Full Stack Java Project

Spring Boot Angular CRUD Application

Employee Management System

Modules

- **Add Employees**
- **List Employees**
- **Update Employees**
- **Delete Employees**

Backend / Server Side

- Create a database empdb;
  mysql>create database empdb;

- Create a Spring Starter Project “EmployeeManagementSystem” in STS

  Click on File → New → Spring Starter Project

  Name: EmployeeManagementSystem

  Type: Maven

  Java Version: 17

  Group: employeemagement

  Artifact: EmployeeManagementSystem

  Package: com.employee.management

  Click Next

  Add the following project dependencies

- Spring Web
- Spring Data JPA
- Lombok
- MySQL Driver
- Spring Boot Dev Tools

Click Finish

- Create a package “com.employee.management.entities” in src/main/java folder of EmployeeManagementSystem
- Create a Entity class “Employee” in com.employee.management.entities package

Employee.java

```
package com.employee.management.entities;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Entity

@Table (name="employees")

@NoArgsConstructor

@AllArgsConstructor

@Data

public class Employee {

    @Id

    @GeneratedValue (strategy=GenerationType.IDENTITY)

    @Column (name="emp_id")

    private int empId;
```

```
    @Column (name="emp_name")
    private String empName;
    @Column (name = "designation")
    private String designation;
    @Column (name="emp_salary")
    private double empSalary;
}
```

- Create a package “com.employee.management.repositories” in src/main/java folder of EmployeeManagementSystem
- Create an interface “EmployeeRepository” in com.employee.management.repositories package

EmployeeRepository.java

```
package com.employee.management.repositories;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.employee.management.entities.Employee;

@Repository
public interface EmployeeRepository extends JpaRepository <Employee,Integer>{
}
```

- Create a package “com.employee.management.exceptions” in src/main/java folder of EmployeeManagementSystem
- Create a class “EmployeeNotFoundException” in com.employee.management.exceptions package

EmployeeNotFoundException.java

```
package com.employee.management.exceptions;
import org.springframework.http.HttpStatus;
```

```
import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus (value = HttpStatus.NOT_FOUND)
public class EmployeeNotFoundException extends RuntimeException {


    public EmployeeNotFoundException(String msg)
    {
            super(msg);
    }
}
```

- Create a package “com.employee.management.controllers” in src/main/java folder of EmployeeManagementSystem
- Create a Rest Controller class “EmployeeController” in com.employee.management.controllers package

EmployeeController.java

```
package com.employee.management.controllers;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
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.RestController;
import com.employee.management.entities.Employee;
import
com.employee.management.exceptions.EmployeeNotFoundExceptio
n;
import
com.employee.management.repositories.EmployeeRepository;
@CrossOrigin(origins = "http://localhost:4200")
```

```
@RestController
public class EmployeeController {
    @Autowired
    private EmployeeRepository employeeRepository;
    @GetMapping ("/employees")
    public List<Employee> getAllEmployees()
    {
            return employeeRepository.findAll();
    }
    @PostMapping("/employees")
    public Employee createEmployee(@RequestBody Employee
employee)
    {
            return employeeRepository.save(employee);
    }
    @GetMapping("/employees/{empId}")
    public ResponseEntity<Employee>
getEmployeeById(@PathVariable int empId) {
            Employee employee =
employeeRepository.findById(empId).orElseThrow(() -> new
EmployeeNotFoundException("Employee with Employee Id
"+empId+" does not exist"));
            return ResponseEntity.ok(employee);
    }

    @PutMapping("/employees/{empId}")
    public ResponseEntity<Employee>
updateEmployee(@PathVariable int empId, @RequestBody Employee
employeeDetails)
    {
            Employee employee =
employeeRepository.findById(empId).orElseThrow(()
 -> new EmployeeNotFoundException("Employee with Employee Id
"+empId+" does not exist"));
            employee.setEmpName(employeeDetails.getEmpName());

    employee.setDesignation(employeeDetails.getDesignation());
```

```
            employee.setEmpSalary(employeeDetails.getEmpSalary());
            employeeRepository.save(employee);
            return ResponseEntity.ok(employee);
        }
        @DeleteMapping("/employees/{empId}")
        public ResponseEntity<Map<String,Boolean>>
    deleteEmployee(@PathVariable int empId){
            Employee employee =
    employeeRepository.findById(empId).orElseThrow(()
    -> new EmployeeNotFoundException("Employee with Employee Id
    "+empId+" does not exist"));
            employeeRepository.delete(employee);
            Map<String,Boolean> response = new
    HashMap<String,Boolean>();
            response.put("Deleted", Boolean.TRUE);
            return ResponseEntity.ok(response);
        }
    }
```

- Update application.properties file of src/main/resources folder of EmployeeManagementSystem

  application.properties

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/empdb
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLD
ialect
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

