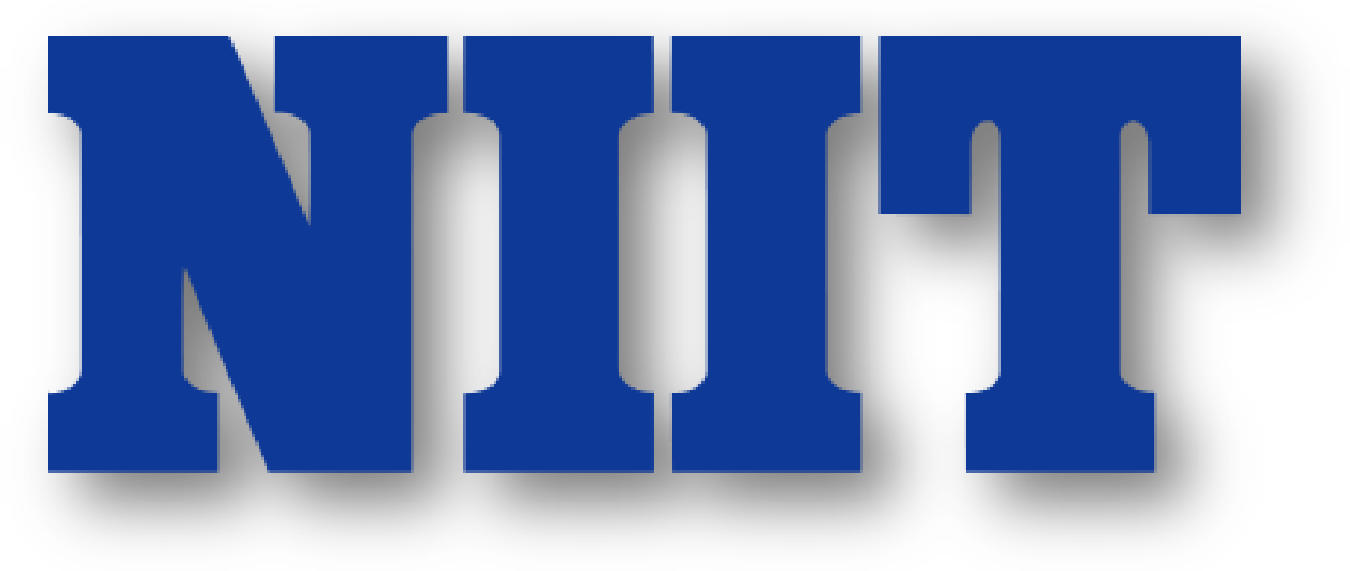
PROJECT ON

**Professionet Consultancy Services (PCS) Application**



**Developed By: - Santhosh Kumar**

**Professionet Consultancy Services (PCS) Application**

Batch Code : S210192

Start Date : 10th December 2020

End Date : 10th January 2021

Name Of The Coordinator : Mrs. Lopamudra Bera

Name Of The Developer : Santhosh Kumar

Date Of Submission : 15th January 2021

**Professionet Consultancy Services (PCS) Application**

CERTIFICATE

This is to certify that this report, titled Professionet Consultancy Services (PCS) Application embodies the original work done by Santhosh Kumar, in partial fulfillment of their course requirement at NIIT.

CO-ORDINATOR:

Acknowledgement

I would like to express my special thanks of gratitude to my teacher Mrs. Lopamudra Bera who gave me the golden opportunity to do this wonderful project on the topic Professionet Consultancy Services (PCS) Application, which also helped me in doing a lot of Research and I came to know about so many new things I am really thankful to them.

Abstract

This project is an application for the Professionet Consultancy Services (PCS), that provides the client to freely be able to manage, store and maintain their company records with a good user interface.

Most of the operation will be done on the user interface designed by awt and the database managed by sql and java.

Configuration

Hardware

Processor : Intel i3 or higher.

RAM : 2GB (minimum)

Speed : 1.5GHz

Secondary

Storage : 10GB

Software :

Java Version : JDK 1.8

Database

Management: MySQL

IDE : Eclipse

OS : Windows 10.

