OBE IMPLEMENTATION:UNIVERSITY SETTING

by

SYNTAX SQUAD
K.PRANAV RITHWIK[AP22110010337]
CH.YASWANTH[AP22110010383]
P.BALAJI[AP22110010381]
HASITH NAGENDRA[AP22110010359]
KOLLIPARA VISHNU[AP22110010391]

A report for the CS307:Mobile Application Development using JAVA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SRM UNIVERSITY AP::AMARAVATI

INDEX

Introduction

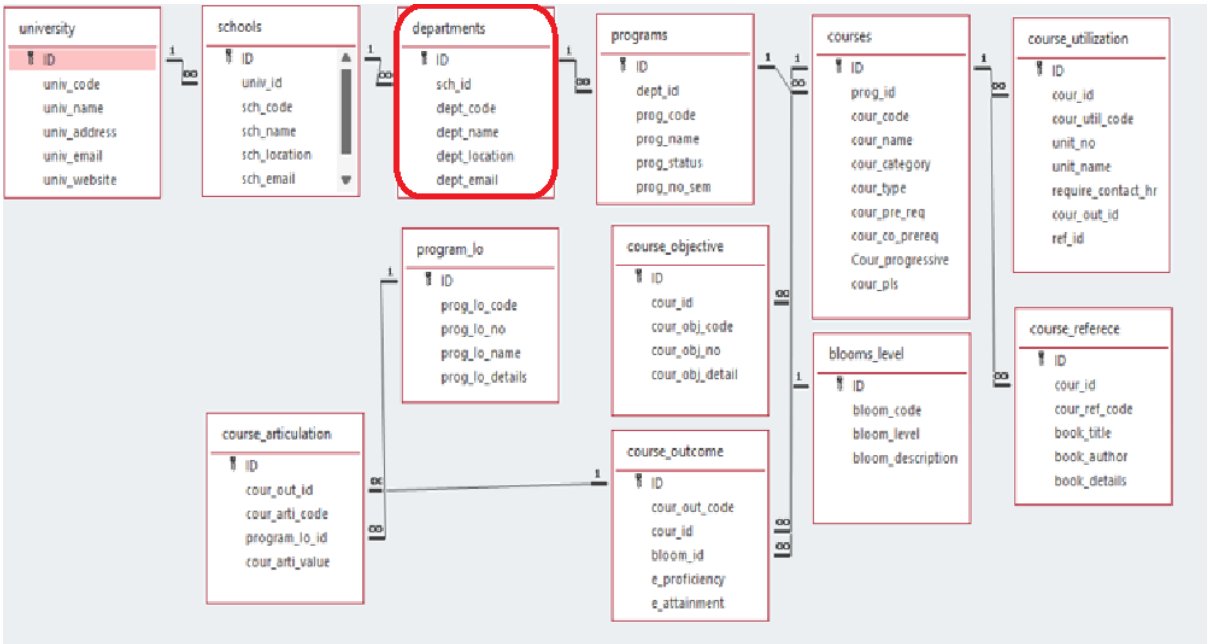
Our University (herewith considered as SRM-AP) is going to implement OBE(Outcome Based Education) in their university and you are assigned in the project to develop a CURD(Create,Update,Retrieve and Delete) windows and mobile application using JAVA programming and Android studio for the same.

Project Modules:

Various Modules available in the project are

1.Blooms Level setting
2.Program Level Objective Setting
3.University
4.Schools
5.Department
6.Programs
7.Courses
8.Course objective setting
9.Course Outcome Setting
10.Course Articulation matrix Setting
11.Course Utilization Setting
12.Course Reference Setting.

Architecture Diagram



Module Description

Module Name: Departments
Module Description:

This module is used to create,Update,Retrieve,Delete(hereafter known as CURD) details of the module and storing the details in the database table(eg.MySQL).

