* jButton1ActionPerformed(): This method is an event handler that is called when the "Submit" button is clicked. It retrieves the values entered in the text fields and performs validation before saving the student's information to a database.
* jLabel11MouseClicked(): This method is an event handler that is called when the "BACK" label is clicked. It takes the user back to the previous screen.
* Other methods used in this code are generated by the NetBeans IDE and are used for layout management of the Swing components. They include setLayout(), setDefaultCloseOperation(), and pack().

Student search delete page –

* initComponents(): This method is generated by the NetBeans IDE and initializes all the components of the GUI such as buttons, text fields, and labels.
* setDefaultCloseOperation(): This method sets the behavior of the JFrame when the user clicks the close button. In this case, the application exits when the JFrame is closed.
* setText(): This method sets the text of the label to the specified string.
* addActionListener(): This method adds an action listener to a button so that it can perform an action when clicked.
* try-catch block: This is used to handle exceptions that may occur when the program is running. In this code, the try block is used to establish a connection to a MySQL database and retrieve data from it. If an exception occurs, the catch block displays an error message.
* JTable and DefaultTableModel: These are used to display the data retrieved from the database in a table format. The DefaultTableModel is used to define the structure of the table, and the JTable is used to display the data.
* PreparedStatement: This is used to execute SQL statements that may contain parameters. In this code, it is used to execute a SELECT statement to retrieve data from the database.
* ResultSet: This is used to store the results of a SQL query. In this code, it is used to store the data retrieved from the database.
* setRowCount(): This method is used to set the number of rows in the table.
* addRow(): This method is used to add a row of data to the table.
* getText(): This method is used to retrieve the text entered by the user in a text field.
* setFont(): This method is used to set the font of the label.

Report card page –

The GUI components used in the report card frame are:

* JScrollPane: Provides a scrollable view of the text component.
* JTextPane: A text component that supports styled text with embedded images and components.
* JLabel: A non-editable text component that displays a string or image.
* JTextField: A text component that allows the editing of a single line of text.
* JButton: A component that can be clicked to trigger an action.
* The layout of the components is set using the GroupLayout manager. The GUI contains several labels and text fields for entering student details, such as ID, roll number, name, and class, and marks for physics, math, and chemistry. A button labeled "Submit" is also present.
* The program also defines action listeners to handle user interactions with the GUI. When the Submit button is clicked, the program handles the event by invoking the actionPerformed() method of the associated listener. Similarly, when the user clicks on the "Back" label, the program invokes the mouseClicked() method of the associated listener.

**Technologies used in the project –**

1) MYSQL DBMS (using xampp server)