Index

|  |  |  |
| --- | --- | --- |
| SNO | Title | Page No. |
| 1 | Introduction | 08 |
| 2 | Aim and Objective | 09 |
| 3 | Case Study | 10 |
| 4 | Project Requirements and Specification | 12 |
| 5 | Project Analysis | 13 |
| 6 | Implementation and Diagrams | 14 |
| 7 | Source Code Snippets | 17 |
| 8 | Outputs | 45 |
| 9 | References | 46 |

Introduction

This project will help you to implement all the concepts. Strategies, techniques, and technologies it’s that you are laming in the current ten. In this project. you will integrate all those concepts and mate live project by following all the aspects of software engineering and project management. You will implement various software engineering and project management concepts in each and every phase of project development. In addition. this project makes you go through the entire software project life cycle, which will help you to work in a real time application development environment.

Aspiring professionals need to apply their knowledge. skills and concepts that they have learned to develop a software project. In real Inv environment. software companies expect their employs to develop their software applications, which incorporate all the pluses of software engineering and project management. This project enables you to work in a simulated environment where you is ill work as a tram and develop a software application. This project provides a detailed study and implementation of project planning. analysts design. development. and documentation. After completing this project, you will get a confidence to apply your experience while developing projects in a software industry. You will appreciate and understand the need and usage of all the theoretical concepts that you have teamed in various semesters.

Aim and objective of the project

Aim:

The aim of this project is to design and develop a application for the

Professionet Consultancy Services (PCS).

Objective:

* To connect to the data base.
* Let the admin add update modify and maintain the database and the records.
* Provide a good user interface.
* Test the application.

Case study

Professionet Consultancy Services (PCS) is a business consultancy that has established itself as a renowned service provider of a wide range of business services to its clients. It was founded on November 25, 1992, by Dr. Monica Roberts, who is now the CEO of the company. PCS is among the top five business services consultancies in London.

The consultancy has its global presence in Asian, African, American, and European markets. It has multiple teams of expert Business Consultants which leverage the business potential of their clients by offering strategic business management services. Some of the prominent services provided by PCS are resource management, process optimization, IT administration, change management, project management, finance management, and risk management. PCS offers business solution to its clients to meet their business needs on a larger scale. So, there is a great significance of aligning PCS employees to their various clients' projects on the basis of skills of each PCS employee and the project requirement. To meet the human resource requirement of their clients' projects, the consultancy follows conventional recruitment or skill mapping process, i.e., step-by-step consequent process, in which one phase initiates only after the previous phase is completed. PCS offers an offline platform for their employees to share their profiles to initiate internal job posting process with the expert Human Resource (HR) consultants. Face-to face meetings and interview rounds are conducted to align talent supply with business needs. Experienced professionals at PCS can directly consult with the HRs and land a suitable project within the same industry vertical or a different domain within PCS. There are over 22,000 new and 50,000 experienced PCS professionals providing their services to 150 clients aligned with the consultancy. The consultancy needs to maintain the information of every PCS employee focusing on their industry verticals and skills/competencies. All PCS employees should have registered profiles. The consultancy should be updated with the latest profiles of the PCS employees and the vacancies of different job roles on each project and industry vertical. Currently, all PCS employees are registered with PCS and are given a unique identification number. Profile validation is done by the HR experts and requirements are fulfilled by mapping skills and requirements manually. It is important to store employees' profile details, which include their current role, skills, and industry verticals, globally. All CVs/employees' profiles and staffing requirements are stored in a database and as hard copies so that the HR. experts of PCS can refer to them and filter manually.

Project requirement specification

For Employers/Project Managers:

* Create an employer profile.
* Register the employer profile.
* Authenticate the employer profile through email ID and password.
* Update the employer profile.
* Archive the employer profile.
* Suspend the employer profile.
* Subscribe to certain keywords.
* Search through keyword(s) for skills separated by comma, returning profile match percentage, contact details, IBU details, and supervisor name.

For Employees:

* Create an employee profile.
* Register the employee profile.
* Authenticate the employee profile through email ID and password.
* Update the employee profile.
* Archive the employee profile.
* Suspend the employee profile.
* View the employee profile.

For HRs:

* Profile Validation.

