

## JAVA

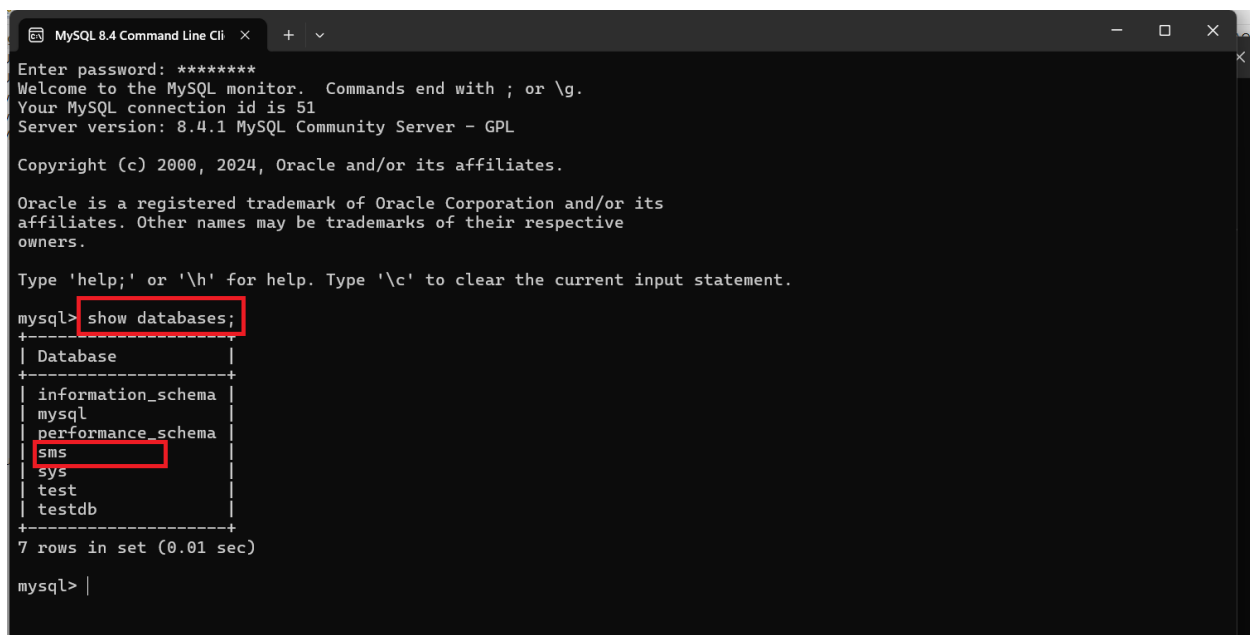
Student ID : C0930321

Student Name : Shreejana Shrestha

# # Student Management System

## # Database and its connectivity to the SMS project

### # showing database



```
MySQL 8.4 Command Line Cli x + v
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 51
Server version: 8.4.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sms |
| sys |
| test |
| testdb |
+-----+
7 rows in set (0.01 sec)

mysql> |
```

## # Structure of student table

```
MySQL 8.4 Command Line Cli | + | -
| performance_schema |
| sms |
| sys |
| test |
| testdb |
+-----+
7 rows in set (0.01 sec)

mysql> desc student;
ERROR 1046 (3D000): No database selected
mysql> use sms;
Database changed
mysql> desc student;
```

Field	Type	Null	Key	Default	Extra
studentID	int	NO	PRI	NULL	auto_increment
firstName	varchar(30)	NO		NULL	
lastName	varchar(30)	NO		NULL	
dateOfBirth	date	NO		NULL	
gender	varchar(10)	NO		NULL	
email	varchar(50)	NO		NULL	
phoneNumber	varchar(15)	YES		NULL	
address	text	YES		NULL	
enrollmentDate	date	NO		NULL	
isActive	tinyint(1)	YES		1	

```
+-----+
10 rows in set (0.01 sec)

mysql> |
> mysql>
```

## # Existing Records on the student table

```
MySQL 8.4 Command Line Cli x + v
```

```
enrollmentDate | date | NO | NULL |  
isActive | tinyint(1) | YES | 1 |  
-----  
10 rows in set (0.01 sec)
```

```
mysql> select * from student;
```

```
studentID | firstName | lastName | dateOfBirth | gender | email | phoneNumber | address | enrollmentDate | isActive |  
-----  
1 | shreejana | stha | 2010-08-05 | Female | shreejana@gmail.com | 3453453457 | ktm | 2010-08-05 | 1  
2 | John | Doe | 2000-01-15 | Male | john.doe@example.com | 123-456-7890 | 123 Main St | 2024-08-09 | 0  
3 | Alice | Smith | 1999-11-30 | Female | alice.smith@example.com | 987-654-3210 | 456 Elm St | 2024-08-01 | 1  
4 | Bob | Johnson | 2001-05-22 | Male | bob.johnson@example.com | 555-123-4567 | 789 Oak St | 2024-09-15 | 1  
5 | Diana | Prince | 2000-02-14 | Female | diana.prince@example.com | 123-789-4560 | 202 Maple St | 2024-08-10 | 0  
7 | shreejana | shrestha | 2010-08-05 | Female | shreejana@gmail.com | 34534534534 | w34sdf sdfgs | 2024-08-08 | 0  
8 | Prabin | Shrestha | 2020-08-12 | Male | test@test.com | 8687687678 | ktm | 2020-08-12 | 0  
10 | Shree | Stha | 2023-08-24 | Male | shree@test.com | 4563214789 | kathmandu | 2024-08-08 | 0  
11 | Raman | Raj | 2013-08-08 | Male | raman@test.com | 2342345678 | Mississauga | 2024-08-07 | 1  
-----  
9 rows in set (0.00 sec)
```

```
mysql> |
```