Programming Details

naming conventions to be used:

- **class name/activity name:**syntaxsquad_departments
- **Function/method name**
  - **Create:**AP22110010337_departments _create
  - **Update:**AP22110010337_departments_update
  - **Retrieve:**AP22110010337_departments_retrive
  - **Delete:**AP22110010337_departments_delete

Table details: departments

Field Name	Data Type
ID	Integer
sch_id	Integer
dept_code	String
dept_name	String
dept_location	String
Dept_email	String

SOURCE CODE

Login.java:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener {
    private JTextField txtUsername;
    private JPasswordField txtPassword;
    private JButton btnLogin, btnReset;

    public Login() {
        setTitle("Login");
        setSize(400, 250);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setResizable(false);

        // Main panel setup
        JPanel panel = new JPanel(new GridBagLayout());
        panel.setBorder(BorderFactory.createEmptyBorder(20, 30, 20, 30));
        panel.setBackground(new Color(245, 245, 245));

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(10, 10, 10, 10);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        Font labelFont = new Font("Segoe UI", Font.BOLD, 14);
        Font fieldFont = new Font("Segoe UI", Font.PLAIN, 13);

        JLabel lblUsername = new JLabel("Username:");
        lblUsername.setFont(labelFont);
        gbc.gridx = 0; gbc.gridy = 0;
        panel.add(lblUsername, gbc);

        txtUsername = new JTextField(15);
        txtUsername.setFont(fieldFont);
        txtUsername.setBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY));
        gbc.gridx = 1;
        panel.add(txtUsername, gbc);

        JLabel lblPassword = new JLabel("Password:");
        lblPassword.setFont(labelFont);
        gbc.gridx = 0; gbc.gridy = 1;
        panel.add(lblPassword, gbc);

        txtPassword = new JPasswordField(15);
        txtPassword.setFont(fieldFont);
```

```
        txtPassword.setBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY));
        gbc.gridx = 1;
        panel.add(txtPassword, gbc);

        btnLogin = createButton("Login");
        gbc.gridx = 0; gbc.gridy = 2;
        panel.add(btnLogin, gbc);

        btnReset = createButton("Reset");
        gbc.gridx = 1;
        panel.add(btnReset, gbc);

        btnLogin.addActionListener(this);
        btnReset.addActionListener(this);

        setContentPane(panel);
        setVisible(true);
    }

    private JButton createButton(String text) {
        JButton btn = new JButton(text);
        btn.setFocusPainted(false);
        btn.setBackground(new Color(66, 133, 244));
        btn.setForeground(Color.WHITE);
        btn.setFont(new Font("Segoe UI", Font.BOLD, 13));
        btn.setPreferredSize(new Dimension(120, 35));
        return btn;
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == btnLogin) {
            String uname = txtUsername.getText().trim();
            String pwd = new String(txtPassword.getPassword()).trim();

            if (uname.isEmpty() || pwd.isEmpty()) {
                JOptionPane.showMessageDialog(this, "Please fill in all fields.");
                return;
            }

            try (Connection conn =
DriverManager.getConnection("jdbc:sqlite:C:/Users/vamsi/Desktop/Apps/javaapp.db");
                 PreparedStatement stmt = conn.prepareStatement("SELECT * FROM users
WHERE uname=? AND pwd=?")) {

                stmt.setString(1, uname);
                stmt.setString(2, pwd);
                ResultSet rs = stmt.executeQuery();

                if (rs.next()) {
                    JOptionPane.showMessageDialog(this, "Login Successful");
                    dispose(); // close login window
                    new syntaxsquad_departments(); // open main UI
                } else {
```

```
                JOptionPane.showMessageDialog(this, "Invalid credentials");
            }

