**Create a spring boot web mvc project for employee management**

**EmployeeApplication.java**

package com.marticus.employee;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class EmployeeApplication {

public static void main(String[] args) {

SpringApplication.run(EmployeeApplication.class, args);

}

}

**AuthController.java**

package com.marticus.employee.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.PostMapping;

import com.marticus.employee.entity.UserEntity;

import com.marticus.employee.service.EmployeeService;

@Controller

public class AuthController {

@Autowired

EmployeeService employeeService;

@PostMapping("/login")

public String login(UserEntity user, Model model) {

boolean isUserPresent = employeeService.authenticateUser(user);

if (isUserPresent) {

return "redirect:/emp/";

}

model.addAttribute("wrongpassword", "Username or Password is wrong");

return "index";

}

}

**EmployeeController.java**

package com.marticus.employee.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import com.marticus.employee.entity.EmployeeEntity;

import com.marticus.employee.repo.EmployeeRepo;

import com.marticus.employee.service.EmployeeService;

@Controller

@RequestMapping("/emp")

public class EmployeeController {

@Autowired

EmployeeService employeeService;

@GetMapping("/")

public String getEmployeeList(Model model) {

List<EmployeeEntity> employees = employeeService.getAllEmployeeList();

model.addAttribute("empList", employees);

return "employeelist";

}

@GetMapping("/add")

public String addEmployee() {

return "addEmployee";

}

@PostMapping("/add")

public String saveEmpInDb(EmployeeEntity employeeEntity) {

employeeService.saveEmpInDB(employeeEntity);

return "redirect:/emp/";

}

@GetMapping("/update/{id}")

public String updateScreen(@PathVariable int id, Model model) {

EmployeeEntity emp = employeeService.findEmployeesById(id);

model.addAttribute("id", emp.getId());

model.addAttribute("name", emp.getEmpName());

model.addAttribute("age", emp.getAge());

model.addAttribute("salary", emp.getSalary());

model.addAttribute("department", emp.getDepartment());

model.addAttribute("aadhar", emp.getAadharNumber());

return "update";

}

@PostMapping("/update")

public String update(EmployeeEntity employeeEntity) {

employeeService.update(employeeEntity);

return "redirect:/emp/";

}

@GetMapping("/delete/{id}")

public String delete(@PathVariable int id) {

employeeService.deleteEmp(id);

return "redirect:/emp/";

}

@GetMapping("/search")

public String search(EmployeeEntity employeeEntity) {

return "search";

}

@PostMapping("/search")

public String searchResults(String search, Model model) {

List<EmployeeEntity> empList = employeeService.search(search);

model.addAttribute("empList", empList);

return "searchresult";

}

}

**RestUser.java**

package com.marticus.employee.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RestController;

import com.marticus.employee.entity.UserEntity;

import com.marticus.employee.service.EmployeeService;

@RestController

@RequestMapping("/api/user")

public class RestUser {

@Autowired

EmployeeService employeeService;

@PostMapping("/create")

public ResponseEntity<String> createUser(@RequestBody UserEntity userEntity)

{

String response = employeeService.createUser(userEntity);

return ResponseEntity.status(HttpStatus.CREATED).body(response);

}

@PutMapping("/update")

public ResponseEntity<String> updateUser(@RequestBody UserEntity userEntity)

{

String response = employeeService.update(userEntity);

return ResponseEntity.ok(response);

}

@DeleteMapping("/delete")

public ResponseEntity<String> deleteUser(@RequestParam int id)

{

String response = employeeService.deleteUser(id);

return ResponseEntity.ok(response);

}

@GetMapping("/users")

public ResponseEntity<String> getUsers()

{

String response;

response = employeeService.findAll();

return ResponseEntity.ok(response);

}

@GetMapping("/id/{id}")

public ResponseEntity<String> getUserById(@PathVariable int id)

{

String response = employeeService.findUsersById(id);

return ResponseEntity.ok(response);

}

@GetMapping("/{userName}")

public ResponseEntity<String> findUsersByUserName(@PathVariable String userName)

{

String response = employeeService.findUsersByUserName(userName);

return ResponseEntity.ok(response);

}

}

**EmployeeEntity.java**

package com.marticus.employee.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

@Table(name = "employees")

public class EmployeeEntity {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "emp\_id" , nullable = true)

private int id;

@Column(name = "emp\_name")

private String empName;

@Column(name = "emp\_salary")

private int salary;

@Column(name = "emp\_age")

private int age;

@Column(name = "emp\_aadhar\_no")

private int aadharNumber;

