

STUDENT DETAILS MANAGEMENT SYSTEM REPORT

INDEX

- 1.) Introduction
- 2.) Objectives
- 3.) System Analysis
 - 3.1 Problem Statement
 - 3.2 Objective of the System
- 4.) Feasibility Analysis
 - 4.1 Technical Feasibility
 - 4.2 Economical Feasibility
 - 4.3 Operational Feasibility
- 5.) Working Screenshots
- 6.) Design and Coding Screenshots
- 7.) Future Scope
- 8.) Bibliography/Refernce/Glossary

INTRODUCTION

The project “STUDENT DETAIL MANAGEMENT SYSTEM” is a Java JSP and MySQL Project which runs on the NetBeans. We have developed this java JSP and MySQL Project on “STUDENT DETAIL MANAGEMENT” for automating the process of Student Information System. The main feature of this project is the teacher can search, add, delete student details after login.

The database used in this project is MySQL hence all the data generate in this project is securely added to MySQL and retrieve from MySQL database.

OBJECTIVE

The main objective of the project is to ease the work of the teachers by adding their student details from where they can access the details of any student and any time. This project is basically a record project in which teacher add the record of their students to access these records when it needed. Teacher can login in this Java JSP project by the default password “teacher” and then he/she can check the list of all students, can add details of more student, can delete details of students and can search the details of any student by there respective Roll numbers.

SYSTEM ANALYSIS

SYSTEM ANALYSIS is the process of observing systems for troubleshooting or development purposes. It is applied to information technology where computer based systems require defined analysis according to their makeup and design.

It is a method of figuring out the basic elements of a project and deciding how to combine them in the best way to solve a problem .

It includes the FEASIBILITY STUDY of the project.

A feasibility study evaluates the project's potential for success. The purpose behind a **project feasibility study** is to know the different variables involved with your business venture and how it will be accepted on the open market along with who will be the target audience . Various types of feasibility that are commonly considered include technical feasibility, operational feasibility, and economic feasibility.

Acc. to Technical Feasibility Study, this application is feasible (or within the bound of possibility) as it requires basic technical skills and not the high upgraded software or tools.

Acc. to Operational Feasibility Study, this application is feasible to much extend as it is up to user's requirement and satisfaction.

Acc. To Economic Feasibility Study, this application is feasible as the development cost is not high & is user friendly, it can produce gains to a great extend.

FEASIBILITY STUDY

Technical Feasibility:

“STUDENT DETAILS MANAGEMENT SYSTEM” application requires very basic knowledge of handling the any software.

User only have to enter the details related to student personal details and login details.

Operational Feasibility:

“STUDENT DETAILS MANAGEMENT SYSTEM” application solve the user problem of maintaining a record of students. User can check the student details anytime with the help of this project.

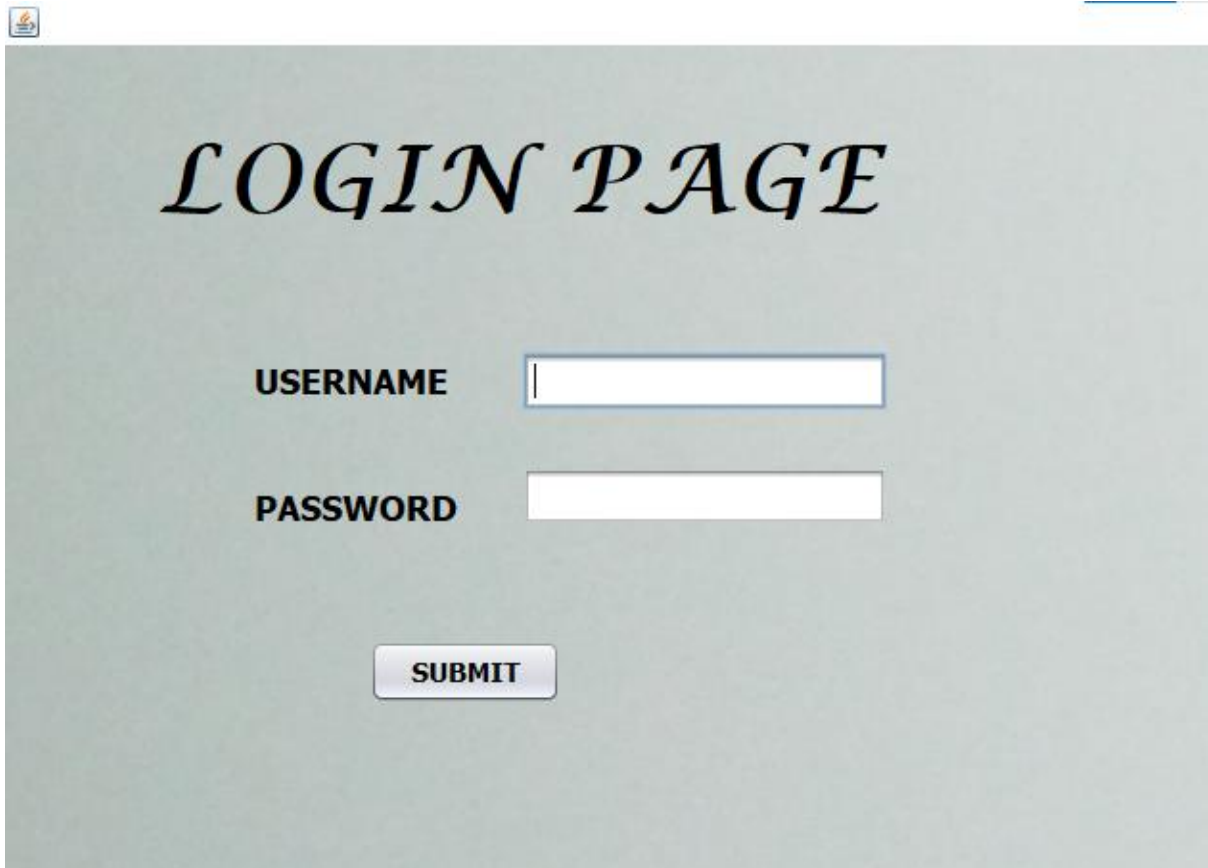
Economic Feasibility:

The cost of Maintaining the “STUDENT DETAILS MANAGEMENT SYSTEM” application is very low. There is no requirement of Internet connection to use the application.