        } catch (SQLException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this, "Database error: " + ex.getMessage());
        }
    } else if (e.getSource() == btnReset) {
        txtUsername.setText("");
        txtPassword.setText("");
    }
  }

  public static void main(String[] args) {
    SwingUtilities.invokeLater(Login::new);
  }
}
```

SyntaxSquad_departments.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class SyntaxSquad_departments extends JFrame {
  private static final String URL = "jdbc:sqlite:C:/Users/vamsi/Desktop/Apps/javaapp.db";
  private JTextField txtDeptId, txtDeptCode, txtDeptName, txtDeptLocation, txtDeptEmail,
txtSchId;
  private JTextArea txtDisplay;
  public SyntaxSquad_departments() {
    setTitle("Department Manager");
    setSize(950, 700);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    JPanel content = new JPanel(new BorderLayout(15, 15));
    content.setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15));
    content.setBackground(new Color(245, 245, 245));

    JPanel formPanel = new JPanel(new GridBagLayout());
    formPanel.setBorder(BorderFactory.createTitledBorder("Department Form"));
    formPanel.setBackground(Color.WHITE);

    GridBagConstraints gbc = new GridBagConstraints();
    gbc.insets = new Insets(10, 10, 10, 10);
```

```
        gbc.fill = GridBagConstraints.HORIZONTAL;

        Font labelFont = new Font("Segoe UI", Font.BOLD, 14);
        Font fieldFont = new Font("Segoe UI", Font.PLAIN, 13);

        int y = 0;
        formPanel.add(createLabel("Department ID:", labelFont), gbc(0, y));
        txtDeptId = createTextField(fieldFont); formPanel.add(txtDeptId, gbc(1, y++));

        formPanel.add(createLabel("Dept Code:", labelFont), gbc(0, y));
        txtDeptCode = createTextField(fieldFont); formPanel.add(txtDeptCode, gbc(1, y++));

        formPanel.add(createLabel("Dept Name:", labelFont), gbc(0, y));
        txtDeptName = createTextField(fieldFont); formPanel.add(txtDeptName, gbc(1, y++));

        formPanel.add(createLabel("Location:", labelFont), gbc(0, y));
        txtDeptLocation = createTextField(fieldFont); formPanel.add(txtDeptLocation, gbc(1,
y++));

        formPanel.add(createLabel("Email:", labelFont), gbc(0, y));
        txtDeptEmail = createTextField(fieldFont); formPanel.add(txtDeptEmail, gbc(1, y++));

        formPanel.add(createLabel("School ID:", labelFont), gbc(0, y));
        txtSchId = createTextField(fieldFont); formPanel.add(txtSchId, gbc(1, y++));

        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 20, 10));
        buttonPanel.setBackground(new Color(245, 245, 245));

        JButton btnAdd = createButton("Add");
        JButton btnRetrieve = createButton("Retrieve");
        JButton btnUpdate = createButton("Update");
        JButton btnDelete = createButton("Delete");

        buttonPanel.add(btnAdd);
        buttonPanel.add(btnRetrieve);
        buttonPanel.add(btnUpdate);
        buttonPanel.add(btnDelete);

        txtDisplay = new JTextArea(15, 60);
        txtDisplay.setFont(new Font("Consolas", Font.PLAIN, 13));
        txtDisplay.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(txtDisplay);
        scrollPane.setBorder(BorderFactory.createTitledBorder("Department Records"));

        content.add(formPanel, BorderLayout.NORTH);
        content.add(buttonPanel, BorderLayout.CENTER);
```

