

DBConnection.java:

```
package src;
```

```
import javax.swing.*;
```

```
import javax.swing.table.DefaultTableModel;
```

```
import java.awt.*;
```

```
import java.awt.event.*;
```

```
import java.util.List;
```

```
public class StudentManagementSystem extends JFrame {
```

```
    private StudentDAO studentDAO;
```

```
    // Components for "View All" tab
```

```
    private.JTable table;
```

```
    private DefaultTableModel tableModel;
```

```
    // Components for "Add Student" tab
```

```
    private JTextField addNameField, addEmailField, addAgeField, addGradeField;
```

```
// Components for "Update Student" tab

private JTextField updateIdField, updateNameField, updateEmailField, updateAgeField,
updateGradeField;


// Components for "Delete Student" tab

private JTextField deleteIdField;


// Components for "Search Student" tab

private JTextField searchField;

private JTable searchTable;

private DefaultTableModel searchTableModel;


public StudentManagementSystem() {

    studentDAO = new StudentDAO();

    initComponents();

}


private void initComponents() {

    setTitle("Student Management System");

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    setSize(800, 600);

    setLocationRelativeTo(null);

    JTabbedPane tabbedPane = new JTabbedPane();

    JPanel viewPanel = new JPanel(new BorderLayout());

    tableModel = new DefaultTableModel(new String[]{"ID", "Name", "Email", "Age",
"Grade"}, 0);
```

```
table = new JTable(tableModel);  
JScrollPane scrollPane = new JScrollPane(table);  
viewPanel.add(scrollPane, BorderLayout.CENTER);  
JButton refreshButton = new JButton("Refresh");  
refreshButton.addActionListener(e -> loadAllStudents());  
viewPanel.add(refreshButton, BorderLayout.SOUTH);  
tabbedPane.addTab("View All", viewPanel);
```

```
JPanel addPanel = new JPanel(new GridLayout(5, 2, 10, 10));  
addPanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));  
addPanel.add(new JLabel("Name:"));  
addNameField = new JTextField();  
addPanel.add(addNameField);
```

```
addPanel.add(new JLabel("Email:"));  
addEmailField = new JTextField();  
addPanel.add(addEmailField);
```

```
addPanel.add(new JLabel("Age:"));  
addAgeField = new JTextField();  
addPanel.add(addAgeField);
```

```
addPanel.add(new JLabel("Grade:"));  
addGradeField = new JTextField();  
addPanel.add(addGradeField);
```

```
JButton addButton = new JButton("Add Student");  
addButton.addActionListener(e -> addStudent());  
addPanel.add(addButton);
```

```
tabbedPane.addTab("Add Student", addPanel);
```

```
JPanel updatePanel = new JPanel(new GridLayout(6, 2, 10, 10));  
updatePanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));  
updatePanel.add(new JLabel("Student ID:"));  
updateIdField = new JTextField();  
updatePanel.add(updateIdField);
```

```
updatePanel.add(new JLabel("New Name:"));  
updateNameField = new JTextField();  
updatePanel.add(updateNameField);
```

```
updatePanel.add(new JLabel("New Email:"));  
updateEmailField = new JTextField();  
updatePanel.add(updateEmailField);
```

```
updatePanel.add(new JLabel("New Age:"));  
updateAgeField = new JTextField();  
updatePanel.add(updateAgeField);
```

```
updatePanel.add(new JLabel("New Grade:"));  
updateGradeField = new JTextField();
```

```
updatePanel.add(updateGradeField);
```

```
JButton updateButton = new JButton("Update Student");
```

```
updateButton.addActionListener(e -> updateStudent());
```

```
updatePanel.add(updateButton);
```

```
tabbedPane.addTab("Update Student", updatePanel);
```

```
JPanel deletePanel = new JPanel(new GridLayout(2, 2, 10, 10));
```

```
deletePanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));
```

```
deletePanel.add(new JLabel("Student ID:"));
```

```
deletelIdField = new JTextField();
```

```
deletePanel.add(deletelIdField);
```

```
JButton deleteButton = new JButton("Delete Student");
```

```
deleteButton.addActionListener(e -> deleteStudent());
```

```
deletePanel.add(deleteButton);
```

```
tabbedPane.addTab("Delete Student", deletePanel);
```

```
JPanel searchPanel = new JPanel(new BorderLayout());
```

```
JPanel searchInputPanel = new JPanel(new FlowLayout());
```

```
searchInputPanel.add(new JLabel("Enter ID or Name:"));
```

```
searchField = new JTextField(20);
```

```
searchInputPanel.add(searchField);
```

```
JButton searchButton = new JButton("Search");
```

```
searchButton.addActionListener(e -> searchStudent());

searchInputPanel.add(searchButton);