All the data will save on user device.

It is user friendly. User don't need to understand it deeply.

WORKING AND SCREENSHOTS OF PROJECT



The screenshot shows a web browser window with a light gray background. At the top center, the text *LOGIN PAGE* is displayed in a large, black, serif font. Below this, there are two input fields. The first field is labeled **USERNAME** in a bold, black, sans-serif font, and it contains a single vertical line cursor. The second field is labeled **PASSWORD** in a bold, black, sans-serif font, and it is empty. Below these fields is a button labeled **SUBMIT** in a bold, black, sans-serif font, with a light gray background and a thin black border. The browser window has a small icon in the top left corner and a title bar in the top right corner.

This is the first page of the project which is login page we have to enter the username and password to login.



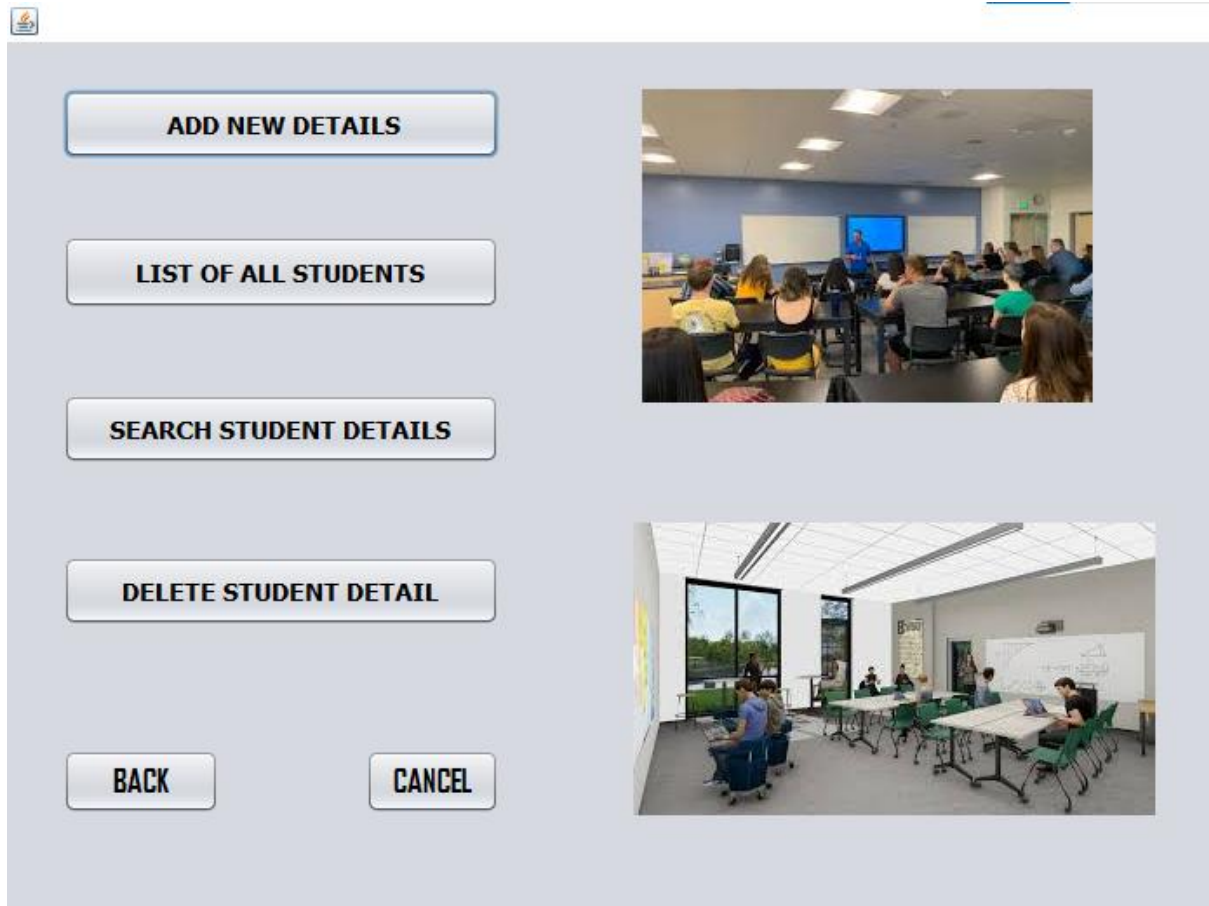
LOGIN PAGE

USERNAME

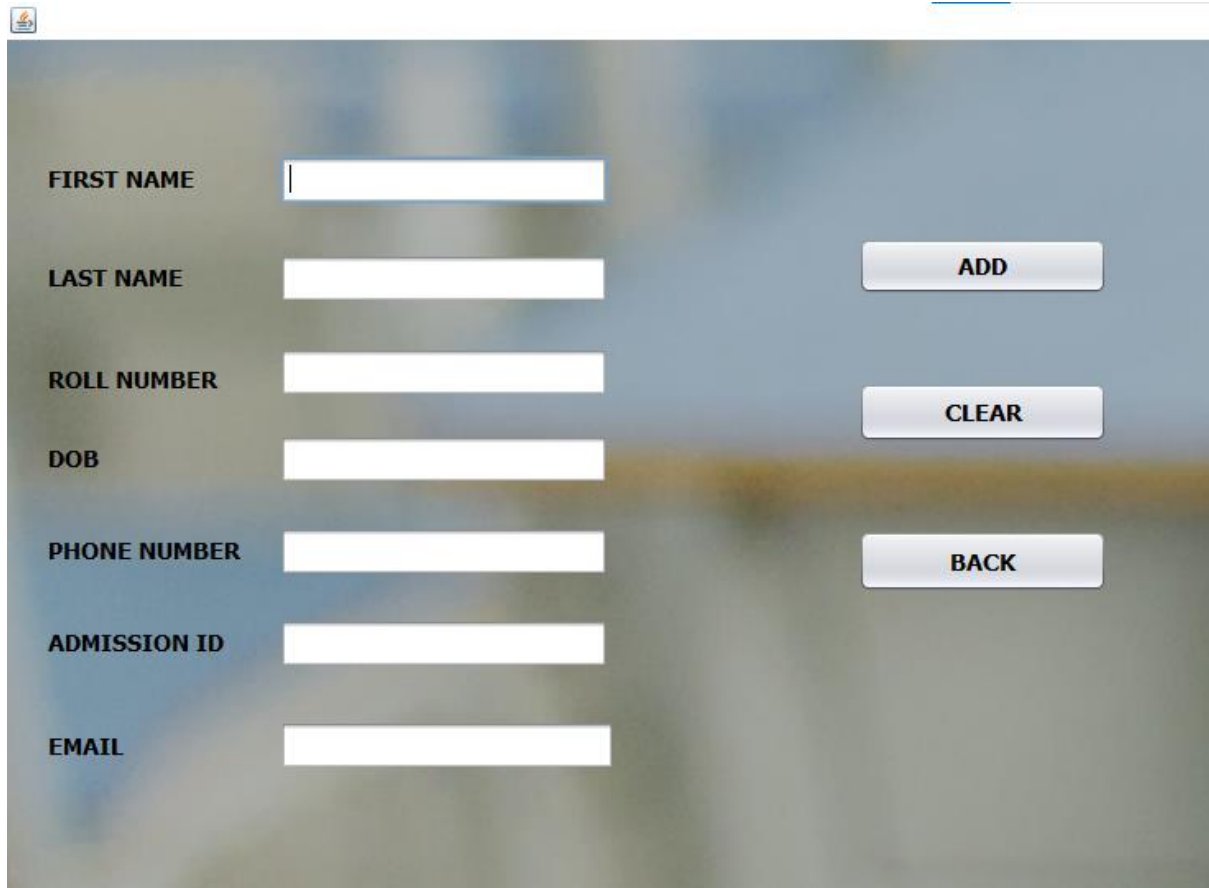
PASSWORD

SUBMIT