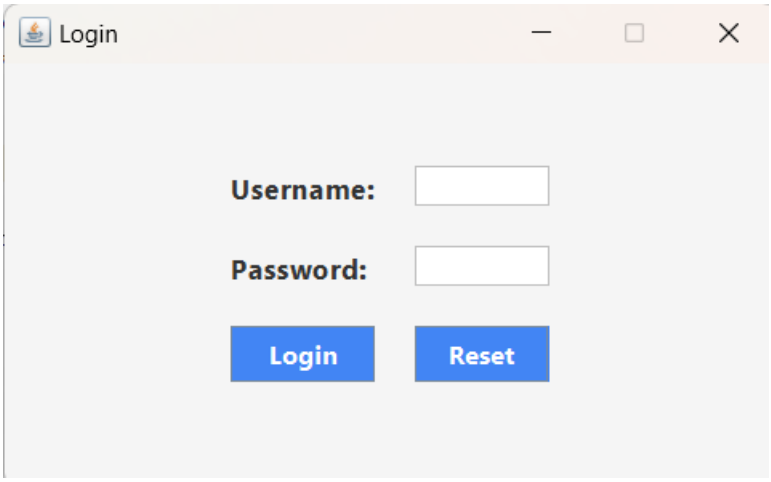
```
    content.add(scrollPane, BorderLayout.SOUTH);
    setContentPane(content);

    btnAdd.addActionListener(e -> AP22110010337_departments_create());
    btnRetrieve.addActionListener(e -> AP22110010337_departments_retrive());
    btnUpdate.addActionListener(e -> AP22110010337_departments_update());
    btnDelete.addActionListener(e -> AP22110010337_departments_delete());

    setVisible(true);
}

private GridBagConstraints gbc(int x, int y) {
    GridBagConstraints gbc = new GridBagConstraints();
    gbc.gridx = x;
    gbc.gridy = y;
    gbc.insets = new Insets(8, 8, 8, 8);
    gbc.fill = GridBagConstraints.HORIZONTAL;
    return gbc;
}

private JLabel createLabel(String text, Font font) {
    JLabel label = new JLabel(text);
    label.setFont(font);
    return label;
}

private JTextField createTextField(Font font) {
    JTextField tf = new JTextField(20);
    tf.setFont(font);
    tf.setBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY));
    return tf;
}

private JButton createButton(String text) {
    JButton btn = new JButton(text);
    btn.setFocusPainted(false);
    btn.setBackground(new Color(66, 133, 244));
    btn.setForeground(Color.WHITE);
    btn.setFont(new Font("Segoe UI", Font.BOLD, 13));
    btn.setPreferredSize(new Dimension(130, 40));
    return btn;
}

private Connection connect() {
    try {
        Class.forName("org.sqlite.JDBC");
```

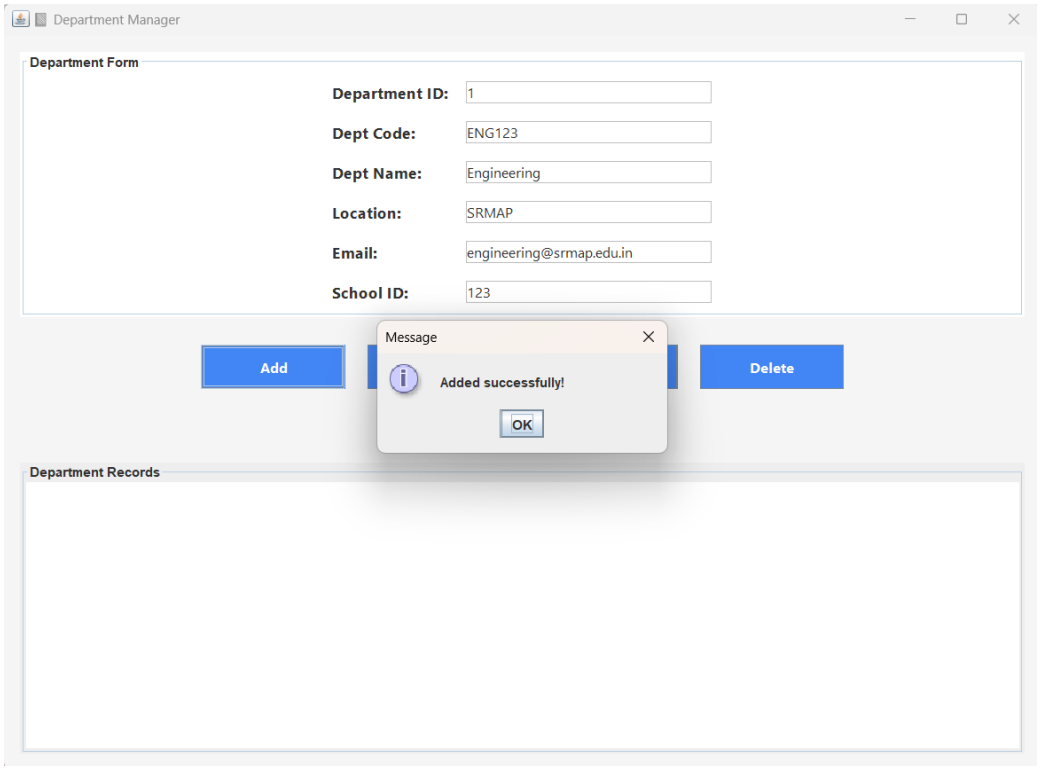
```