- Run “EmployeeManagementSystem” as Spring Boot App

  Right click on EmployeeManagementSystem → Run As → Spring Boot App

Front End – Angular Side

- Create an Angular Project “FrontEnd-EmployeeManagement”
  >ng new FrontEnd-EmployeeManagement --no-standalone

- Add Bootstrap into the Angular Project
  C:\ FrontEnd-EmployeeManagement>npm install bootstrap

- Add Bootstrap CDN in index.html file of src folder of Angular Project

index.html

```
<!doctype html>
<html lang="en">
<head>
 <meta charset="utf-8">
 <title>FrontendEmpmgt</title>
 <base href="/">
   <meta name="viewport" content="width=device-width, initial-scale=1">
 <link rel="icon" type="image/x-icon" href="favicon.ico">
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">
</head>
<body>
 <app-root></app-root>
</body>
</html>
```

styles.css

```
/* You can add global styles to this file, and also import other style files */
.footer {
   position : absolute;
   bottom: 0;
   width: 100%;
   height: 30px;
```

```
    background-color: green;
    text-align: center;
    color: white;
  }
  .header {
    position : relative;
    top: 0;
    width: 100%;
    height: 30px;
    background-color: green;
    text-align: center;
    color: white;
  }
```

- Create a class “Employee” in app folder
  >ng g class Employee

  Employee.ts

```
export class Employee {
    empId : number=0;
    empName : string = "";
    designation : string = "";
    empSalary : number = 0;
}
```

- Create a Service class “employee”
  >ng g s employee

  employee.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Employee } from './employee';

@Injectable({
  providedIn: 'root'
})
export class EmployeeService {
```

```
  private baseURL = "http://localhost:8080/employees";

  constructor(private httpClient : HttpClient) { }

  getEmployeeList():Observable<Employee[]>{
    return this.httpClient.get<Employee[]>(`${this.baseURL}`);
  }




  createEmployee(employee : Employee) : Observable<object>{
    return this.httpClient.post(`${this.baseURL}`,employee);
  }

  getEmployeeById(empId : number) : Observable<Employee>{
    return this.httpClient.get<Employee>(`${this.baseURL}/${empId}`);
  }

  updateEmployee(empId:number,employee:Employee):Observable<E
mployee>{
 return
this.httpClient.put<Employee>(`${this.baseURL}/${empId}`,employee)
;
  }

  deleteEmployeeById(empId:number):Observable<object>{
 return
this.httpClient.delete<Employee>(`${this.baseURL}/${empId}`);
  }
}
```

- Create a new component “list-employee”
  >ng g c list-employee

  list-employee.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Employee } from '../employee';
import { EmployeeService } from '../employee.service';
import { Router } from '@angular/router';

@Component({
  selector: 'app-list-employee',
  templateUrl: './list-employee.component.html',
  styleUrls: ['./list-employee.component.css']
})
export class ListEmployeeComponent implements OnInit {
  employees : Employee[] = [];
    constructor(private employeeService : EmployeeService,private
router:Router) { }

  ngOnInit(): void {
    this.getEmployees();
  }
  private getEmployees(){
    this.employeeService.getEmployeeList().subscribe(data => {
      this.employees = data;
    });
  }

  updateEmployee(empId : number)
  {
    this.router.navigate(['update-employee',empId]);
  }

  deleteEmployee(empId : number)
  {
    this.employeeService.deleteEmployeeById(empId).subscribe(data
=> {
      console.log(data);
      this.getEmployees();
    })
  }
```

```
}
```

list-employee.component.html

```
<table class="table table-striped">
  <thead>
    <tr>
      <th>Employee Name</th>
      <th>Designation</th>
      <th>Salary</th>
      <th>Action</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let employee of employees">
      <td>{{employee.empName}}</td>
      <td>{{employee.designation}}</td>
      <td>{{employee.empSalary}}</td>
      <td>
                <button (click)="updateEmployee(employee.empId)"
class="btn btn-success">Update</button>
                 <button (click)="deleteEmployee(employee.empId)"
class="btn btn-danger" style="margin-left:10px">Delete</button>
      </td>
    </tr>
  </tbody>
</table>
```

app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ListEmployeeComponent } from './list-employee/list-
employee.component';
import { CreateEmployeeComponent } from './create-
employee/create-employee.component';
```

```
import { UpdateEmployeeComponent } from './update-employee/update-employee.component';

const routes: Routes = [
 {path:'employees',component:ListEmployeeComponent},
 {path:'',redirectTo:'employees',pathMatch:'full'},
 {path:'create-employee',component:CreateEmployeeComponent},
 {path:'update-employee/:id',component:UpdateEmployeeComponent},

];

@NgModule({
 imports: [RouterModule.forRoot(routes)],
 exports: [RouterModule]
})
export class AppRoutingModule { }
```

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { ListEmployeeComponent } from './list-employee/list-employee.component';
import { HttpClientModule } from '@angular/common/http';
import { CreateEmployeeComponent } from './create-employee/create-employee.component';
import { FormsModule } from '@angular/forms';
import { UpdateEmployeeComponent } from './update-employee/update-employee.component';

@NgModule({
 declarations: [
   AppComponent,
   ListEmployeeComponent,
```

```
    CreateEmployeeComponent,
    UpdateEmployeeComponent,

  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

app.component.html