## # Making connection to the database

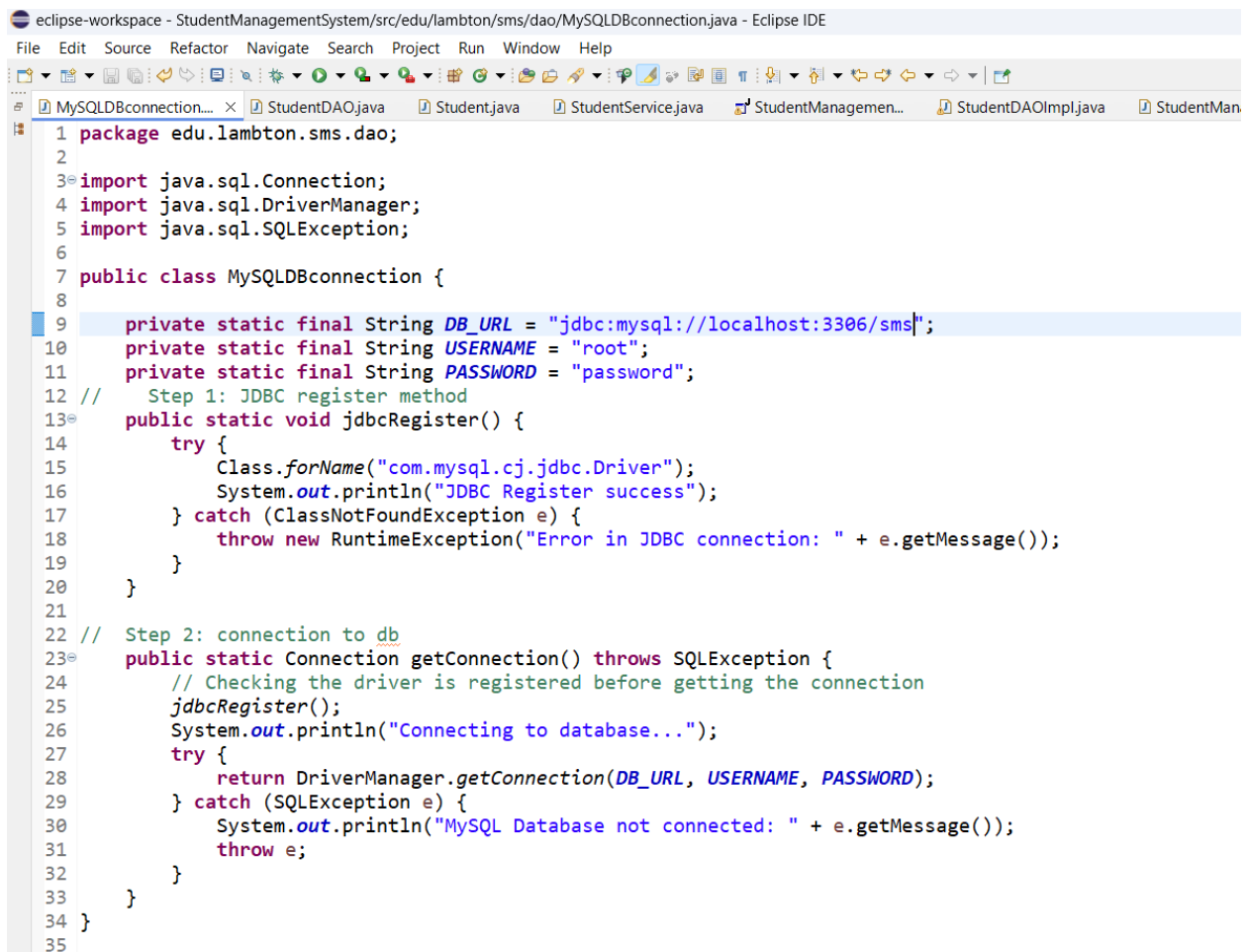
The screenshot displays the Eclipse IDE environment. The background shows the source code for `StudentManagementApp.java`, which includes imports for database connectivity and GUI components, and a `main` method that initializes the application and attempts to connect to a database. The foreground features a window titled "Student Management System" with a search bar and a table of student records.

```
3 import edu.lambton.sms.dao.MySQLDBConnection;
4 import edu.lambton.sms.dao.StudentDAO;
5 import edu.lambton.sms.dao.StudentDAOImpl;
6 import edu.lambton.sms.service.StudentService;
7 import edu.lambton.sms.gui.StudentManagementView;
8
9 import javax.swing.*;
10 import java.sql.Connection;
11
12 public class StudentManagementApp {
13
14     public static void main(String[] args) {
15         // Initialize Swing application on the Event Dispatch Thread
16         SwingUtilities.invokeLater(() -> {
17             Connection connection = null;
18             try {
19                 // Establish database connection
20                 connection = MySQLDBConnection.getConnection();
21
22                 // Create DAO and service instances
23                 StudentDAO studentDAO = new StudentDAOImpl();
24                 StudentService studentService = new StudentService(studentDAO);
25
26                 // Launch the GUI
27                 StudentManagementView view = new StudentManagementView(studentService);
28                 view.setVisible(true);
29
30             } catch (Exception e) {
31                 e.printStackTrace();
32                 // Handle error and display generic message
33                 JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());
34             }
35         });
36     }
37
38 }
39
```

The "Student Management System" window includes a search bar and a table of student records. The table has columns for ID, First Name, Last Name, Date of Birth, Gender, Email, Phone, Address, Enrollment Date, and Active status.

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone	Address	Enrollment Date	Active
3	Alice	Smith	1999-1-1	Female	alice.s.	987-654	456 Elm St.	2024-08	true
4	Bob	Johnson	2001-0-0	Male	bob.joh.	555-123	789 Oak Ave.	2024-09	true
5	Diana	Prince	2000-0-0	Female	diana.pr.	123-789	202 Main St.	2024-08	false
2	John	Doe	2000-0-0	Male	john.do.	123-456	123 Main St.	2024-08	false
8	Prabin	Shrestha	2020-0-0	Male	test@te	868768	ktm	2020-08	false
11	Raman	Raj	2013-0-0	Male	raman..	234234	Mississ.	2024-08	true
10	Shree	Sitha	2023-0-0	Male	shree..	456321	kathma.	2024-08	false
1	shreeja	sitha	2010-0-0	Female	shreeja.	345345	ktm	2010-08	true
7	shreeja	shrestha	2010-0-0	Female	shreeja.	345345	w34adf	2024-08	false