            return DriverManager.getConnection(URL);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "Connection Error: " + e.getMessage());
            return null;
        }
    }

    private void clearFields() {
        txtDeptId.setText("");
        txtDeptCode.setText("");
        txtDeptName.setText("");
        txtDeptLocation.setText("");
        txtDeptEmail.setText("");
        txtSchId.setText("");
    }

    private boolean validateIdField() {
        if (txtDeptId.getText().trim().isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please enter Department ID.");
            return false;
        }
        try {
            Integer.parseInt(txtDeptId.getText());
            return true;
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this, "Department ID must be numeric.");
            return false;
        }
    }

    private void AP22110010337_departments_create() {
        String sql = "INSERT INTO departments (dept_code, dept_name, dept_location,
dept_email, sch_id) VALUES (?, ?, ?, ?, ?)";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(sql))
{
            ps.setString(1, txtDeptCode.getText());
            ps.setString(2, txtDeptName.getText());
            ps.setString(3, txtDeptLocation.getText());
            ps.setString(4, txtDeptEmail.getText());
            ps.setInt(5, Integer.parseInt(txtSchId.getText()));
            ps.executeUpdate();
            JOptionPane.showMessageDialog(this, "Added successfully!");
            clearFields();
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Insert Error: " + e.getMessage());
        }
```

