OBE IMPLEMENTATION: SCHOOL SETTING

by

PEKETI JASWANTH SRI PHANIDAR REDDY [AP22110010304]

YUVA KOMARA [AP221100100327]

KATRAGADDA PRUDHVI RAJU [AP22110010307]

USSV SATHWIK NAIDU [AP22110010293]

NIKITHA SRI KOMMALAPATI [AP22110010351]

VISHNU PRIYA VAYYA [AP22110010390]

A report for the CS307:Mobile Application Development using JAVA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SRM UNIVERSITY AP:: AMARAVATI

INDEX

INTRODUCTION

Outcome-Based Education (OBE) is a student-centric teaching and learning model that focuses on measuring student performance through outcomes. It emphasizes the achievement of specific competencies at the end of educational experiences. SRM University - Andhra Pradesh has adopted the OBE framework across its academic infrastructure to align its curriculum and evaluation system with international standards.

The purpose of this document is to present a detailed project report on the development of the "Schools" module, which is a part of the comprehensive OBE Implementation System. The Schools module is one of the foundational layers of the system, enabling the management of various schools under the university's administration. Developed using Java and integrated with a MySQL database, this module supports Create, Update, Retrieve, and Delete (CURD) operations for school entities.

Project Modules

The complete OBE system comprises multiple modules, each handling a distinct component of the academic and outcome assessment process. These modules are:

1. Blooms Level Setting
2. Program Level Objective Setting
3. University
4. **Schools (Current Module)**
5. Departments
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

Each of these modules works in tandem with one another to deliver a cohesive academic management experience. The Schools module, in particular, plays a crucial role in managing the academic structure by facilitating the administration of different schools within SRM-AP.

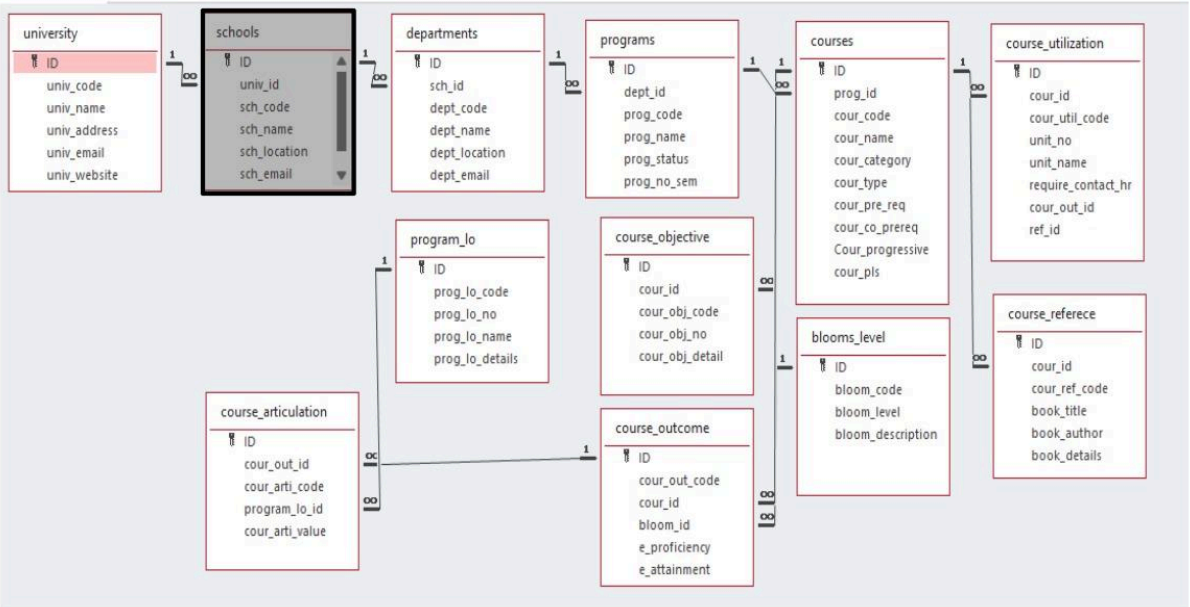
ARCHITECTURE DIAGRAM

The architecture of the system follows a modular and layered design pattern. Each module is loosely coupled with others, which allows for independent development and deployment. The Schools module interacts primarily with the University module and serves as a prerequisite for the Departments module.

