OBE IMPLEMENTATION:BLOOM LEVEL SETTING

by

K.Manaswi[AP22110010007]
S.Aswartha Harshitha[AP22110010012]
K.Swathi[AP22110010029]
P.Sudhamai[AP22110010031]
K.Madhavi[AP22110010042]

A report for the CSE307 : Mobile Application Development using JAVA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SRM UNIVERSITY AP::AMARAVATI

INDEX

INTRODUCTION

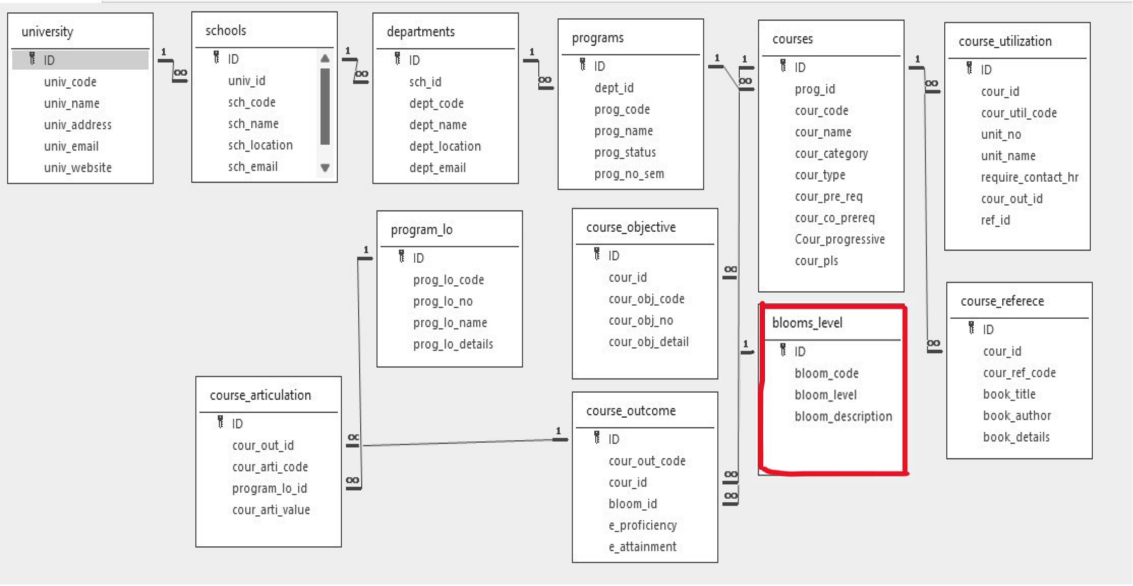
SRM University-AP is implementing the Outcome Based Education (OBE) system to improve educational effectiveness by aligning curriculum and assessments with intended outcomes. As part of this initiative, various modules are being developed to manage academic entities. I have been assigned the “Blooms Level Setting” module and developed a Windows application using Java AWT/Swing and SQLite to perform CRUD (Create, Retrieve, Update, Delete) operations.

Project Modules:

The OBE Implementation System comprises the following modules:

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:
Blooms Level Setting Module

Description:
This module allows the admin to create, retrieve, update, and delete Bloom's Taxonomy levels. Each level contains a unique code, a descriptive level name, and its description. The data is stored in an SQLite database. This module facilitates the academic team in aligning course outcomes with appropriate Bloom's levels.

Programming Details naming conventions to be used:

- **Class name/activity name:** BLOOMSLEVEL
- **Func on/method name :**
    - **Create:** addBloom()
    - **Update:** updateBloom()
    - **Retrieve:** viewBloom()
    - **Delete:** deleteBloom()

Table details:[bloomslevel]

Field Name	**Data type**
id	integer
bloom_code	String
bloom_level	String
bloom_description	String

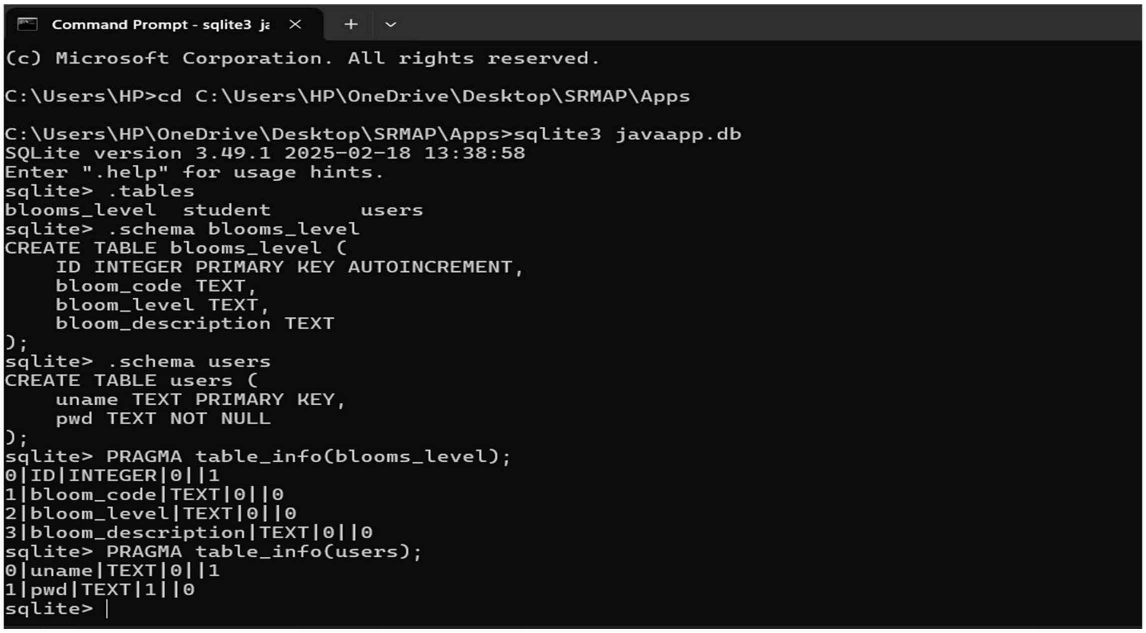
SOUCRCE CODE

Code:

