**Objectives**

## Create Model

### Country.java

package com.example.demo.model;

public class Country {

private String code;

private String name;

// Constructor

public Country(String code, String name) {

this.code = code;

this.name = name;

}

// Getters and Setters

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

Create DAO Layer

### CountryDao.java

package com.example.demo.dao;

import com.example.demo.model.Country;import org.springframework.stereotype.Repository;

import java.util.\*;

@Repositorypublic class CountryDao {

private static Map<String, Country> countries = new HashMap<>();

static {

countries.put("IN", new Country("IN", "India"));

countries.put("US", new Country("US", "United States"));

countries.put("UK", new Country("UK", "United Kingdom"));

}

public Country getCountryByCode(String code) {

return countries.get(code.toUpperCase());

}

public List<Country> getAllCountries() {

return new ArrayList<>(countries.values());

}

}

Create Service Layer

### CountryService.java

package com.example.demo.service;

import com.example.demo.dao.CountryDao;import com.example.demo.model.Country;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.stereotype.Service;

import java.util.List;

@Servicepublic class CountryService {

@Autowired

private CountryDao countryDao;

public Country getCountryByCode(String code) {

return countryDao.getCountryByCode(code);

}

public List<Country> getAllCountries() {

return countryDao.getAllCountries();

}

}

Create Controller

### CountryController.java

package com.example.demo.controller;

import com.example.demo.model.Country;import com.example.demo.service.CountryService;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestControllerpublic class CountryController {

@Autowired

private CountryService countryService;

// GET all countries

@GetMapping("/countries")

public List<Country> getAllCountries() {

return countryService.getAllCountries();

}

// GET country by code

@GetMapping("/countries/{code}")

public Country getCountry(@PathVariable String code) {

Country country = countryService.getCountryByCode(code);

if (country == null) {

throw new RuntimeException("Country Not Found");

}

return country;

}

}

Main Class

### DemoApplication.java

package com.example.demo;

import org.springframework.boot.SpringApplication;import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplicationpublic class DemoApplication {

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

}

Test in Postman

### Get All Countries

**Method:** GET

**URL:** http://localhost:8080/countries

Output: JSON array of countries[

{ "code": "IN", "name": "India" },

{ "code": "US", "name": "United States" },

{ "code": "UK", "name": "United Kingdom" }]

## **Problem Statement - Display Employee List and Edit Employee form using RESTful Web Service**  STEP 1: Create Static Employee List in XML

### Employee.java (Model)

package com.example.model;

public class Employee {

private int id;

private String name;

private String department;

// Constructor, Getters & Setters

public Employee() {}

public Employee(int id, String name, String department) {

this.id = id;

this.name = name;

this.department = department;

}

// Getters & setters

}

### employee-config.xml (Place inside resources)

<?xml version="1.0" encoding="UTF-8"?><beans xmlns="http://www.springframework.org/schema/beans"

xmlns:util="http://www.springframework.org/schema/util"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/util

http://www.springframework.org/schema/util/spring-util.xsd">

<util:list id="employeeList">

<bean class="com.example.model.Employee">

<property name="id" value="1"/>

<property name="name" value="John Doe"/>

<property name="department" value="HR"/>

</bean>

<bean class="com.example.model.Employee">

<property name="id" value="2"/>

<property name="name" value="Jane Smith"/>

<property name="department" value="Finance"/>

</bean>

</util:list></beans>

STEP 2: Create REST Service in Spring Boot

### Enable XML Bean Loading in SpringConfig.java

package com.example.config;

import org.springframework.context.annotation.\*;import org.springframework.context.support.ClassPathXmlApplicationContext;

@Configurationpublic class SpringConfig {

@Bean(name = "employeeList")

public List<?> employeeList() {

ApplicationContext context = new ClassPathXmlApplicationContext("employee-config.xml");

return (List<?>) context.getBean("employeeList");

}

}

EmployeeService.java

package com.example.service;

import com.example.model.Employee;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.stereotype.Service;

import java.util.List;

