CRUD Employee

create database employeedb;

use employeedb;

create table tbl\_employee(

id int not null primary key auto\_increment,

username varchar(50) not null,

password varchar(255) not null,

name varchar(50) not null,

dob varchar(50) not null,

department varchar(5) not null

);

INSERT INTO `tbl\_employee` (`id`, `username`, `password`, `name`, `dob`, `department`) VALUES (NULL, 'hoa', 'hoa', 'Nguyen Hoa', '2003-03-02', 'IT'), (NULL, 'binh', 'binh', 'Tran Binh', '2005-04-05', 'HR'), (NULL, 'thien', 'thien', 'Nguyen Thien', '2003-05-01', 'IT'), (NULL, 'thoi', 'thoi', 'Tran thoi', '2005-09-02', 'HR'), (NULL, 'dia', 'dia', 'Tran dia', '2001-02-05', 'HR'), (NULL, 'loi', 'loi', 'Phan loi', '2005-12-05', 'IT'),(NULL, 'nhan', 'nhan', 'Tran nhan', '2005-09-07', 'HR'), (NULL, 'le', 'le', 'Tran le', '2005-09-12', 'HR'), (NULL, 'nghia', 'nghia', 'Tran nghia', '2005-09-20', 'HR');;

create table tbl\_department(

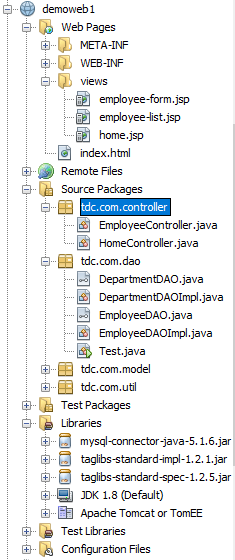
id varchar(5) not null primary key,

name varchar(30) not null,

description varchar(50) not null

);

INSERT INTO `tbl\_department` (`id`, `name`, `description`) VALUES ('IT', 'Công nghệ thông tin', 'Phòng hỗ trợ IT'), ('HR', 'Nhân sự', 'Phòng quản lý nhân sự');



package tdc.com.util;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class DBConnectionUtil {

private static final String URL = "jdbc:mysql://localhost:3306/employeedb";

private static final String DRIVER = "com.mysql.jdbc.Driver";

private static final String USERNAME = "root";

private static final String PASSWORD = "root";

private static Connection connection = null;

public static Connection openConnection() {

if (connection != null) {

return connection;

} else {

try {

Class.forName(DRIVER);

connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

return connection;

}

}

public static void main(String[] args) {

System.out.println(DBConnectionUtil.openConnection());

}

} }

}

package tdc.com.model;

public class Employee {

private Integer id;

private String username;

private String password;

private String name;

private String department;

private String dob;

public Employee() {

}

public Employee(Integer id, String username, String password, String name, String department, String dob) {

this.id = id;

this.username = username;

this.password = password;

this.name = name;

this.department = department;

this.dob = dob;

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

public String getDob() {

return dob;

}

public void setDob(String dob) {

this.dob = dob;

}

@Override

public String toString() {

return "Employee{" + "id=" + id + ", username=" + username + ", password=" + password + ", name=" + name + ", department=" + department + ", dob=" + dob + '}';

}

}

package tdc.com.dao;

import java.util.List;

import tdc.com.model.Employee;

public interface EmployeeDAO {

List<Employee> getAllEmployees();

Employee get(int id);

boolean save(Employee employee);

boolean delete(int id);

boolean update(Employee employee);

}

package tdc.com.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import tdc.com.model.Employee;

import tdc.com.util.DBConnectionUtil;

public class EmployeeDAOImpl implements EmployeeDAO{

Connection connection = null;

ResultSet resultSet = null;

Statement statement = null;

PreparedStatement preparedStatement = null;

public List<Employee> getAllEmployees() {

List<Employee> list = null;

