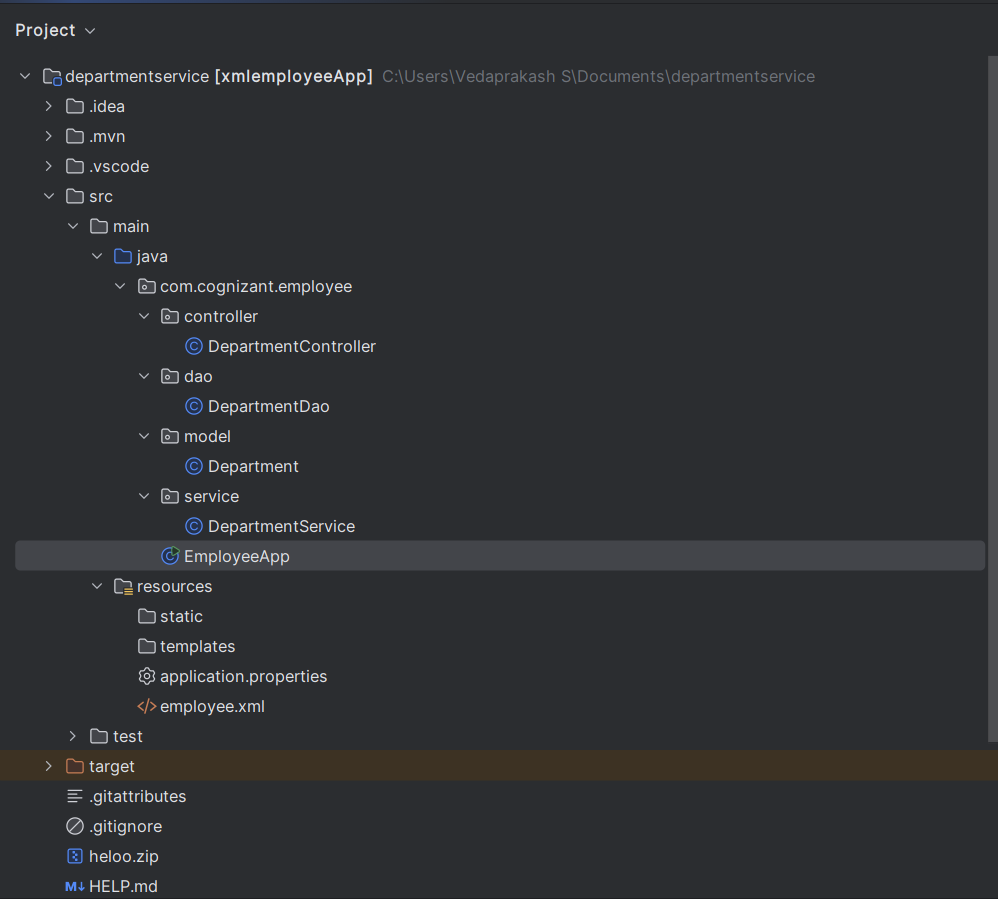
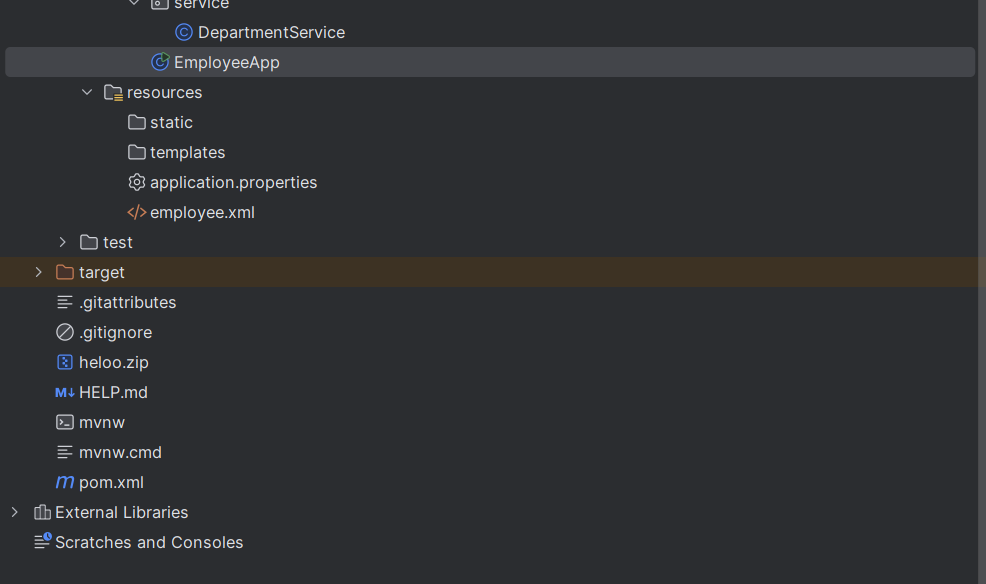
**Module 7 - Spring REST using Spring Boot 3**