@Servicepublic class EmployeeService {

@Autowired

private List<Employee> employeeList;

public List<Employee> getAllEmployees() {

return employeeList;

}

public Employee getEmployeeById(int id) {

return employeeList.stream()

.filter(emp -> emp.getId() == id)

.findFirst()

.orElse(null);

}

}

EmployeeController.java

package com.example.controller;

import com.example.model.Employee;import com.example.service.EmployeeService;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController@CrossOrigin // Needed for Angular to accesspublic class EmployeeController {

@Autowired

private EmployeeService employeeService;

@GetMapping("/employees")

public List<Employee> getAllEmployees() {

return employeeService.getAllEmployees();

}

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

public Employee getEmployeeById(@PathVariable int id) {

return employeeService.getEmployeeById(id);

}

}

STEP 3: Angular Component Changes

## employee.service.ts

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

export interface Employee {

id: number;

name: string;

department: string;

}

@Injectable({

providedIn: 'root'

})export class EmployeeService {

private baseUrl = 'http://localhost:8080/employees';

constructor(private http: HttpClient) {}

getEmployees(): Observable<Employee[]> {

return this.http.get<Employee[]>(this.baseUrl);

}

getEmployeeById(id: number): Observable<Employee> {

return this.http.get<Employee>(`${this.baseUrl}/${id}`);

}

}

employee-list.component.ts

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

@Component({

selector: 'app-employee-list',

templateUrl: './employee-list.component.html'

})export class EmployeeListComponent implements OnInit {

employees: Employee[] = [];

constructor(private service: EmployeeService) {}

ngOnInit(): void {

this.service.getEmployees().subscribe(data => {

this.employees = data;

});

}

}

employee-list.component.html

<table>

<tr \*ngFor="let emp of employees">

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

<td>{{ emp.name }}</td>

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

<td><button (click)="edit(emp.id)">Edit</button></td>

</tr></table>

edit() function in component (navigate to edit form)

edit(id: number) {

// Navigate to /edit/{id}

}

edit-employee.component.ts

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

@Component({

selector: 'app-edit-employee',

templateUrl: './edit-employee.component.html'

})export class EditEmployeeComponent implements OnInit {

employee!: Employee;

constructor(private route: ActivatedRoute, private service: EmployeeService) {}

ngOnInit(): void {

const id = Number(this.route.snapshot.paramMap.get('id'));

this.service.getEmployeeById(id).subscribe(data => {

this.employee = data;

});

}

}

edit-employee.component.html

<div \*ngIf="employee">

<form>

<label>ID:</label> <input type="text" [value]="employee.id" readonly><br>

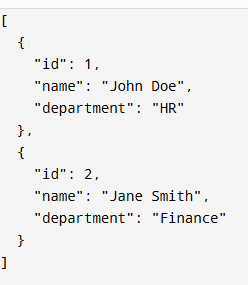
<label>Name:</label> <input type="text" [(ngModel)]="employee.name"><br>

<label>Department:</label> <input type="text" [(ngModel)]="employee.department"><br>

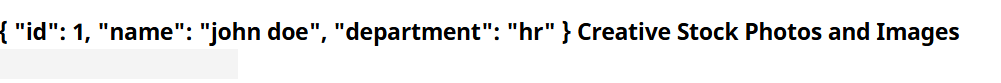
<button type="submit">Save</button>

</form></div>

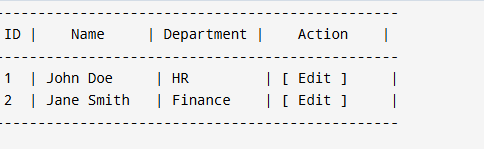
**Spring Boot REST API Output ( browser)**



**URL: <http://localhost:8080/employees/1>**



****Angular Output****



**Create static employee list data using spring xml configuration**

## 1. Define Model Classes

### Department.java

package com.example.model;s

public class Department {

private int id;

private String name;

public Department() {}

public Department(int id, String name) {

this.id = id;

this.name = name;