searchPanel.add(searchInputPanel, BorderLayout.NORTH);


searchTableModel = new DefaultTableModel(new String[]{"ID", "Name", "Email", "Age",
"Grade"}, 0);

searchTable = new JTable(searchTableModel);

JScrollPane searchScrollPane = new JScrollPane(searchTable);

searchPanel.add(searchScrollPane, BorderLayout.CENTER);

tabbedPane.addTab("Search Student", searchPanel);


add(tabbedPane);


loadAllStudents();
}
```

```
private void loadAllStudents() {
    tableModel.setRowCount(0);

    List<Student> students = studentDAO.getAllStudents();

    for (Student student : students) {
        tableModel.addRow(new Object[]{
            student.getId(),
            student.getName(),
            student.getEmail(),
            student.getAge(),
            student.getGrade()
        });
    }
}
```

```
    });  
}  
}
```

```
private void addStudent() {  
    String name = addNameField.getText().trim();  
    String email = addEmailField.getText().trim();  
    int age;  
    String grade = addGradeField.getText().trim();  
    try {  
        age = Integer.parseInt(addAgeField.getText().trim());  
    } catch (NumberFormatException e) {  
        JOptionPane.showMessageDialog(this, "Invalid age. Please enter a valid number.");  
        return;  
    }  
}
```

```
Student student = new Student(name, email, age, grade);  
boolean success = studentDAO.insertStudent(student);  
if (success) {  
    JOptionPane.showMessageDialog(this, "Student added successfully.");  
    addNameField.setText("");  
    addEmailField.setText("");  
    addAgeField.setText("");  
    addGradeField.setText("");  
    loadAllStudents();  
} else {
```

```
JOptionPane.showMessageDialog(this, "Failed to add student. Check logs for details.");
```

```
}
```

```
}
```

```
private void updateStudent() {
```

```
    int id;
```

```
    try {
```

```
        id = Integer.parseInt(updateIdField.getText().trim());
```

```
    } catch (NumberFormatException e) {
```

```
        JOptionPane.showMessageDialog(this, "Invalid ID. Please enter a valid number.");
```

```
        return;
```

```
    }
```

```
    String name = updateNameField.getText().trim();
```

```
    String email = updateEmailField.getText().trim();
```

```
    int age;
```

```
    try {
```

```
        age = Integer.parseInt(updateAgeField.getText().trim());
```

```
    } catch (NumberFormatException e) {
```

```
        JOptionPane.showMessageDialog(this, "Invalid age. Please enter a valid number.");
```

```
        return;
```

```
    }
```

```
    String grade = updateGradeField.getText().trim();
```

```
    Student student = new Student(id, name, email, age, grade);
```

```
    boolean success = studentDAO.updateStudent(student);
```



```
if (success) {  
    JOptionPane.showMessageDialog(this, "Student updated successfully.");  
    updateIdField.setText("");  
    updateNameField.setText("");  
    updateEmailField.setText("");  
    updateAgeField.setText("");  
    updateGradeField.setText("");  
    loadAllStudents();  
} else {  
    JOptionPane.showMessageDialog(this, "Failed to update student. Check logs for  
details.");  
}  
}
```

```
private void deleteStudent() {  
    int id;  
    try {  
        id = Integer.parseInt(deletIdField.getText().trim());  
    } catch (NumberFormatException e) {  
        JOptionPane.showMessageDialog(this, "Invalid ID. Please enter a valid number.");  
        return;  
    }  
    boolean success = studentDAO.deleteStudent(id);  
    if (success) {  
        JOptionPane.showMessageDialog(this, "Student deleted successfully.");  
        deletIdField.setText("");  
    }  
}
```

```
        loadAllStudents();
    } else {
        JOptionPane.showMessageDialog(this, "Failed to delete student. Check logs for details.");
    }
}
```

```
private void searchStudent() {
    String keyword = searchField.getText().trim();
    List<Student> students = studentDAO.getStudentByIdOrName(keyword);
    searchTableModel.setRowCount(0);
    for (Student student : students) {
        searchTableModel.addRow(new Object[]{
            student.getId(),
            student.getName(),
            student.getEmail(),
            student.getAge(),
            student.getGrade()
        });
    }
    if (students.isEmpty()) {
        JOptionPane.showMessageDialog(this, "No matching student found.");
    }
}
```

```
public static void main(String[] args) {
```

```
SwingUtilities.invokeLater() -> {  
    new StudentManagementSystem().setVisible(true);  
    });  
}  
}
```