Project analysis

11th December 2020: Installed and make a database for the project in MySQL [zero errors]

15th December 2020: Learned about the CRUD Program. In Eclipse. [zero errors]

20th December 2020: Implemented the CRUD Programs and review the performance till then to Co-Ordinator. [zero errors]

29th December 2020: Implemented to connect to the database to manipulate the records through eclipse java code. [zero error]

07th January 2021: Learned to use the awt package and created some frames for the application.

10th January 2021: Tested the login and authentication and perform operation on the database using the UI.

14th January 2021: Completion of Documentation of PCSApplication.

15th January 2021: Submission of the Project PCSApplication.

Implementation and diagrams

Java has been one of the most popular programming languages for many years. Java is Object Oriented. However, it is not considered as pure object oriented as it provides support for primitive data types (like int, char, etc.) The Java codes are first compiled into byte code (machine independent code). Then the byte code is run on Java Virtual Machine (JVM) regardless of the underlying architecture. Java syntax is similar to C/C++. But Java does not provide low level programming functionalities like pointers. Also, Java codes are always written in the form of classes and objects.

Java is used in all kind of applications like Mobile Applications (Android is Java based), desktop applications, web applications, client server applications, enterprise applications and many more.

There is current two sets of Javas APIs for graphics programming: AWT (Abstract Windowing Toolkit), Swing AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components. Swing API, a much more comprehensive set of

graphics libraries that enhance the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC consists of Swing, Java2D,

1. Accessibility, Internationalization, and Pluggable Look-and-Feel Support APIs. JFC has been integrated into core Java since JDK 1.2.

AWT

AWT is huge! It consists of 12 packages of 370 classes (Swing is even bigger, with 18 packages of 737 classes as of JDK 8). Fortunately, only 2 packages - java.awt and java.awt.event - are commonly-used.

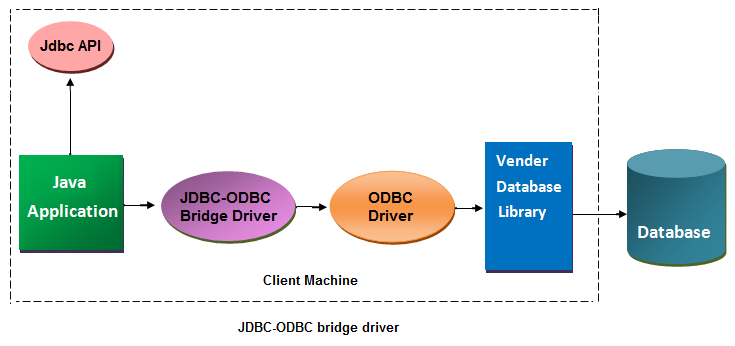
1. The java.awt package contains the core AWT graphics classes:

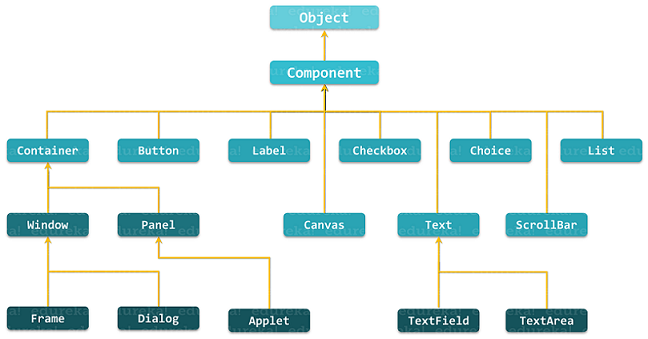
* GUI Component classes, such as Button, TextField, and Label.
* GUI Container classes, such as Frame and Panel.
* Layout managers, such as FlowLayout, BorderLayout and GridLayout.
* Custom graphics classes, such as Graphics, Color and Font.

2. The java.awt.event package supports event handling:

* Event classes, such as ActionEvent, MouseEvent, KeyEvent and WindowEvent,
* Event Listener Interfaces, such as ActionListener, MouseListener, MouseMotionListener, KeyListener and WindowListener,
* Event Listener Adapter classes, such as MouseAdapter, KeyAdapter, and WindowAdapter.

AWT provides a platform-independent and device-independent interface to develop graphic programs that runs on all platforms, including Windows, Mac OS X, and Unixes.