}

// Getters and Setters

}

Skill.java

package com.example.model;

public class Skill {

private int id;

private String name;

public Skill() {}

public Skill(int id, String name) {

this.id = id;

this.name = name;

}

// Getters and Setters

}

Employee.java

package com.example.model;

import java.util.List;

public class Employee {

private int id;

private String name;

private double salary;

private boolean permanent;

private Department department;

private List<Skill> skillList;

public Employee() {}

public Employee(int id, String name, double salary, boolean permanent, Department department, List<Skill> skillList) {

this.id = id;

this.name = name;

this.salary = salary;

this.permanent = permanent;

this.department = department;

this.skillList = skillList;

}

// Getters and Setters

}

2. Define XML Configuration (employee.xml)

Place this file in src/main/resources.

<?xml version="1.0" encoding="UTF-8"?><beans xmlns="http://www.springframework.org/schema/beans"

xmlns:util="http://www.springframework.org/schema/util"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/util

http://www.springframework.org/schema/util/spring-util.xsd">

<!-- Departments -->

<bean id="hrDept" class="com.example.model.Department">

<property name="id" value="1"/>

<property name="name" value="HR"/>

</bean>

<bean id="financeDept" class="com.example.model.Department">

<property name="id" value="2"/>

<property name="name" value="Finance"/>

</bean>

<!-- Reusable Skills -->

<bean id="javaSkill" class="com.example.model.Skill">

<property name="id" value="1"/>

<property name="name" value="Java"/>

</bean>

<bean id="angularSkill" class="com.example.model.Skill">

<property name="id" value="2"/>

<property name="name" value="Angular"/>

</bean>

<bean id="sqlSkill" class="com.example.model.Skill">

<property name="id" value="3"/>

<property name="name" value="SQL"/>

</bean>

<!-- Skill Lists -->

<util:list id="skillList1">

<ref bean="javaSkill"/>

<ref bean="angularSkill"/>

</util:list>

<util:list id="skillList2">

<ref bean="javaSkill"/>

<ref bean="sqlSkill"/>

</util:list>

<!-- Employees -->

<bean id="employee1" class="com.example.model.Employee">

<property name="id" value="1"/>

<property name="name" value="John Doe"/>

<property name="salary" value="50000"/>

<property name="permanent" value="true"/>

<property name="department" ref="hrDept"/>

<property name="skillList" ref="skillList1"/>

</bean>

<bean id="employee2" class="com.example.model.Employee">

<property name="id" value="2"/>

<property name="name" value="Jane Smith"/>

<property name="salary" value="60000"/>

<property name="permanent" value="true"/>

<property name="department" ref="financeDept"/>

<property name="skillList" ref="skillList2"/>

</bean>

<bean id="employee3" class="com.example.model.Employee">

<property name="id" value="3"/>

<property name="name" value="Robert Fox"/>

<property name="salary" value="55000"/>

<property name="permanent" value="false"/>

<property name="department" ref="hrDept"/>

<property name="skillList" ref="skillList1"/>

</bean>

<bean id="employee4" class="com.example.model.Employee">

<property name="id" value="4"/>

<property name="name" value="Emily Davis"/>

<property name="salary" value="62000"/>

<property name="permanent" value="true"/>

<property name="department" ref="financeDept"/>

<property name="skillList" ref="skillList2"/>

</bean>

<!-- Employee List -->

<util:list id="employeeList">

<ref bean="employee1"/>

<ref bean="employee2"/>

<ref bean="employee3"/>

<ref bean="employee4"/>

</util:list>

</beans>

3. Load XML in Spring and Configure EmployeeDao

### EmployeeDao.java

package com.example.dao;

import com.example.model.Employee;import org.springframework.context.support.ClassPathXmlApplicationContext;import org.springframework.stereotype.Repository;

import java.util.ArrayList;import java.util.List;

@Repositorypublic class EmployeeDao {

private static ArrayList<Employee> EMPLOYEE\_LIST;

public EmployeeDao() {

ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("employee.xml");