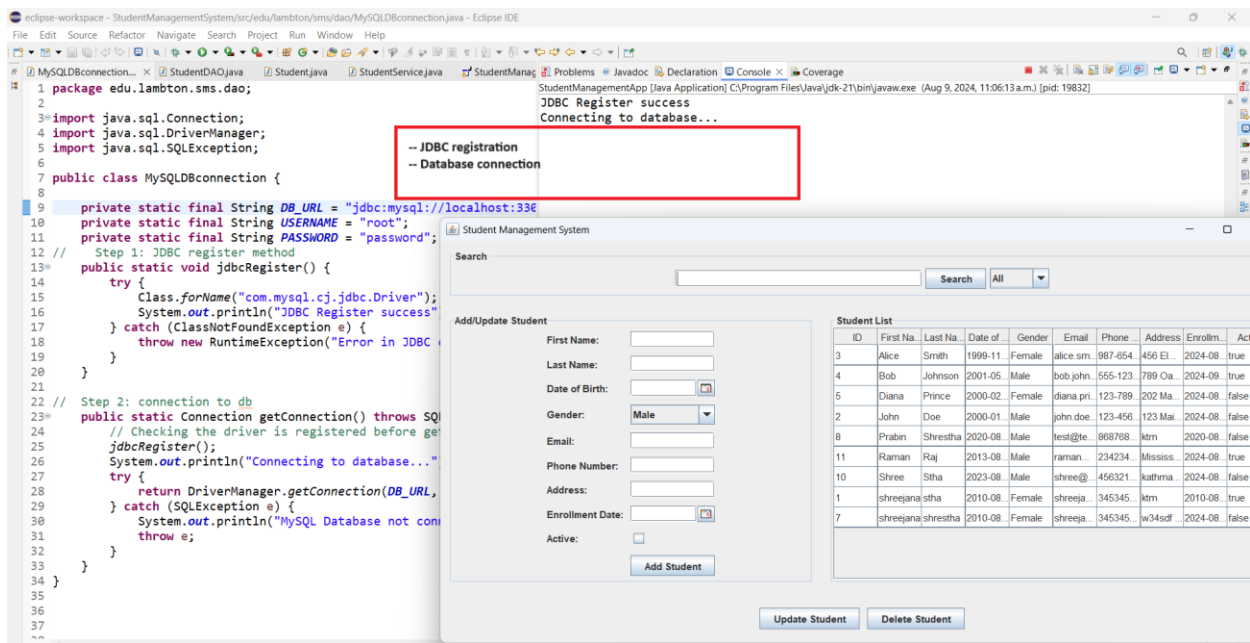
## # Database connectivity code snippet



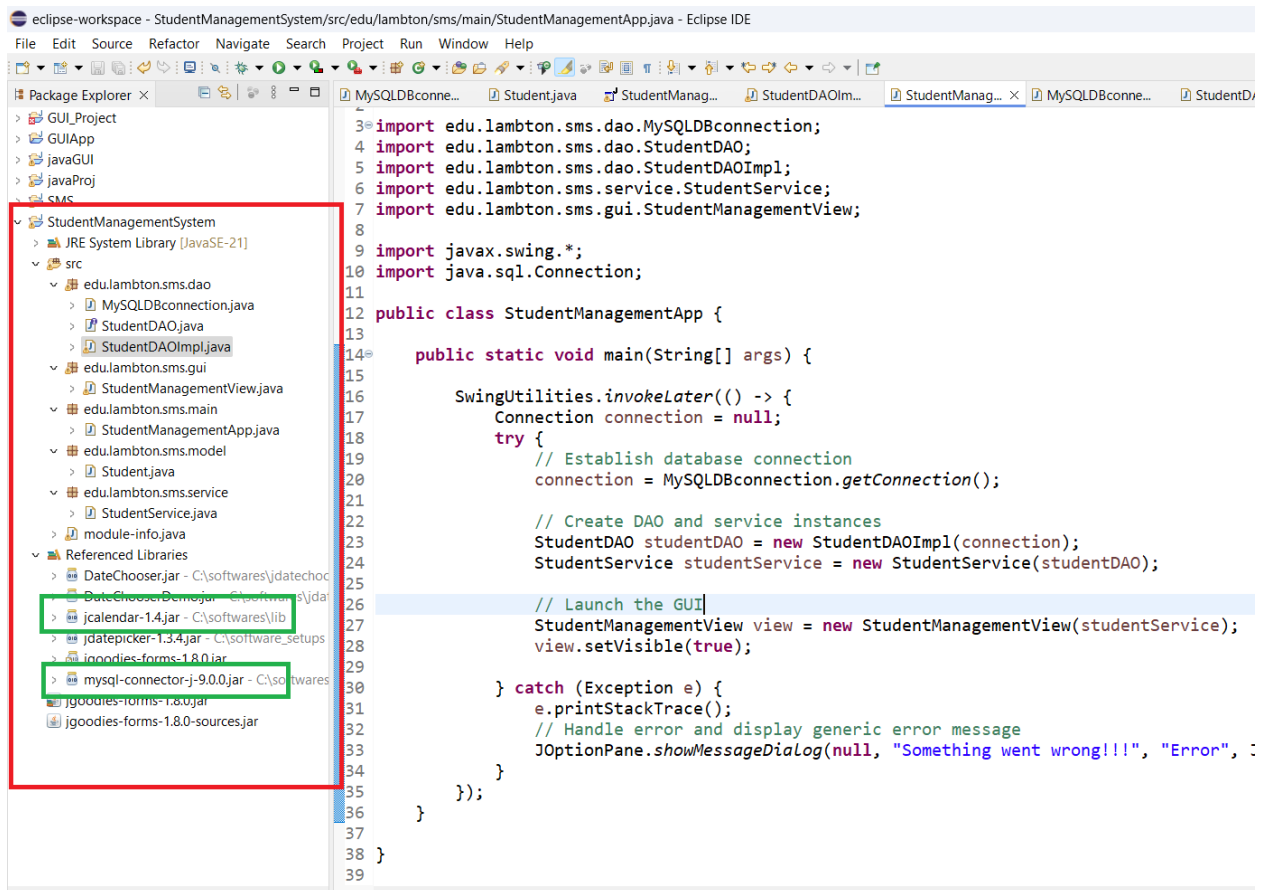
The screenshot shows the Eclipse IDE interface with the file `MySQLDBconnection.java` open. The package is `edu.lambton.sms.dao`. The code includes imports for `java.sql.Connection`, `java.sql.DriverManager`, and `java.sql.SQLException`. The `MySQLDBconnection` class contains two static methods: `jdbcRegister()` and `getConnection()`. `jdbcRegister()` registers the MySQL JDBC driver. `getConnection()` checks if the driver is registered and then establishes a connection using `DriverManager.getConnection(DB_URL, USERNAME, PASSWORD)`. Both methods include try-catch blocks to handle exceptions.

```
1 package edu.lambton.sms.dao;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class MySQLDBconnection {
8
9     private static final String DB_URL = "jdbc:mysql://localhost:3306/sms";
10    private static final String USERNAME = "root";
11    private static final String PASSWORD = "password";
12    // Step 1: JDBC register method
13    public static void jdbcRegister() {
14        try {
15            Class.forName("com.mysql.cj.jdbc.Driver");
16            System.out.println("JDBC Register success");
17        } catch (ClassNotFoundException e) {
18            throw new RuntimeException("Error in JDBC connection: " + e.getMessage());
19        }
20    }
21
22    // Step 2: connection to db
23    public static Connection getConnection() throws SQLException {
24        // Checking the driver is registered before getting the connection
25        jdbcRegister();
26        System.out.println("Connecting to database...");
27        try {
28            return DriverManager.getConnection(DB_URL, USERNAME, PASSWORD);
29        } catch (SQLException e) {
30            System.out.println("MySQL Database not connected: " + e.getMessage());
31            throw e;
32        }
33    }
34 }
35
```

⇒ **MySQLDBconnection** class handles the database connection setup which is initialized from the **StudentManagementApp** class. Here, we have registered the jdbc and make database connection using `DriverManager.getConnection(DB_URL, USERNAME, PASSWORD)`.



## # Folder structure in the project



Here, we have used edu.lambton.sms as the root package for the project SMS. All the other packages gui, dao, service, main, model are organized under this. We have created a separate package and respective class for each functionality.

- ➔ Gui contains the components viewed by the admin/user. It manages the user interface, including forms and table views in our application.
- ➔ Dao contains data access object interface and implementation. It defines the data access methods for the student entity and contains all the SQL queries and database interaction logic.
- ➔ Model contains domain classes that represent the data model i.e Student in our application
- ➔ Service contains the service layer where the main business logic resides such as adding, updating, retrieving and deleting. It interacts with DAO layer to perform these operations.
- ➔ Main contains the main entry point of our application

## # First GUI of Student Management System

The screenshot shows the 'Student Management System' window. It features a search bar at the top with a 'Search' button and a dropdown menu set to 'All'. Below the search bar, there are two main sections: 'Add/Update Student' on the left and 'Student List' on the right. The 'Add/Update Student' section contains input fields for First Name, Last Name, Date of Birth (with a calendar icon), Gender (a dropdown menu currently showing 'Male'), Email, Phone Number, Address, Enrollment Date (with a calendar icon), and an 'Active' checkbox. An 'Add Student' button is at the bottom of this section. The 'Student List' section displays a table with columns: ID, First Name, Last Name, Date of Birth, Gender, Email, Phone Nu..., Address, Enrollment..., and Active. The table is currently empty. At the bottom of the window, there are 'Update Student' and 'Delete Student' buttons.