@Column(name = "department")

private String department;

}

**RestData.java**

package com.marticus.employee.entity;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

public class RestData {

private String activity;

private String type;

private int participants;

private float price;

private String link;

private String key;

private float accessibility;

// private String name;

// private int age;

// private int count;

}

//{"name":"meelad","age":28,"count":21}

//{"activity":"Learn about the Golden Ratio",

// "type":"education",

// "participants":1,

// "price":0.1,

// "link":"https://en.wikipedia.org/wiki/Golden\_ratio",

// "key":"2095681",

// "accessibility":0.2}

**UserEntity.java**

package com.marticus.employee.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

@Table(name = "users")

public class UserEntity {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "user\_id" , nullable = true)

private int id;

@Column(name = "user\_name")

private String userName;

@Column(name = "password")

private String password;

}

**EmployeeRepo.java**

package com.marticus.employee.repo;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.marticus.employee.entity.EmployeeEntity;

@Repository

public interface EmployeeRepo extends JpaRepository<EmployeeEntity,Long> {

EmployeeEntity findEmployeesById(int id);

List<EmployeeEntity> findEmployeesByEmpNameContaining(String search);

}

**UserRepo.java**

package com.marticus.employee.repo;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.marticus.employee.entity.UserEntity;

@Repository

public interface UserRepo extends JpaRepository<UserEntity, Integer> {

UserEntity findUsersById(int id);

UserEntity findUsersByUserName(String userName);

UserEntity findUsersByUserNameAndPassword(String userName, String password);

}

**EmployeeService.java**

package com.marticus.employee.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.marticus.employee.entity.EmployeeEntity;

import com.marticus.employee.entity.UserEntity;

import com.marticus.employee.repo.EmployeeRepo;

import com.marticus.employee.repo.UserRepo;

@Service

public class EmployeeService {

@Autowired

UserRepo userRepo;

@Autowired

EmployeeRepo employeeRepo;

public String createUser(UserEntity userEntity) {

String response = " user created successfully";

UserEntity result = userRepo.save(userEntity);

if (result == null) {

response = "User cannot be created ";

return response;

}

return response;

}

public String update(UserEntity userEntity) {

String response = "User updated Successfully";

UserEntity user = userRepo.findUsersById(userEntity.getId());

if (user != null) {

userRepo.save(userEntity);

return response;

}

response = "No user found";

return response;

}

public String deleteUser(int id) {

String response = "User deleted Successfully";

UserEntity user = userRepo.findUsersById(id);

if (user != null) {

userRepo.deleteById(user.getId());

return response;

}

response = "User Not Found";

return response;

}

public String findAll() {

String temp;

String response = "";

List<UserEntity> users = userRepo.findAll();

if (users.size() > 0) {

for (UserEntity user : users) {

temp = "\n Id : " + user.getId() + "\n Name : " + user.getUserName() + "\n";

response = response + temp;

}

return response;

}

response = "There are no users";

return response;

}

public String findUsersById(int id) {

String response;

UserEntity user = userRepo.findUsersById(id);

if (user != null) {

response = " Id : " + user.getId() + "\n Name : " + user.getUserName();

return response;

}

response = "User Not Found";

return response;

}

public String findUsersByUserName(String userName) {

String response;

UserEntity user = userRepo.findUsersByUserName(userName);

if (user != null) {

response = " Id : " + user.getId() + "\n Name : " + user.getUserName();

return response;

}

response = "User Not Found";

return response;

}

public boolean authenticateUser(UserEntity user) {

UserEntity userEntity = userRepo.findUsersByUserNameAndPassword(user.getUserName(), user.getPassword());

if (userEntity != null) {

return true;

}

return false;

}

public List<EmployeeEntity> getAllEmployeeList() {

List<EmployeeEntity> employees = employeeRepo.findAll();

return employees;

}

public void saveEmpInDB(EmployeeEntity employeeEntity) {

employeeRepo.save(employeeEntity);

}

public EmployeeEntity findEmployeesById(int id) {

return employeeRepo.findEmployeesById(id);

}

public void deleteEmp(int id) {

EmployeeEntity employeeEntity = employeeRepo.findEmployeesById(id);

if (employeeEntity != null) {

employeeRepo.delete(employeeEntity);

}

}

public List<EmployeeEntity> search(String search) {

return employeeRepo.findEmployeesByEmpNameContaining(search);

}

public void update(EmployeeEntity employeeEntity) {

EmployeeEntity emp = employeeRepo.findEmployeesById(employeeEntity.getId());

if(emp!=null)