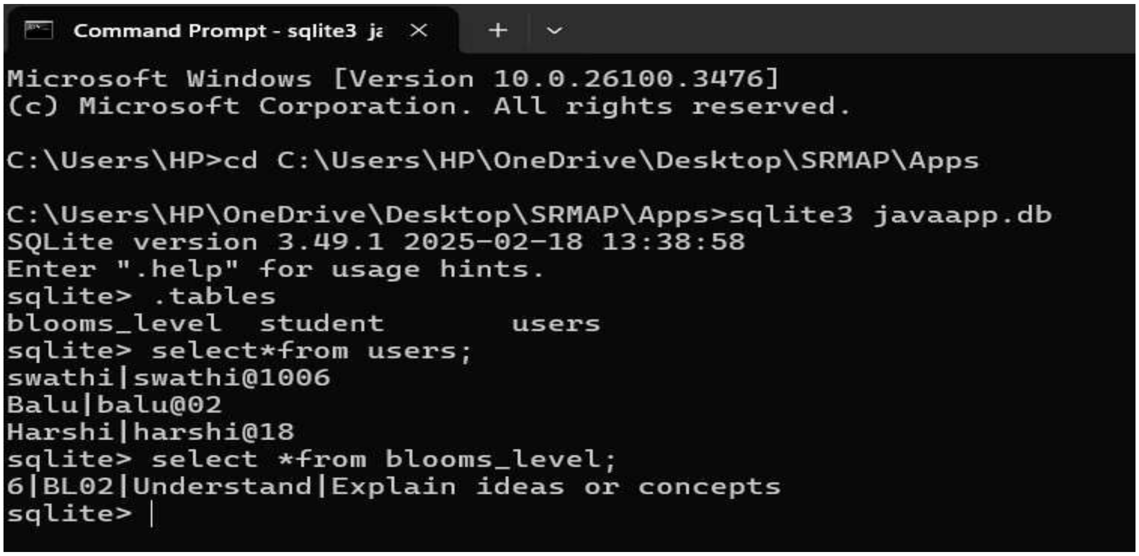
```
package bloomslevel;

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

public class BLOOMSLEVEL {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(LoginFrame::new);
    }
}

class LoginFrame extends JFrame implements ActionListener {
    JTextField txtUser;
    JPasswordField txtPass;
    JButton btnLogin;
    Connection conn;

    public LoginFrame() {
        setTitle("User - Login");
        setSize(380, 250);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setResizable(false);

        JPanel panel = new JPanel();
        panel.setLayout(null);
        panel.setBackground(new Color(245, 250, 255));

        JLabel title = new JLabel("Welcome");
        title.setFont(new Font("Arial", Font.BOLD, 18));
        title.setBounds(130, 20, 250, 30);
        panel.add(title);

        JLabel lblUser = new JLabel("Username:");
        lblUser.setFont(new Font("SansSerif", Font.PLAIN, 14));
        lblUser.setBounds(50, 70, 80, 25);
        panel.add(lblUser);
        txtUser = new JTextField();
        txtUser.setBounds(140, 70, 180, 25);
        panel.add(txtUser);

        JLabel lblPass = new JLabel("Password:");
        lblPass.setFont(new Font("SansSerif", Font.PLAIN, 14));
        lblPass.setBounds(50, 110, 80, 25);
        panel.add(lblPass);

        txtPass = new JPasswordField();
        txtPass.setBounds(140, 110, 180, 25);
        panel.add(txtPass);

        btnLogin = new JButton("Login");
        btnLogin.setFont(new Font("Tahoma", Font.BOLD, 13));
        btnLogin.setBackground(new Color(100, 149, 237));
        btnLogin.setForeground(Color.WHITE);
        btnLogin.setFocusPainted(false);
        btnLogin.setBounds(140, 160, 180, 30);
        btnLogin.addActionListener(this);
        panel.add(btnLogin);

        add(panel);
        connectDB();
        setVisible(true);
    }

    void connectDB() {
        try {
            Class.forName("org.sqlite.JDBC");
            conn =
DriverManager.getConnection("jdbc:sqlite:C:/Users/HP/OneDrive/Desktop/SRMAP
/Apps/javaapp.db");
            System.out.println("Connected to DB - LoginFrame");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "Database Connection Failed: " +
e);
        }
    }

    public void actionPerformed(ActionEvent e) {
        String username = txtUser.getText().trim();
        String password = String.valueOf(txtPass.getPassword()).trim();

        try {
            String query = "SELECT * FROM users WHERE uname = ? AND pwd = ?";
            PreparedStatement pst = conn.prepareStatement(query);
            pst.setString(1, username);
            pst.setString(2, password);
            ResultSet rs = pst.executeQuery();

            if (rs.next()) {
                JOptionPane.showMessageDialog(this, "Login successful!");
                dispose();
                new BloomLevelFrame(conn);
            } else {
                JOptionPane.showMessageDialog(this, "Invalid username or
password.");
            }
        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Login Error: " + ex);
        }
    }
}

class BloomLevelFrame extends JFrame implements ActionListener {
    JLabel lblCode, lblLevel, lblDesc;
    JTextField txtCode;
    JComboBox<String> comboLevel;
    JTextArea txtDesc;
    JButton btnAdd, btnUpdate, btnDelete, btnSearch;
    JTable table;
    DefaultTableModel tableModel;
    Connection conn;

    public BloomLevelFrame(Connection connection) {
        this.conn = connection;

        setTitle("Bloom's Level Manager");
        setSize(600, 600);
        setLayout(new GridBagLayout());
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(8, 8, 8, 8);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        lblCode = new JLabel("Bloom Code:");
        txtCode = new JTextField();

        lblLevel = new JLabel("Bloom Level:");
        String[] levels = {"-- Select Bloom Level --", "Remember", "Understand",
"Apply", "Analyze", "Evaluate", "Create"};
        comboLevel = new JComboBox<>(levels);

        lblDesc = new JLabel("Description:");