The default Password of the project is “teacher” you have to enter the username and password to login.



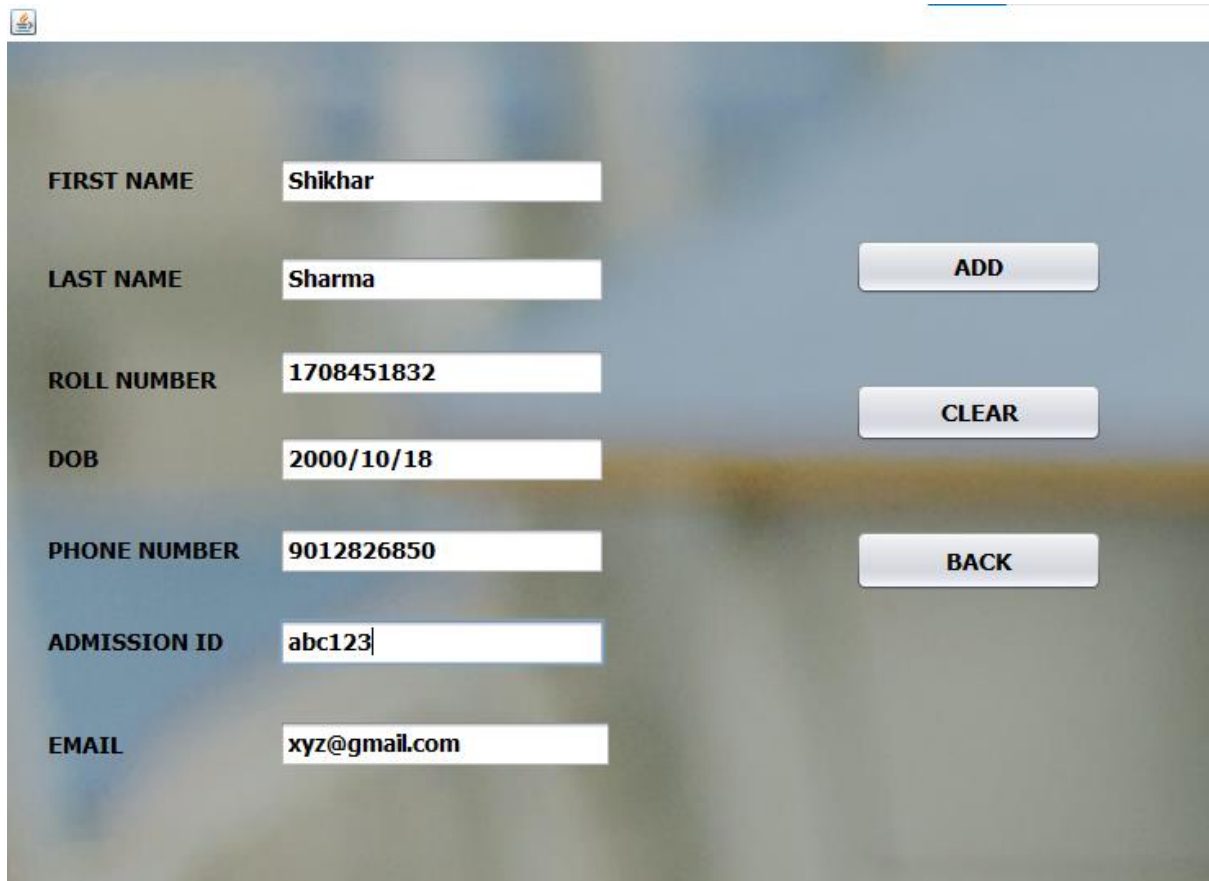
After login this page of the project opens. In which you have four options with the help of which you can add, search and delete the student deatails and check the list of all student whose data is added successfully.



A screenshot of a web application interface for adding student details. The form is displayed on a blurred background. It consists of seven input fields arranged vertically, each preceded by a label: 'FIRST NAME', 'LAST NAME', 'ROLL NUMBER', 'DOB', 'PHONE NUMBER', 'ADMISSION ID', and 'EMAIL'. To the right of the input fields, there are three buttons: 'ADD' (positioned next to the 'LAST NAME' field), 'CLEAR' (positioned next to the 'ROLL NUMBER' field), and 'BACK' (positioned next to the 'PHONE NUMBER' field). The 'ADD' button is a light blue button with a dark blue border. The 'CLEAR' and 'BACK' buttons are light blue buttons with dark blue borders. The 'FIRST NAME' input field has a small cursor visible at the end of the text.