2) JSWINGS

3) APACHE NETBEANS IDE 17

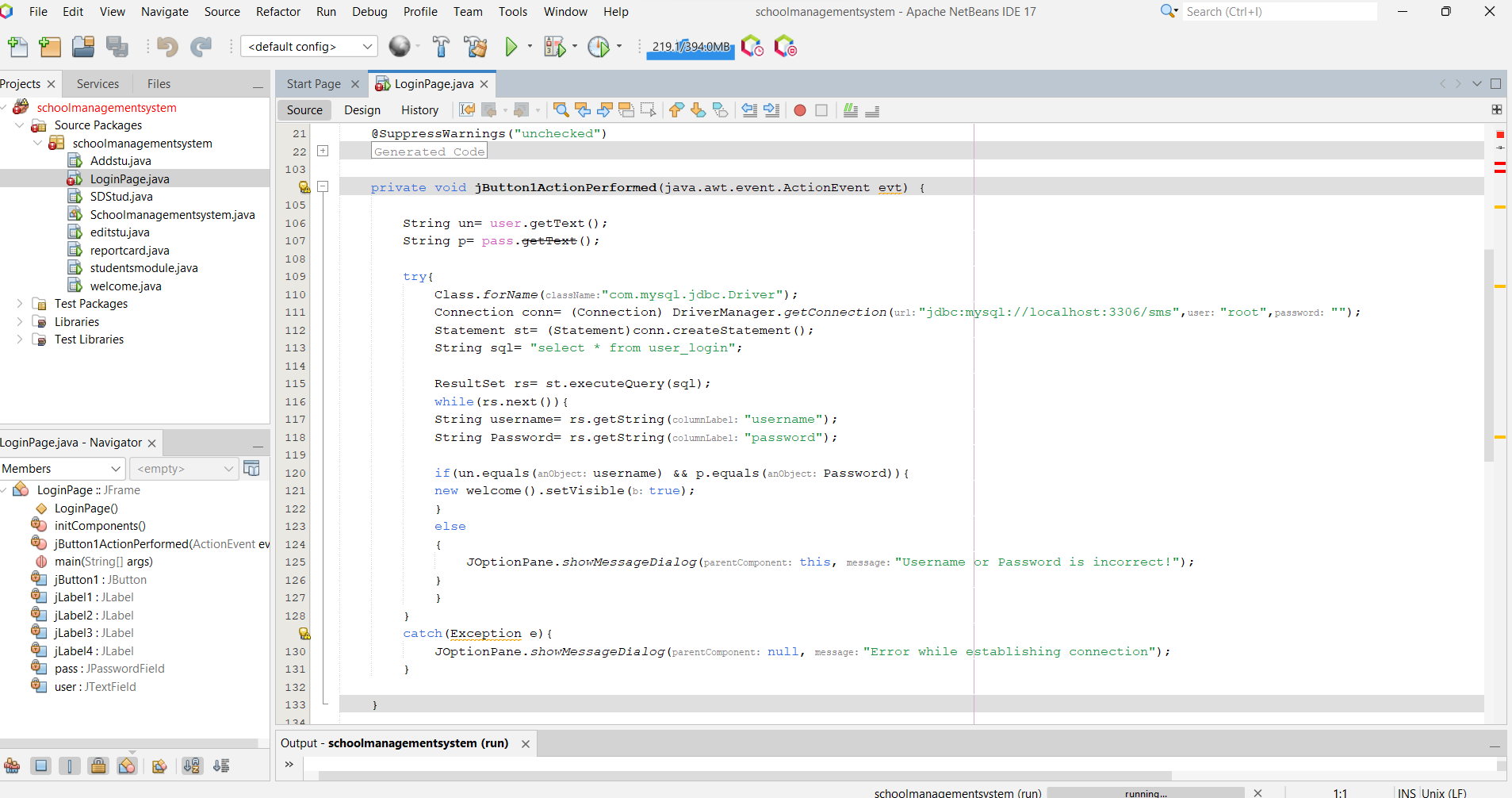
**Advantage–**

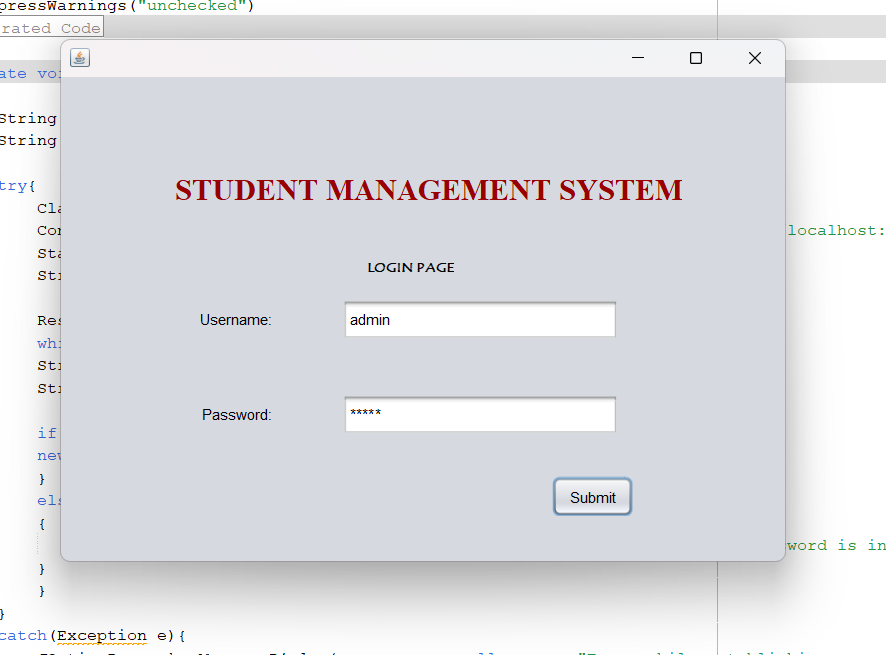
* The admin can access the portal and can manage the students data very easily and with efficient manner such as id, name, parents name, phone number, roll number, address, class etc.
* The admin can also update the data in the application
* Report cards can be formed for the students which has automatic grading system in the project
* As being a mini project , the application can perform crud operation on the admin side.