Student.java:

```
package src;  
  
public class Student {  
    private int id;  
    private String name;  
    private String email;  
    private int age;  
    private String grade;  
  
    public Student() {}  
  
    public Student(int id, String name, String email, int age, String grade) {  
        this.id = id;  
        this.name = name;  
        this.email = email;  
        this.age = age;  
        this.grade = grade;  
    }  
  
    public Student(String name, String email, int age, String grade) {  
        this.name = name;
```

```
    this.email = email;

    this.age = age;

    this.grade = grade;
}
```

```
public int getId() { return id; }

public void setId(int id) { this.id = id; }
```

```
public String getName() { return name; }

public void setName(String name) { this.name = name; }
```

```
public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }
```

```
public int getAge() { return age; }

public void setAge(int age) { this.age = age; }
```

```
public String getGrade() { return grade; }

public void setGrade(String grade) { this.grade = grade; }
```

```
@Override
```

```
public String toString() {

    return "Student [ID=" + id + ", Name=" + name + ", Email=" + email

        + ", Age=" + age + ", Grade=" + grade + "]";

}

}
```

StudentDAO.java

```
package src;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class StudentDAO {

    public boolean insertStudent(Student student) {

        String sql = "INSERT INTO students (name, email, age, grade) VALUES (?, ?, ?, ?)";

        try (Connection conn = DBConnection.getConnection();

            PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setString(1, student.getName());
            pstmt.setString(2, student.getEmail());
            pstmt.setInt(3, student.getAge());
            pstmt.setString(4, student.getGrade());

            int rowsAffected = pstmt.executeUpdate();

            return rowsAffected > 0;

        } catch (SQLException | ClassNotFoundException e) {

            e.printStackTrace();

            return false;

        }
    }
}
```

```
}
```

```
// Retrieve and return all students from the database.
```

```
public List<Student> getAllStudents() {
```

```
    List<Student> students = new ArrayList<>();
```

```
    String sql = "SELECT * FROM students";
```

```
    try (Connection conn = DBConnection.getConnection();
```

```
        PreparedStatement pstmt = conn.prepareStatement(sql);
```

```
        ResultSet rs = pstmt.executeQuery()) {
```

```
        while (rs.next()) {
```

```
            Student student = new Student(
```

```
                rs.getInt("id"),
```

```
                rs.getString("name"),
```

```
                rs.getString("email"),
```

```
                rs.getInt("age"),
```

```
                rs.getString("grade")
```

```
            );
```

```
            students.add(student);
```

```
        }
```

```
    } catch (SQLException | ClassNotFoundException e) {
```

```
        e.printStackTrace();
```

```
    }
```

```
    return students;
```

```
}
```

```
public boolean updateStudent(Student student) {
```

```
    String sql = "UPDATE students SET name = ?, email = ?, age = ?, grade = ? WHERE id = ?";
```

```
    try (Connection conn = DBConnection.getConnection();
```

```
        PreparedStatement pstmt = conn.prepareStatement(sql)) {
```

```
        pstmt.setString(1, student.getName());
```

```
        pstmt.setString(2, student.getEmail());
```

```
        pstmt.setInt(3, student.getAge());
```

```
        pstmt.setString(4, student.getGrade());
```

```
        pstmt.setInt(5, student.getId());
```

```
        int rowsAffected = pstmt.executeUpdate();
```

```
        return rowsAffected > 0;
```

```
    } catch (SQLException | ClassNotFoundException e) {
```

```
        e.printStackTrace();
```

```
        return false;
```

```
    }
```

```
}
```

```
public boolean deleteStudent(int id) {
```

```
    String sql = "DELETE FROM students WHERE id = ?";
```

```
    try (Connection conn = DBConnection.getConnection();
```

```

        PreparedStatement pstmt = conn.prepareStatement(sql) {

            pstmt.setInt(1, id);

            int rowsAffected = pstmt.executeUpdate();

            return rowsAffected > 0;

        } catch (SQLException | ClassNotFoundException e) {

            e.printStackTrace();

            return false;

        }
    }

    public List<Student> getStudentByIdOrName(String keyword) {

        List<Student> students = new ArrayList<>();

        String sql = "SELECT * FROM students WHERE id = ? OR name LIKE ?";

        try (Connection conn = DBConnection.getConnection();

            PreparedStatement pstmt = conn.prepareStatement(sql)) {

            try {

                int id = Integer.parseInt(keyword);

                pstmt.setInt(1, id);

            } catch (NumberFormatException ex) {

                pstmt.setInt(1, -1);

            }

            pstmt.setString(2, "%" + keyword + "%");