FIRST NAME	<input type="text"/>	
LAST NAME	<input type="text"/>	ADD
ROLL NUMBER	<input type="text"/>	CLEAR
DOB	<input type="text"/>	
PHONE NUMBER	<input type="text"/>	BACK
ADMISSION ID	<input type="text"/>	
EMAIL	<input type="text"/>	

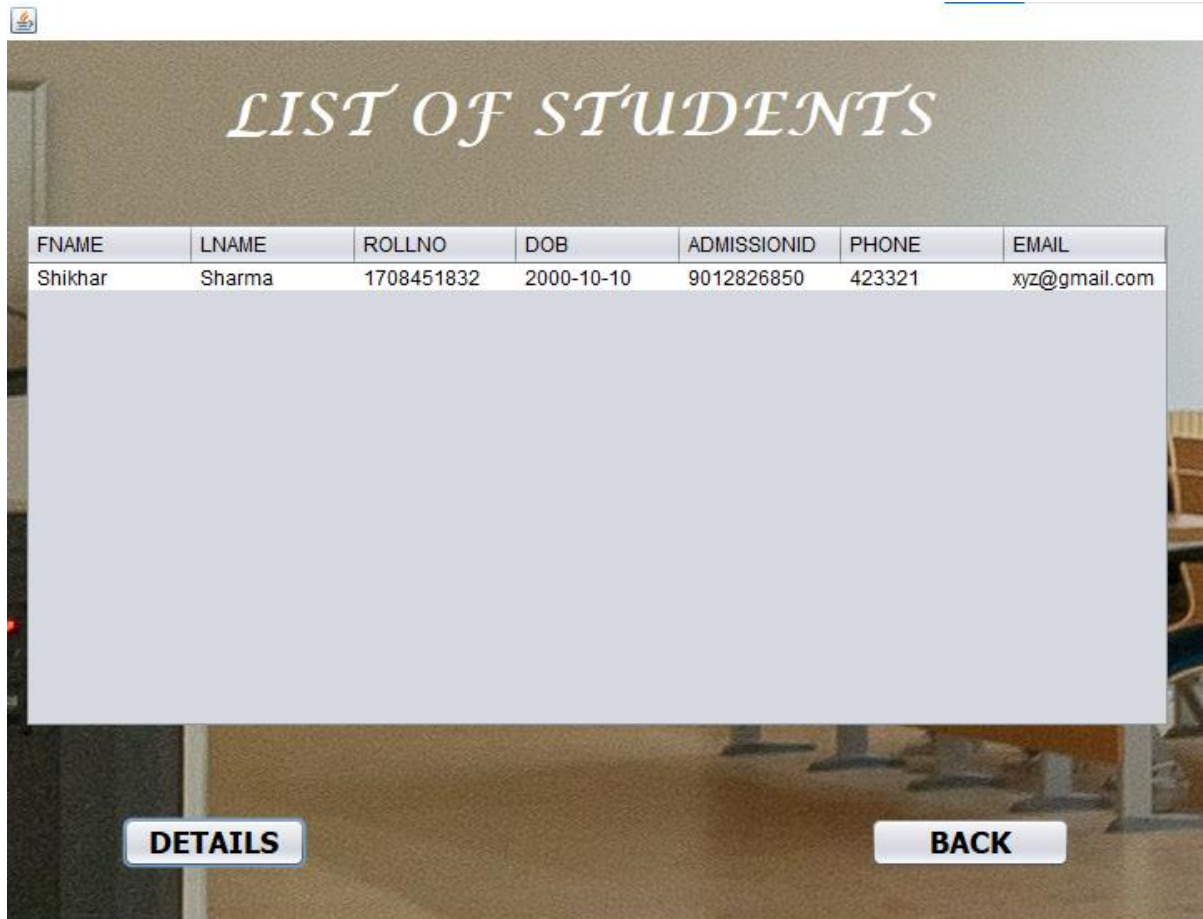
When user click on first option “Add New Details” then this page open in which user have to enter all the related details of the students correctly.



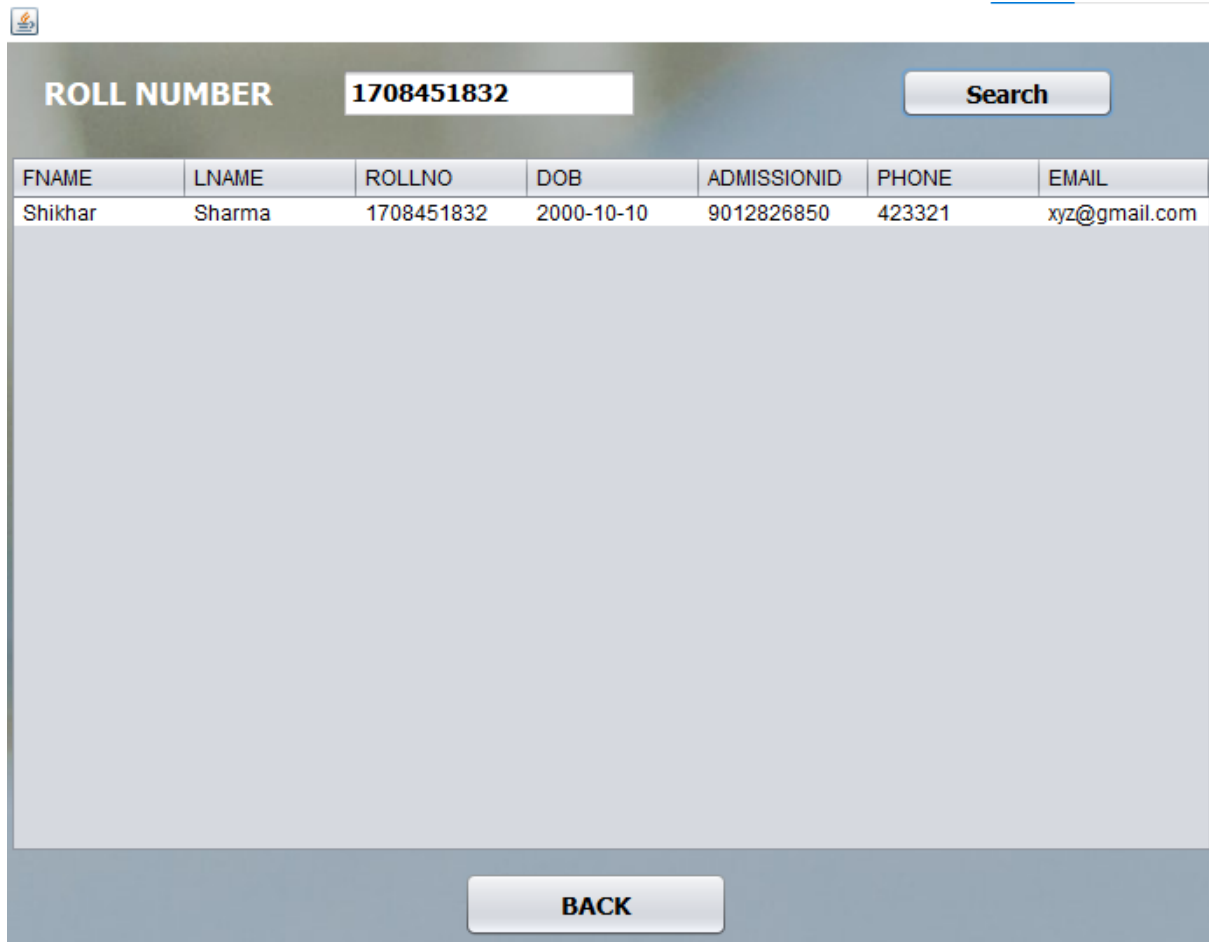
A screenshot of a web application interface for adding a new record. The form is set against a blurred background. It contains seven input fields, each with a label to its left. The labels are in all caps. The input fields contain the following text: 'Shikhar', 'Sharma', '1708451832', '2000/10/18', '9012826850', 'abc123', and 'xyz@gmail.com'. To the right of the input fields are three buttons: 'ADD' (positioned next to the Last Name field), 'CLEAR' (positioned next to the Roll Number field), and 'BACK' (positioned next to the Phone Number field). The buttons are light gray with rounded corners and black text.