        txtDesc = new JTextArea(3, 20);
        JScrollPane scrollPane = new JScrollPane(txtDesc);

        btnAdd = new JButton("Add");
        btnUpdate = new JButton("Update");
        btnDelete = new JButton("Delete");
        btnSearch = new JButton("Search");

        btnAdd.addActionListener(this);
        btnUpdate.addActionListener(this);
        btnDelete.addActionListener(this);
        btnSearch.addActionListener(this);

        gbc.gridx = 0; gbc.gridy = 0; add(lblCode, gbc);
        gbc.gridx = 1; add(txtCode, gbc);

        gbc.gridx = 0; gbc.gridy = 1; add(lblLevel, gbc);
        gbc.gridx = 1; add(comboLevel, gbc);

        gbc.gridx = 0; gbc.gridy = 2; add(lblDesc, gbc);
        gbc.gridx = 1; add(scrollPane, gbc);

        JPanel btnPanel = new JPanel(new GridLayout(1, 4, 10, 0));
        btnPanel.add(btnAdd);
        btnPanel.add(btnUpdate);
        btnPanel.add(btnDelete);
        btnPanel.add(btnSearch);

        gbc.gridx = 0; gbc.gridy = 3; gbc.gridwidth = 2;
        add(btnPanel, gbc);

        String[] columns = {"ID", "Bloom Code", "Bloom Level", "Description"};
        tableModel = new DefaultTableModel(columns, 0);
        table = new JTable(tableModel);

        JTableHeader header = table.getTableHeader();
        header.setBackground(new Color(100, 149, 237)); // blue
        header.setForeground(Color.WHITE);              // white text
        header.setFont(new Font("SansSerif", Font.BOLD, 14)); // bold font

        table.setRowHeight(25);
```

```
        table.setFont(new Font("SansSerif", Font.PLAIN, 13));

        table.setDefaultRenderer(Object.class, new DefaultTableCellRenderer() {
            @Override
            public Component getTableCellRendererComponent(JTable table, Object
value,
                    boolean isSelected, boolean hasFocus, int row, int column) {
                Component c = super.getTableCellRendererComponent(table, value,
isSelected, hasFocus, row, column);
                if (!isSelected) {
                    c.setBackground(row % 2 == 0 ? Color.WHITE : new Color(230, 240,
255));
                } else {
                    c.setBackground(new Color(173, 216, 230)); // highlight on select
                }
                return c;
            }
        });

        JScrollPane tableScroll = new JScrollPane(table);
        tableScroll.setPreferredSize(new Dimension(550, 150));
        gbc.gridx = 0; gbc.gridy = 4;
        gbc.gridwidth = 2;
        gbc.fill = GridBagConstraints.BOTH;
        add(tableScroll, gbc);

        viewBlooms();
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == btnAdd) addBloom();
        else if (e.getSource() == btnUpdate) updateBloom();
        else if (e.getSource() == btnDelete) deleteBloom();
        else if (e.getSource() == btnSearch) searchBloom();
    }

    void clearFields() {
        txtCode.setText("");
        comboLevel.setSelectedIndex(0);
        txtDesc.setText("");
    }

    void addBloom() {
        try {
            String code = txtCode.getText().trim();
            String level = (String) comboLevel.getSelectedItem();
            String desc = txtDesc.getText().trim();

            if (code.isEmpty() || level.equals("-- Select Bloom Level --") ||
desc.isEmpty()) {
                JOptionPane.showMessageDialog(this, "All fields are required!");
                return;
            }

            String query = "INSERT INTO blooms_level (bloom_code, bloom_level,
bloom_description) VALUES (?, ?, ?)";
            PreparedStatement pst = conn.prepareStatement(query);
            pst.setString(1, code);
            pst.setString(2, level);
            pst.setString(3, desc);
            pst.executeUpdate();
            pst.close();

            JOptionPane.showMessageDialog(this, "Bloom's Level Added!");
            clearFields();
            viewBlooms();

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Error: " + ex);
        }
    }

    void searchBloom() {
        try {
            String code = txtCode.getText().trim();
            if (code.isEmpty()) {