- **Frontend**: Java Swing (AWT)
- **Backend**: JDBC for database connectivity
- **Database**: MySQL

Key features of the architecture:

- User interface built with Java Swing
- CRUD operations handled via event-driven programming
- Data persisted in MySQL with standard queries
- Reusable components for uniform design



Module Description

Module Name: Schools

Objective: The primary goal of the Schools module is to enable university staff to **create**, **retrieve**, **update**, and **delete** (CRUD) information associated with schools under SRM-AP. This includes storing metadata such as the school code, name, location, email, and its link to the university entity via a unique university ID (uni_id).

Functionalities:

- Add new school details
- Update existing school information
- Delete school records
- Retrieve and display school details based on school code

Inter- Module Dependency:

- Depends on University module (for uni_id linkage)
- Provides data to Departments module (for sch_code reference)

Target Users:

- University administrators
- Academic coordinators
- IT system administrators

Programming Details

The Schools module has been implemented using Java programming language. The graphical user interface (GUI) is built using Java AWT (Abstract Window Toolkit), providing a clean and functional layout for user interaction.

Naming Conventions Used:

- Class/Activity: schools_Schools
- Functions:
    - Create: create.schools()
    - Update: update.schools()
    - Retrieve: retrieve.schools()
    - Delete: delete.schools()

These functions handle the corresponding CRUD operations via button clicks in the Java Swing application.

Table Details: schools

The module uses a MySQL database with the following schema for the "schools" table:

Field Name	**Data Type**
id	Integer
uni_id	String
sch_code	String
sch_name	String
sch_location	String
sch_email	String

SOURCE CODE

```
package schools;


import java.awt.*;
import java.awt.event.*;
import java.sql.*;


public class Schools extends Frame {


    // Labels
    Label l1, l2, l3, l4, l5, headerLabel, footerLabel;


    // TextFields
    TextField tfUniID, tfSchCode, tfSchName, tfSchLocation, tfSchEmail;


    // Buttons
    Button btnAdd, btnUpdate, btnDelete, btnRetrieve;


    // Database Connection
    Connection con;
    PreparedStatement pst;


    public Schools() {
        setTitle("University School Management");
```

```
// Use BorderLayout for screen adaptation
setLayout(new BorderLayout());

// ===== Header =====
Panel header = new Panel();
header.setBackground(new Color(0, 100, 0));
headerLabel = new Label("University School Management", Label.CENTER);
headerLabel.setFont(new Font("Arial", Font.BOLD, 20));
headerLabel.setForeground(Color.WHITE);
header.add(headerLabel);
add(header, BorderLayout.NORTH);

// ===== Footer =====
Panel footer = new Panel();
footer.setBackground(new Color(0, 100, 0));
footerLabel = new Label("© 2025 University Management System");
footerLabel.setFont(new Font("Arial", Font.PLAIN, 14));
footerLabel.setForeground(Color.WHITE);
footer.add(footerLabel);
add(footer, BorderLayout.SOUTH);

// ===== Center Form Panel =====
Panel formPanel = new Panel(null); // Absolute layout
formPanel.setPreferredSize(new Dimension(1000, 400));
```

```
// Initialize Labels
l1 = new Label("University ID:");
l2 = new Label("School Code:");
l3 = new Label("School Name:");
l4 = new Label("School Location:");
l5 = new Label("School Email:");

// Initialize TextFields
tfUniID = new TextField();
tfSchCode = new TextField();
tfSchName = new TextField();
tfSchLocation = new TextField();
tfSchEmail = new TextField();

// Initialize Buttons
btnAdd = new Button("Add School");
btnUpdate = new Button("Update School");
btnDelete = new Button("Delete School");
btnRetrieve = new Button("Retrieve School");

int screenWidth = Toolkit.getDefaultToolkit().getScreenSize().width;
int xLabel = (screenWidth / 2) - 180;
int xField = (screenWidth / 2);
int y = 250;
```

```
int hGap = 40;

l1.setBounds(xLabel, y, 150, 25); tfUniID.setBounds(xField, y, 200, 25); y += hGap;
l2.setBounds(xLabel, y, 150, 25); tfSchCode.setBounds(xField, y, 200, 25); y += hGap;
l3.setBounds(xLabel, y, 150, 25); tfSchName.setBounds(xField, y, 200, 25); y += hGap;
l4.setBounds(xLabel, y, 150, 25); tfSchLocation.setBounds(xField, y, 200, 25); y += hGap;
l5.setBounds(xLabel, y, 150, 25); tfSchEmail.setBounds(xField, y, 200, 25); y += hGap;

btnAdd.setBounds(xLabel, y, 100, 30);
btnUpdate.setBounds(xLabel + 110, y, 100, 30);
btnDelete.setBounds(xLabel + 220, y, 100, 30);
btnRetrieve.setBounds(xLabel + 330, y, 100, 30);

// Add components to formPanel
formPanel.add(l1); formPanel.add(tfUniID);
formPanel.add(l2); formPanel.add(tfSchCode);
formPanel.add(l3); formPanel.add(tfSchName);
formPanel.add(l4); formPanel.add(tfSchLocation);
formPanel.add(l5); formPanel.add(tfSchEmail);
formPanel.add(btnAdd); formPanel.add(btnUpdate);
formPanel.add(btnDelete); formPanel.add(btnRetrieve);

add(formPanel, BorderLayout.CENTER);
```

```
// Connect to DB
        connectToDatabase();

        // Event Listeners
        btnAdd.addActionListener(e -> addSchool());
        btnUpdate.addActionListener(e -> updateSchool());
        btnDelete.addActionListener(e -> deleteSchool());
        btnRetrieve.addActionListener(e -> retrieveSchool());

        // Frame Settings
        setSize(1000, 500);
        setVisible(true);
        setLocationRelativeTo(null); // Center the window
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    private void connectToDatabase() {
        String url = "jdbc:mysql://localhost:3306/university_db?useSSL=false";
        String user = "root";
        String password = "Nikitha@2030";

        try {
```

```
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to MySQL database successfully!");
        } catch (ClassNotFoundException e) {
            System.out.println("JDBC Driver not found!");
            e.printStackTrace();
        } catch (SQLException e) {
            System.out.println("Failed to connect to MySQL!");
            e.printStackTrace();
        }
    }

    private void clearForm() {
        tfUniID.setText("");
        tfSchCode.setText("");
        tfSchName.setText("");
        tfSchLocation.setText("");
        tfSchEmail.setText("");
    }
    private void addSchool() {
        try {
                String query = "INSERT INTO schools (uni_id, sch_code, sch_name, sch_location, sch_email) VALUES (?, ?, ?, ?, ?)";
            pst = con.prepareStatement(query);
            pst.setString(1, tfUniID.getText());
```

```
            pst.setString(2, tfSchCode.getText());
            pst.setString(3, tfSchName.getText());
            pst.setString(4, tfSchLocation.getText());
            pst.setString(5, tfSchEmail.getText());
            pst.executeUpdate();
            System.out.println("School Added Successfully!");
            clearForm();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    private void updateSchool() {
        try {
            String query = "UPDATE schools SET uni_id = ?, sch_name = ?, sch_location = ?, sch_email
= ? WHERE sch_code = ?";
            pst = con.prepareStatement(query);
            pst.setString(1, tfUniID.getText());
            pst.setString(2, tfSchName.getText());
            pst.setString(3, tfSchLocation.getText());
            pst.setString(4, tfSchEmail.getText());
            pst.setString(5, tfSchCode.getText());
            int rowsAffected = pst.executeUpdate();
            if (rowsAffected > 0) {
                System.out.println("School Updated Successfully!");
                clearForm();
```