Employee employee = null;

try {

list = new ArrayList<Employee>();

String sql = "SELECT \* FROM tbl\_employee";

connection = DBConnectionUtil.openConnection();

statement = connection.createStatement();

resultSet = statement.executeQuery(sql); //executeUpdate()

while (resultSet.next()) {

employee = new Employee();

employee.setId(resultSet.getInt("id"));

employee.setUsername(resultSet.getString("username"));

employee.setPassword(resultSet.getString("password"));

employee.setName(resultSet.getString("name"));

employee.setDob(resultSet.getString("dob"));

employee.setDepartment(resultSet.getString("department"));

list.add(employee);

}

} catch (SQLException e) {

e.printStackTrace();

}

return list;

}

public Employee get(int id) {

Employee employee = null;

try {

employee = new Employee();

String sql = "SELECT \* FROM tbl\_employee where id=" + id;

connection = DBConnectionUtil.openConnection();

statement = connection.createStatement();

resultSet = statement.executeQuery(sql);

if (resultSet.next()) {

employee.setId(resultSet.getInt("id"));

employee.setUsername(resultSet.getString("username"));

employee.setPassword(resultSet.getString("password"));

employee.setName(resultSet.getString("name"));

employee.setDob(resultSet.getString("dob"));

employee.setDepartment(resultSet.getString("department"));

}

} catch (SQLException e) {

e.printStackTrace();

}

return employee;

}

public boolean save(Employee e) {

boolean flag = false;

try {

String sql = "INSERT INTO tbl\_employee(name, username, password, dob, department)VALUES"

+ "('" + e.getName() + "', '" + e.getUsername()+ "', '"+ e.getPassword()

+ "', '" + e.getDob() +"', '"+ e.getDepartment() +"')";

connection = DBConnectionUtil.openConnection();

preparedStatement = connection.prepareStatement(sql);

preparedStatement.executeUpdate();

flag = true;

} catch (SQLException ex) {

ex.printStackTrace();

}

return flag;

}

public boolean delete(int id) {

boolean flag = false;

try {

String sql = "DELETE FROM tbl\_employee where id=" + id;

connection = DBConnectionUtil.openConnection();

preparedStatement = connection.prepareStatement(sql);

preparedStatement.executeUpdate();

flag = true;

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

public boolean update(Employee employee) {

boolean flag = false;

try {

String sql = "UPDATE tbl\_employee SET username = ?, password = ?, name = ?, dob = ?, department = ? where id= ?";

connection = DBConnectionUtil.openConnection();

preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, employee.getUsername());

preparedStatement.setString(2, employee.getPassword());

preparedStatement.setString(3, employee.getName());

preparedStatement.setString(4, employee.getDob());

preparedStatement.setString(5, employee.getDepartment());

preparedStatement.setInt(6, employee.getId());

int ud = preparedStatement.executeUpdate();

System.out.println("ud = "+ud);

flag = true;

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

}

Controller

package tdc.com.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import tdc.com.dao.EmployeeDAO;

import tdc.com.dao.EmployeeDAOImpl;

import tdc.com.model.Employee;

@WebServlet(name = "employeecontroller", urlPatterns = {"/employeecontroller"})

public class EmployeeController extends HttpServlet {

private static final long serialVersionUID = 1L;

RequestDispatcher dispatcher = null;

EmployeeDAO employeeDAO = null;

public EmployeeController() {

employeeDAO = new EmployeeDAOImpl();

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

String action = request.getParameter("action");

if(action == null) {

action = "LIST";

}

switch(action) {

case "LIST":

listEmployee(request, response);

break;

case "EDIT":

getSingleEmployee(request, response);

break;

case "DELETE":

deleteEmployee(request, response);

break;

default:

listEmployee(request, response);

break;

}

}

private void deleteEmployee(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

String id = request.getParameter("id");

if(employeeDAO.delete(Integer.parseInt(id))) {

request.setAttribute("NOTIFICATION", "Employee Deleted Successfully!");