List<Employee> list = (List<Employee>) context.getBean("employeeList");

EMPLOYEE\_LIST = new ArrayList<>(list);

context.close();

}

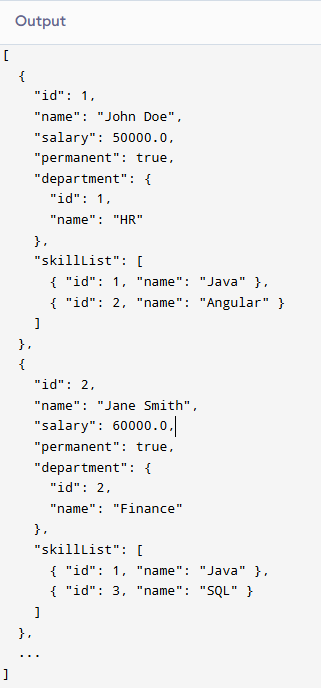
public List<Employee> getAllEmployees() {

return EMPLOYEE\_LIST;

}

}

**OUTPUT:**



**Create REST service to gets all employees**

### 1. ****EmployeeService.java****

Update the class to:

Annotate with @Service

Define getAllEmployees() method

Annotate the method with @Transactional

package com.example.service;

import com.example.dao.EmployeeDao;import com.example.model.Employee;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.stereotype.Service;import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Servicepublic class EmployeeService {

@Autowired

private EmployeeDao employeeDao;

@Transactional

public List<Employee> getAllEmployees() {

return employeeDao.getAllEmployees();

}

}

2. **EmployeeController.java**

Update this class to:

Add @Autowired for EmployeeService

Define a @GetMapping("/employees") method that returns the list

package com.example.controller;

import com.example.model.Employee;import com.example.service.EmployeeService;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController@CrossOriginpublic class EmployeeController {

@Autowired

private EmployeeService employeeService;

@GetMapping("/employees")

public List<Employee> getAllEmployees() {

return employeeService.getAllEmployees();

}

}

**OUTPUT:**



**Create REST service for department**

### 1. ****Update**** department-config.xml ****(or reuse existing**** employee.xml****)****

If you already have departments in employee.xml, you can add this section:

<!-- Add in employee.xml or a separate department-config.xml --><util:list id="departmentList">

<ref bean="hrDept"/>

<ref bean="financeDept"/></util:list>

2. **DepartmentDao.java**

package com.example.dao;

import com.example.model.Department;import org.springframework.context.support.ClassPathXmlApplicationContext;import org.springframework.stereotype.Repository;

import java.util.ArrayList;import java.util.List;

@Repositorypublic class DepartmentDao {

private static List<Department> DEPARTMENT\_LIST;

public DepartmentDao() {

ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("employee.xml");

List<Department> deptList = (List<Department>) context.getBean("departmentList");

DEPARTMENT\_LIST = new ArrayList<>(deptList);

context.close();

System.out.println("DepartmentDao loaded from XML.");

}

public List<Department> getAllDepartments() {

System.out.println("getAllDepartments() called from DepartmentDao");

return DEPARTMENT\_LIST;

}

}

3. **DepartmentService.java**

package com.example.service;

import com.example.dao.DepartmentDao;import com.example.model.Department;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.stereotype.Service;

import java.util.List;

@Servicepublic class DepartmentService {

@Autowired

private DepartmentDao departmentDao;

public List<Department> getAllDepartments() {

System.out.println("getAllDepartments() called from DepartmentService");

return departmentDao.getAllDepartments();

}

}

4. **DepartmentController.java**

package com.example.controller;

import com.example.model.Department;import com.example.service.DepartmentService;import org.springframework.beans.factory.annotation.Autowired;import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController@CrossOriginpublic class DepartmentController {

@Autowired

private DepartmentService departmentService;

@GetMapping("/departments")

public List<Department> getAllDepartments() {

System.out.println("GET /departments called in DepartmentController");

return departmentService.getAllDepartments();

}

}

**OUTPUT:**