## # GUI with all from the student table listed below

This screenshot shows the same 'Student Management System' window, but now the 'Student List' table is populated with 10 student records. The search dropdown menu is highlighted with a red box and is set to 'All'. The 'Add/Update Student' section remains the same. The 'Student List' table contains the following data:

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45	202 Maple	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
8	Prabin	Shrestha	2020-08-12	Male	akodfidas	8687687678	ktm	2024-08-08	false
10	Shree	Stha	2023-08-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana...	3453453457	ktm	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana...	345345345	w34sdf sdf	2024-08-08	true

Below the table, there are 'Update Student' and 'Delete Student' buttons. In the background, a code editor is visible, showing Java code for the 'Student' class and its methods. The system tray at the bottom shows the date and time as 10:24 AM on 2024-08-09.

```

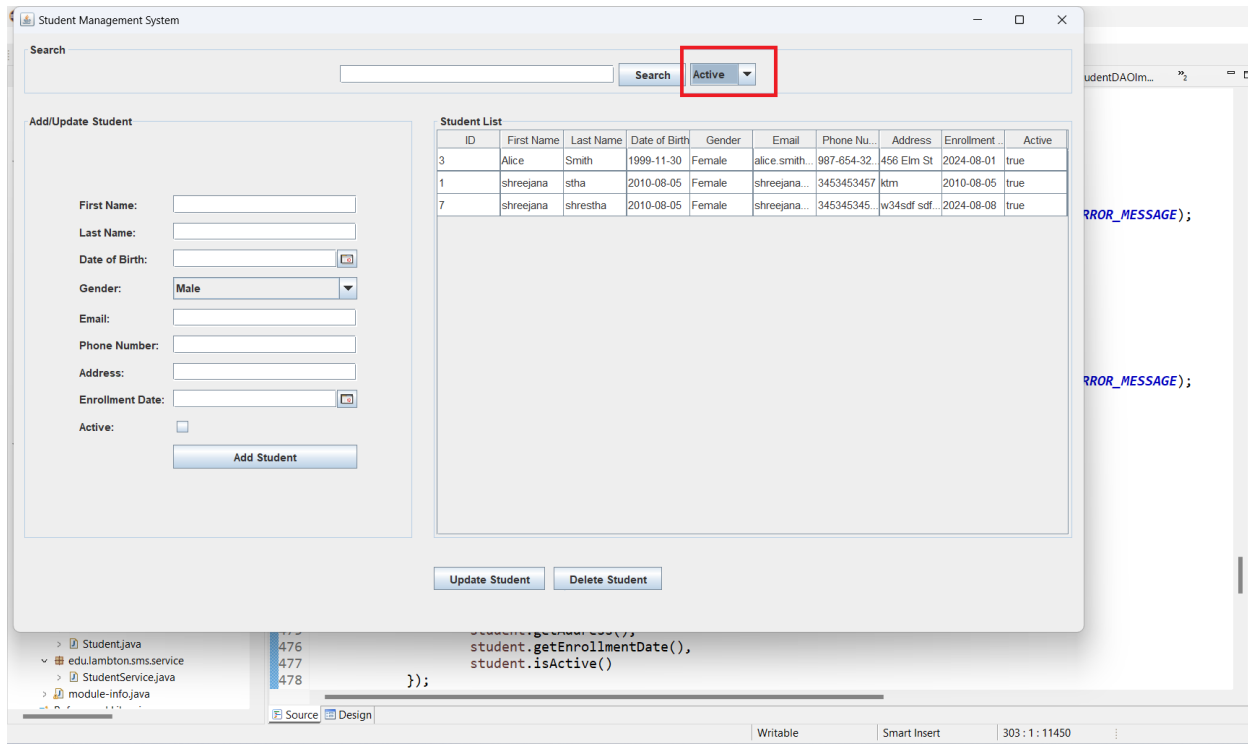
MySQLDBconnection... StudentDAO.java Student.java StudentService.java StudentManagemen... x StudentDAOImpl.java StudentManagemen... MySQLD
444 private void refreshTable() {
445     try {
446         List<Student> students = studentService.getAllStudents();
447         refreshTable(students);
448     } catch (SQLException e) {
449         e.printStackTrace();
450         JOptionPane.showMessageDialog(this, "Error refreshing table.", "Error", JOptionPane.ERROR_MESSAGE);
451     }
452 }
453
454 private void refreshTableByStatus(String status) {
455     try {
456         List<Student> students = studentService.getAllStudentsByStatus(status);
457         refreshTable(students);
458     } catch (SQLException e) {
459         e.printStackTrace();
460         JOptionPane.showMessageDialog(this, "Error refreshing table.", "Error", JOptionPane.ERROR_MESSAGE);
461     }
462 }
463
464 private void refreshTable(List<Student> students) {
465     tableModel.setRowCount(0);
466     for (Student student : students) {
467         tableModel.addRow(new Object[]{
468             student.getId(),
469             student.getFirstName(),
470             student.getLastName(),
471             student.getDateOfBirth(),
472             student.getGender(),
473             student.getEmail(),
474             student.getPhoneNumber(),
475             student.getAddress(),
476             student.getEnrollmentDate(),
477             student.isActive()
478         });
479 }

```

⇒ This is the logic implemented to display all the records from the student table. The table is listed with data if exists when the application is first run.



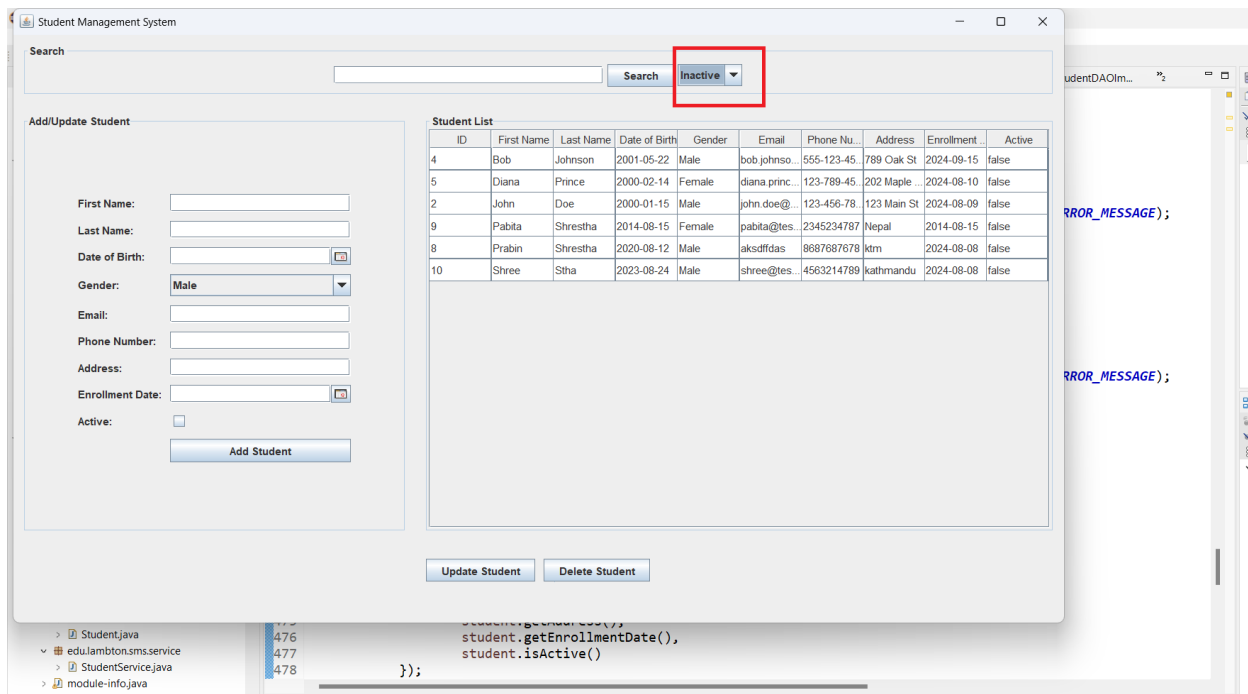
## # Table listing only the active students record



The screenshot shows the 'Student Management System' window. The 'Search' dropdown menu is open, and the 'Active' option is selected and highlighted with a red box. The 'Student List' table displays three active students.