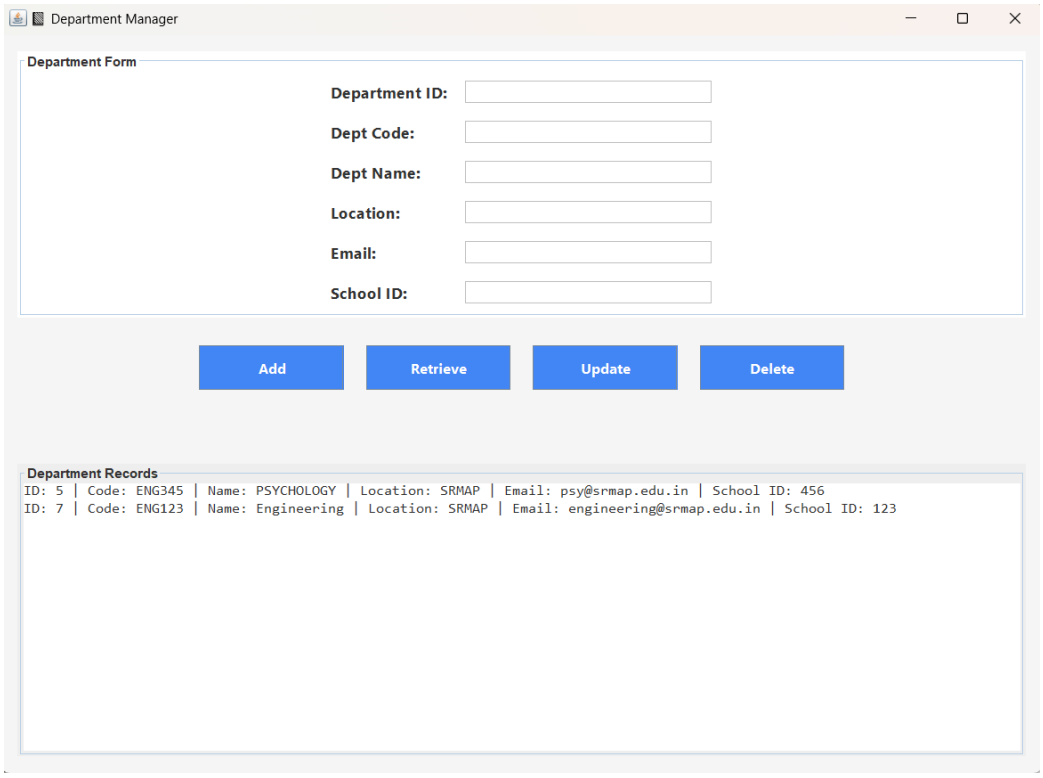
```
    }

    private void AP22110010337_departments_retrive() {
        String sql = "SELECT * FROM departments";
        txtDisplay.setText("");
        try (Connection conn = connect(); Statement stmt = conn.createStatement(); ResultSet rs
= stmt.executeQuery(sql)) {
            while (rs.next()) {
                txtDisplay.append("ID: " + rs.getInt("ID") +
                        " | Code: " + rs.getString("dept_code") +
                        " | Name: " + rs.getString("dept_name") +
                        " | Location: " + rs.getString("dept_location") +
                        " | Email: " + rs.getString("dept_email") +
                        " | School ID: " + rs.getInt("sch_id") + "\n");
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Fetch Error: " + e.getMessage());
        }
    }

    private void AP22110010337_departments_update() {
        if (!validateIdField()) return;
        String sql = "UPDATE departments SET dept_code=?, dept_name=?, dept_location=?,
dept_email=?, sch_id=? WHERE ID=?";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(sql))
{
            ps.setString(1, txtDeptCode.getText());
            ps.setString(2, txtDeptName.getText());
            ps.setString(3, txtDeptLocation.getText());
            ps.setString(4, txtDeptEmail.getText());
            ps.setInt(5, Integer.parseInt(txtSchId.getText()));
            ps.setInt(6, Integer.parseInt(txtDeptId.getText()));
            int rows = ps.executeUpdate();
            if (rows > 0) {
                JOptionPane.showMessageDialog(this, "Updated successfully!");
                clearFields();
            } else {
                JOptionPane.showMessageDialog(this, "No matching record found.");
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Update Error: " + e.getMessage());
        }
    }

    private void AP22110010337_departments_delete() {
        if (!validateIdField()) return;
```