                JOptionPane.showMessageDialog(this, "Please enter Bloom Code to
search.");
                return;
            }

            String query = "SELECT * FROM blooms_level WHERE bloom_code = ?";
            PreparedStatement pst = conn.prepareStatement(query);
            pst.setString(1, code);
            ResultSet rs = pst.executeQuery();

            tableModel.setRowCount(0);
            if (rs.next()) {
                comboLevel.setSelectedItem(rs.getString("bloom_level"));
                txtDesc.setText(rs.getString("bloom_description"));
                tableModel.addRow(new Object[]{
                    rs.getInt("ID"),
                    rs.getString("bloom_code"),
                    rs.getString("bloom_level"),
                    rs.getString("bloom_description")
                });
            } else {
                JOptionPane.showMessageDialog(this, "No record found.");
            }

            pst.close();

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Search Error: " + ex);
        }
    }

    void updateBloom() {
        try {
            String code = txtCode.getText().trim();
            String level = (String) comboLevel.getSelectedItem();
            String desc = txtDesc.getText().trim();

            if (code.isEmpty() || level.equals("-- Select Bloom Level --") ||
desc.isEmpty()) {
                JOptionPane.showMessageDialog(this, "All fields are required to
update!");
                return;
            }

            String selectQuery = "SELECT * FROM blooms_level WHERE
bloom_code = ?";
            PreparedStatement pstSelect = conn.prepareStatement(selectQuery);
            pstSelect.setString(1, code);
            ResultSet rs = pstSelect.executeQuery();

            if (rs.next()) {
                String updateQuery = "UPDATE blooms_level SET bloom_level = ?,
bloom_description = ? WHERE bloom_code = ?";
                PreparedStatement pstUpdate = conn.prepareStatement(updateQuery);
                pstUpdate.setString(1, level);
                pstUpdate.setString(2, desc);
                pstUpdate.setString(3, code);
                pstUpdate.executeUpdate();
                pstUpdate.close();

                JOptionPane.showMessageDialog(this, "Updated Successfully!");
                viewBlooms();
            } else {
                JOptionPane.showMessageDialog(this, "No record found to update.");
            }

            pstSelect.close();

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Update Error: " + ex);
        }
    }

    void deleteBloom() {
        String code = JOptionPane.showInputDialog(this, "Enter Bloom Code to
Delete:");
        if (code != null && !code.trim().isEmpty()) {
            try {
                String query = "DELETE FROM blooms_level WHERE bloom_code =
?";
                PreparedStatement pst = conn.prepareStatement(query);
                pst.setString(1, code.trim());

                int result = pst.executeUpdate();
                pst.close();

                if (result > 0) {
                    JOptionPane.showMessageDialog(this, "Deleted Successfully.");
                    viewBlooms();
                } else {
                    JOptionPane.showMessageDialog(this, "No record found.");
                }

            } catch (Exception ex) {
                JOptionPane.showMessageDialog(this, "Delete Error: " + ex);
            }
        }
    }

    void viewBlooms() {
        try {
            String query = "SELECT * FROM blooms_level";
            PreparedStatement pst = conn.prepareStatement(query);
            ResultSet rs = pst.executeQuery();

            tableModel.setRowCount(0);
            while (rs.next()) {
                tableModel.addRow(new Object[]{
                        rs.getInt("ID"),
                        rs.getString("bloom_code"),
                        rs.getString("bloom_level"),
                        rs.getString("bloom_description")
                });
            }

            pst.close();

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "View Error: " + ex);
        }
    }
}
```