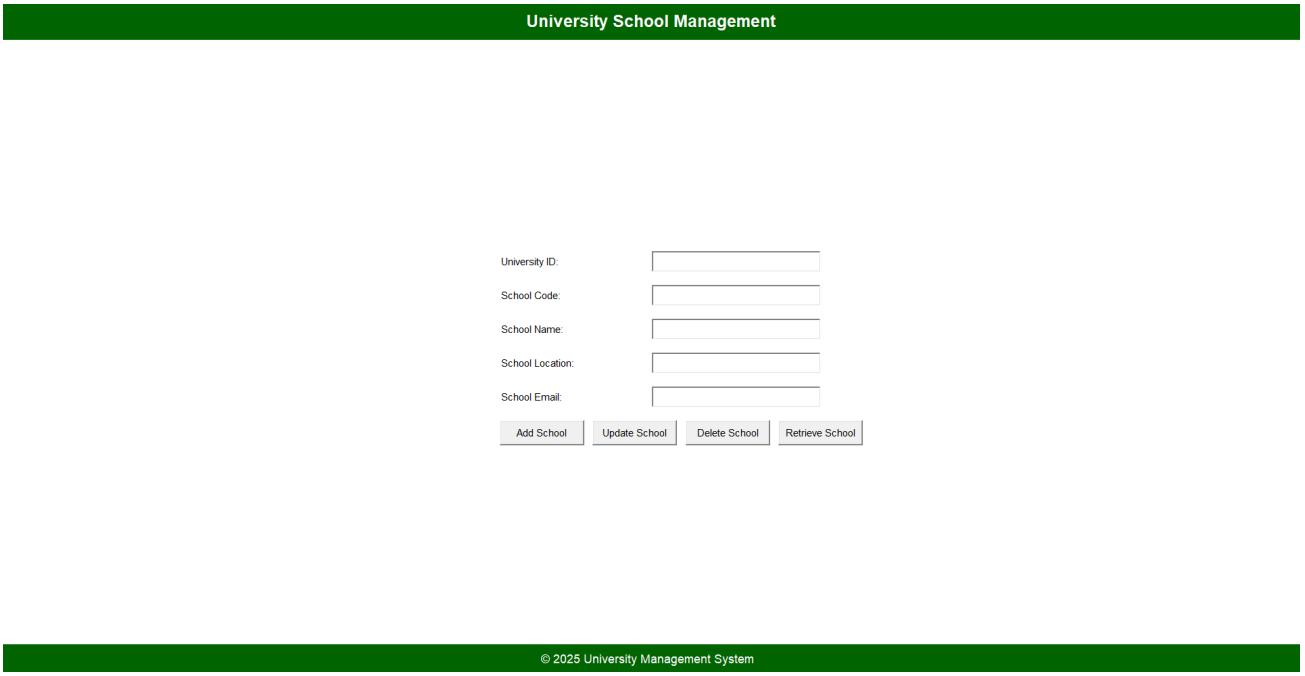
```
        } else {
            System.out.println("No matching school found for update.");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void deleteSchool() {
    try {
        String query = "DELETE FROM schools WHERE sch_code = ?";
        pst = con.prepareStatement(query);
        pst.setString(1, tfSchCode.getText());
        pst.executeUpdate();
        System.out.println("School Deleted Successfully!");
        clearForm();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
private void retrieveSchool() {
    try {
        String query = "SELECT * FROM schools WHERE sch_code = ?";
        pst = con.prepareStatement(query);
        pst.setString(1, tfSchCode.getText());
```

