**Cognizant Deep Nurture 4.0 Hands-on Exercise - Week 4**

Problem Statement - Display Employee List and EditEmployee form using RESTful Web ServiceIn the previous angular module, we developed a screen that lists employeesand it was populated with hard coded values. Now this angular application hasbe changed to get the data from RESTful Web Service developed in Spring.The following are the high level activities that needs to be done to accomplishthis:Create static employee list data using spring xml configurationCreate a REST Service that reads data from xml configuration and returns itMake changes in angular component to consume the created REST ServiceOnce above activities are completed, clicking on the Edit button against eachemployee should display Edit Employee form with values retrieved fromRESTful Web Service. This will also involve activities similar to the onespecified above.NOTE: There is no specific activity as part of this hands on, refer the nexthands ons that covers above three activities in detail.    
   
   
Create static employee list data using spring xmlconfigurationFollow steps below to accomplish this activity:Incorporate the following in employee.xml:oCreate one or two more departmentsoCreate four more instances of Employee. (use employee sample data from angular)oReuse existing skills instead of creating new onesoInclude all four employee instances in an ArrayList.In EmployeeDao, incorporate the following:oCreate static variable with name EMPLOYEE\_LIST of type ArrayList<Employee>oInclude constructor that reads employee list from xml config and set the EMPLOYEE\_LISToCreate method getAllEmployees() that returns the EMPLOYEE\_LIST    
   
Create REST service to gets all employeesFollow steps below to accomplish this activity:In EmployeeService, incorporate the following:oChange the annotation for this class from @Component to @ServiceoCreate method getAllEmployees() that invokes employeeDao.getAllEmployees() and return the employee listoDefine @Transactional annotation for this method.In EmployeeController, incorporate the following:oInclude a new get method with name getAllEmployees() that returns the employee listoMark this method as GetMapping annotation with the URL as '/employees'oWithin this method invoke employeeService.getAllEmployees() and return the same.Test the service using postman.    
   
Create REST service for departmentCreate a new service to get all the departments.Follow steps below to achieve this:Create a new REST Service, define below list of classes and respective methods:oDepartmentControllergetAllDepartments() with URL "/departments", this method will return array of departmentsoDepartmentServicegetAllDepartments()oDepartmentDaogetAllDepartments() - Create a static variable DEPARTMENT\_LIST, this should be populated from spring xml configurationTest the service using postman.Also verify if department REST service is called by looking into the logs.    
   
   
   
   
Step 1: employee.xml (Spring XML Config)

**Path**: src/main/resources/employee.xml

xml

CopyEdit

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

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

xsi:schemaLocation="http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd">

<!-- Departments -->

<bean id="deptHR" class="com.cognizant.model.Department">

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

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

</bean>

<bean id="deptIT" class="com.cognizant.model.Department">

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

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

</bean>

<!-- Employees -->

<bean id="empList" class="java.util.ArrayList">

<constructor-arg>

<list>

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

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

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

<property name="salary" value="25000"/>

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

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

<property name="skills">

<list>

<value>Java</value>

<value>Spring</value>

</list>

</property>

</bean>

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

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

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

<property name="salary" value="22000"/>

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

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

<property name="skills">

<list>

<value>Excel</value>

<value>Communication</value>

</list>

</property>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

Employee.java

java

CopyEdit

public class Employee {

private int id;

private String name;

private double salary;

private boolean permanent;

private Department department;

private List<String> skills;

// Getters, Setters, toString

}

Department.java

java

CopyEdit

public class Department {

private int id;

private String name;

// Getters, Setters, toString

}

EmployeeDao.java

java

CopyEdit

@Repository

public class EmployeeDao {

private static List<Employee> EMPLOYEE\_LIST;

@Autowired

public EmployeeDao(ApplicationContext context) {

EMPLOYEE\_LIST = (List<Employee>) context.getBean("empList");

}

public List<Employee> getAllEmployees() {

return EMPLOYEE\_LIST;

}

}

EmployeeService.java

java

CopyEdit

@Service

public class EmployeeService {

@Autowired

private EmployeeDao employeeDao;

@Transactional

public List<Employee> getAllEmployees() {

return employeeDao.getAllEmployees();

}

}

EmployeeController.java

java

CopyEdit

@RestController

public class EmployeeController {

@Autowired

private EmployeeService employeeService;

@GetMapping("/employees")

public List<Employee> getAllEmployees() {

return employeeService.getAllEmployees();

}

}

DepartmentDao.java

java

CopyEdit

@Repository

public class DepartmentDao {

private static List<Department> DEPARTMENT\_LIST;

@Autowired

public DepartmentDao(ApplicationContext context) {

DEPARTMENT\_LIST = Arrays.asList(

new Department(1, "HR"),

new Department(2, "IT")

);

}

public List<Department> getAllDepartments() {

return DEPARTMENT\_LIST;

}

}

DepartmentService.java

java

CopyEdit

@Service

public class DepartmentService {

@Autowired

private DepartmentDao departmentDao;

public List<Department> getAllDepartments() {

return departmentDao.getAllDepartments();

}

}

DepartmentController.java

java

CopyEdit

@RestController

public class DepartmentController {

@Autowired

private DepartmentService departmentService;

@GetMapping("/departments")

public List<Department> getAllDepartments() {

return departmentService.getAllDepartments();

}

}

application.properties

properties

CopyEdit

server.port=8083

How to Test (Postman / Angular)

* **GET** http://localhost:8083/employees → returns JSON list of employees
* **GET** http://localhost:8083/departments → returns department list
* In Angular, you consume these with HttpClient.

Sample Output (JSON from /employees)

json

CopyEdit

[

{

"id": 1,

"name": "Alice",

"salary": 25000.0,

"permanent": true,

"department": { "id": 2, "name": "IT" },

"skills": ["Java", "Spring"]

},

{

"id": 2,

"name": "Bob",

"salary": 22000.0,

"permanent": false,

"department": { "id": 1, "name": "HR" },

"skills": ["Excel", "Communication"]

}

]