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32	456 Elm St	2024-09-01	true
1	shreejana	stha	2010-08-05	Female	shreejana...	3453453457	ktrm	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana...	345345345	w34sdf sdf	2024-08-08	true

## # Table listing only the inactive students record



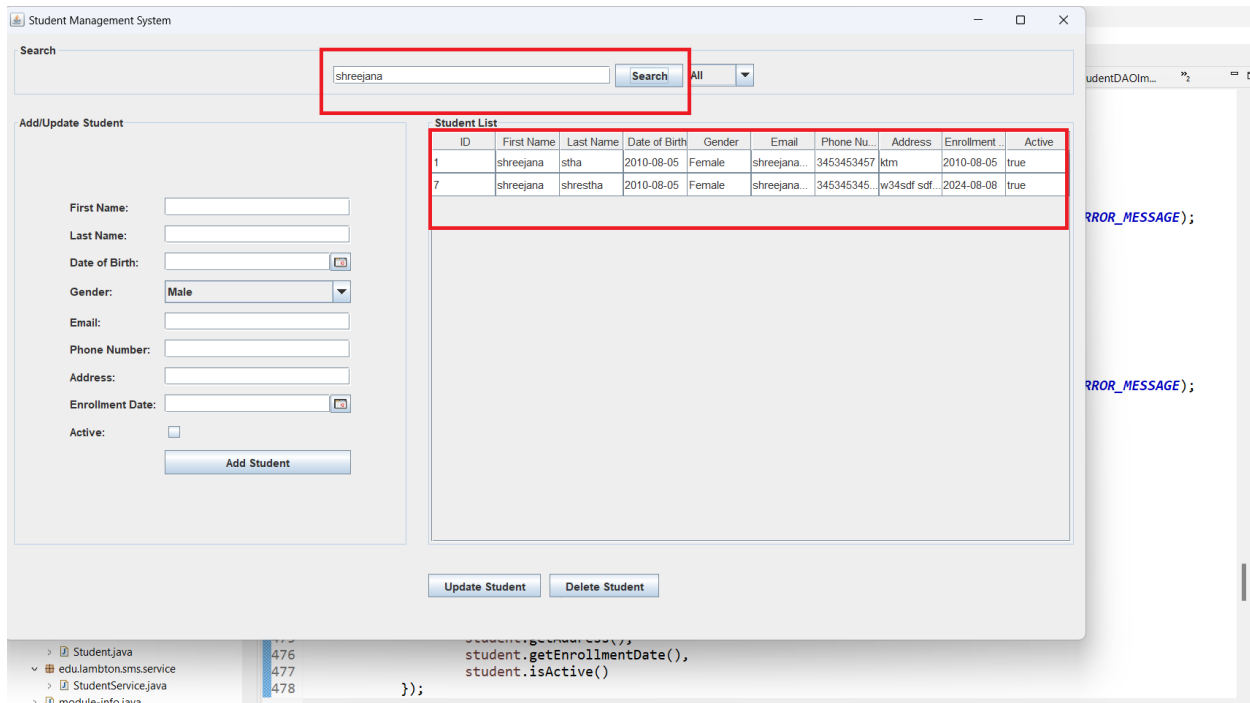
The screenshot shows the 'Student Management System' window. The 'Search' dropdown menu is open, and the 'Inactive' option is selected and highlighted with a red box. The 'Student List' table displays five inactive students.

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment...	Active
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45	202 Maple	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78	123 Main St	2024-09-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
8	Prabin	Shrestha	2020-08-12	Male	aksdfdas	9867687678	ktrm	2024-08-08	false
10	Shree	Stha	2023-08-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false

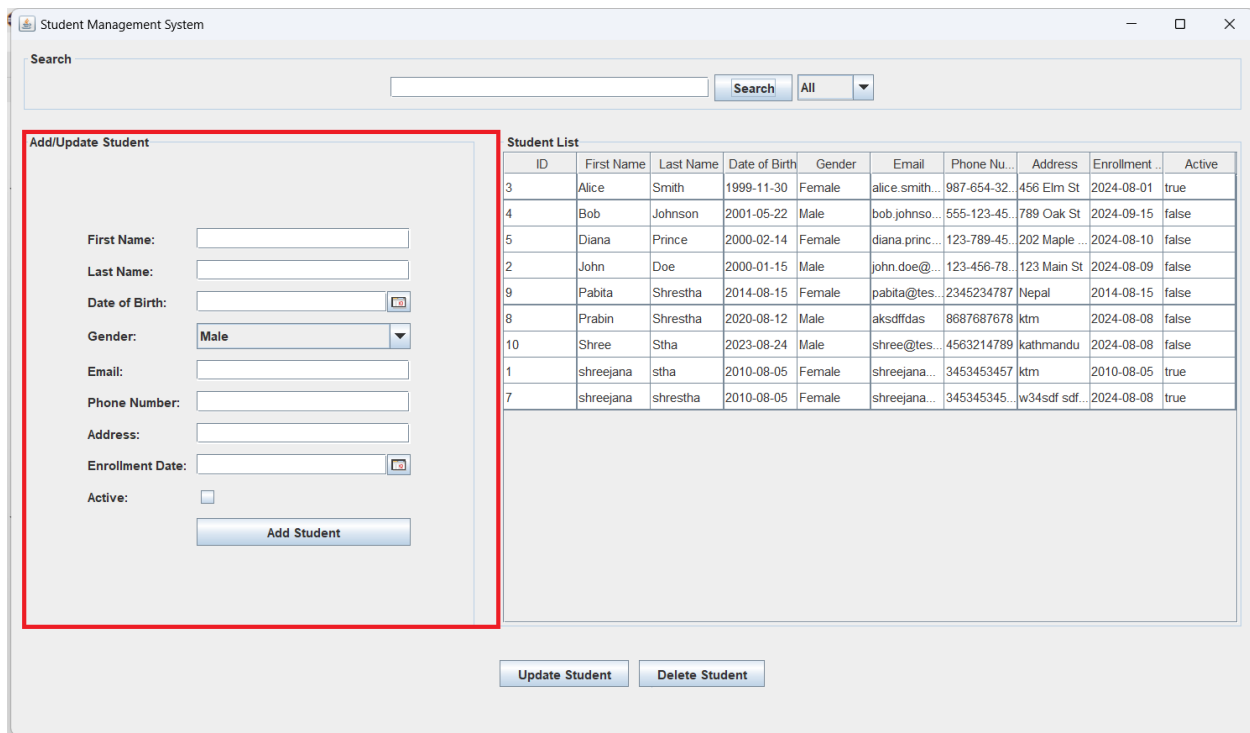
## # Search filter

```
115
116 @Override
117 public List<Student> searchStudents(String searchTerm) throws SQLException {
118     List<Student> students = new ArrayList<>();
119     String sql = "SELECT * FROM student WHERE firstName LIKE ? OR lastName LIKE ? OR email LIKE ? ORDER BY firstName";
120     try (PreparedStatement statement = connection.prepareStatement(sql)) {
121         String pattern = "%" + searchTerm + "%";
122         statement.setString(1, pattern);
123         statement.setString(2, pattern);
124         statement.setString(3, pattern);
125         try (ResultSet resultSet = statement.executeQuery()) {
126             while (resultSet.next()) {
127                 Student student = mapRowToStudent(resultSet);
128                 students.add(student);
129             }
130         }
131     }
132     return students;
133 }
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273 private void searchStudents() {
274     try {
275         String searchTerm = txtSearch.getText();
276         List<Student> students = studentService.searchStudents(searchTerm);
277         refreshTable(students);
278     } catch (SQLException e) {
279         e.printStackTrace();
280         JOptionPane.showMessageDialog(this, "Error searching students.", "Error", JOptionPane.ERROR_MESSAGE);
281     }
282 }
283
```