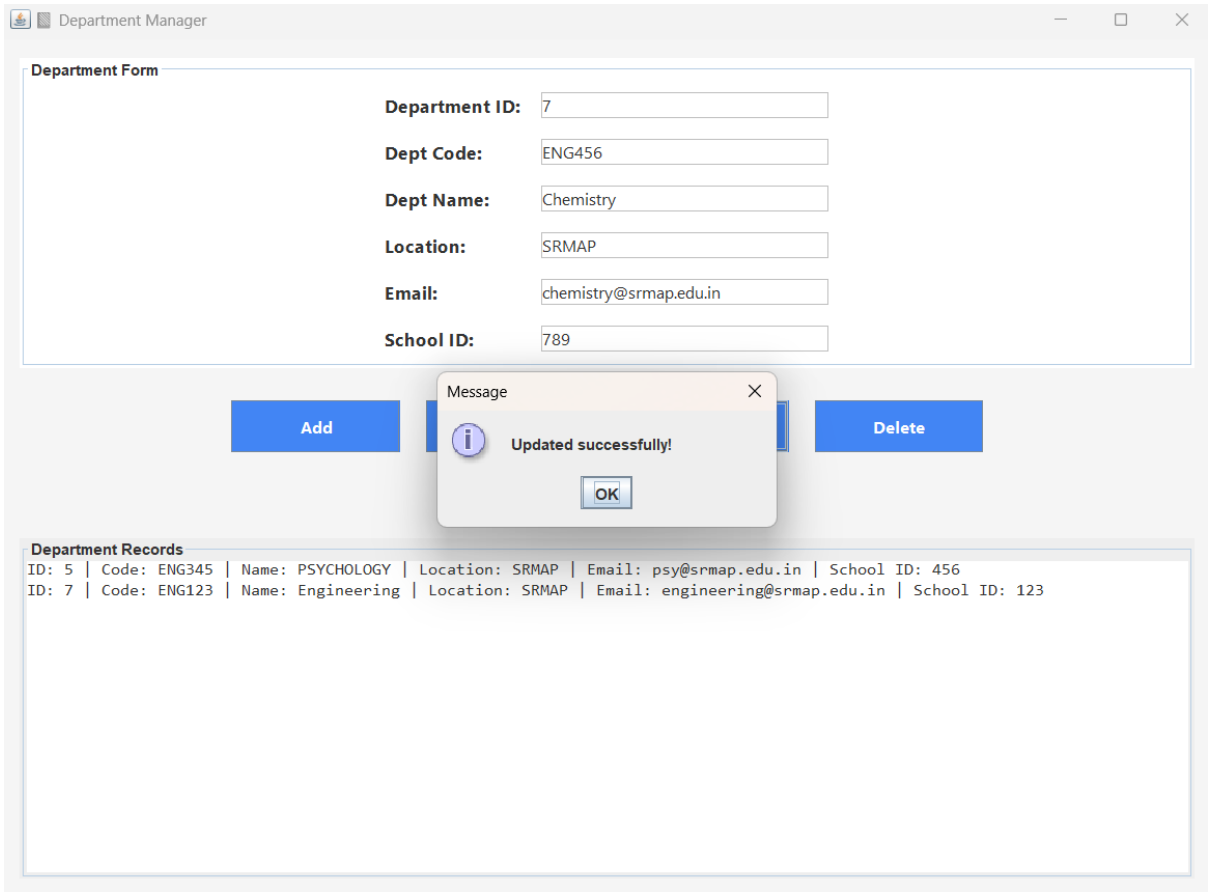
```
        String sql = "DELETE FROM departments WHERE ID=?";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(sql))
{
            ps.setInt(1, Integer.parseInt(txtDeptId.getText()));
            int rows = ps.executeUpdate();
            if (rows > 0) {
                JOptionPane.showMessageDialog(this, "Deleted successfully!");
                clearFields();
            } else {
                JOptionPane.showMessageDialog(this, "No matching record found.");
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Delete Error: " + e.getMessage());
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(SyntaxSquad_departments::new);
    }
}
```