```



```

try (ResultSet rs = pstmt.executeQuery()) {
    while (rs.next()) {
        Student student = new Student(
            rs.getInt("id"),
            rs.getString("name"),
            rs.getString("email"),
            rs.getInt("age"),
            rs.getString("grade")
        );
        students.add(student);
    }
}

} catch (SQLException | ClassNotFoundException e) {
    e.printStackTrace();
}

return students;
}
}

```

StudentManagementSystem.java

```

package src;

import javax.swing.*;

import javax.swing.table.DefaultTableModel;

```

```
import java.awt.*;

import java.awt.event.*;

import java.util.List;


public class StudentManagementSystem extends JFrame {

    private StudentDAO studentDAO;


    // Components for "View All" tab

    private JTable table;

    private DefaultTableModel tableModel;


    // Components for "Add Student" tab

    private JTextField addNameField, addEmailField, addAgeField, addGradeField;


    // Components for "Update Student" tab

    private JTextField updateIdField, updateNameField, updateEmailField, updateAgeField,
updateGradeField;


    // Components for "Delete Student" tab

    private JTextField deleteIdField;


    // Components for "Search Student" tab

    private JTextField searchField;

    private JTable searchTable;

    private DefaultTableModel searchTableModel;
```

```
public StudentManagementSystem() {  
    studentDAO = new StudentDAO();  
    initComponents();  
}
```

```
private void initComponents() {  
    setTitle("Student Management System");  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setSize(800, 600);  
    setLocationRelativeTo(null);  
    JTabbedPane tabbedPane = new JTabbedPane();  
    JPanel viewPanel = new JPanel(new BorderLayout());  
    tableModel = new DefaultTableModel(new String[]{"ID", "Name", "Email", "Age",  
"Grade"}, 0);  
    table = new JTable(tableModel);  
    JScrollPane scrollPane = new JScrollPane(table);  
    viewPanel.add(scrollPane, BorderLayout.CENTER);  
    JButton refreshButton = new JButton("Refresh");  
    refreshButton.addActionListener(e -> loadAllStudents());  
    viewPanel.add(refreshButton, BorderLayout.SOUTH);  
    tabbedPane.addTab("View All", viewPanel);  
  
    JPanel addPanel = new JPanel(new GridLayout(5, 2, 10, 10));  
    addPanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));  
    addPanel.add(new JLabel("Name:"));  
    addNameField = new JTextField();
```

```
addPanel.add(addNameField);
```

```
addPanel.add(new JLabel("Email:"));
```

```
addEmailField = new JTextField();
```

```
addPanel.add(addEmailField);
```

```
addPanel.add(new JLabel("Age:"));
```

```
addAgeField = new JTextField();
```

```
addPanel.add(addAgeField);
```

```
addPanel.add(new JLabel("Grade:"));
```

```
addGradeField = new JTextField();
```

```
addPanel.add(addGradeField);
```

```
JButton addButton = new JButton("Add Student");
```

```
addButton.addActionListener(e -> addStudent());
```

```
addPanel.add(addButton);
```

```
tabbedPane.addTab("Add Student", addPanel);
```

```
JPanel updatePanel = new JPanel(new GridLayout(6, 2, 10, 10));
```

```
updatePanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));
```

```
updatePanel.add(new JLabel("Student ID:"));
```

```
updateIdField = new JTextField();
```

```
updatePanel.add(updateIdField);
```

```
updatePanel.add(new JLabel("New Name:"));  
updateNameField = new JTextField();  
updatePanel.add(updateNameField);
```

```
updatePanel.add(new JLabel("New Email:"));  
updateEmailField = new JTextField();  
updatePanel.add(updateEmailField);
```

```
updatePanel.add(new JLabel("New Age:"));  
updateAgeField = new JTextField();  
updatePanel.add(updateAgeField);
```

```
updatePanel.add(new JLabel("New Grade:"));  
updateGradeField = new JTextField();  
updatePanel.add(updateGradeField);
```

```
JButton updateButton = new JButton("Update Student");  
updateButton.addActionListener(e -> updateStudent());  
updatePanel.add(updateButton);
```

```
tabbedPane.addTab("Update Student", updatePanel);
```

```
JPanel deletePanel = new JPanel(new GridLayout(2, 2, 10, 10));  
deletePanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));  
deletePanel.add(new JLabel("Student ID:"));  
deleteIdField = new JTextField();
```

```
deletePanel.add(deleteIdField);
```

```
JButton deleteButton = new JButton("Delete Student");
```

```
deleteButton.addActionListener(e -> deleteStudent());
```

```
deletePanel.add(deleteButton);
```

```
tabbedPane.addTab("Delete Student", deletePanel);
```

```
JPanel searchPanel = new JPanel(new BorderLayout());
```

```