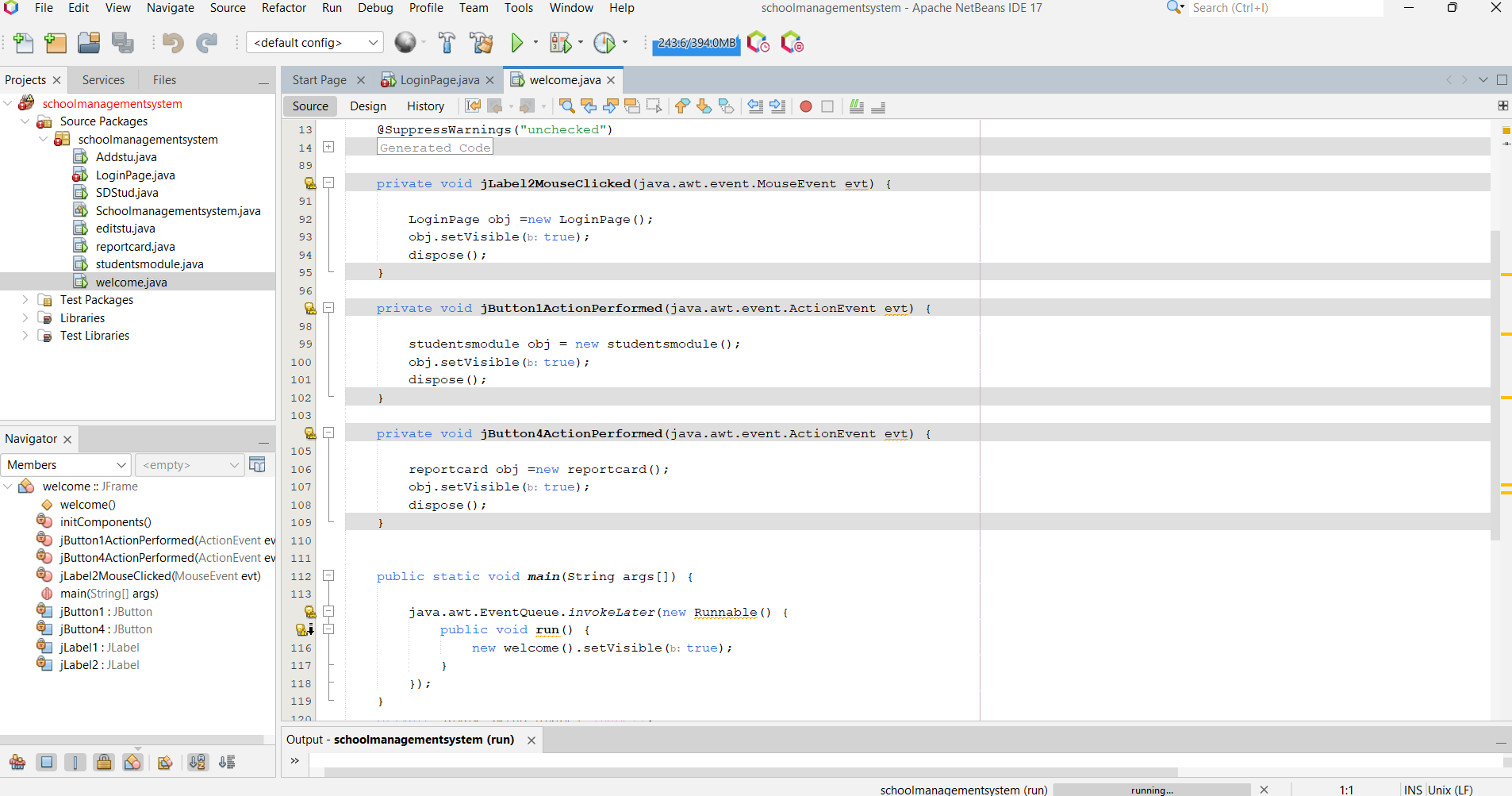
**Limitations–**

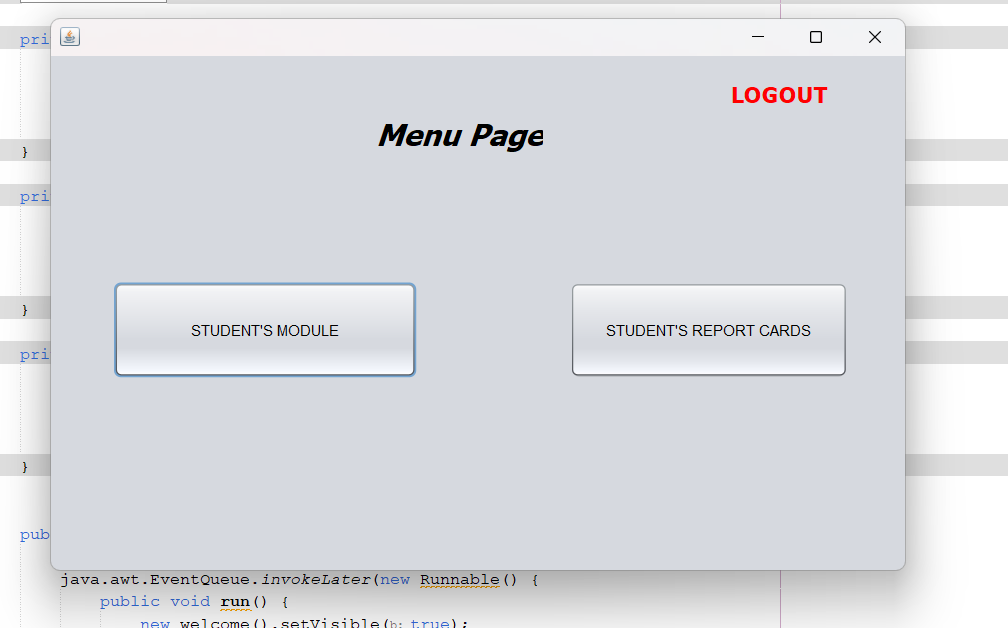
Our project only deals with the administrations section of the management so we can add more sections like teachers , student login , subject module etc.

