**Dr. Mahalingam College of Engineering and Technology, Pollachi-642003**

**Department of Information Technology**

**Academic Year: 2024 - 2025**

**TRL and SDG Certificate**

**Project Title :**

**Course Code & Name : 23ITI402 Database Management Systems**

**Department and Semester: Information Technology & IV SEM**

**Technology Readiness Level (TRL) of the Project : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Sustainability Development Goals (SDG)-Goal Name : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Names and Roll Numbers of Student** | **Signature of the Student** |
| 1. | VEERALAKSHMI P , 727623BIT046 |  |

|  |  |
| --- | --- |
| **Preparation (25 marks)** |  |
| **Interpretation (30 marks)** |  |
| **Output & Result (15 marks)** |  |
| **Viva (5 marks)** |  |
| **Total (75 marks)** |  |
| **Signature of the faculty incharge** |  |

**1.Objective of the Project (1 page)**

**2.Entire Project Code**

**3.Database view**

**4.Output screenshots**

**5.Result**

**1.OBJECTIVE OF THE PROJECT:**

The objective of the MyBank Database Management System project is to build an effective and user-friendly application that helps in managing various banking operations in a digital format. The system is intended to serve as a solution that replaces traditional, paper-based banking with a modern and automated approach. It focuses on essential features such as account creation, secure login, deposits, withdrawals, balance checking, and viewing transaction records. The project leverages Java Swing for the graphical user interface, Java for the backend logic, and MySQL as the database to store and manage customer and transaction information efficiently. This integration ensures that every transaction is processed accurately and that users always view the most up-to-date account information. The system is designed with data consistency, integrity, and security in mind, so that banking operations can be carried out with confidence. Through this project, the aim is to provide a basic but realistic simulation of a banking environment that helps students understand how software and databases are used together to solve real-world financial problems. It also sets the foundation for learning more advanced concepts such as authentication, real-time data handling, and secure system design.

**2.ENTIRE PROJECT CODE:**

**BankDashboard Class :**

package bb;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class BankDashboard extends JFrame {

private JButton depositButton, withdrawButton;

private JTextField amountField;

private String username;

private static final Color BUTTON\_COLOR = new Color(0, 123, 255); // Blue color for buttons

private static final Color BUTTON\_HOVER\_COLOR = new Color(0, 105, 217); // Hover effect color

private static final Color FIELD\_COLOR = new Color(240, 240, 240); // Light gray color for text fields

private static final Color PANEL\_BACKGROUND\_COLOR = new Color(255, 255, 255); // White color for panel

public BankDashboard(String username) {

this.username = username;

setTitle("Welcome, " + username);

setSize(400, 200);

setLocationRelativeTo(null);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

// Setting up panel and layout

JPanel panel = new JPanel();

panel.setLayout(null);

panel.setBackground(PANEL\_BACKGROUND\_COLOR); // Set background color for panel

// Label for amount input

JLabel label = new JLabel("Amount:");

label.setBounds(10, 20, 100, 25);

panel.add(label);

// Text field for amount

amountField = new JTextField();

amountField.setBounds(100, 20, 165, 25);

amountField.setBackground(FIELD\_COLOR); // Set background color for text field

panel.add(amountField);

// Deposit button

depositButton = new JButton("Deposit");

depositButton.setBounds(10, 60, 120, 25);

depositButton.setBackground(BUTTON\_COLOR); // Set background color for deposit button

depositButton.setForeground(Color.WHITE); // Set text color

depositButton.setFocusPainted(false); // Remove focus border

depositButton.setBorder(BorderFactory.createLineBorder(BUTTON\_COLOR)); // Border color

depositButton.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseEntered(java.awt.event.MouseEvent evt) {

depositButton.setBackground(BUTTON\_HOVER\_COLOR); // Change color on hover

}

public void mouseExited(java.awt.event.MouseEvent evt) {

depositButton.setBackground(BUTTON\_COLOR); // Reset color when mouse leaves

}

});

panel.add(depositButton);