AWT Implementation

Code snippets

//JDBCConnection.java

package config;

import java.sql.\*;

public class JDBCConnection {

static String url="jdbc:mysql://localhost:3306/pcsdb1";

static String username="root";

static String password="niit@123";

static Connection conn=null;

public static Connection getDBConnection() {

try {

Class.forName("com.mysql.jdbc.Driver"); // Loading driver

conn=DriverManager.getConnection(url, username, password);

}

catch(ClassNotFoundException ex) {

System.out.println(ex);

}

catch(SQLException ex) {

System.out.println(ex);

}

return conn;

}

}

//Employee.java

package model;

public class Employee {

private int employeeId;

private String firstName;

private String lastName;

private String email;

private String userId;

private String password;

private String gender;

private String role;

private String active;

//default constructor method

public Employee() {

super();

// TODO Auto-generated constructor stub

}

//parameterized constructor method will all fields

public Employee(int employeeId, String firstName, String lastName, String email, String userId, String password,

String gender, String role, String active) {

super();

this.employeeId = employeeId;

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

this.userId = userId;

this.password = password;

this.gender = gender;

this.role = role;

this.active = active;

}

//parameterized constructor method without employeeid and active fields

public Employee(String firstName, String lastName, String email, String userId, String password, String gender,

String role) {

super();

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

this.userId = userId;

this.password = password;

this.gender = gender;

this.role = role;

}

//getter and setter methods

public int getEmployeeId() {

return employeeId;

}

public void setEmployeeId(int employeeId) {

this.employeeId = employeeId;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public String getActive() {

return active;

}

public void setActive(String active) {

this.active = active;

}

@Override

public String toString() {

return "Employee [employeeId=" + employeeId + ", firstName=" + firstName + ", lastName=" + lastName + ", email="

+ email + ", userId=" + userId + ", password=" + password + ", gender=" + gender + ", role=" + role

+ ", active=" + active + "]";

}

}

//IEmployeeDao.java

package dao;

import java.util.List;

import model.Employee;

public interface IEmployeeDao {

List<Employee> getAllEmployees();

void addEmployee(Employee emp);

Employee getEmployeeById(int id);

void updateEmployee(Employee emp);

void deactivateEmployee(Employee emp);

void activateEmployee(Employee emp);

void deleteEmployee(int id);

}

//EmployeeDaoImpl.java

package daoImpl;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

import config.JDBCConnection;

import dao.IEmployeeDao;

import model.Employee;

public class EmployeeDaoImpl implements IEmployeeDao{

Connection conn=null;

public EmployeeDaoImpl() {

conn=JDBCConnection.getDBConnection();

}

@Override

public List<Employee> getAllEmployees() {

List<Employee> empList=new ArrayList<Employee>();

try {

String query="select \* from Employee";

Statement stmt=conn.createStatement();

ResultSet rst=stmt.executeQuery(query);

if(rst!=null) {

while(rst.next()) {

Employee emp=new Employee();

emp.setEmployeeId(rst.getInt(1));

emp.setFirstName(rst.getString(2));

emp.setLastName(rst.getString(3));

emp.setEmail(rst.getString(4));

emp.setUserId(rst.getString(5));

emp.setPassword(rst.getString(6));

emp.setRole(rst.getString(7));

emp.setGender(rst.getString(8));

emp.setActive(rst.getString(9));

empList.add(emp);

}

}

}catch(SQLException ex) {

System.out.println(ex);

}

return empList;

}

@Override

public void addEmployee(Employee emp) {

try {

String query="insert into Employee(FirstName, LastName, Email, UserId, Password, Role, Gender, Active) values(?,?,?,?,?,?,?,?)";

PreparedStatement pst=conn.prepareStatement(query);

pst.setString(1,emp.getFirstName());

pst.setString(2,emp.getLastName());

pst.setString(3,emp.getEmail());

pst.setString(4,emp.getUserId());

pst.setString(5,emp.getPassword());

pst.setString(6,emp.getRole());

pst.setString(7,emp.getGender());

pst.setString(8,emp.getActive());

int i=pst.executeUpdate();

if(i==1) {

System.out.println("1 record inserted...");

}

else {

System.out.println("Insertion failed...");

}

}catch(SQLException ex) {

System.out.println(ex);