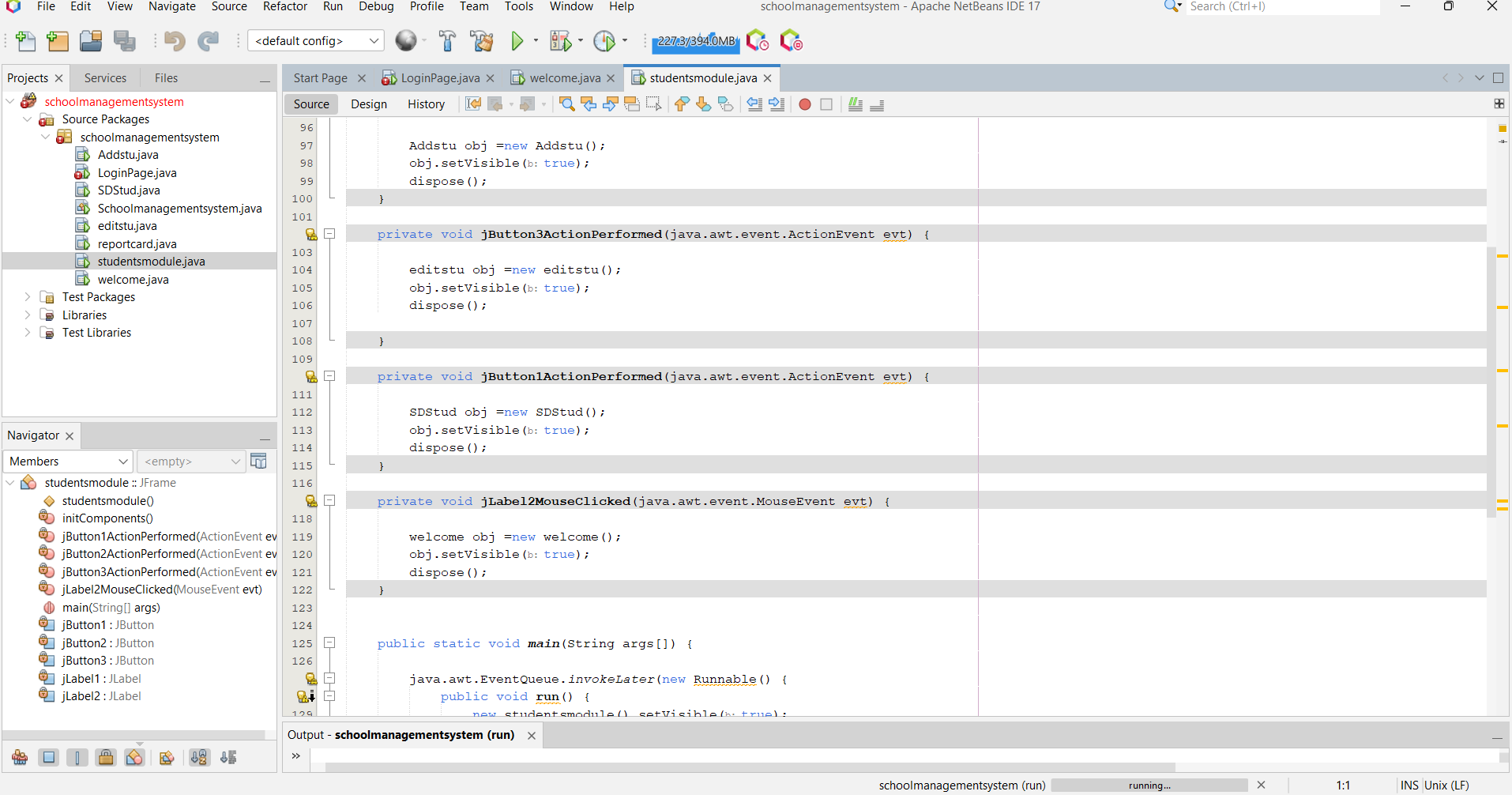
**Code and snapshots of the project –**

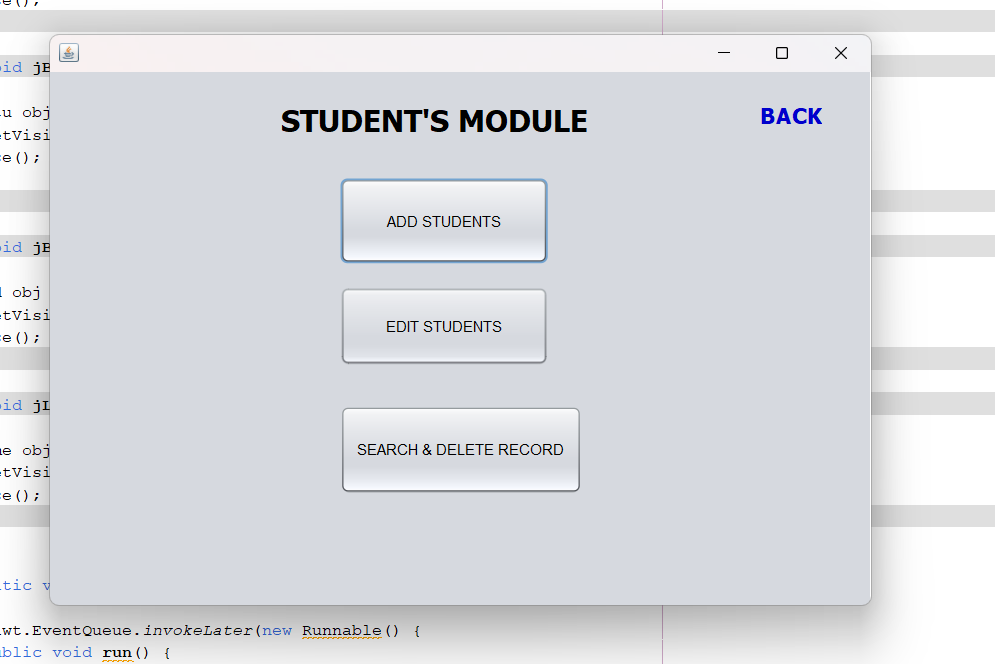


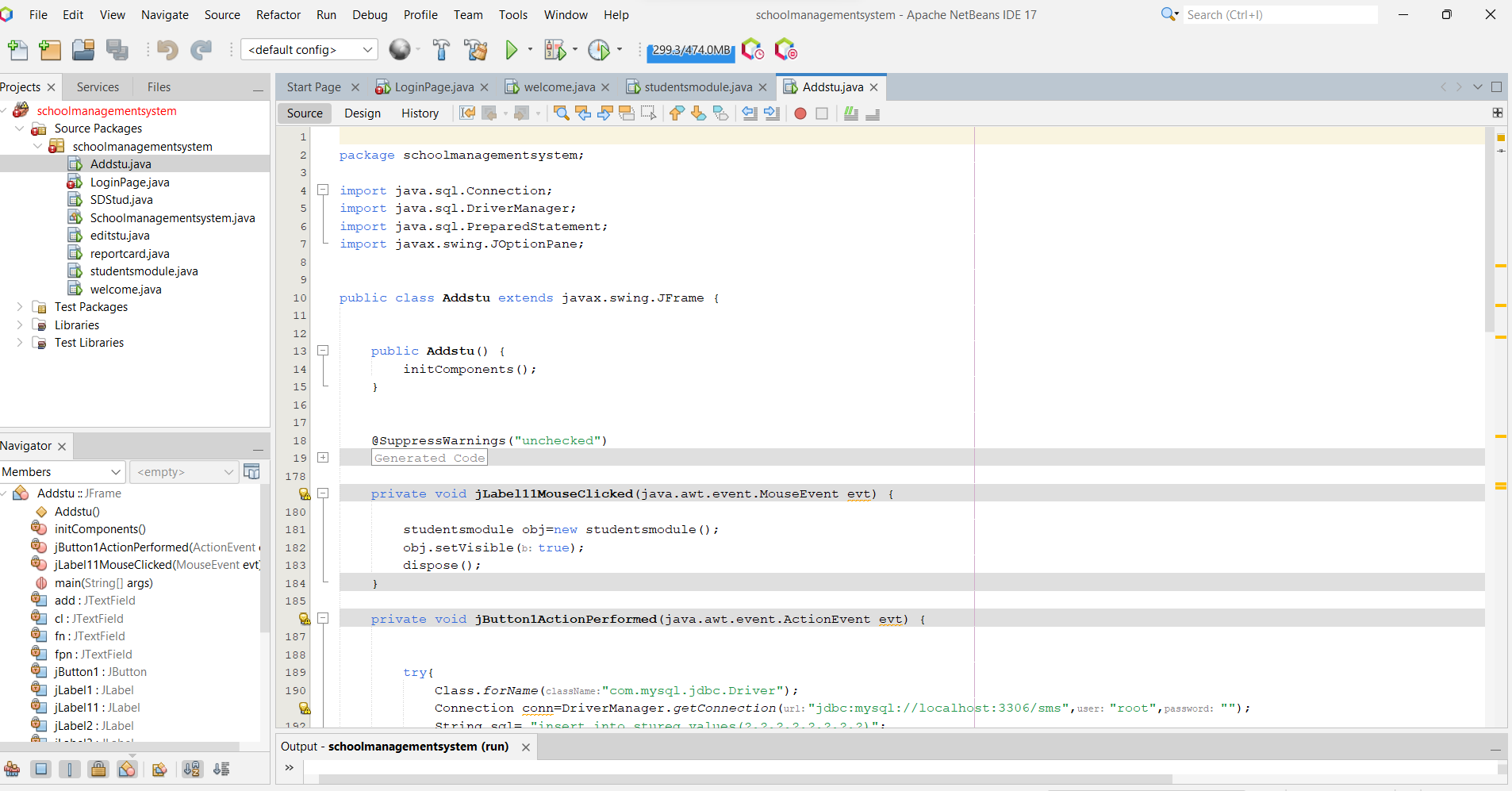


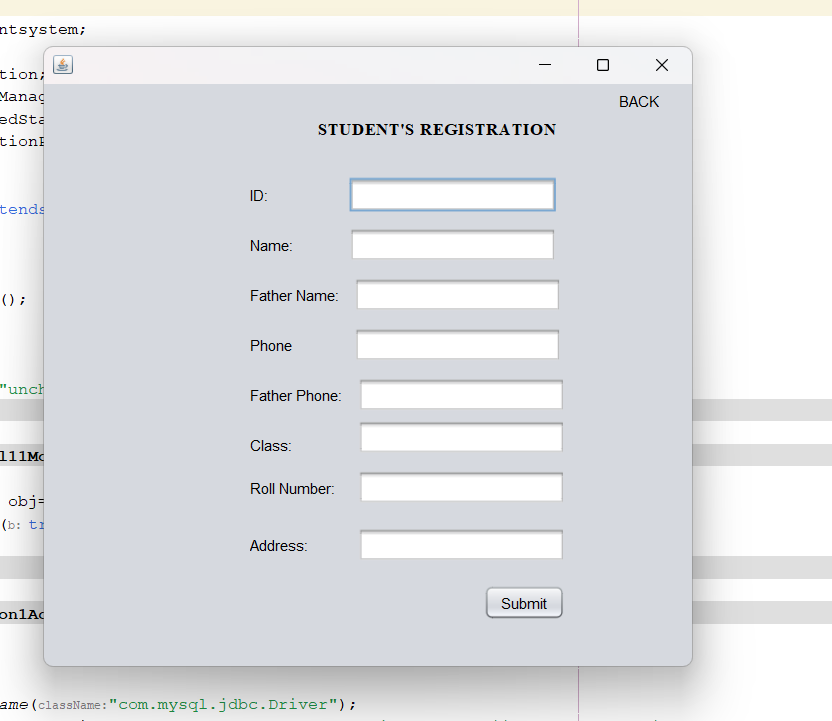


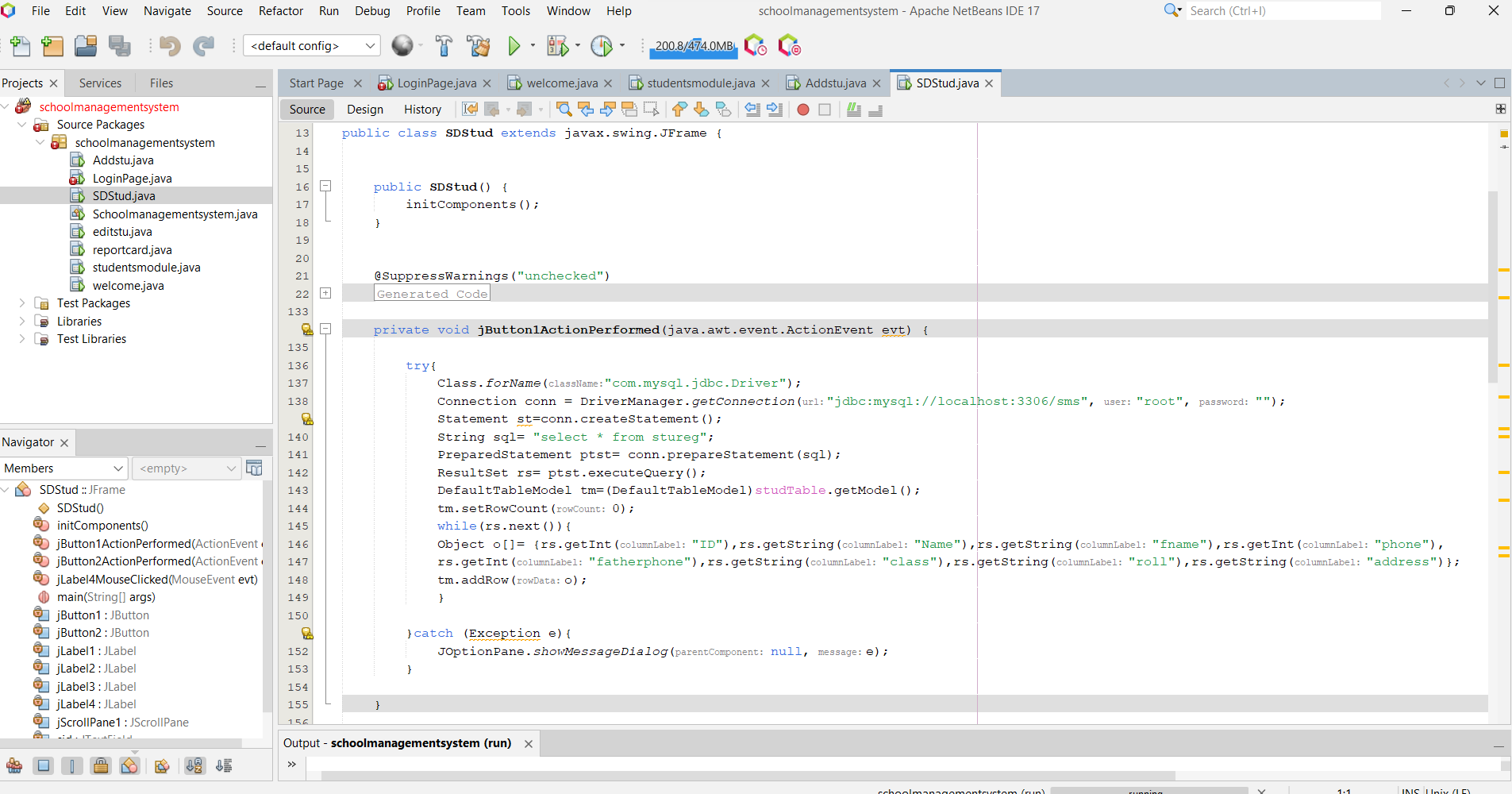


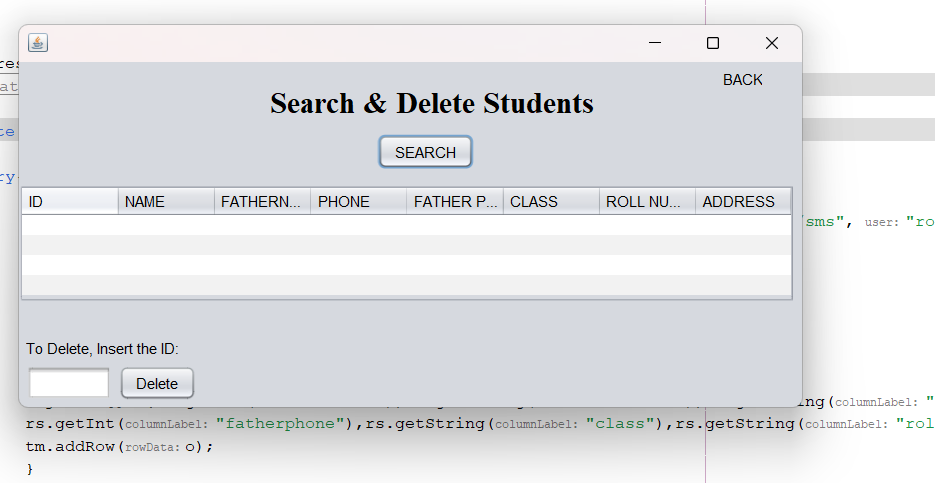