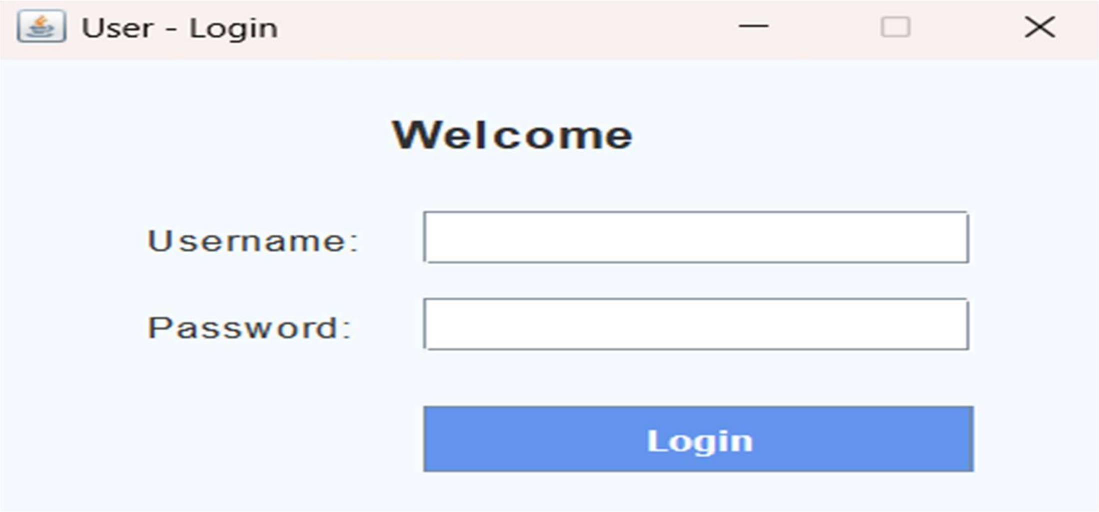
SCREEN SHOTS

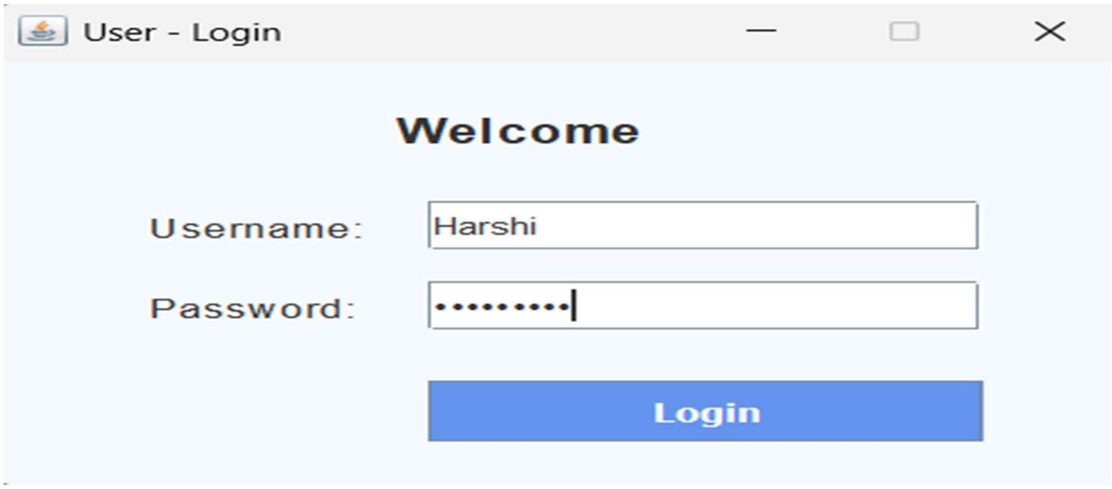


```
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\SRMAP\Apps

C:\Users\HP\OneDrive\Desktop\SRMAP\Apps>sqlite3 javaapp.db
SQLite version 3.49.1 2025-02-18 13:38:58
Enter ".help" for usage hints.
sqlite> .tables
blooms_level  student       users
sqlite> .schema blooms_level
CREATE TABLE blooms_level (
    ID INTEGER PRIMARY KEY AUTOINCREMENT,
    bloom_code TEXT,
    bloom_level TEXT,
    bloom_description TEXT
);
sqlite> .schema users
CREATE TABLE users (
    uname TEXT PRIMARY KEY,
    pwd TEXT NOT NULL
);
sqlite> PRAGMA table_info(blooms_level);
0|ID|INTEGER|0||1
1|bloom_code|TEXT|0||0
2|bloom_level|TEXT|0||0
3|bloom_description|TEXT|0||0
sqlite> PRAGMA table_info(users);
0|uname|TEXT|0||1
1|pwd|TEXT|1||0
sqlite> |
```



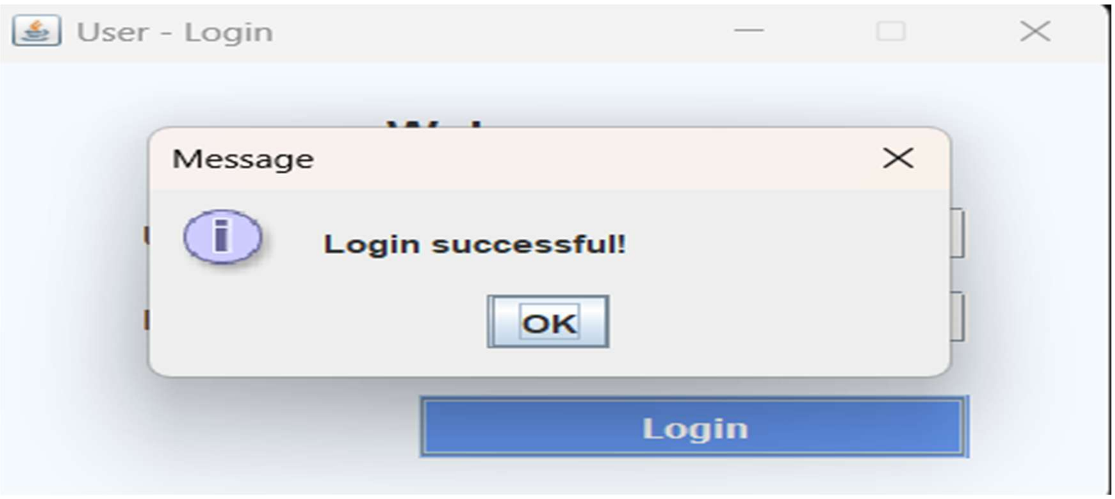
```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\SRMAP\Apps

C:\Users\HP\OneDrive\Desktop\SRMAP\Apps>sqlite3 javaapp.db
SQLite version 3.49.1 2025-02-18 13:38:58
Enter ".help" for usage hints.
sqlite> .tables
blooms_level  student       users
sqlite> select*from users;
swathi|swathi@1006
Balu|balu@02
Harshi|harshi@18
sqlite> select *from blooms_level;
6|BL02|Understand|Explain ideas or concepts
sqlite> |
```