JPanel searchInputPanel = new JPanel(new FlowLayout());
```

```
searchInputPanel.add(new JLabel("Enter ID or Name:"));
```

```
searchField = new JTextField(20);
```

```
searchInputPanel.add(searchField);
```

```
JButton searchButton = new JButton("Search");
```

```
searchButton.addActionListener(e -> searchStudent());
```

```
searchInputPanel.add(searchButton);
```

```
searchPanel.add(searchInputPanel, BorderLayout.NORTH);
```

```
searchTableModel = new DefaultTableModel(new String[]{"ID", "Name", "Email", "Age",  
"Grade"}, 0);
```

```
searchTable = new JTable(searchTableModel);
```

```
JScrollPane searchScrollPane = new JScrollPane(searchTable);
```

```
searchPanel.add(searchScrollPane, BorderLayout.CENTER);
```

```
tabbedPane.addTab("Search Student", searchPanel);
```

```
add(tabbedPane);
```

```
loadAllStudents();  
}
```

```
private void loadAllStudents() {  
    tableModel.setRowCount(0);  
    List<Student> students = studentDAO.getAllStudents();  
    for (Student student : students) {  
        tableModel.addRow(new Object[]{  
            student.getId(),  
            student.getName(),  
            student.getEmail(),  
            student.getAge(),  
            student.getGrade()  
        });  
    }  
}
```

```
private void addStudent() {  
    String name = addNameField.getText().trim();  
    String email = addEmailField.getText().trim();  
    int age;  
    String grade = addGradeField.getText().trim();  
    try {  
        age = Integer.parseInt(addAgeField.getText().trim());  
    } catch (NumberFormatException e) {
```

```
JOptionPane.showMessageDialog(this, "Invalid age. Please enter a valid number.");  
return;  
}
```

```
Student student = new Student(name, email, age, grade);  
boolean success = studentDAO.insertStudent(student);  
if (success) {  
    JOptionPane.showMessageDialog(this, "Student added successfully.");  
    addNameField.setText("");  
    addEmailField.setText("");  
    addAgeField.setText("");  
    addGradeField.setText("");  
    loadAllStudents();  
} else {  
    JOptionPane.showMessageDialog(this, "Failed to add student. Check logs for  
details.");  
}  
}
```

```
private void updateStudent() {  
    int id;  
    try {  
        id = Integer.parseInt(updateIdField.getText().trim());  
    } catch (NumberFormatException e) {  
        JOptionPane.showMessageDialog(this, "Invalid ID. Please enter a valid number.");  
        return;  
    }  
}
```



```
}

String name = updateNameField.getText().trim();

String email = updateEmailField.getText().trim();

int age;

try {

    age = Integer.parseInt(updateAgeField.getText().trim());

} catch (NumberFormatException e) {

    JOptionPane.showMessageDialog(this, "Invalid age. Please enter a valid number.");

    return;

}

String grade = updateGradeField.getText().trim();


Student student = new Student(id, name, email, age, grade);

boolean success = studentDAO.updateStudent(student);

if (success) {

    JOptionPane.showMessageDialog(this, "Student updated successfully.");

    updateIdField.setText("");

    updateNameField.setText("");

    updateEmailField.setText("");

    updateAgeField.setText("");

    updateGradeField.setText("");

    loadAllStudents();

} else {

    JOptionPane.showMessageDialog(this, "Failed to update student. Check logs for details.");

}
```

```
}
```

```
private void deleteStudent() {  
    int id;  
    try {  
        id = Integer.parseInt(deleteldField.getText().trim());  
    } catch (NumberFormatException e) {  
        JOptionPane.showMessageDialog(this, "Invalid ID. Please enter a valid number.");  
        return;  
    }  
    boolean success = studentDAO.deleteStudent(id);  
    if (success) {  
        JOptionPane.showMessageDialog(this, "Student deleted successfully.");  
        deleteldField.setText("");  
        loadAllStudents();  
    } else {  
        JOptionPane.showMessageDialog(this, "Failed to delete student. Check logs for details.");  
    }  
}
```

```
private void searchStudent() {  
    String keyword = searchField.getText().trim();  
    List<Student> students = studentDAO.getStudentByIdOrName(keyword);  
    searchTableModel.setRowCount(0);  
    for (Student student : students) {
```

```
searchTableModel.addRow(new Object[]{
    student.getId(),
    student.getName(),
    student.getEmail(),
    student.getAge(),
    student.getGrade()
});
}
if (students.isEmpty()) {
    JOptionPane.showMessageDialog(this, "No matching student found.");
}
}
```

```
public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        new StudentManagementSystem().setVisible(true);
    });
}
}
```