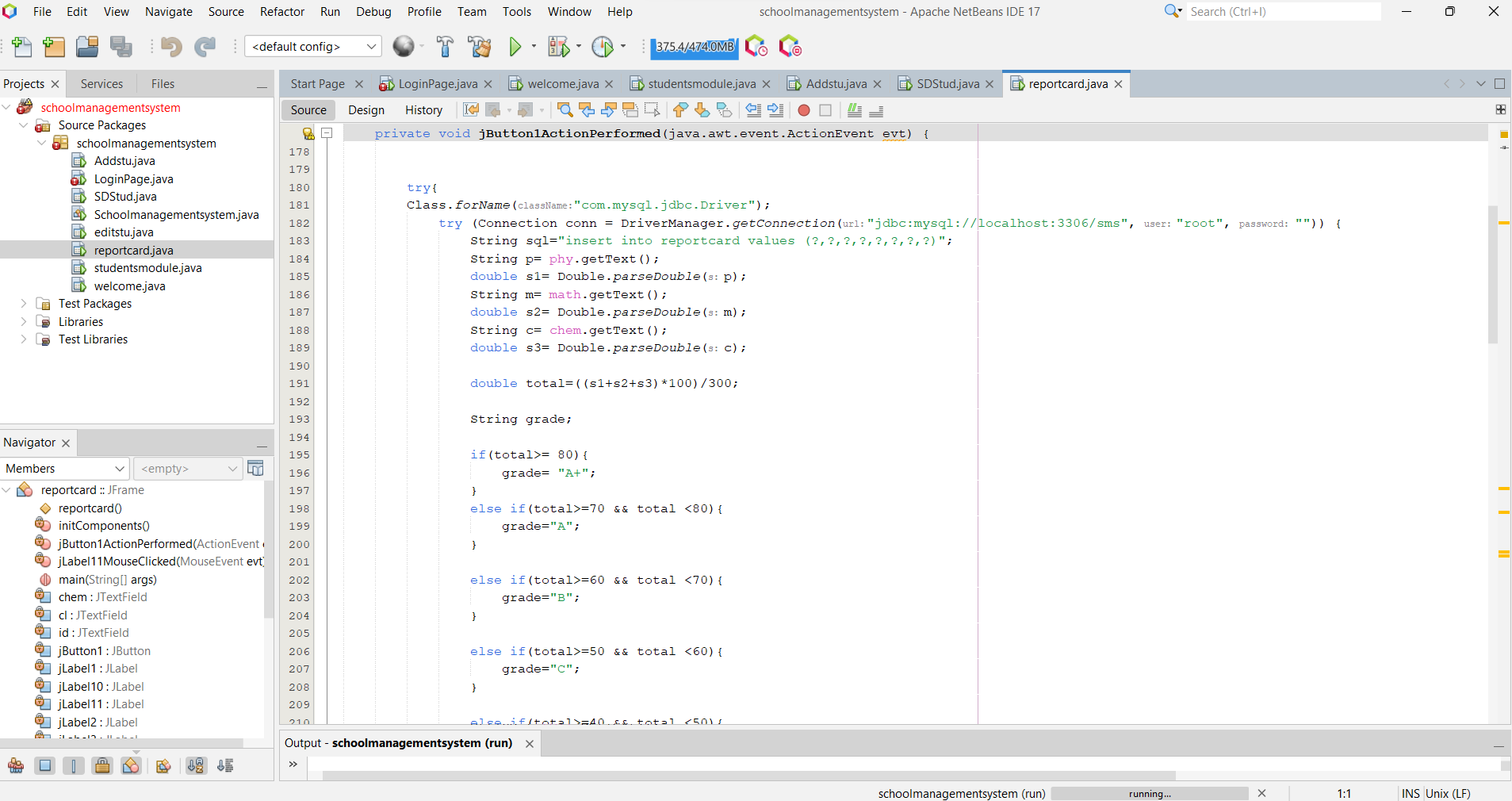


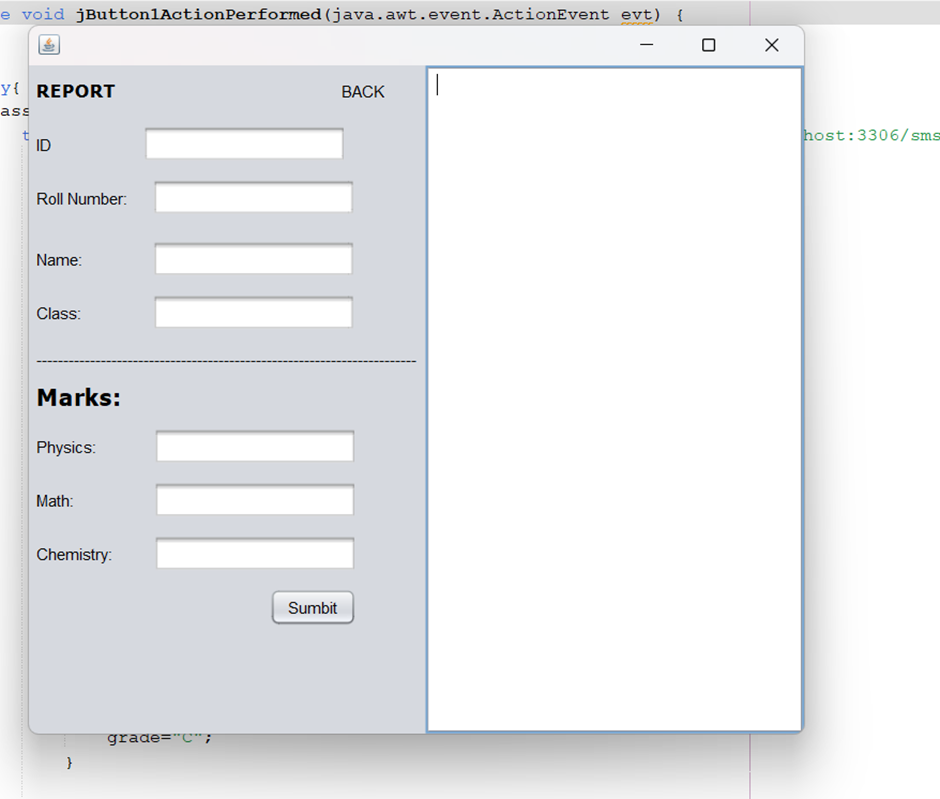


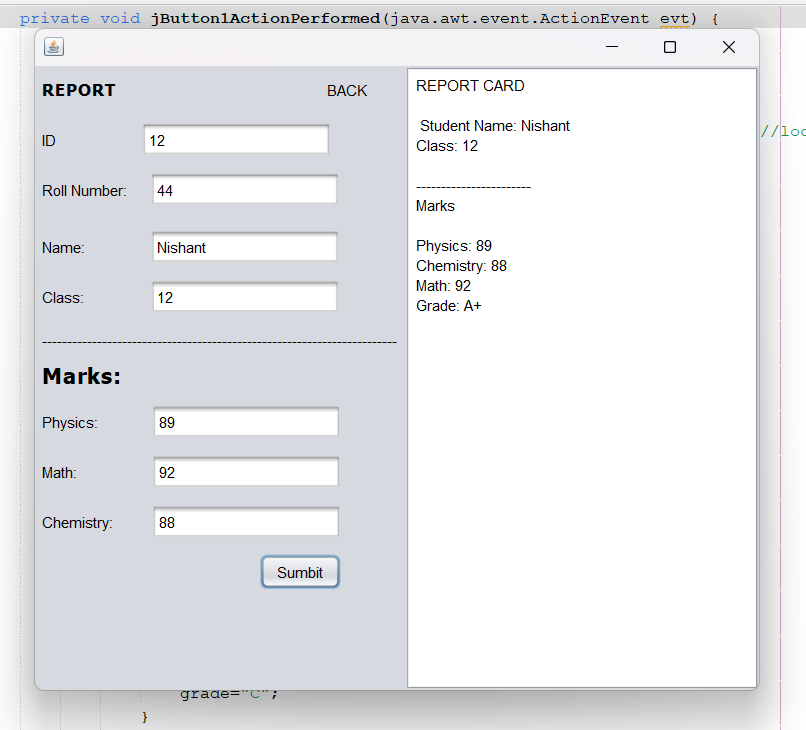




Report card section –







**Future scope –**

* Advanced Analytics: Incorporating advanced analytics into the student management system can provide a better understanding of student performance, learning patterns, and help identify areas of improvement.
* Integration with Social Media: Integrating the student management system with social media can provide a better platform for communication between students, teachers, and parents. This could include features like discussion forums, chat rooms, and group projects.
* Automated Attendance Tracking: Implementing automated attendance tracking can save time and reduce errors by eliminating the need for manual attendance taking.
* Online fee payment module

**References –**

The idea and methods for the project has been taken after researching and going through various websites and by discussion among all our group members.

There is no specific reference as we have gone through ample of projects to get the basic idea and implement the methods and code.