```
<header class="header">
  <div class="container">
    <h4>Employee Management System</h4>
  </div>
</header>
<nav class="navbar navbar-expand-sm bg-primary navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item">
      <a routerLink="employees" routerLinkActive="active" class="nav-
link">Employee List</a>
    </li>
    <li class="nav-item">
          <a routerLink="create-employee" routerLinkActive="active"
class="nav-link">Add Employee</a>
    </li>
  </ul>
</nav>
<h3 class="text-center">Welcome to Employee Management
System</h3>
<div class="container">
```

```
<router-outlet></router-outlet>
</div>
<footer class="footer">
 <div class="container">
  <span>All Rights Reserved Raj Technologies</span>
 </div>
</footer>
```

- Create a new component “create-employee”
  >ng g c create-employee

  create-employee.component.html

```
<div class="col-md-8 offset-md-3">
<h3>Create Employee</h3>
<form (ngSubmit)="onSubmit()">
  <div class="form-group">
    <label>Employee Name</label>
      <input type="text" class="form-control" placeholder="Enter Employee Name" id="empName" [(ngModel)]="employee.empName" name="empName">
  </div>
  <div class="form-group">
    <label>Designation</label>
      <input type="text" class="form-control" placeholder="Enter Employee Designation" id="designation" [(ngModel)]="employee.designation" name="designation">
  </div>
  <div class="form-group">
    <label>Employee Salary</label>
      <input type="text" class="form-control" placeholder="Enter Employee Salary" id="salary" [(ngModel)]="employee.empSalary" name="salary">
  </div>
  <div style="margin-top: 10px;">
        <button type="submit" class="btn btn-primary">Save Employee</button>
```

```
  </div>
</form>
</div>
```

create-employee.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Employee } from '../employee';
import { EmployeeService } from '../employee.service';
import { Router } from '@angular/router';

@Component({
 selector: 'app-create-employee',
 templateUrl: './create-employee.component.html',
 styleUrls: ['./create-employee.component.css']
})
export class CreateEmployeeComponent implements OnInit {
 employee : Employee = new Employee();
  constructor(private employeeService : EmployeeService, private route:Router) { }

 ngOnInit(): void {
 }
 onSubmit(){
  this.insertEmployee();
  console.log(this.employee)
 }
 insertEmployee(){
   this.employeeService.createEmployee(this.employee).subscribe(data=>{
    this.goToEmployeeList();
    console.log(data);
  })
 }

 goToEmployeeList(){
  this.route.navigate(['/employees']);
 }
```

}

- Create a new component “update-employee”
  >ng g c update-employee

update-employee.component.html

```
<div class="col-md-8 offset-md-3">
  <h3>Update Employee</h3>
  <form (ngSubmit)="onSubmit()">
    <div class="form-group">
      <label>Employee Name</label>
        <input type="text" class="form-control" placeholder="Enter Employee Name" id="empName" [(ngModel)]="employee.empName" name="empName">
    </div>
    <div class="form-group">
      <label>Designation</label>
        <input type="text" class="form-control" placeholder="Enter Employee Designation" id="designation" [(ngModel)]="employee.designation" name="designation">
    </div>
    <div class="form-group">
      <label>Employee Salary</label>
        <input type="text" class="form-control" placeholder="Enter Employee Salary" id="salary" [(ngModel)]="employee.empSalary" name="salary">
    </div>
    <div style="margin-top: 10px;">
        <button type="submit" class="btn btn-primary">Update Employee</button>
    </div>
  </form>
  </div>
```

update-employee.component.ts

```
import { Component, OnInit } from '@angular/core';
import { EmployeeService } from '../employee.service';
```

```
import { ActivatedRoute, Router } from '@angular/router';
import { Employee } from '../employee';

@Component({
  selector: 'app-update-employee',
  templateUrl: './update-employee.component.html',
  styleUrls: ['./update-employee.component.css']
})
export class UpdateEmployeeComponent implements OnInit {
  empId:number=0;
  employee:Employee = new Employee();
     constructor(private employeeService:EmployeeService, private
route:ActivatedRoute,private router:Router) { }

  ngOnInit(): void {
    this.empId=this.route.snapshot.params['id'];
     this.employeeService.getEmployeeById(this.empId).subscribe(data
=>{
      this.employee = data;
    })
  }

    onSubmit(){
      this.employeeService.updateEmployee(this.empId,this.employee).
subscribe(data=>{
        this.employee=data;
        this.goToEmployeeList();
      })
    }

    goToEmployeeList(){
      this.router.navigate(['/employees']);
    }
}
```

- Run Angular Project
  >ng serve

  Note: Before we run Angular Project, Spring Boot project should be running

- Open browser and type the following url
  http://localhost:4200