FIRST NAME	Shikhar	
LAST NAME	Sharma	ADD
ROLL NUMBER	1708451832	CLEAR
DOB	2000/10/18	
PHONE NUMBER	9012826850	BACK
ADMISSION ID	abc123	
EMAIL	xyz@gmail.com	

After entering the details you have to press ADD button if the details are not correct then the dialog box pop up and show the error if the details are correct then dialog box through the message "Record successfully added" as shown in the screenshots below.



If user choose the second option means "List of All Students" then this page opens and after clicking the details button all the details of student are showed up on the table whose data is entered by the user. As we only enter one data so, only one row is showed up in the above table.

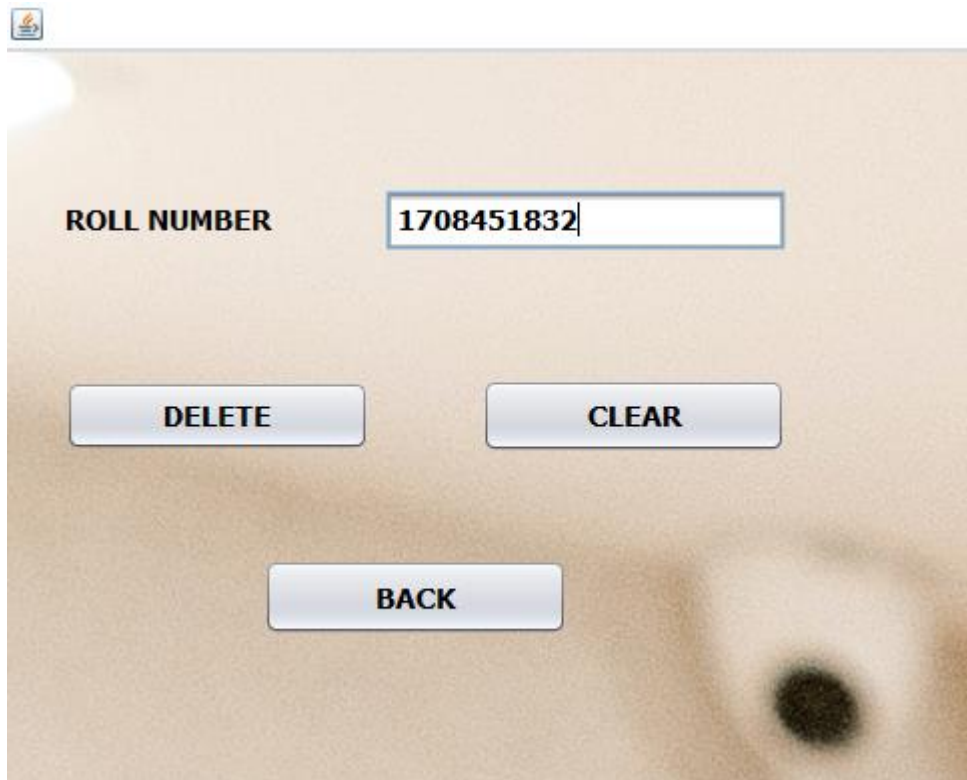


ROLL NUMBER **1708451832** **Search**

FNAME	LNAME	ROLLNO	DOB	ADMISSIONID	PHONE	EMAIL
Shikhar	Sharma	1708451832	2000-10-10	9012826850	423321	xyz@gmail.com