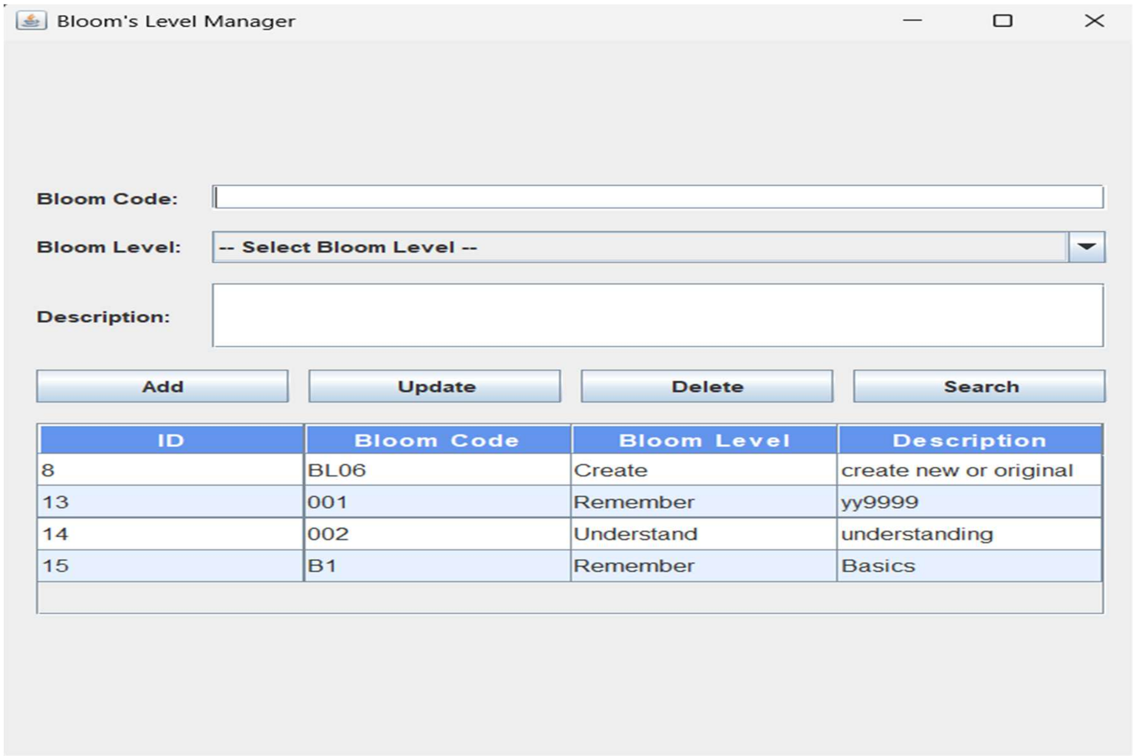


Username: Harshi

Password: •••••••••



Login successful!



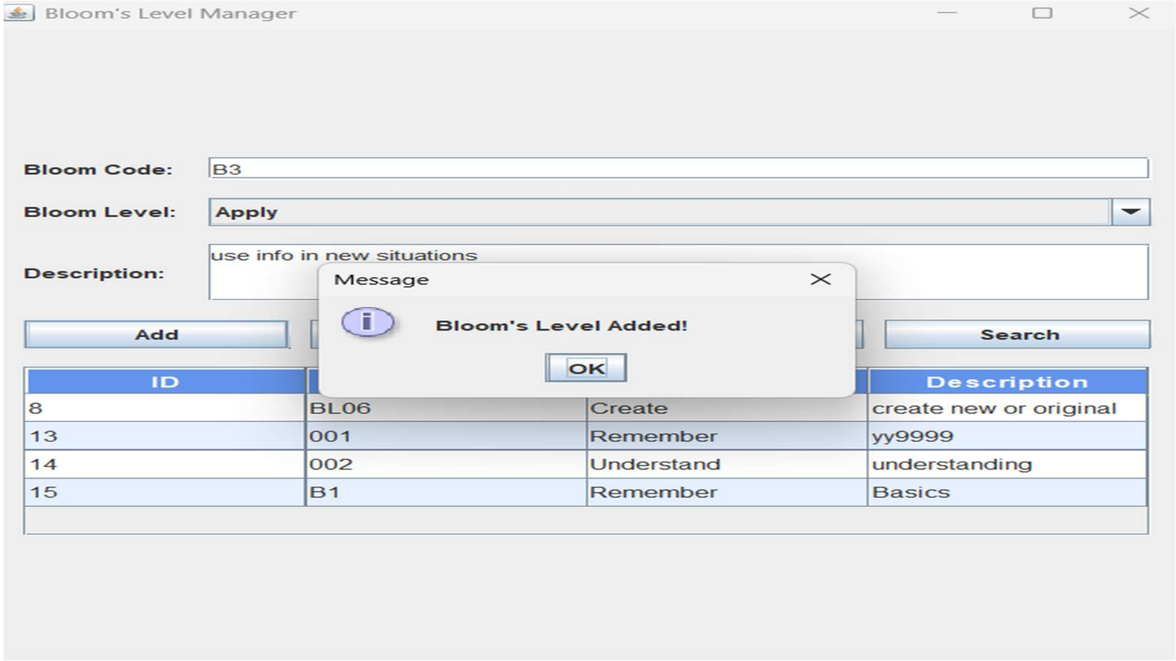
Bloom Code:

Bloom Level: -- Select Bloom Level --

Description:

ID	Bloom Code	Bloom Level	Description
8	BL06	Create	create new or original
13	001	Remember	yy9999
14	002	Understand	understanding
15	B1	Remember	Basics