}

listEmployee(request, response);

}

private void getSingleEmployee(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{

String id = request.getParameter("id");

Employee theEmployee = employeeDAO.get(Integer.parseInt(id));

request.setAttribute("employee", theEmployee);

dispatcher = request.getRequestDispatcher("/views/employee-form.jsp");

dispatcher.forward(request, response);

}

private void listEmployee(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

List<Employee> theList = employeeDAO.getAllEmployees();

request.setAttribute("list", theList);

dispatcher = request.getRequestDispatcher("/views/employee-list.jsp");

dispatcher.forward(request, response);

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

String id = request.getParameter("id");

Employee e = new Employee();

e.setUsername(request.getParameter("username"));

e.setPassword(request.getParameter("password"));

e.setName(request.getParameter("name"));

e.setDob(request.getParameter("dob"));

e.setDepartment(request.getParameter("department"));

if(id.isEmpty() || id == null) {

if(employeeDAO.save(e)) {

request.setAttribute("NOTIFICATION", "Employee Saved Successfully!");

}

}else {

e.setId(Integer.parseInt(id));

if(employeeDAO.update(e)) {

request.setAttribute("NOTIFICATION", "Employee Updated Successfully!");

}

}

listEmployee(request, response);

}

}

employee-list.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Employee Directory</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

</head>

<body>

<div class = "container">

<h1>Employee CRUD</h1>

<hr/>

<p>${NOTIFICATION}</p>

<p>

<button class = "btn btn-primary" onclick="window.location.href = 'views/employee-form.jsp'">Add Employee</button>

</p>

<table class = "table table-striped table-bordered">

<tr class = "thead-dark">

<th>UserName</th>

<th>Name</th>

<th>Department</th>

<th>Date of birth</th>

<th>Actions</th>

</tr>

<c:forEach items="${list}" var="employee">

<tr>

<td>${employee.username}</td>

<td>${employee.name}</td>

<td>${employee.department}</td>

<td>${employee.dob}</td>

<td>

<a href = "${pageContext.request.contextPath}/employeecontroller?action=EDIT&id=${employee.id}">Edit</a>

|

<a href = "${pageContext.request.contextPath}/employeecontroller?action=DELETE&id=${employee.id}">Delete</a>

</td>

</tr>

</c:forEach>

</table>

</div>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

</body>

</html>

Employee-form.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Insert title here</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

</head>

<body>

<div class = "container">

<h1>Employee Register</h1>

<hr/>

<div class = "row">

<div class = "col-md-4">

<form action = "${pageContext.request.contextPath}/employeecontroller" method="POST">

<div class = "form-group">

<input type = "text" class = "form-control" name = "username" placeholder = "Enter UserName" value = "${employee.username}"/>

<div class = "form-group">

<input type = "text" class = "form-control" name = "password" placeholder = "Enter Password" value = "${employee.password}"/>

<div class = "form-group">

<input type = "text" class = "form-control" name = "name" placeholder = "Enter Name" value = "${employee.name}"/>

</div>

<div class = "form-group">

<input type = "date" class = "form-control" name = "dob" value = "${employee.dob}"/>

</div>

<div class="form-group">

<input type = "text" class = "form-control" name = "department" placeholder = "Enter Department" value = "${employee.department}"/>

</div>

<input type = "hidden" name = "id" value = "${employee.id}"/>

<button type = "submit" class = "btn btn-primary">Save</button>

</form>

</div>

</div>

<a href = "${pageContext.request.contextPath}/employeecontroller?action=LIST">Back to List</a>