BACK

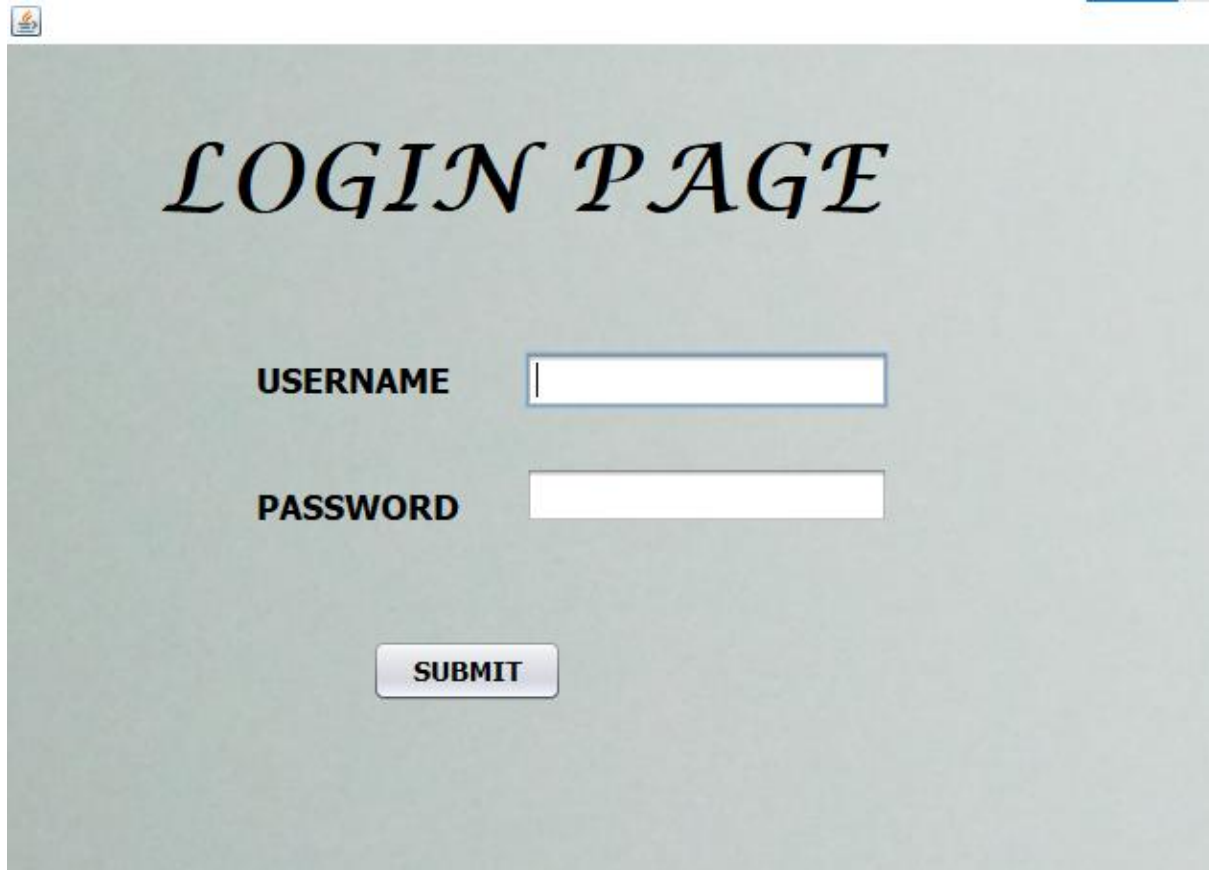
We can search the details of any student by entering their respective Roll Number if we choose the third option "Search Student Details".



A screenshot of a web application interface. At the top left, there is a small icon of a person. Below it, the text "ROLL NUMBER" is displayed in bold. To the right of this text is a text input field containing the number "1708451832". Below the input field, there are three buttons: "DELETE", "CLEAR", and "BACK". The "DELETE" button is on the left, "CLEAR" is on the right, and "BACK" is centered below them. All buttons are light blue with black text and rounded corners.

If we choose the last option means “Delete Student Details” then we can delete the details of any student by providing the Roll Number of the student and simply Clicking on the Delete button.

DESIGN AND CODE SCREENSHOTS OF THE PROJECT



```
Warning: Do not modify this code. The content of this method is always
* regenerated by the Form Editor.
*/
@SuppressWarnings("unchecked")
Generated Code

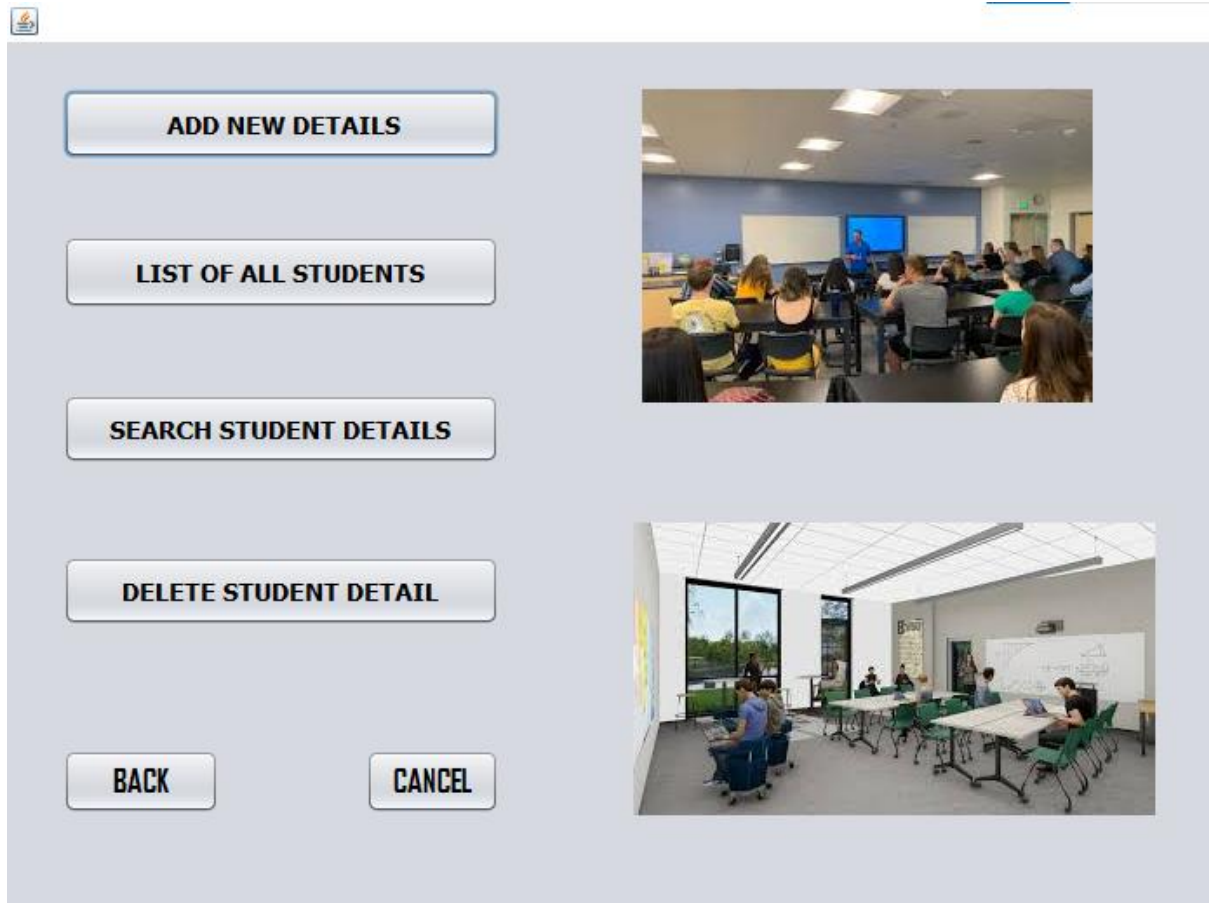
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    String pwd = new String(p1.getPassword());

    if(pwd.equals("teacher"))
    {
        login.this.setVisible(false);
        new sl().setVisible(true);
    }
    else
    {
        JOptionPane.showMessageDialog(jDialog1, "Error! Try Again.", "Error Message Box", JOptionPane.ERROR_MESSAGE);
    } // TODO add your handling code here:
}

private void t1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)
```



```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    sl.this.setVisible(false);  
    new login().setVisible(true);  
    // TODO add your handling code here:  
}  
  
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
    sl.this.setVisible(false);  
    new search().setVisible(true);    // TODO add your handling code here:  
}  
  
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    sl.this.setVisible(false);  
    new delete().setVisible(true);    // TODO add your handling code here:  
}  
  
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    sl.this.setVisible(false);  
    new add().setVisible(true);    // TODO add your handling code here:  
}  
  
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    sl.this.setVisible(false);  
    new list().setVisible(true);    // TODO add your handling code here:  
}  
  
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {  
    System.exit(0);    // TODO add your handling code here:  
}
```


