package mysqldbconnection;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class StudentCRUD extends JFrame {

private JTextField studentIdField, nameField, ageField;

private Connection connection;

public StudentCRUD() {

super("Student Database CRUD GUI");

JLabel studentIdLabel = new JLabel("Student ID:");

JLabel nameLabel = new JLabel("Name:");

JLabel ageLabel = new JLabel("Age:");

studentIdField = new JTextField(10);

nameField = new JTextField(20);

ageField = new JTextField(5);

JButton insertButton = new JButton("Insert");

JButton updateButton = new JButton("Update");

JButton deleteButton = new JButton("Delete");

JButton retrieveButton = new JButton("Retrieve");

setLayout(new GridLayout(5, 2));

add(studentIdLabel);

add(studentIdField);

add(nameLabel);

add(nameField);

add(ageLabel);

add(ageField);

add(insertButton);

add(updateButton);

add(deleteButton);

add(retrieveButton);

insertButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

insertData();

}

});

updateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

updateData();

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

deleteData();

}

});

retrieveButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

retrieveData();

}

});

setSize(400, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

setVisible(true);

}

**Task1:**

private Connection getConnection() throws SQLException {

return DriverManager.getConnection("jdbc:mysql://localhost:3306/scdlab10", "root", "fast123");

}

**Task2 - Task8:**

private void insertData() {

try {

connection = getConnection();

connection.setAutoCommit(false);

String studentId = studentIdField.getText();

String name = nameField.getText();

int age = Integer.parseInt(ageField.getText());

String insertQuery = "INSERT INTO students (student\_id, name, age) VALUES (?, ?, ?)";

try (PreparedStatement preparedStatement = connection.prepareStatement(insertQuery)) {

preparedStatement.setString(1, studentId);

preparedStatement.setString(2, name);

preparedStatement.setInt(3, age);

preparedStatement.executeUpdate();

connection.commit();

JOptionPane.showMessageDialog(this, "Record inserted successfully!");

}

} catch (SQLException | NumberFormatException ex) {

handleSQLException((SQLException) ex, "Error inserting record: ");

} finally {

closeConnection();

}

}

private void updateData() {

try {

connection = getConnection();

connection.setAutoCommit(false);

String studentId = studentIdField.getText();

int age = Integer.parseInt(ageField.getText());

String updateQuery = "UPDATE students SET age = ? WHERE student\_id = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(updateQuery)) {

preparedStatement.setInt(1, age);

preparedStatement.setString(2, studentId);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

connection.commit();

JOptionPane.showMessageDialog(this, "Record updated successfully!");

} else {

JOptionPane.showMessageDialog(this, "No record found for the provided Student ID.", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

} catch (SQLException | NumberFormatException ex) {

handleSQLException((SQLException) ex, "Error updating record: ");

} finally {

closeConnection();

}

}

private void deleteData() {

try {

connection = getConnection();

connection.setAutoCommit(false);

String studentId = studentIdField.getText();

String deleteQuery = "DELETE FROM students WHERE student\_id = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(deleteQuery)) {

preparedStatement.setString(1, studentId);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

connection.commit();

JOptionPane.showMessageDialog(this, "Record deleted successfully!");

} else {

JOptionPane.showMessageDialog(this, "No record found for the provided Student ID.", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

} catch (SQLException ex) {

handleSQLException(ex, "Error deleting record: ");

} finally {

closeConnection();

}

}

private void retrieveData() {

try {

connection = getConnection();

String query = "SELECT \* FROM students";

try (Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(query)) {

StringBuilder result = new StringBuilder("Student Records:\n");

while (resultSet.next()) {

String studentId = resultSet.getString("student\_id");

String name = resultSet.getString("name");

int age = resultSet.getInt("age");

result.append("Student ID: ").append(studentId).append(", Name: ").append(name).append(", Age: ").append(age).append("\n");

}

JOptionPane.showMessageDialog(this, result.toString());

}

} catch (SQLException ex) {

handleSQLException(ex, "Error retrieving records: ");

} finally {

closeConnection();

}

}

private void handleSQLException(SQLException ex, String errorMessage) {

ex.printStackTrace();

// Check for duplicate key violation

if (ex instanceof SQLIntegrityConstraintViolationException && ex.getErrorCode() == 1062) {

JOptionPane.showMessageDialog(this, "Duplicate key violation. Record with the same ID already exists.", "Error", JOptionPane.ERROR\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, errorMessage + ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);

}

try {

if (connection != null) {

connection.rollback();

System.out.println("Transaction rolled back.");

}

} catch (SQLException rollbackException) {

rollbackException.printStackTrace();

}

}

private void closeConnection() {

// Close the connection in the finally block to ensure proper cleanup

try {

if (connection != null) {

connection.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

new StudentCRUD();

}

});

}

}

**Task7:**