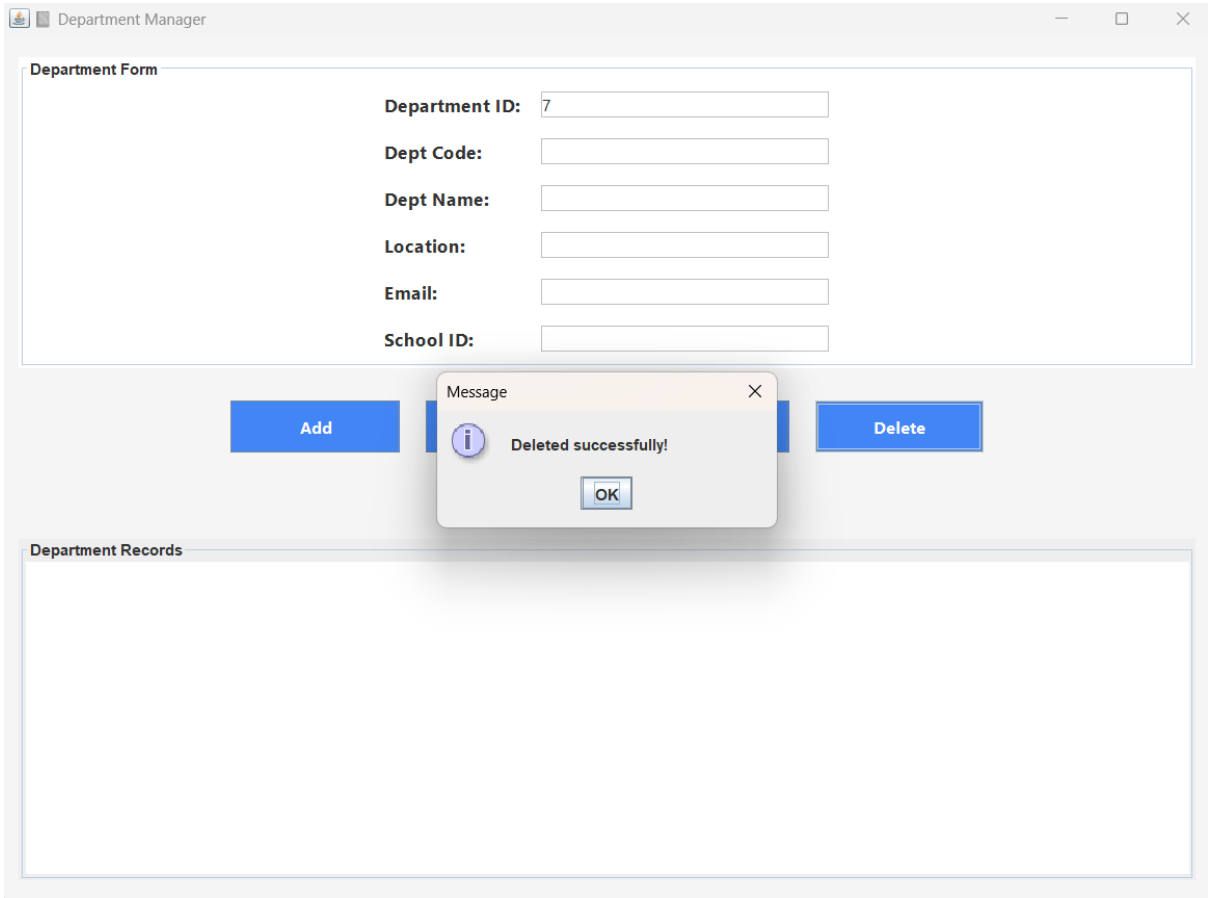
Screen Shots




Department Manager
Department Form
Department ID:
1
Dept Code:
ENG123
Dept Name:
Engineering
Location:
SRMAP
Email:
engineering@srmap.edu.in
School ID:
123
Add
Delete
Message
Added successfully!
OK
Department Records


Department Manager
Department Form
Department ID:
Dept Code:
Dept Name:
Location:
Email:
School ID:
Add
Retrieve
Update
Delete
Department Records
ID: 5 | Code: ENG345 | Name: PSYCHOLOGY | Location: SRMAP | Email: psy@srmap.edu.in | School ID: 456
ID: 7 | Code: ENG123 | Name: Engineering | Location: SRMAP | Email: engineering@srmap.edu.in | School ID: 123


Department Manager
Department Form
Department ID: 7
Dept Code: ENG456
Dept Name: Chemistry
Location: SRMAP
Email: chemistry@srmap.edu.in
School ID: 789
Add
Delete
Message
Updated successfully!
OK
Department Records
ID: 5 | Code: ENG345 | Name: PSYCHOLOGY | Location: SRMAP | Email: psy@srmap.edu.in | School ID: 456
ID: 7 | Code: ENG123 | Name: Engineering | Location: SRMAP | Email: engineering@srmap.edu.in | School ID: 123


Department Manager
Department Form
Department ID: 7
Dept Code:
Dept Name:
Location:
Email:
School ID:
Add
Delete
Message
Deleted successfully!
OK
Department Records

Conclusion

This project successfully demonstrates the design and development of a desktop-based Academic Department Management System using Java Swing and SQLite. It incorporates essential CRUD (Create, Read, Update, Delete) functionalities that allow users to manage departmental records efficiently. The integration of a secure login system adds an additional layer of authentication, ensuring that only authorized users can access and manipulate the data.The use of Java Swing provided a user-friendly graphical interface, while SQLite offered a lightweight, yet powerful, backend database suitable for standalone applications.Overall, the project lays a strong foundation for further development, such as adding roles (Admin/User), enhancing UI design, implementing validation checks, and scaling up to networked or cloud-based systems for larger institutions.