

MINI PROJECT

TITLE : INCOME MANAGEMENT

DONE BY

LAKSHMINARAYANAN K

LAKSHMIKANTH K

NITHEESH K

AIM: To create a java program using swing programming on the income management.

LOGIN PAGE:

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

public class LoginPage extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    JTextField txtUsername;
    JPasswordField txtPassword;
    JButton btnLogin, btnRegister;

    public LoginPage() { setTitle("Login
    Page");
    setLayout(new GridLayout(3, 2));

    add(new JLabel("Username:"));
    txtUsername = new JTextField();
    add(txtUsername);
```

```
add(new JLabel("Password:"));
txtPassword = new JPasswordField();
add(txtPassword);

btnLogin = new JButton("Login");
btnLogin.addActionListener(this);
add(btnLogin);

btnRegister = new JButton("Register");
btnRegister.addActionListener(this);
add(btnRegister);

setSize(300,150);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setVisible(true);
}

public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btnLogin) { loginUser();
    } else if (e.getSource() == btnRegister) {
        new RegisterPage(); this.dispose();
    }
}

private void loginUser() { try (Connection conn =
    DBConnection.getConnection()) { String username =
    txtUsername.getText();
    String password = new String(txtPassword.getPassword());
```

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

try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
    pstmt.setString(1, username); pstmt.setString(2, password);

    ResultSet rs = pstmt.executeQuery();
    if (rs.next()) {
        new SalaryManagementSystem();
        this.dispose();
    } else {
        JOptionPane.showMessageDialog(this, "Invalid credentials!");
    }
}

} catch (SQLException ex) { ex.printStackTrace();
    JOptionPane.showMessageDialog(this, "Database Error: " + ex.getMessage());
}

}

public static void main(String[] args) { new
LoginPage();
}
}
```

MAIN PAGE:

```
import javax.swing.*;
```

```
import java.awt.*; import
```

```
java.awt.event.*; import
```

```
java.sql.*;
```

```
public class SalaryManagementSystem extends JFrame implements ActionListener {
```

```
    private static final long serialVersionUID = 1L;
```

```
    JTextField txtName, txtDepartment, txtSalary, txtId;
```

```
    JButton btnAdd, btnUpdate, btnDelete, btnView, btnViewTable;
```

```
    public SalaryManagementSystem() { setTitle("Salary
```

```
    Management System"); setLayout(new
```

```
    GridLayout(7, 2));
```

```
    add(new JLabel("Employee ID:"));
```

```
    txtId = new JTextField(); add(txtId);
```

```
    add(new JLabel("Name:"));
```

```
    txtName = new JTextField(); add(txtName);
```

```
    add(new JLabel("Department:")); txtDepartment
```

```
    = new JTextField(); add(txtDepartment);
```

```
    add(new JLabel("Salary:")); txtSalary
```

```
    = new JTextField(); add(txtSalary);
```

```
btnAdd = new JButton("Add Employee");  
btnAdd.addActionListener(this); add(btnAdd);
```

```
btnUpdate = new JButton("Update Salary");  
btnUpdate.addActionListener(this); add(btnUpdate);
```

```
btnDelete = new JButton("Delete Employee");  
btnDelete.addActionListener(this); add(btnDelete);
```

```
btnView = new JButton("View Employee");  
btnView.addActionListener(this); add(btnView);
```

```
btnViewTable = new JButton("View Salary Table"); btnViewTable.addActionListener(this);  
add(btnViewTable);
```

```
setSize(500, 400);  
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); setVisible(true);  
}
```

```
public void actionPerformed(ActionEvent e) { try  
(Connection conn = DBConnection.getConnection()) {  
String id = txtId.getText();  
String name = txtName.getText();  
String department = txtDepartment.getText();  
String salary = txtSalary.getText();  
  
if (e.getSource() == btnAdd) {
```

```

String sql = "INSERT INTO employees (name, department, salary) VALUES (?, ?, ?)";
try (PreparedStatement pstmt = conn.prepareStatement(sql)) { pstmt.setString(1,
name); pstmt.setString(2, department); pstmt.setDouble(3,
Double.parseDouble(salary)); pstmt.executeUpdate();
JOptionPane.showMessageDialog(this, "Employee added successfully!");
}
} else if (e.getSource() == btnUpdate) {
String sql = "UPDATE employees SET salary = ? WHERE id = ?";
try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
pstmt.setDouble(1, Double.parseDouble(salary));
pstmt.setInt(2, Integer.parseInt(id)); int rowsUpdated =
pstmt.executeUpdate();
JOptionPane.showMessageDialog(this, rowsUpdated > 0 ?
"Salary updated!" : "Employee not found.");
}
} else if (e.getSource() == btnDelete) {
String sql = "DELETE FROM employees WHERE id = ?"; try
(PreparedStatement pstmt = conn.prepareStatement(sql)) {
pstmt.setInt(1, Integer.parseInt(id)); int rowsDeleted =
pstmt.executeUpdate();
JOptionPane.showMessageDialog(this, rowsDeleted > 0 ? "Employee deleted!" : "Employee not found.");
}
} else if (e.getSource() == btnView) {
String sql = "SELECT * FROM employees WHERE id = ?"; try
(PreparedStatement pstmt = conn.prepareStatement(sql)) {
pstmt.setInt(1, Integer.parseInt(id)); ResultSet rs =
pstmt.executeQuery();
if (rs.next()) {

```

```

txtName.setText(rs.getString("name"));
txtDepartment.setText(rs.getString("department"));
txtSalary.setText(String.valueOf(rs.getDouble("salary")));
} else {
JOptionPane.showMessageDialog(this, "Employee not found.");
}
}
} else if (e.getSource() == btnViewTable) { viewSalaryTable(conn);
}
} catch (SQLException ex) { ex.printStackTrace();
JOptionPane.showMessageDialog(this, "Database Error: " + ex.getMessage());
}
}

```

```

private void viewSalaryTable(Connection conn) { String sql = "SELECT *
FROM employees"; try (PreparedStatement pstmt =
conn.prepareStatement( sql, ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
ResultSet rs = pstmt.executeQuery()) {

```

```

ResultSetMetaData metaData = rs.getMetaData(); int
columnCount = metaData.getColumnCount();

```

```

String[] columnNames = new String[columnCount]; for
(int i = 1; i <= columnCount; i++) {
columnNames[i - 1] = metaData.getColumnName(i);
}

```

```

rs.last();

```



```
int rowCount = rs.getRow(); rs.beforeFirst();
```

```
String[][] data = new String[rowCount][columnCount]; int  
rowIndex = 0; while (rs.next()) { for (int colIndex = 1;  
colIndex <= columnCount; colIndex++) {  
data[rowIndex][colIndex - 1] = rs.getString(colIndex);  
}  
rowIndex++;  
}
```

```
JTable table = new JTable(data, columnNames);  
JScrollPane scrollPane = new JScrollPane(table);  
JFrame tableFrame = new JFrame("Salary Table");  
tableFrame.add(scrollPane); tableFrame.setSize(600,  
400); tableFrame.setVisible(true);
```

```
} catch (SQLException ex) { ex.printStackTrace();  
JOptionPane.showMessageDialog(this, "Error fetching data: " + ex.getMessage());  
}  
}  
}
```