}

}

@Override

public Employee getEmployeeById(int id) {

Employee emp=new Employee();

try {

String query="select \* from Employee where EmployeeId=?";

PreparedStatement pst=conn.prepareStatement(query);

pst.setInt(1,id);

ResultSet rst=pst.executeQuery();

if(rst!=null) {

while(rst.next()) {

emp.setEmployeeId(rst.getInt(1));

emp.setFirstName(rst.getString(2));

emp.setLastName(rst.getString(3));

emp.setEmail(rst.getString(4));

emp.setUserId(rst.getString(5));

emp.setPassword(rst.getString(6));

emp.setRole(rst.getString(7));

emp.setGender(rst.getString(8));

emp.setActive(rst.getString(9));

}

}

}catch(SQLException ex) {

System.out.println(ex);

}

return emp;

}

@Override

public void updateEmployee(Employee emp) {

try {

String query = "UPDATE Employee SET password=?, email=? WHERE EmployeeId=?";

PreparedStatement statement = conn.prepareStatement(query);

statement.setString(1, emp.getPassword());

statement.setString(2, emp.getEmail());

statement.setInt(3,emp.getEmployeeId());

int rowsUpdated = statement.executeUpdate();

if (rowsUpdated > 0) {

System.out.println("An existing user was updated successfully!");

}

else {

System.out.println("updation failed...");

}

}catch(SQLException ex) {

System.out.println(ex);

}

}

@Override

public void deactivateEmployee(Employee emp) {

try {

//creating PreparedStatement object by passing query string

PreparedStatement pst=conn.prepareStatement("update Employee set Active=? where EmployeeId=?");

pst.setString(1, "Deactive");

pst.setInt(2, emp.getEmployeeId());

int i=pst.executeUpdate();

if(i==1){

System.out.println("Employee deactivated...");

}

else {

System.out.println("updation failed...");

}

}

catch(SQLException ex) {

System.out.println(ex.getMessage());

}

}

@Override

public void activateEmployee(Employee emp) {

try {

//creating PreparedStatement object by passing query string

PreparedStatement pst=conn.prepareStatement("update Employee set Active=? where EmployeeId=?");

pst.setString(1, "Active");

pst.setInt(2, emp.getEmployeeId());

int i=pst.executeUpdate();

if(i==1){

System.out.println("Employee Activated...");

}

else {

System.out.println("updation failed...");

}

}

catch(SQLException ex) {

System.out.println(ex.getMessage());

}

}

@Override

public void deleteEmployee(int id) {

try {

//creating PreparedStatement object by passing query string

PreparedStatement pst=conn.prepareStatement("delete from Employee where EmployeeId=?");

pst.setInt(1, id);

int i=pst.executeUpdate();

if(i==1){

System.out.println("Employee deleted...");

}

else {

System.out.println("deletion failed...");

}

}

catch(SQLException ex) {

System.out.println(ex.getMessage());

}

}

}

//EmployeeController.java

package controller;

import java.util.List;

import dao.IEmployeeDao;

import daoImpl.EmployeeDaoImpl;

import model.Employee;

public class EmployeeController {

IEmployeeDao empDao=null;

public EmployeeController() {

empDao=new EmployeeDaoImpl();

}

public List<Employee> getAllEmployees(){

List<Employee> empList=empDao.getAllEmployees();

return empList;

}

public void addEmployee(Employee emp) {

empDao.addEmployee(emp);

}

public Employee getEmployeeById(int id) {

Employee emp=empDao.getEmployeeById(id);

return emp;

}

public void updateEmployee(Employee emp) {

empDao.updateEmployee(emp);

}

public void deleteEmployee(int id) {

empDao.deleteEmployee(id);

}

public void deactivateEmployee(Employee emp) {

empDao.deactivateEmployee(emp);

}

public void activateEmployee(Employee emp) {

empDao.activateEmployee(emp);

}

}

//EmployeeExec.java

package excDao;

import java.util.Iterator;

import java.util.List;

import java.io.\*;

import controller.EmployeeController;

import model.Employee;

public class EmployeeExec {

EmployeeController empController=null;

public EmployeeExec() {

empController=new EmployeeController();