- ➔ The search filter button enables the functionality to filter and retrieve student records based on specific criteria such as students' firstName, lastName and email. When a search query is entered, the system queries the database and returns a list of students that match the criteria and only the relevant student information is displayed.



## # GUI to get the student data



## # Input validation for the form fields

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Last Name:

Date of Birth:

📅

Gender:

Male

▼

Email:

Phone Number:

Address:

Enrollment Date:

📅

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
-12	Male	aksdffdas	8687687678	ktm	2024-08-08	false			
-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
-05	Female	shreejana...	3453453457	ktm	2010-08-05	true			
-05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true			

Validation Error

✖

✖

Date of Birth cannot be empty.

OK

Update Student

Delete Student

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Last Name:

Date of Birth:

Aug 13, 2020

📅

Gender:

Male

▼

Email:

Phone Number:

Address:

Enrollment Date:

📅

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
-12	Male	aksdffdas	8687687678	ktm	2024-08-08	false			
-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
-05	Female	shreejana...	3453453457	ktm	2010-08-05	true			
-05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true			

Validation Error

✖

✖

Enrollment Date cannot be empty.

OK

Update Student

Delete Student

Student Management System

Search

Search

All

Add/Update Student

First Name:

Last Name:

Date of Birth:

Aug 13, 2020

Gender:

Male

Email:

Phone Number:

Address:

Enrollment Date:

Aug 8, 2024

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
-12	Male	aksdffd...	8687687678	ktm	2024-08-08	false			
-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
-05	Female	shreejana...	3453453457	ktm	2010-08-05	true			
-05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true			

Validation Error

First Name cannot be empty.

OK

Update Student

Delete Student

## # Adding functionality

```

private void addStudent() {
    try {
        String firstName = txtFirstName.getText();
        String lastName = txtLastName.getText();

        String gender = (String) cbGender.getSelectedItem();
        String email = txtEmail.getText();
        String phoneNumber = txtPhoneNumber.getText();
        String address = txtAddress.getText();

        boolean isActive = chkActive.isSelected();

        if (dateOfBirthChooser.getDate() == null) {
            showValidationMessage("Date of Birth cannot be empty.");
            return;
        }

        if (enrollmentDateChooser.getDate() == null) {
            showValidationMessage("Enrollment Date cannot be empty.");
            return;
        }

        LocalDate dateOfBirth = new java.sql.Date(dateOfBirthChooser.getDate().getTime()).toLocalDate();
        LocalDate enrollmentDate = new java.sql.Date(enrollmentDateChooser.getDate().getTime()).toLocalDate();

        boolean valid = validateFields(firstName, lastName, dateOfBirth, email, phoneNumber, address, enrollmentDate);
        if (!valid) {
            return;
        }

        Student student = new Student(0, firstName, lastName, dateOfBirth, gender, email, phoneNumber, address, enrollmentDate);
        studentService.addStudent(student);
        refreshTable();
        String msg = "Student " + student.getFullName() + " added successfully!!!";
        JOptionPane.showMessageDialog(this, msg, "Success", JOptionPane.INFORMATION_MESSAGE);
    }
}

```

The screenshot displays the Eclipse IDE with the `StudentDAOImpl.java` file open. The code implements the `StudentDAO` interface, including methods for adding a student and querying the database. The GUI, titled "Student Management System", features a search bar, an "Add/Update Student" form, a "Student List" table, and a success modal.

**Student List Table:**

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson...	555-123-45...	789 Oak St	2024-09-15	true
5	Diana	Prince	2000-02-14	Female	diana.prince...	123-789-45...	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78...	123 Main St	2024-08-09	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.com	8687687678	ktn	2020-08-12	false
11	Raman	Raj	2013-08-08	Male	raman@test...	2342345678	Mississauga	2024-08-07	true
10	Shree	Stha	2023-08-24	Male	shree@test.c...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana@g...	3453453457	ktn	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	false

**Add/Update Student Form:**

First Name: Raman  
 Last Name: Raj  
 Date of Birth: Aug 8, 2013  
 Gender: Male  
 Email: raman@test.com  
 Phone Number: 2342345678  
 Address: Mississauga  
 Enrollment Date: Aug 7, 2024  
 Active: ☒  
 Add Student

**Success Modal:**

Student Raman Raj added successfully!!!

**Notes:**

- Adding student from the form field and the success modal is shown on successful insertion.
- Likewise, the record is inserted to the student table.
- The form will reset once the OK button from the success modal is clicked.

➔ The "Add Student" functionality in our project allows to input student details through a form in the GUI and save them to the database. When the admin user submits the form, the input data is validated, ensuring that all required fields are filled out correctly. Once validated, a Student object is created using the form data, and the StudentService interacts with the StudentDAO to insert the new student record into the database. If the insertion is successful, the form is reset, and the student table in the GUI is refreshed to display the newly added student.

## # Table after successful addition of record

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Last Name:

Date of Birth:

📅

Gender:

Male

▼

Email:

Phone Number:

Address:

Enrollment Date:

📅

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson@...	555-123-45...	789 Oak St	2024-09-15	true
5	Diana	Prince	2000-02-14	Female	diana.prince@...	123-789-45...	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78...	123 Main St	2024-08-09	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.com	8687687678	ktm	2020-08-12	false
11	Raman	Raj	2013-08-08	Male	raman@test...	2342345678	Mississauga	2024-08-07	true
10	Shree	Stha	2023-08-24	Male	shree@test.c...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana@g...	3453453457	ktm	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	false

Update Student

Delete Student

## # Selecting the record to perform update/delete or activate/deactivate operation

Student Management System

Search

Search

All

Add/Update Student

First Name:

Last Name:

Date of Birth:

Gender:

Male

Email:

Phone Number:

Address:

Enrollment Date:

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@ges...	23456234767	nepal	2014-08-15	false
8	Prabin	Shrestha	2020-08-12	Male	aksdffdass	8687687678	ktm	2024-08-08	false
10	Shree	Sina	2023-06-24	Male	shree@ges...	4303214768	kathmandu	2024-06-06	false
1	shreejana	stha	2010-08-05	Female	shreejana...	3453453457	ktm	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true

- Selecting table row to perform update or delete.

- on the table row selection, Activate/Deactivate button will be displayed based on the isActive status.