C-Create



Bloom Code: B3

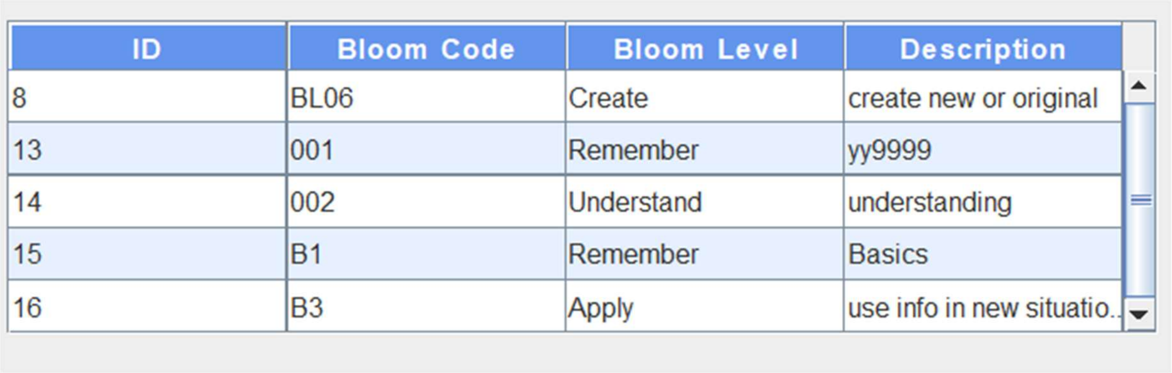
Bloom Level: Apply

Description: use info in new situations

Bloom's Level Added!

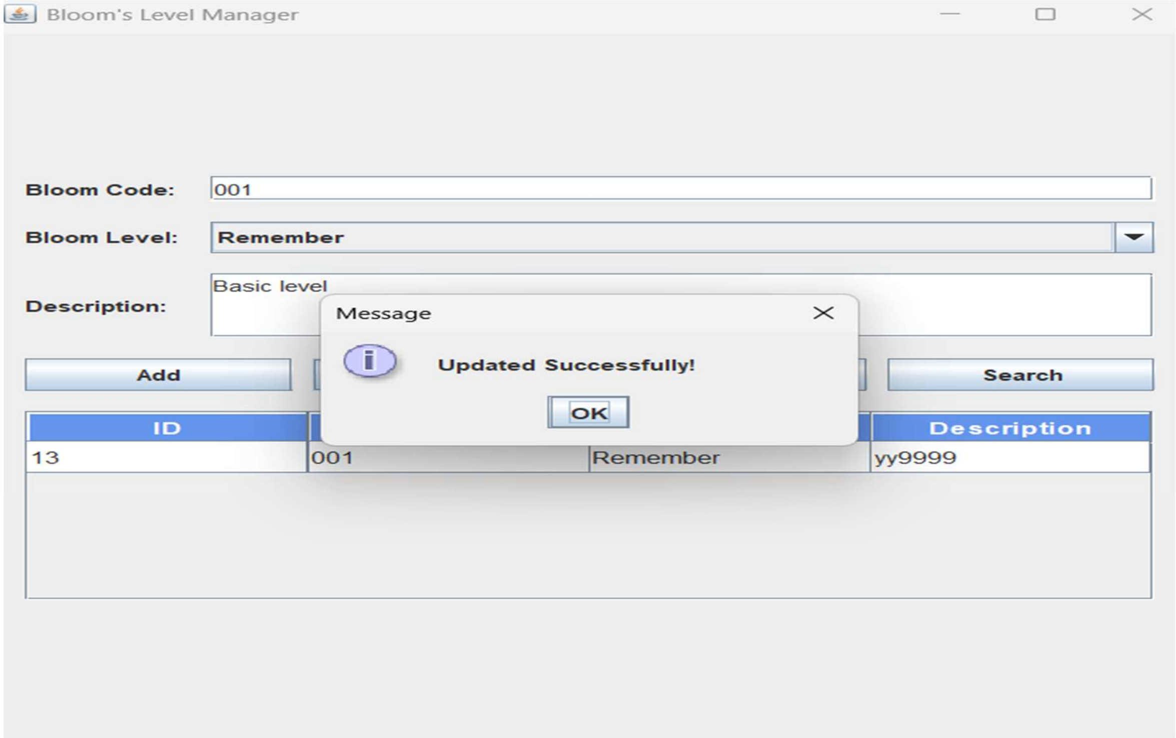
ID			Description
8	BL06	Create	create new or original
13	001	Remember	yy9999
14	002	Understand	understanding
15	B1	Remember	Basics

R-Read



ID	Bloom Code	Bloom Level	Description
8	BL06	Create	create new or original
13	001	Remember	yy9999
14	002	Understand	understanding
15	B1	Remember	Basics
16	B3	Apply	use info in new situatio...