// Withdraw button

withdrawButton = new JButton("Withdraw");

withdrawButton.setBounds(140, 60, 120, 25);

withdrawButton.setBackground(BUTTON\_COLOR); // Set background color for withdraw button

withdrawButton.setForeground(Color.WHITE); // Set text color

withdrawButton.setFocusPainted(false); // Remove focus border

withdrawButton.setBorder(BorderFactory.createLineBorder(BUTTON\_COLOR)); // Border color

withdrawButton.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseEntered(java.awt.event.MouseEvent evt) {

withdrawButton.setBackground(BUTTON\_HOVER\_COLOR); // Change color on hover

}

public void mouseExited(java.awt.event.MouseEvent evt) {

withdrawButton.setBackground(BUTTON\_COLOR); // Reset color when mouse leaves

}

});

panel.add(withdrawButton);

// Action for deposit button

depositButton.addActionListener(e -> {

double amount = Double.parseDouble(amountField.getText());

if (User.deposit(username, amount)) {

JOptionPane.showMessageDialog(this, "Deposit Successful");

new TransactionDetailsForm(username).setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Failed");

}

});

// Action for withdraw button

withdrawButton.addActionListener(e -> {

double amount = Double.parseDouble(amountField.getText());

if (User.withdraw(username, amount)) {

JOptionPane.showMessageDialog(this, "Withdraw Successful");

new TransactionDetailsForm(username).setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Failed or Insufficient Balance");

}

});

add(panel);

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

new BankDashboard("JohnDoe").setVisible(true); // Replace with actual username

});

}

}

**DBconnection class:**

package bb;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnection {

private static final String URL = "jdbc:mysql://localhost:3306/d"; // Replace with your DB URL

private static final String USER = "root"; // Replace with your DB username

private static final String PASSWORD = "dbms"; // Replace with your DB password

// Method to establish a connection to the database

public static Connection getConnection() throws SQLException {

try {

// Register the JDBC driver

Class.forName("com.mysql.cj.jdbc.Driver");

// Establish and return the connection

return DriverManager.getConnection(URL, USER, PASSWORD);

} catch (ClassNotFoundException e) {

// Handle exception if driver is not found

throw new SQLException("JDBC Driver not found.", e);

} catch (SQLException e) {

// Handle any SQL exceptions

throw new SQLException("Failed to establish connection to the database.", e);

}

}

}

**Register Class:**

package bb;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

import java.awt.Color;

public class Register extends JFrame {

private JTextField usernameField;

private JPasswordField passwordField;

private JButton registerButton;

// Define colors for the UI

private static final Color BUTTON\_COLOR = new Color(0, 123, 255); // Blue color for buttons

private static final Color BUTTON\_HOVER\_COLOR = new Color(0, 105, 217); // Hover effect color

private static final Color FIELD\_COLOR = new Color(240, 240, 240); // Light gray color for text fields

private static final Color PANEL\_BACKGROUND\_COLOR = new Color(255, 255, 255); // White color for panel

public Register() {

setTitle("Register New User");

setSize(300, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

JPanel panel = new JPanel();

panel.setLayout(null);

panel.setBackground(PANEL\_BACKGROUND\_COLOR); // Set panel background color

// Username label

JLabel usernameLabel = new JLabel("Username:");

usernameLabel.setBounds(10, 20, 80, 25);

panel.add(usernameLabel);

// Username input field

usernameField = new JTextField();

usernameField.setBounds(100, 20, 165, 25);

usernameField.setBackground(FIELD\_COLOR); // Set background color for text field

panel.add(usernameField);

// Password label

JLabel passwordLabel = new JLabel("Password:");

passwordLabel.setBounds(10, 50, 80, 25);

panel.add(passwordLabel);

// Password input field

passwordField = new JPasswordField();

passwordField.setBounds(100, 50, 165, 25);

passwordField.setBackground(FIELD\_COLOR); // Set background color for text field

panel.add(passwordField);