}

public void getAllEmployees() {

List<Employee> empList=empController.getAllEmployees();

for(Employee emp:empList) {

System.out.println(emp);

}

}

public void getEmployeeById() {

int id=0;

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter employee id whose record you want to view:");

id=Integer.parseInt(reader.readLine());

}

catch(IOException ex){

System.out.println(ex);

}

Employee emp=empController.getEmployeeById(id);

System.out.println(emp);

}

public void addEmployee() {

Employee emp=new Employee();

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter Employee Detail:");

System.out.println("First Name:");

emp.setFirstName(reader.readLine());

System.out.println("Last Name:");

emp.setLastName(reader.readLine());

System.out.println("Email:");

emp.setEmail(reader.readLine());

System.out.println("User Id:");

emp.setUserId(reader.readLine());

System.out.println("Password:");

emp.setPassword(reader.readLine());

System.out.println("Role:");

String role=reader.readLine();

emp.setRole(role);

System.out.println("Gender:");

emp.setGender(reader.readLine());

if(role.equals("HRA")) {

emp.setActive("Active");

}

else {

emp.setActive("Deactive");

}

}catch(IOException ex){

System.out.println(ex);

}

empController.addEmployee(emp);

}

public void updateEmployee() {

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

int id;

String email, password, confirmpassword;

System.out.println("Enter EmployeeId whose record you want to update:");

id=Integer.parseInt(reader.readLine());

Employee emp=empController.getEmployeeById(id);

System.out.println("Enter your new email:");

email=reader.readLine();

System.out.println("Enter your new password:");

password=reader.readLine();

System.out.println("Re-enter same password to confirm:");

confirmpassword=reader.readLine();

if(password.equals(confirmpassword)) {

emp.setPassword(password);

emp.setEmail(email);

empController.updateEmployee(emp);

}

else {

System.out.println("Sorry! you have entered different password!");

}

}

catch(IOException ex) {

System.out.println(ex.getMessage());

}

}

public void deactiveEmployee() {

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

int id;

System.out.println("Enter EmployeeId whose record you want to deactivate:");

id=Integer.parseInt(reader.readLine());

Employee emp=empController.getEmployeeById(id);

empController.deactivateEmployee(emp);

}

catch(IOException ex) {

System.out.println(ex.getMessage());

}

}

public void activeEmployee() {

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

int id;

System.out.println("Enter EmployeeId whose record you want to activate:");

id=Integer.parseInt(reader.readLine());

Employee emp=empController.getEmployeeById(id);

empController.activateEmployee(emp);

}

catch(IOException ex) {

System.out.println(ex.getMessage());

}

}

public void deleteEmployee() {

try {

BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));

int id;

System.out.println("Enter EmployeeId whose record you want to delete:");

id=Integer.parseInt(reader.readLine());

empController.deleteEmployee(id);

}

catch(IOException ex) {

System.out.println(ex.getMessage());

}

}

}

//TestApp.java

package entry;

import java.sql.SQLException;

import java.util.Scanner;

import config.JDBCConnection;

import excDao.EmployeeExec;

public class TestApp {

public void testConnection() {

try {

if(JDBCConnection.getDBConnection().isClosed()) {

System.out.println("Connection is closed");

}

else {

System.out.println("Connection is Opened");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public void processMenu() {

EmployeeExec obj=new EmployeeExec();

Scanner sc=new Scanner(System.in);

int option=0;

char ch='y';

do {

System.out.println("----------CRUD Operation-----------");

System.out.println("1. View all Employees");

System.out.println("2. View single Employee");

System.out.println("3. Add Employee");

System.out.println("4. Update Employee");

System.out.println("5. Delete Employee");

System.out.println("6. Deactivate Employee");

System.out.println("7. Activate Employee");

System.out.println("8. Quit");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

System.out.println("Enter your choice:");

option=sc.nextInt();

switch(option) {

case 1:

obj.getAllEmployees();

break;

case 2:

obj.getEmployeeById();

break;

case 3:

obj.addEmployee();

break;

case 4:

obj.updateEmployee();

break;

case 5:

obj.deleteEmployee();

break;

case 6:

obj.deactiveEmployee();

break;

case 7:

obj.activeEmployee();

break;

case 8:

System.exit(0);

break;

default:

System.out.println("Wrong input!");

break;

}

System.out.println("Do you want to continue(y/n)?");

ch=sc.next().charAt(0);

}while(ch=='y' || ch=='Y');

}

public static void main(String[] args) {

TestApp test=new TestApp();

//test.testConnection();

test.processMenu();

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class ActivateFrame extends JFrame {

Container container;

public ActivateFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("ACTIVATE USER");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false);

}

private void addComponentsToContainer() {

// TODO Auto-generated method stub

}

private void setLocationAndSize() {

// TODO Auto-generated method stub

}

private void setLayoutManager() {

// TODO Auto-generated method stub

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class AddSkillFrame extends JFrame {

Container container;

public AddSkillFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("ADD SKILL");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false);

}

private void addComponentsToContainer() {

// TODO Auto-generated method stub

}

private void setLocationAndSize() {

// TODO Auto-generated method stub

}

private void setLayoutManager() {

// TODO Auto-generated method stub

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class EmpHomeFrame extends JFrame{

Container container;

public EmpHomeFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("EMP HOME");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false); // fixed size frame

}

public void setLayoutManager() {

container.setLayout(null);

}

public void setLocationAndSize() {

}

public void addComponentsToContainer() {

}

public void callDispose() {

this.dispose();

}

}

package view;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

public class HraHomeFrame extends JFrame{

Container container;

JButton bLogout, bActivate, bViewAllEmployee, bAddSkill;

public HraHomeFrame() {

container=getContentPane();

bLogout=new JButton("LOGOUT");

bLogout.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

callDispose();

new LoginFrame();

}

});

bActivate=new JButton("ACTIVATE USERS");

bActivate.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

callDispose();

new ActivateFrame();

}

});

bViewAllEmployee=new JButton("VIEW ALL EMPLOYESS");

bViewAllEmployee.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

callDispose();

new ViewAllEmployeeFrame();

}

});

bAddSkill=new JButton("ADD SKILL");

bViewAllEmployee.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

callDispose();

new AddSkillFrame();

}

});

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("HRA HOME");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false); // fixed size frame

}

public void setLayoutManager() {

container.setLayout(null);

}

public void setLocationAndSize() {

bLogout.setBounds(100, 250, 300, 30);

bActivate.setBounds(100, 150, 300, 30);

bViewAllEmployee.setBounds(100, 200, 300, 30);

bAddSkill.setBounds(100, 100, 300, 30);

}

public void addComponentsToContainer() {

container.add(bLogout);

container.add(bActivate);

container.add(bViewAllEmployee);

container.add(bAddSkill);

}

public void callDispose() {

this.dispose();

}

}

package view;

import java.awt.Container;

import java.awt.event.\*;

import javax.swing.\*;

import controller.EmployeeController;

import model.Employee;

public class LoginFrame extends JFrame{

Container container;

JLabel lUserId, lPassword, lMessage;

JTextField tUserId;

JPasswordField tPassword;

JButton bLogin, bRegister;

JCheckBox cShowPassword;

EmployeeController empController=null;

public LoginFrame() {

container=getContentPane();

lUserId=new JLabel("USERNAME");

lPassword=new JLabel("PASSWORD");

tUserId=new JTextField();

tPassword=new JPasswordField();

lMessage=new JLabel();

empController=new EmployeeController();

bLogin=new JButton("LOGIN");

bLogin.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String userId, password;

userId=tUserId.getText();

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

Employee emp=new Employee();

emp=empController.checkLogin(userId, password);

if(emp==null) {

lMessage.setText("You are not authorised user! Retry or Register!");

}

else {

if(emp.getActive().equals("Active")) {

if(emp.getRole().equals("HRA")) {

callDispose();

new HraHomeFrame();

}

else if(emp.getRole().equals("PME")) {

callDispose();

new PmeHomeFrame();

}

else {

callDispose();

new EmpHomeFrame();

}

}

}

}

});

bRegister=new JButton("REGISTER");

bRegister.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

callDispose();

new RegisterFrame();

}

});

cShowPassword=new JCheckBox("Show Password");

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("LOGIN FORM");

this.setVisible(true);

this.setBounds(10, 10, 500, 600);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false); // fixed size frame