The screenshot shows a Java Swing application window with a light blue background. On the left, there are seven text input fields with labels: FIRST NAME (Shikhar), LAST NAME (Sharma), ROLL NUMBER (1708451832), DOB (2000/10/10), PHONE NUMBER (9012826850), ADMISSION ID (423321), and EMAIL (xyz@gmail.com). To the right of these fields are two buttons: 'ADD' and 'CLEAR'. A 'Message' dialog box is open in the foreground, showing an information icon and the text 'Record successfully added.' with an 'OK' button.

```

} private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try
    {
        Class.forName("java.sql.Driver");
        Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "abhi12317");

        Statement stmt = (Statement) con.createStatement();
        String query = "INSERT INTO sd VALUES ('"+t1.getText()+"', '"+t2.getText()+"', '"+t3.getText()+"', '"+t4.getText()+"', "
            + "'"+t5.getText()+"', '"+t6.getText()+"', '"+t7.getText()+"')";

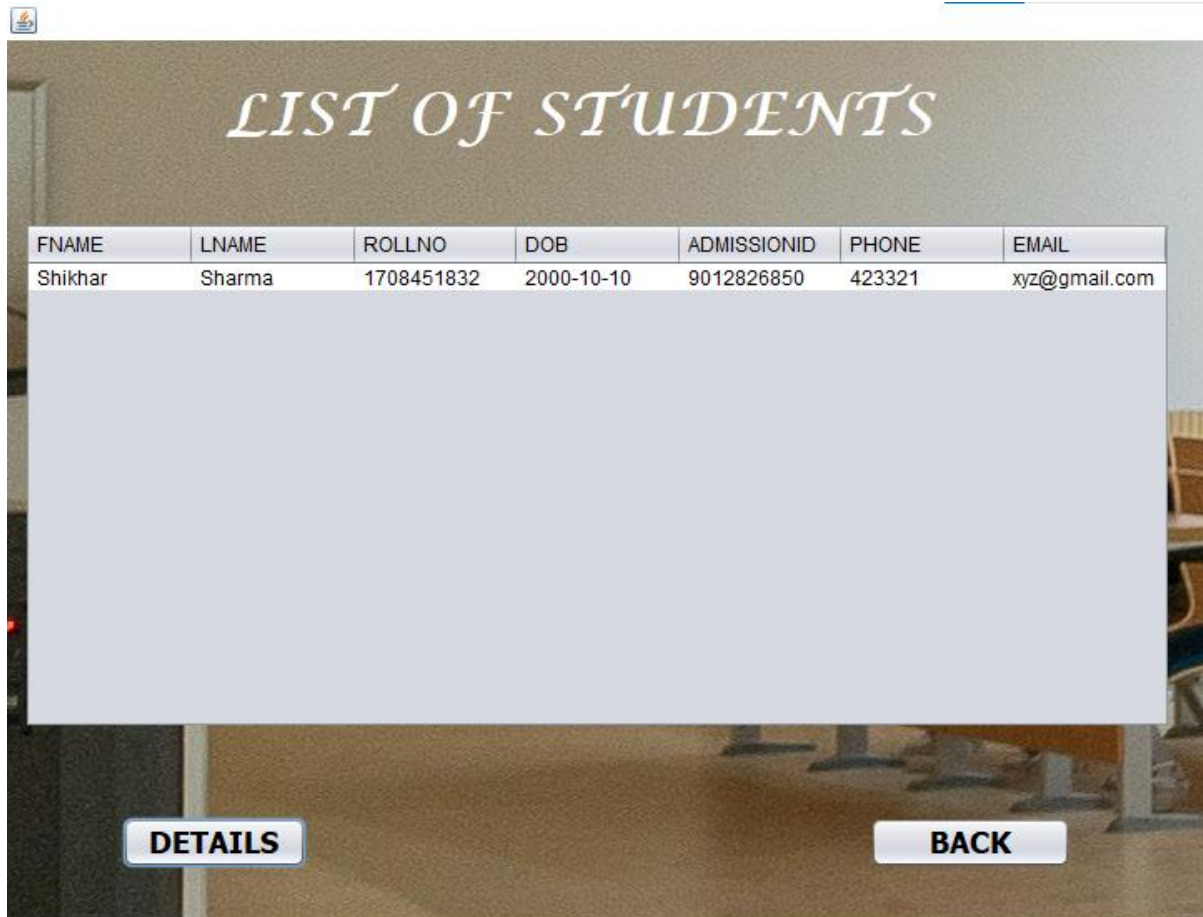
        stmt.executeUpdate(query);
        JOptionPane.showMessageDialog(null, "Record successfully added.");
    }

    catch (HeadlessException | ClassNotFoundException | SQLException e) {
        JOptionPane.showConfirmDialog(null, "Error in Adding");
    }


    // TODO add your handling code here:
}

} private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    t1.setText("");
    t2.setText("");
    t3.setText("");
    t4.setText("");
    t5.setText("");
    t6.setText("");
    t7.setText("");
}