{

emp.setEmpName(employeeEntity.getEmpName());

emp.setAge(employeeEntity.getAge());

emp.setSalary(employeeEntity.getSalary());

emp.setDepartment(employeeEntity.getDepartment());

emp.setAadharNumber(employeeEntity.getAadharNumber());

employeeRepo.save(emp);

}

}

}

**application.properties**

## Spring view resolver set up

spring.mvc.view.prefix=/WEB-INF/jsp/

spring.mvc.view.suffix=.jsp

spring.datasource.url=jdbc:mysql://localhost:3306/training?useSSL=false

spring.datasource.username=root

spring.datasource.password=priya@22

spring.datasourc.Driver=com.mysql.jdbc.Driver

spring.datasource.pool-size=30

spring.jpa.show-sql=true

server.port=8000

spring.jpa.properties.hibernate.hbm2ddl.auto=update

spring.jpa.properties.hibernate.hbm2ddl.dialect=org.hibernate.dialect.MySQL8Dialect

**addEmployee.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

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

<%@ page session="false"%>

<html>

<head>

</head>

<body>

<form action="add" method="POST">

<p> Name :<input type="text" name="empName"><br>

<p> age :<input type="text" name="age"><br>

<p> salary :<input type="text" name="salary"><br>

<p> aadharNumber :<input type="text" name="aadharNumber"><br>

<p> department :<input type="text" name="department"><br>

<input

type="submit" value="Save">

</form>

</body>

</html>

**Employeelist.java**

<%@ include file="common/header.jspf"%>

<table border="1">

<tr>

<th>Id</th>

<th>Name</th>

<th>Age</th>

<th>Salary</th>

<th>Aadhar</th>

<th>Department</th>

<th>Update</th>

<th>Delete</th>

</tr>

<c:forEach items="${empList}" var="emp">

<tr>

<td>${emp.id}</td>

<td>${emp.empName}</td>

<td>${emp.age}</td>

<td>${emp.salary}</td>

<td>${emp.aadharNumber}</td>

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

<td><a href="/emp/update/${emp.id}">Update</a></td>

<td><a href="/emp/delete/${emp.id}">Delete</a></td>

</tr>

</c:forEach>

</table>

<%@ include file="common/navigation.jspf"%>

<%@ include file="common/footer.jspf"%>

**index.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

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

<%@ page session="false"%>

<html>

<head>

<title>Home</title>

</head>

<body>

<center> <h1>Employee Management Portal </h1> </center>

<h2> Enter The Details </h2>

<form action="login" method="POST">

<p> User Name :<input type="text" name="userName"><br>

<p> Password :<input type="text" name="password"><br>

<p>${wrongpassword}</p>

<input

type="submit" value="login">

</form>

</body>

</html>

**Search.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

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

<%@ page session="false"%>

<html>

<head>

<title>Search</title>

</head>

<body>

<P>The time on the server is ${serverTime}.</p>

<form action="/emp/search" method="Post">

Search : <input type="text" name="search">

<input

type="submit" value="search">

</form>

</body>

</html>

**Searchresult.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

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

<%@ page session="false"%>

<html>

<head>

<title>Home</title>

</head>

<body>

<table border="1">

<tr>

<th>Id</th>

<th>Name</th>

<th>Age</th>

<th>Salary</th>

<th>Aadhar</th>

<th>Department</th>

<th>Update</th>

<th>Delete</th>

</tr>

<c:forEach items="${empList}" var="emp">

<tr>

<td>${emp.id}</td>

<td>${emp.empName}</td>

<td>${emp.age}</td>

<td>${emp.salary}</td>

<td>${emp.aadharNumber}</td>

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

<td><a href = "/emp/update/${emp.id}">Update</a></td>

<td><a href = "/emp/delete/${emp.id}">Delete</a></td>

</tr>

</c:forEach>

</table>

</body>

</html>

**Update.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

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

<%@ page session="false"%>

<html>

<head>

<title>UpdateInfo</title>

</head>

<body>

<h1>Please Update Info</h1>

<form action="/emp/update" method="POST">

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

<p> ID :<input type="text" name="id" value = "${id}" disabled><br>

<p> Name :<input type="text" name="empName" value = "${name}"><br>

<p> age :<input type="text" name="age" value = "${age}"><br>

<p> salary :<input type="text" name="salary" value = "${salary}"><br>

<p> aadharNumber :<input type="text" name="aadharNumber" value = "${aadhar}" ><br>

<p> department :<input type="text" name="department" value ="${department}"><br>

<input

type="submit" value="Update">

</form>

</body>

</html>