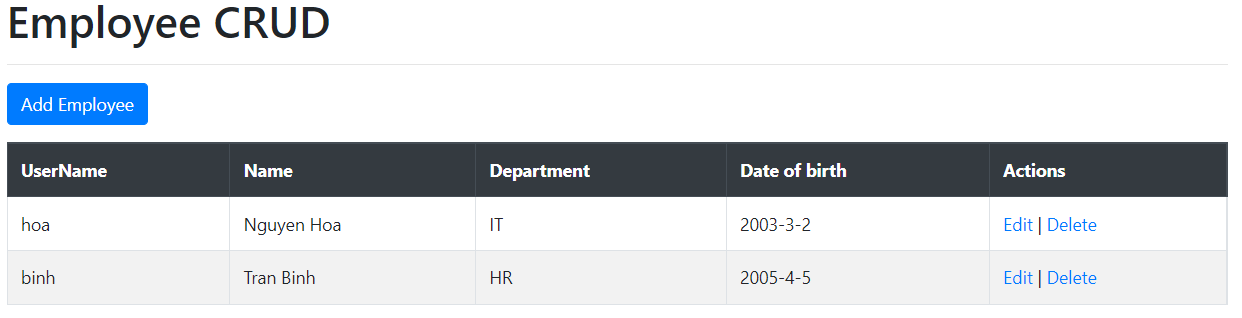
</div>

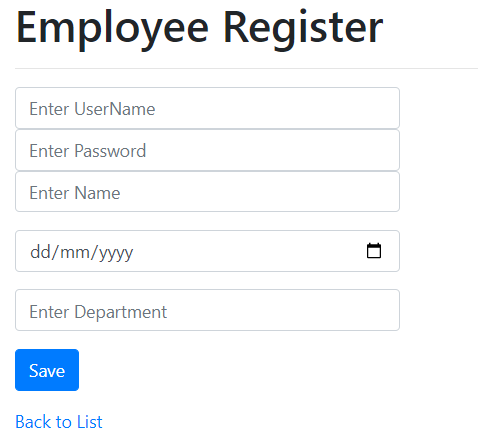
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

</body>

</html>





Yêu cầu:

1. Làm tương tự cho trang Department CRUD
2. Làm trang home.jsp cho hiển thị tất cả các Department và Employee. Hiển thị Employee theo Department.
3. Điều chỉnh form Đăng ký Employee cho phép chọn tên Department được load lên từ CSDL. Và Danh sách Employee hiển thị tên Department. Lưu ý trong CSDL tbl\_employee vẫn lưu mã Department.

List Employee by Department

Employee-form.jsp

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Home</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"></script>

<style>

ul.breadcrumb {

padding: 10px 16px;

list-style: none;

background-color: #eee;

border-radius: 10px;

}

</style>

</head>

<body>

<div class="container">

<div class="row">

<div class="col">

<nav aria-label="breadcrumb">

<ul class="breadcrumb">

<li class="breadcrumb-item"><a href="${pageContext.request.contextPath}/homecontroller">Home</a></li>

<li class="breadcrumb-item"><a href="#"></a><%= (request.getParameter("depid")==null)? "" : request.getParameter("depid") %></li>

</ul>

</nav>

</div>

</div>

<div class="row">

<div class="col-sm-3">

<div class="card bg-light mb-3">

<div class="card-header bg-primary text-white text-uppercase"><i class="fa fa-list"></i> Departments</div>

<ul class="list-group category\_block">

<c:forEach items="${listDepartment}" var="d">

<li class="list-group-item text-white"><a href="homecontroller?depid=${d.id}">${d.name}</a></li>

</c:forEach>

</ul>

</div>

<div class="card bg-light mb-3">

<div class="card-header bg-primary text-white text-uppercase">\*\*\*</div>

<div class="card-body">

</div>

</div>

</div>

<div class="col-sm-9">

<div class="row">

<c:forEach items="${listEmployee}" var="o">

<div class="col-12 col-md-6 col-lg-4">

<div class="card">

<div class="card-body">

<h4 class="card-title show\_txt"><a href="#" title="View Product">${o.name}</a></h4>

<p class="card-text show\_txt">${o.dob}</p>

</div>

</div>

</div>

</c:forEach>

</div>

</div>

</div>

</div>

</body>

</html>