U-Update



Bloom Code: 001

Bloom Level: Remember

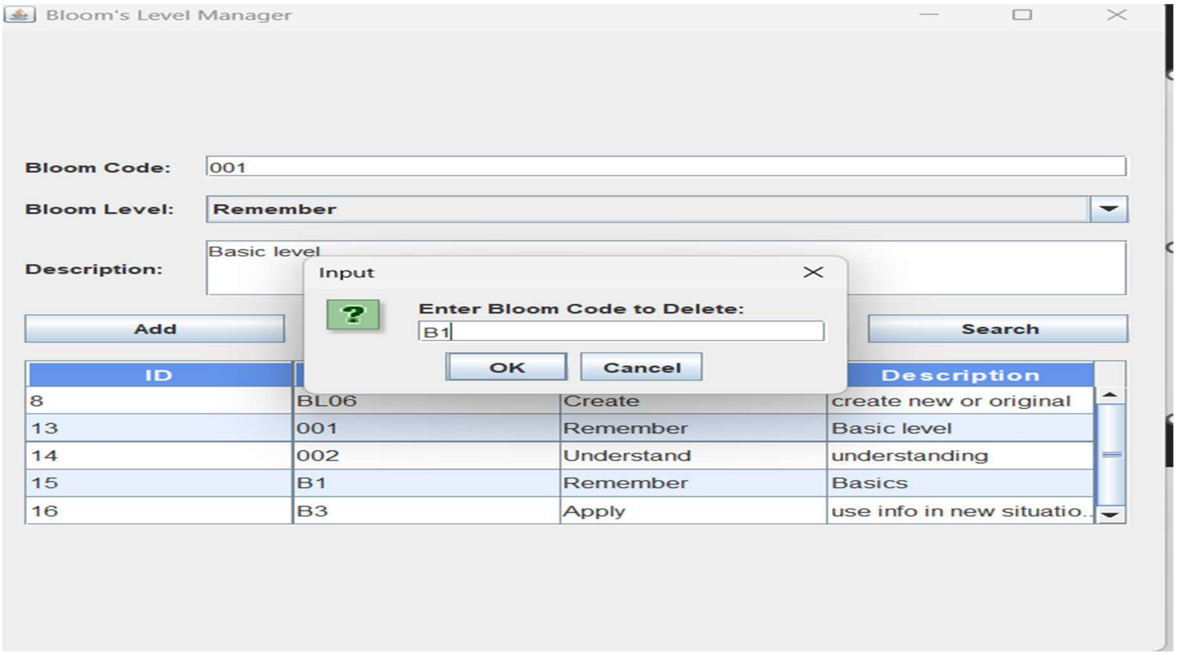
Description: Basic level

Message

Updated Successfully!

ID			Description
13	001	Remember	yy9999

D-Delete



Bloom Code: 001

Bloom Level: Remember

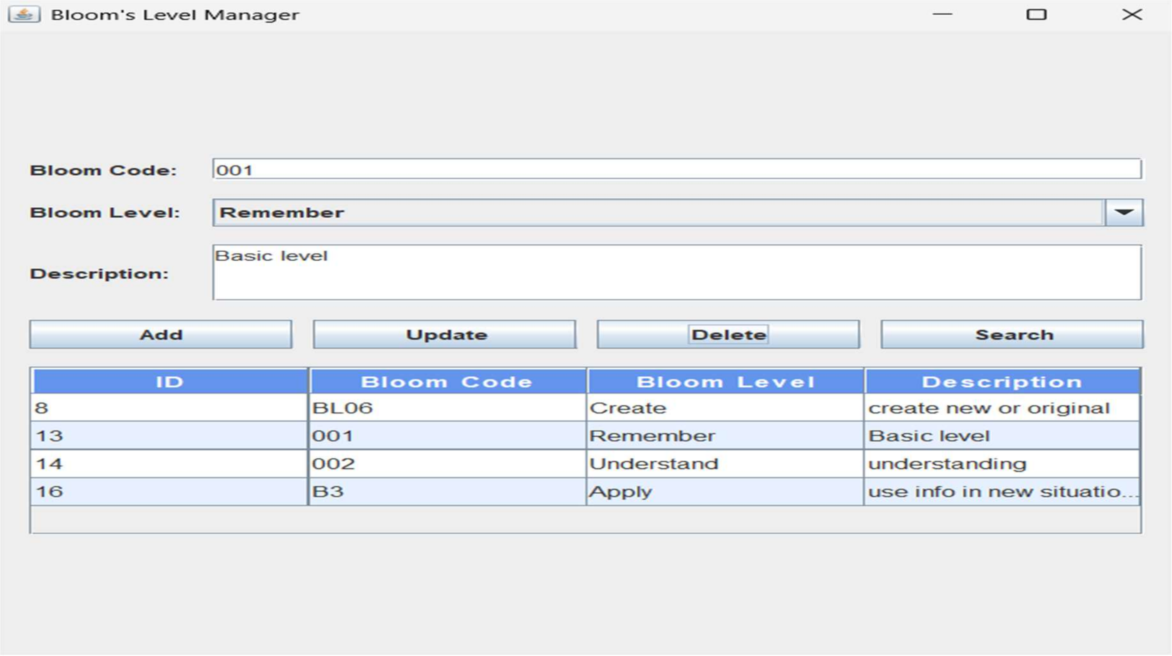
Description: Basic level

Input

Enter Bloom Code to Delete: B1

ID			Description
8	BL06	Create	create new or original
13	001	Remember	Basic level
14	002	Understand	understanding
15	B1	Remember	Basics
16	B3	Apply	use info in new situatio...

After deleting:



Bloom Code: 001

Bloom Level: Remember

Description: Basic level

ID	Bloom Code	Bloom Level	Description
8	BL06	Create	create new or original
13	001	Remember	Basic level
14	002	Understand	understanding
16	B3	Apply	use info in new situatio...

CONCLUSION

The Blooms Level Setting module provides an easy-to-use interface for managing Bloom's taxonomy levels, which are crucial in outcome-based curriculum design. This application fulfills the CRUD functionalities and demonstrates practical application development using Java Swing and SQLite. By integrating this module with other OBE modules, the university can track and assess learning outcomes more effectively and implement OBE standards at every academic level.