// Register button

registerButton = new JButton("Register");

registerButton.setBounds(10, 80, 255, 25);

registerButton.setBackground(BUTTON\_COLOR); // Set background color for button

registerButton.setForeground(Color.WHITE); // Set text color to white

registerButton.setFocusPainted(false); // Remove focus border

registerButton.setBorder(BorderFactory.createLineBorder(BUTTON\_COLOR)); // Border color

registerButton.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseEntered(java.awt.event.MouseEvent evt) {

registerButton.setBackground(BUTTON\_HOVER\_COLOR); // Change color on hover

}

public void mouseExited(java.awt.event.MouseEvent evt) {

registerButton.setBackground(BUTTON\_COLOR); // Reset color when mouse leaves

}

});

panel.add(registerButton);

// Register button action

registerButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

registerUser();

}

});

// Add panel to the frame

add(panel);

}

// Method to handle user registration

private void registerUser() {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

// Insert new user into the database

if (User.registerUser(username, password)) {

JOptionPane.showMessageDialog(this, "User registered successfully!");

this.dispose(); // Close the register frame after successful registration

} else {

JOptionPane.showMessageDialog(this, "User registration failed!");

}

}

// Main method to launch the Register frame

public static void main(String[] args) {

new Register().setVisible(true);

}

}

**TRANSATION CLASS:**

Transacackage bb;

import javax.swing.\*;

import java.awt.\*;

import java.sql.\*;

public class TransactionDetailsForm extends JFrame {

private JTextArea textArea;

public TransactionDetailsForm(String username) {

setTitle("Transaction Details");

setSize(400, 400); // Increased size to accommodate balance and transactions

setLocationRelativeTo(null);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

textArea = new JTextArea();

textArea.setEditable(false);

JScrollPane pane = new JScrollPane(textArea);

add(pane, BorderLayout.CENTER);

try (Connection conn = DBConnection.getConnection()) {

String sql = "SELECT \* FROM transactions WHERE username = ? ORDER BY timestamp DESC";

PreparedStatement ps = conn.prepareStatement(sql);

ps.setString(1, username);

ResultSet rs = ps.executeQuery();

double totalDeposit = 0.0;

double totalWithdraw = 0.0;

// Loop through all the transactions

while (rs.next()) {

String type = rs.getString("type");

double amount = rs.getDouble("amount");

// Add deposit amounts and subtract withdrawal amounts

if ("Deposit".equals(type)) {

totalDeposit += amount;

} else if ("Withdraw".equals(type)) {

totalWithdraw += amount;

}

}

// Calculate total balance after all deposits and withdrawals

double totalBalance = totalDeposit - totalWithdraw;

// Display the transaction details and the total balance

StringBuilder transactionDetails = new StringBuilder();

transactionDetails.append("Total Deposit: ").append(totalDeposit).append("\n")

.append("Total Withdraw: ").append(totalWithdraw).append("\n\n")

.append("Total Balance: ").append(totalBalance);

textArea.setText(transactionDetails.toString());

} catch (SQLException e) {

textArea.setText("Error retrieving transaction.");

}

}

public static void main(String[] args) {

new TransactionDetailsForm("username").setVisible(true); // Replace "username" with actual username

}

}

**User Class:**

package bb;

import java.sql.\*;

public class User {

private static final String RESET = "\033[0m"; // Reset color

private static final String GREEN = "\033[32m"; // Green color

private static final String RED = "\033[31m"; // Red color

private static final String BLUE = "\033[34m"; // Blue color

public static boolean registerUser(String username, String password) {

try (Connection conn = DBConnection.getConnection()) {

String sql = "INSERT INTO users (username, password, balance) VALUES (?, ?, 0)";

PreparedStatement ps = conn.prepareStatement(sql);

ps.setString(1, username);

ps.setString(2, password);

boolean success = ps.executeUpdate() > 0;

if (success) {

System.out.println(GREEN + "User registered successfully!" + RESET);

} else {

System.out.println(RED + "Failed to register user." + RESET);

}

return success;

} catch (SQLException e) {

System.out.println(RED + "Error: " + e.getMessage() + RESET);

return false;