}

public void setLayoutManager() {

container.setLayout(null);

}

public void setLocationAndSize() {

lUserId.setBounds(50, 150, 100, 30);

lPassword.setBounds(50,250,100,30);

tUserId.setBounds(250,150,150,30);

tPassword.setBounds(250,250,150,30);

cShowPassword.setBounds(250,300,150,30);

lMessage.setBounds(100, 350, 250, 30);

bLogin.setBounds(100,400,100,30);

bRegister.setBounds(200,400,100,30);

}

public void addComponentsToContainer() {

container.add(lUserId);

container.add(tUserId);

container.add(lPassword);

container.add(tPassword);

container.add(cShowPassword);

container.add(bLogin);

container.add(bRegister);

container.add(lMessage);

}

public void callDispose() {

this.dispose();

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class PmeHomeFrame extends JFrame{

Container container;

public PmeHomeFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("PME HOME");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false); // fixed size frame

}

public void setLayoutManager() {

container.setLayout(null);

}

public void setLocationAndSize() {

}

public void addComponentsToContainer() {

}

public void callDispose() {

this.dispose();

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class RegisterFrame extends JFrame{

Container container;

public RegisterFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("REGISTRATION FORM");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false); // fixed size frame

}

public void setLayoutManager() {

container.setLayout(null);

}

public void setLocationAndSize() {

}

public void addComponentsToContainer() {

}

public void callDispose() {

this.dispose();

}

}

package view;

import java.awt.Container;

import javax.swing.\*;

public class ViewAllEmployeeFrame extends JFrame {

Container container;

public ViewAllEmployeeFrame() {

container=getContentPane();

setLayoutManager();

setLocationAndSize();

addComponentsToContainer();

this.setTitle("ACTIVATE USER");

this.setVisible(true);

this.setBounds(10, 10, 500, 700);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setResizable(false);

}

private void addComponentsToContainer() {

// TODO Auto-generated method stub

}

private void setLocationAndSize() {

// TODO Auto-generated method stub

}

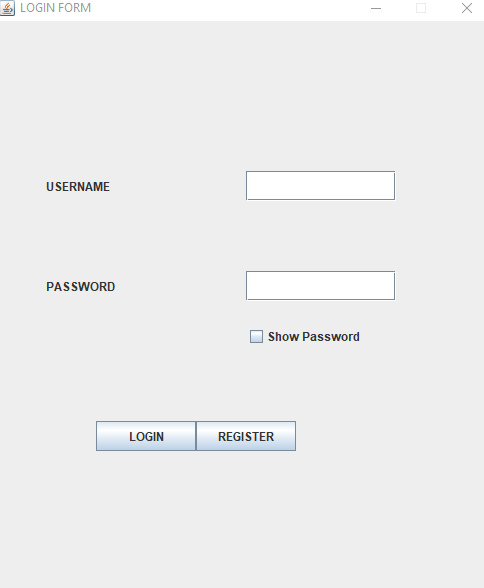
private void setLayoutManager() {

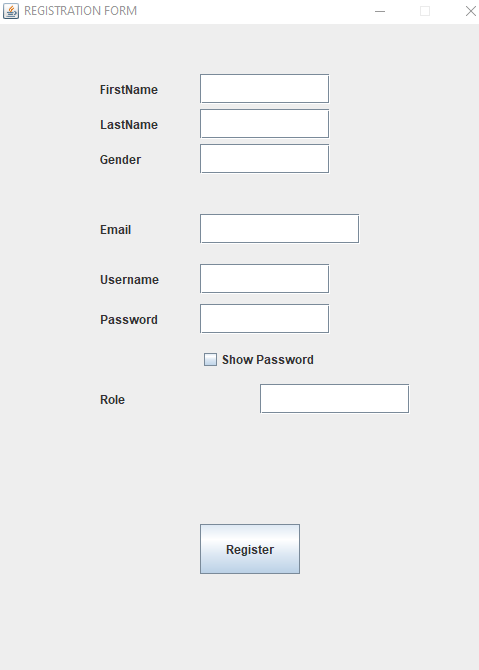
// TODO Auto-generated method stub

}

}

Output





References

* NIIT Project Guide
* Google