```

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
  
    DefaultTableModel model = (DefaultTableModel) T1.getModel();  
    try{  
        Class.forName("java.sql.Driver");  
        Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "abhil2317");  
        Statement stmt = (Statement) con.createStatement();  
        String query = "SELECT * FROM SD;";  
        ResultSet rs = stmt.executeQuery(query);  
        while(rs.next()){  
            String fname = rs.getString("FNAME");  
            String lname = rs.getString("LNAME");  
            String roll = rs.getString("ROLLNO");  
            String dob = rs.getString("DOB");  
            String admission = rs.getString("ADMISSION_ID");  
            String phone = rs.getString("PHONE");  
            String email = rs.getString("EMAIL");  
  
            model.addRow(new Object[] {fname, lname, roll, dob, admission, phone, email});  
        }  
    }  
    catch(ClassNotFoundException | SQLException e){  
        JOptionPane.showMessageDialog(null, "Error");  
    }  
    // TODO add your handling code here:  
}
```



ROLL NUMBER

1708451832

Search

FNAME	LNAME	ROLLNO	DOB	ADMISSIONID	PHONE	EMAIL
Shikhar	Sharma	1708451832	2000-10-10	9012826850	423321	xyz@gmail.com

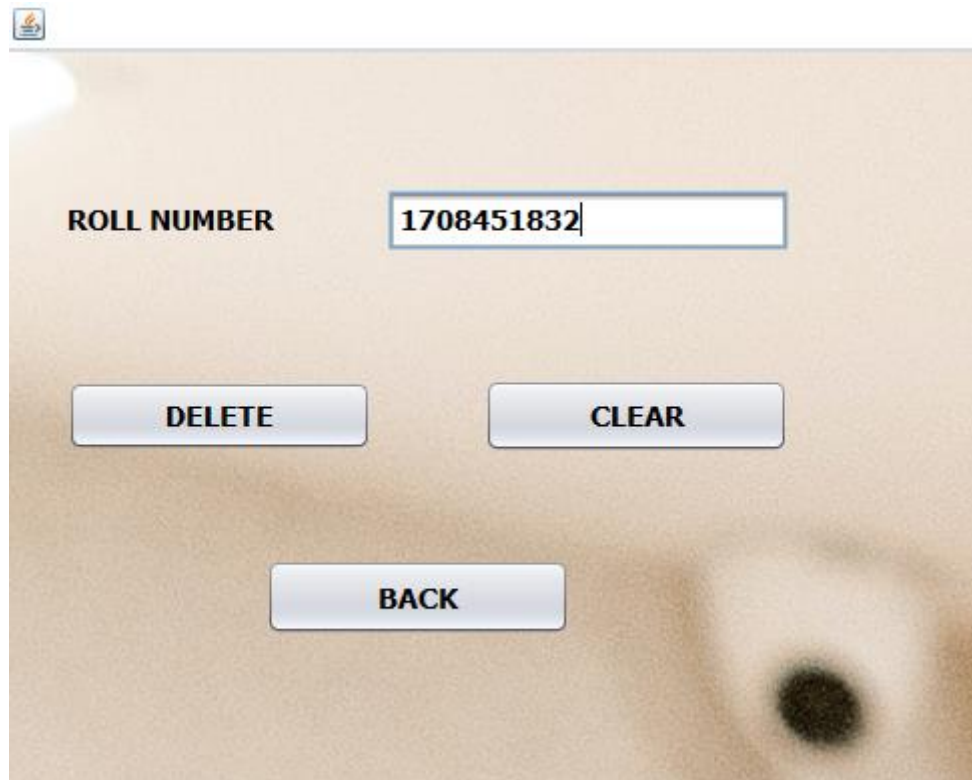
BACK

```

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    DefaultTableModel model = (DefaultTableModel) T1.getModel();
    try{
        Class.forName("java.sql.Driver");
        Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "abhil2317");
        Statement stmt = (Statement) con.createStatement();
        String query = "SELECT * FROM SD WHERE ROLLNO = "+t1.getText()+" ";
        ResultSet rs = stmt.executeQuery(query);
        while(rs.next()){
            String fname = rs.getString("FNAME");
            String lname = rs.getString("LNAME");
            String roll = rs.getString("ROLLNO");
            String dob = rs.getString("DOB");
            String admission = rs.getString("ADMISSION_ID");
            String phone = rs.getString("PHONE");
            String email = rs.getString("EMAIL");

            model.addRow(new Object[] {fname, lname, roll, dob, admission, phone, email});
        }
    }
    catch(ClassNotFoundException | SQLException e){
        JOptionPane.showMessageDialog(null,"Error");
    }
    // TODO add your handling code here:
}

```



```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
  
    try  
    {  
        Class.forName("java.sql.Driver");  
        Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "abhi12317");  
  
        Statement stmt = (Statement) con.createStatement();  
        String query = "Delete from SD WHERE ROLLNO="+t3.getText()+" ";  
  
        stmt.executeUpdate(query);  
        JOptionPane.showMessageDialog(null, "Record successfully deleted.");  
    }  
  
    catch (HeadlessException | ClassNotFoundException | SQLException e) {  
        JOptionPane.showMessageDialog(null, "Error in Adding");  
    }  
    // TODO add your handling code here:  
}
```

IMPLEMENTATION AND MAINTENANCE

There is an apk file and zip file. Install apk file directly to device and run.

Zip file for netbeans. Before running the project in netbeans you have to create table name "SD" in mysql in database "test". To store and retrieve data.

FUTURE SCOPE

Graphical Analysis , download and share option.

Saving data online through user id.

REFERENCES

<https://netbeans.org/kb/docs/java/quickstart.html>

<https://github.com/topics/netbeans-project>