OUTPUT:

Student Management System				
View All Add Student Update Student Delete Student Search Student				
ID	Name	Email	Age	Grade
1	Md Noorullah Khn	md2079@davnepal.edu.np	24	A
2	Amit Mahato	amit1@gmail.com	21	B
3	Jyoti Yadav	jyoti123@gmail.com	22	A
4	Sujina Maharjan	anjsnamarly@gmail.com	21	F

Insert:

Student Management System	
View All Add Student Update Student Delete Student Search Student	
Name:	Yubraj Adhikari
Email:	yubraj@davnepal.edu.np
Age:	22
Grade:	A
Add Student	

After Insert:

Student Management System				
View All Add Student Update Student Delete Student Search Student				
ID	Name	Email	Age	Grade
1	Md Noorullah Khn	md2079@davnepal.edu.np	24	A
2	Amit Mahato	amit1@gmail.com	21	B
3	Jyoti Yadav	jyoti123@gmail.com	22	A
4	Sujina Maharjan	anjsnamarly@gmail.com	21	F
5	Yubraj Adhikari	yubraj@davnepal.edu.np	22	A

Update:

Student Management System	
View All Add Student Update Student Delete Student Search Student	
Student ID:	2
New Name:	Amit Kushuwa
New Email:	amit@davnepal.edu.np
New Age:	26
New Grade:	D
Update Student	

After Update:

Student Management System

View All	Add Student	Update Student	Delete Student	Search Student
ID	Name	Email	Age	Grade
1	Md Noorullah Khn	md2079@davnepal.edu.np	24	A
2	Amit Kushuwa	amit@davnepal.edu.np	26	D
3	Jyoti Yadav	jyoti123@gmail.com	22	A
4	Sujina Maharjan	anjsnamarly@gmail.com	21	F
5	Yubraj Adhikari	yubraj@davnepal.edu.np	22	A

Delete:

Student Management System

View All Add Student Update Student Delete Student Search Student

Student ID: 3

Message

Student deleted successfully.

OK


Delete Student

After Delete:

Student Management System

View All	Add Student	Update Student	Delete Student	Search Student
ID	Name	Email	Age	Grade
1	Md Noorullah Khn	md2079@davnepal.edu.np	24	A
2	Amit Kushuwa	amit@davnepal.edu.np	26	D
4	Sujina Maharjan	anjsnamarly@gmail.com	21	F
5	Yubraj Adhikari	yubraj@davnepal.edu.np	22	A

Search:

 Student Management System— □ ×

View AllAdd StudentUpdate StudentDelete StudentSearch Student

Enter ID or Name:

ID	Name	Email	Age	Grade
2	Amit Kushuwa	amit@davnepal.edu.np	26	D