Update Student

Delete Student

Activate



## # Error message if no record is not selected

Student Management System

Search   All

**Add/Update Student**

First Name:

Last Name:

Date of Birth:

Gender:

Email:

Phone Number:

Address:

Enrollment Date:

Active: ☐

**Student List**

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
-12	Male	test@test.c...	8687687678	ktrn	2020-08-12	false			
-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
-05	Female	shreejana...	3453453457	ktrn	2010-08-05	true			
-05	Female	shreejana...	345345345...	w34sdf sdf.	2024-08-08	true			

Warning

Please select a student to delete.

-- warning message when trying to delete without selecting the record from the table

## # Update Operation

```
40         statement.executeUpdate();
41     }
42 }
43
44 @Override
45 public void updateStudentStatus(int studentId, boolean isActive) throws SQLException {
46     String sql = "UPDATE student SET isActive = ? WHERE studentId = ?";
47     try (PreparedStatement statement = connection.prepareStatement(sql)) {
48         statement.setBoolean(1, isActive);
49         statement.setInt(2, studentId);
50         statement.executeUpdate();
51     }
52 }
53
```

```

356 private void updateStudent() {
357     int selectedRow = tblStudents.getSelectedRow();
358     if (selectedRow != -1) {
359         int studentId = (int) tblStudents.getValueAt(selectedRow, 0);
360         try {
361             Student student = studentService.getStudentById(studentId);
362             if (student != null) {
363                 populateFormWithStudentData(student); // Populate form with selected student's data
364             }
365         } catch (SQLException e) {
366             e.printStackTrace();
367             JOptionPane.showMessageDialog(this, "Error loading student data.", "Error", JOptionPane.ERROR_MESSAGE);
368         }
369     } else {
370         JOptionPane.showMessageDialog(this, "Please select a student to update.", "Warning", JOptionPane.WARNING_MESSAGE);
371     }
372 }
373
374 private void updateStudentRecord() {
375     int selectedRow = tblStudents.getSelectedRow();
376     if (selectedRow != -1) {
377         int studentId = (int) tblStudents.getValueAt(selectedRow, 0);
378         try {
379             Student student = studentService.getStudentById(studentId);
380
381             if (student != null) {
382                 String firstName = txtFirstName.getText();
383                 String lastName = txtLastName.getText();
384                 LocalDate dateOfBirth = new java.sql.Date(dateOfBirthChooser.getDate().getTime()).toLocalDate();
385                 String email = txtEmail.getText();
386                 String phoneNumber = txtPhoneNumber.getText();
387                 String address = txtAddress.getText();
388                 LocalDate enrollmentDate = new java.sql.Date(dateOfBirthChooser.getDate().getTime()).toLocalDate();
389                 boolean isActive = chkActive.isSelected();
390
391                 String phoneNumer = txtPhoneNumber.getText();
392                 String address = txtAddress.getText();
393                 LocalDate enrollmentDate = new java.sql.Date(dateOfBirthChooser.getDate().getTime()).toLocalDate();
394                 boolean isActive = chkActive.isSelected();
395
396                 boolean valid = validateFields(firstName, lastName, dateOfBirth, email, phoneNumer, address, enrollmentDate, isActive);
397                 if (!valid) {
398                     return;
399                 }
400
401                 student.setFirstName(firstName);
402                 student.setLastName(lastName);
403                 student.setDateOfBirth(dateOfBirth);
404                 student.setGender((String) cbGender.getSelectedItem());
405                 student.setEmail(email);
406                 student.setPhoneNumber(phoneNumber);
407                 student.setAddress(address);
408                 student.setEnrollmentDate(enrollmentDate);
409                 student.setActive(isActive);
410
411                 studentService.updateStudent(student);
412                 refreshTable();
413
414                 String msg = "Student " + student.getFullName() + " updated successfully!!!";
415                 JOptionPane.showMessageDialog(this, msg, "Success", JOptionPane.INFORMATION_MESSAGE);
416                 resetForm(); // Reset the form after successful update
417             }
418         } catch (SQLException e) {
419             e.printStackTrace();
420             JOptionPane.showMessageDialog(this, "Error updating student.", "Error", JOptionPane.ERROR_MESSAGE);
421         }
422     } else {
423         JOptionPane.showMessageDialog(this, "Please select a student to update.", "Warning", JOptionPane.WARNING_MESSAGE);
424     }
425 }

```

Student Management System

Search

Search

All

Add/Update Student

First Name:

Prabin

Last Name:

Shrestha

Date of Birth:

Aug 12, 2020

Gender:

Male

Email:

aksdfdas

Phone Number:

8687687678

Address:

ktrn

Enrollment Date:

Aug 8, 2024

Active:

☐

Update Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
8	Prabin	Shrestha	2020-08-12	Male	aksdfdas	8687687678	ktrn	2024-08-08	false
10	Shree	Stha	2023-08-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana...	3453453457	ktrn	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana...	345345345...	w34sdf sdf.	2024-08-08	true

-- The form is populated with the selected student data on click on the update button below student list table.  
-- Likewise, the add Student button is changed to update student.  
-- after updating the values and clicking on the update student button of the form, the values will be updated and the table will be refreshed simultaneously.

Update Student

Delete Student

Activate

- ➔ With the "Update Student" functionality, admin user can modify an existing student's details. When a row in the student table is selected, the student's current data is populated into the form fields. The user can then edit these fields and submit the form. Upon submission, the data is validated to ensure correctness. The StudentService then interacts with the StudentDAO to update the student's record in the database using the modified data. After a successful update, the form is reset, the table is refreshed to show the updated information, and the button text reverts to "Add Student."

# # Input validation

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Prabin

Last Name:

Shrestha

Date of Birth:

Aug 12, 2020

📅

Gender:

Male

▼

Email:

aksdffdass

Phone Number:

8687687678

Address:

ktm

Enrollment Date:

Aug 8, 2024

📅

Active:

☐

Update Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
12	Male	aksdffdass	8687687678	ktm	2024-08-08	false			
24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
05	Female	shreejana...	3453453457	ktm	2010-08-05	true			
05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true			

Validation Error

✖

Email is not valid.

OK

Update Student

Delete Student

Activate

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Prabin

Last Name:

Shrestha

Date of Birth:

Aug 12, 2020

📅

Gender:

Male

▼

Email:

test@test.com

Phone Number:

86876876782

Address:

ktm

Enrollment Date:

Aug 8, 2024

📅

Active:

☐

Update Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
12	Male	aksdffdass	8687687678	ktm	2024-08-08	false			
24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false			
05	Female	shreejana...	3453453457	ktm	2010-08-05	true			
05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true			

Validation Error

✖

Phone Number must be 10 digits.

OK

Update Student

Delete Student

Activate

## # Successful update of data

Student Management System

Search

Add/Update Student

First Name: Prabin

Last Name: Shrestha

Date of Birth: Aug 12, 2020

Gender: Male

Email: test@test.com

Phone Number: 8687687678

Address: ktm

Enrollment Date: Aug 8, 2024

Active: ☐

Update Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
				ale	test@test.c...	8687687678	ktm	2020-08-12	false
				ale	shree@tes...	4563214789	kathmandu	2024-08-08	false
				male	shreejana...	3453453457	ktm	2010-08-05	true
				female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true

Success

Student Prabin Shrestha updated successfully!!!