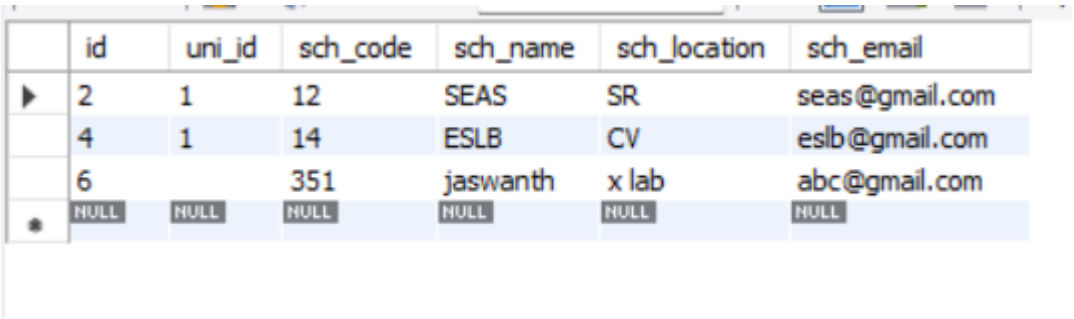
```
            ResultSet rs = pst.executeQuery();
            if (rs.next()) {
                tfUniID.setText(rs.getString("uni_id"));
                tfSchName.setText(rs.getString("sch_name"));
                tfSchLocation.setText(rs.getString("sch_location"));
                tfSchEmail.setText(rs.getString("sch_email"));
                System.out.println("School Retrieved Successfully!");
            } else {
                System.out.println("No School Found!");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    public static void main(String[] args) {
        new Schools();
    }
}
```

OUTPUT

GUI Screen Shots :



SQL Screen Shots :



	id	uni_id	sch_code	sch_name	sch_location	sch_email
▶	2	1	12	SEAS	SR	seas@gmail.com
	4	1	14	ESLB	CV	eslb@gmail.com
	6		351	jaswanth	x lab	abc@gmail.com
*	NULL	NULL	NULL	NULL	NULL	NULL

Conclusion

The Schools module plays an integral role in the OBE Implementation System at SRM-AP. It provides university administrators with the ability to manage core academic entities in a structured and secure way. By leveraging Java's capabilities for desktop application development and MySQL's robustness for data storage, the module ensures reliability, usability, and scalability.

Future enhancements may include:

- Role-based access control
- Enhanced UI/UX using JavaFX
- RESTful APIs for web integration
- Validation enhancements for field inputs

In summary, the Schools module not only simplifies school management but also acts as a foundation for the broader academic structure under the OBE framework. The successful implementation of this module paves the way for a smooth rollout of additional modules such as Departments, Programs, and Course Management.

This module has been developed with modularity, reusability, and future scalability in mind, aligning well with the long-term goals of the SRM-AP academic ecosystem.