}

}

public static boolean validateLogin(String username, String password) {

try (Connection conn = DBConnection.getConnection()) {

String sql = "SELECT \* FROM users WHERE username = ? AND password = ?";

PreparedStatement ps = conn.prepareStatement(sql);

ps.setString(1, username);

ps.setString(2, password);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

System.out.println(GREEN + "Login successful!" + RESET);

return true;

} else {

System.out.println(RED + "Invalid username or password." + RESET);

return false;

}

} catch (SQLException e) {

System.out.println(RED + "Error: " + e.getMessage() + RESET);

return false;

}

}

public static boolean deposit(String username, double amount) {

try (Connection conn = DBConnection.getConnection()) {

conn.setAutoCommit(false);

String updateSql = "UPDATE users SET balance = balance + ? WHERE username = ?";

PreparedStatement ps = conn.prepareStatement(updateSql);

ps.setDouble(1, amount);

ps.setString(2, username);

if (ps.executeUpdate() > 0) {

String insertTxn = "INSERT INTO transactions (username, type, amount) VALUES (?, 'Deposit', ?)";

PreparedStatement txnStmt = conn.prepareStatement(insertTxn);

txnStmt.setString(1, username);

txnStmt.setDouble(2, amount);

txnStmt.executeUpdate();

conn.commit();

System.out.println(GREEN + "Deposit successful. Amount: " + amount + RESET);

return true;

}

System.out.println(RED + "Deposit failed. Please try again." + RESET);

return false;

} catch (SQLException e) {

System.out.println(RED + "Error: " + e.getMessage() + RESET);

return false;

}

}

public static boolean withdraw(String username, double amount) {

try (Connection conn = DBConnection.getConnection()) {

conn.setAutoCommit(false);

String updateSql = "UPDATE users SET balance = balance - ? WHERE username = ? AND balance >= ?";

PreparedStatement ps = conn.prepareStatement(updateSql);

ps.setDouble(1, amount);

ps.setString(2, username);

ps.setDouble(3, amount);

if (ps.executeUpdate() > 0) {

String insertTxn = "INSERT INTO transactions (username, type, amount) VALUES (?, 'Withdraw', ?)";

PreparedStatement txnStmt = conn.prepareStatement(insertTxn);

txnStmt.setString(1, username);

txnStmt.setDouble(2, amount);

txnStmt.executeUpdate();

conn.commit();

System.out.println(GREEN + "Withdrawal successful. Amount: " + amount + RESET);

return true;

}

System.out.println(RED + "Insufficient balance for withdrawal." + RESET);

return false;

} catch (SQLException e) {

System.out.println(RED + "Error: " + e.getMessage() + RESET);

return false;

}

}

// An unused method but keeping it for compatibility with the previous code structure

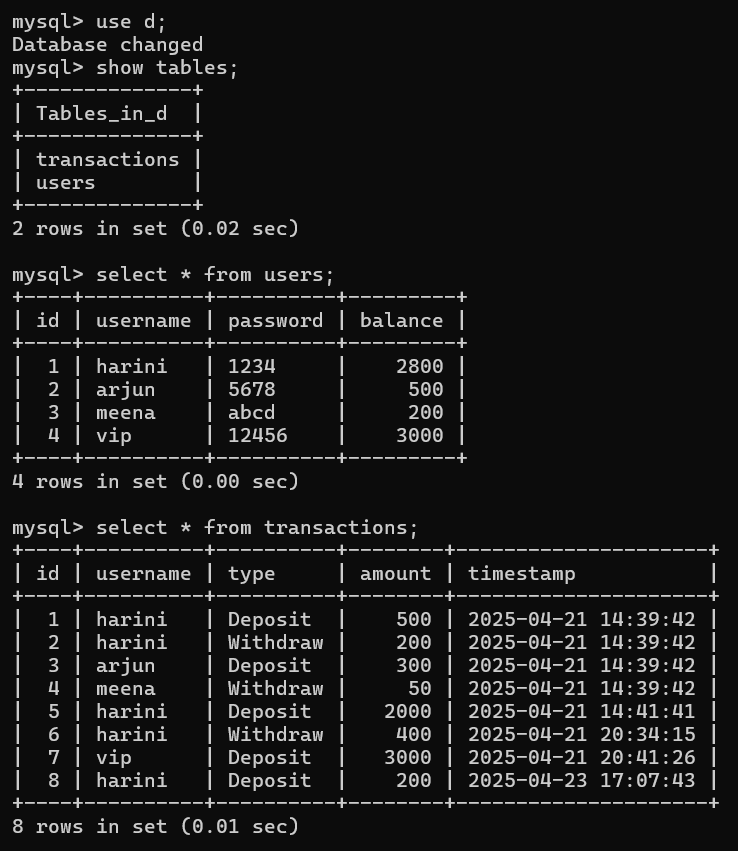
static boolean Register(String username, String password) {

throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody

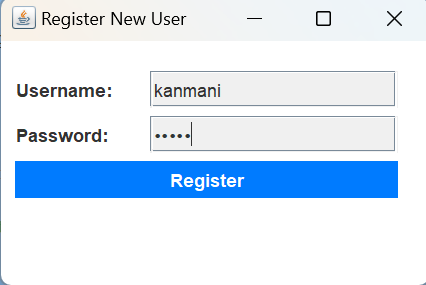
}

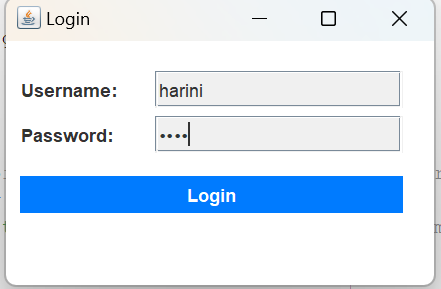
}

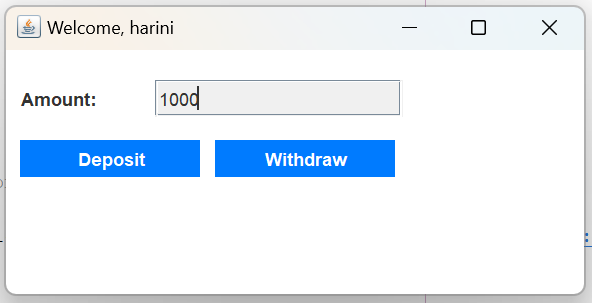
**3.DATABASE VIEW:**

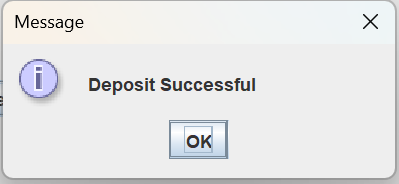


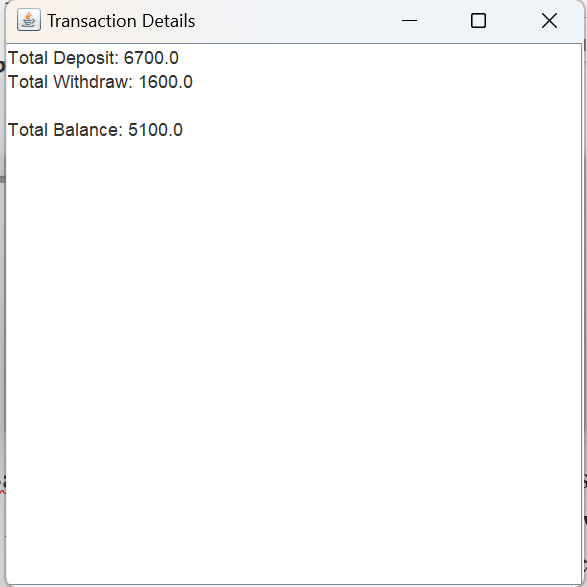
**4.OUTPUT SCREENSHOTS:**

****

****

****

****

****

**5.RESULT:**

The MyBank Database Management System was successfully developed to perform key banking operations like deposit, withdrawal, and balance inquiry. The system runs smoothly with accurate data handling and a user-friendly interface.