Update Student

Delete Student

Student Management System

Search

Add/Update Student

First Name: Prabin

Last Name: Shrestha

Date of Birth: Aug 12, 2020

Gender: Male

Email: test@test.com

Phone Number: 8687687678

Address: ktm

Enrollment Date: Aug 8, 2024

Active: ☐

Update Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnso...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.princ...	123-789-45...	202 Maple ...	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@...	123-456-78...	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@tes...	2345234787	Nepal	2014-08-15	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.c...	8687687678	ktm	2020-08-12	false
10	Shree	Stha	2023-08-24	Male	shree@tes...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana...	3453453457	ktm	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana...	345345345...	w34sdf sdf...	2024-08-08	true

Success

Student Prabin Shrestha updated successfully!!!

Update Student

Delete Student

-- on successful update, success modal is displayed and similarly the record from the table is also updated.

## # Delete operation

```
422     }
423
424     private void deleteStudent() {
425         int selectedRow = tblStudents.getSelectedRow();
426         if (selectedRow != -1) {
427             int studentId = (int) tblStudents.getValueAt(selectedRow, 0);
428             String fullName = (String) tblStudents.getValueAt(selectedRow, 1) + " " + (String) tblStudents.getValueAt(selectedRow, 2);
429
430             try {
431                 studentService.deleteStudent(studentId);
432                 String msg = "Student " + fullName + " deleted successfully!!!";
433                 JOptionPane.showMessageDialog(this, msg, "Success", JOptionPane.INFORMATION_MESSAGE);
434                 refreshTable();
435             } catch (SQLException e) {
436                 e.printStackTrace();
437                 JOptionPane.showMessageDialog(this, "Error deleting student.", "Error", JOptionPane.ERROR_MESSAGE);
438             }
439         } else {
440             JOptionPane.showMessageDialog(this, "Please select a student to delete.", "Warning", JOptionPane.WARNING_MESSAGE);
441         }
442     }
443 }
```

```
33     }
34
35     @Override
36     public void deleteStudent(int studentId) throws SQLException {
37         String sql = "DELETE FROM student WHERE studentId = ?";
38         try (PreparedStatement statement = connection.prepareStatement(sql)) {
39             statement.setInt(1, studentId);
40             statement.executeUpdate();
41         }
42     }
43 }
```

Student Management System

Search:

**Add/Update Student**

First Name:

Last Name:

Date of Birth:

Gender:

Email:

Phone Number:

Address:

Enrollment Date:

Active: ☐

**Student List**

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Number	Address	Enrollment Date	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson@...	555-123-45	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.prince@...	123-789-45	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78	123 Main St	2024-08-09	false
9	Pabita	Shrestha	2014-08-15	Female	pabita@test.c...	2345234787	Nepal	2014-08-15	false
	ale	test@test.com	8687687678	ktm	2020-08-12	false			
	ale	shree@test.c...	4563214789	kathmandu	2024-08-08	false			
	emale	shreejana@g...	3453453457	ktm	2010-08-05	true			
	emale	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	true			

**Success**

Student Pabita Shrestha deleted successfully!!!

- ➔ The 'Delete Student' functionality remove a student's record from the database permanently. When a user selects a row in the student table and chooses the delete option, the system prompts the user for confirmation to prevent accidental deletions. If confirmed, the StudentService interacts with the StudentDAO to execute a delete operation on the selected student's record in the database. Once the record is deleted, the table is refreshed to reflect the removal, and the form is cleared to reset the interface for the next operation.

## # Table after successful deletion

Student Management System

Search

Search

All

▼

Add/Update Student

First Name:

Last Name:

Date of Birth:

📅

Gender:

Male

▼

Email:

Phone Number:

Address:

Enrollment Date:

📅

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson...	555-123-45...	789 Oak St	2024-09-15	false
5	Diana	Prince	2000-02-14	Female	diana.prince...	123-789-45...	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78...	123 Main St	2024-08-09	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.com	8687687678	ktrn	2020-08-12	false
10	Shree	Stha	2023-08-24	Male	shree@test.c...	4563214789	kathmandu	2024-08-08	false
1	shreejana	stha	2010-08-05	Female	shreejana@g...	3453453457	ktrn	2010-08-05	true
7	shreejana	shrestha	2010-08-05	Female	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	true

Update Student

Delete Student

## # Activate / Deactivate functionality

```
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```



```

67
68
69 @Override
70 public List<Student> getAllStudentsByStatus(String status) throws SQLException {
71     String sql;
72     List<Student> students = new ArrayList<>();
73     Map<String, Boolean> dictionary = new HashMap<String, Boolean>() {{
74         put("Active", true);
75         put("Inactive", false);
76     }};
77
78     if (status.equals("All")) {
79         sql = "SELECT * FROM student ORDER BY firstName";
80     } else {
81         sql = "SELECT * FROM student WHERE isActive = ? ORDER BY firstName";
82     }
83
84     try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
85         if (!status.equals("All")) {
86             preparedStatement.setBoolean(1, dictionary.get(status));
87         }
88
89         try (ResultSet resultSet = preparedStatement.executeQuery()) {
90             while (resultSet.next()) {
91                 Student student = mapRowToStudent(resultSet);
92                 students.add(student);
93             }
94         }
95     }
96
97     return students;
98 }

```

- ➔ We have activate/deactivate functionality and the corresponding logic to show the data in the table based on the status selected from the drop down filter.
- ➔ It allows admin to activate or deactivate the students which can be achieved from the toggle of the button on the bottom section. If the student is currently active, clicking the "Deactivate" button will set the isActive field in the database to false. Conversely, if the student is inactive, clicking the "Activate" button will set the isActive field to true. The system uses the StudentService to call the StudentDAO, which executes the update query to toggle the isActive status in the database.

Student Management System

Search

Search

All

Add/Update Student

First Name:

Last Name:

Date of Birth:

Gender:

Male

Email:

Phone Number:

Address:

Enrollment Date:

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson...	555-123-45...	789 Oak St	2024-09-15	true
5	Diana	Prince	2000-02-14	Female	diana.prince...	123-789-45...	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78...	123 Main St	2024-08-09	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.com	8687687678	ktm	2020-08-12	false
24	Male	shree@test.c...	4563214789	kathmandu	2024-08-08	false			
05	Female	shreejana@g...	3453453457	ktm	2010-08-05	true			
05	Female	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	false			

Success

Student Bob Johnson is Activated

OK

Update Student

Delete Student

Student Management System

Search

Search

All

Add/Update Student

First Name:

Last Name:

Date of Birth:

Gender:

Male

Email:

Phone Number:

Address:

Enrollment Date:

Active:

☐

Add Student

Student List

ID	First Name	Last Name	Date of Birth	Gender	Email	Phone Nu...	Address	Enrollment ...	Active
3	Alice	Smith	1999-11-30	Female	alice.smith@...	987-654-32...	456 Elm St	2024-08-01	true
4	Bob	Johnson	2001-05-22	Male	bob.johnson...	555-123-45...	789 Oak St	2024-09-15	true
5	Diana	Prince	2000-02-14	Female	diana.prince...	123-789-45...	202 Maple St	2024-08-10	false
2	John	Doe	2000-01-15	Male	john.doe@ex...	123-456-78...	123 Main St	2024-08-09	false
8	Prabin	Shrestha	2020-08-12	Male	test@test.com	8687687678	ktm	2020-08-12	false
	Male	shree@test.c...	4563214789	kathmandu	2024-08-08	false			
	Female	shreejana@g...	3453453457	ktm	2010-08-05	true			
	Female	shreejana@g...	345345345...	w34sdf sdfgs	2024-08-08	false			

Success

Student shreejana shrestha is Deactivated

OK

Update Student

Delete Student