**Create REST service for department**

**Structure:**





**DepartmentController.java:**

package com.cognizant.employee.controller;  
  
import com.cognizant.employee.model.Department;  
import com.cognizant.employee.service.DepartmentService;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/api")  
public class DepartmentController {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(DepartmentController.class);  
  
 private final DepartmentService departmentService;  
  
 public DepartmentController() {  
 this.departmentService = new DepartmentService();  
 }  
  
 @GetMapping("/departments")  
 public List<Department> getAllDepartments() {  
 *LOGGER*.info("⚙️ /departments endpoint called");  
 return departmentService.getAllDepartments();  
 }  
}

**DepartmentDao.java:**

package com.cognizant.employee.dao;  
  
import com.cognizant.employee.model.Department;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class DepartmentDao {  
  
 private static final String *CONFIG\_LOCATION* = "employee.xml";  
 public static List<Department> *DEPARTMENT\_LIST*;  
  
 public DepartmentDao() {  
 ApplicationContext context = new ClassPathXmlApplicationContext(*CONFIG\_LOCATION*);  
 *DEPARTMENT\_LIST* = (ArrayList<Department>) context.getBean("departmentList");  
 }  
  
 public List<Department> getAllDepartments() {  
 return *DEPARTMENT\_LIST*;  
 }  
}

**Department.java:**

package com.cognizant.employee.model;  
  
public class Department {  
 private int id;  
 private String name;  
  
 // Getters and setters  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
  
 public String getName() { return name; }  
 public void setName(String name) { this.name = name; }  
}

**DepartmentService.java:**

package com.cognizant.employee.service;  
  
import com.cognizant.employee.dao.DepartmentDao;  
import com.cognizant.employee.model.Department;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
  
import java.util.List;  
  
@Service  
public class DepartmentService {  
  
 private final DepartmentDao departmentDao;  
  
 public DepartmentService() {  
 this.departmentDao = new DepartmentDao();  
 }  
  
 @Transactional  
 public List<Department> getAllDepartments() {  
 return departmentDao.getAllDepartments();  
 }  
}

**EmployeeApp.java:**

package com.cognizant.employee;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication   
public class EmployeeApp {  
 public static void main(String[] args) {  
 SpringApplication.*run*(EmployeeApp.class, args);  
 }  
}

**application.properties:**server.port=8083

**employee.xml:**

<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="dept1" class="com.cognizant.employee.model.Department">  
 <property name="id" value="101"/>  
 <property name="name" value="IT"/>  
 </bean>  
  
 <bean id="dept2" class="com.cognizant.employee.model.Department">  
 <property name="id" value="102"/>  
 <property name="name" value="HR"/>  
 </bean>  
  
 <!-- Department list -->  
 <bean id="departmentList" class="java.util.ArrayList">  
 <constructor-arg>  
 <list>  
 <ref bean="dept1"/>  
 <ref bean="dept2"/>  
 </list>  
 </constructor-arg>  
 </bean>  
</beans>

**pom.xml:**

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  
 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.1.5</version> <!-- You can change to latest stable -->  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
  
 <groupId>com.cognizant</groupId>  
 <artifactId>xmlemployeeApp</artifactId>  
 <version>1.0.0</version>  
 <packaging>jar</packaging>  
  
 <name>xmlemployeeApp</name>  
 <description>Spring REST App loading employee list from XML</description>  
  
 <properties>  
 <java.version>17</java.version>  
 <spring.boot.version>3.1.5</spring.boot.version> <!-- use compatible version -->  
 </properties>  
  
 <dependencies>  
 <!-- Spring Boot Web Starter -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <!-- Spring TX for @Transactional -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-tx</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-tx</artifactId>  
 </dependency>  
  
 <!-- Spring Context (needed for XML-based context loading) -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 </dependency>  
  
 <!-- Optional: Faster XML for JSON support (Spring Boot already includes it) -->  
 <dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 </dependency>  
  
 <!-- Optional: Lombok for cleaner code -->  
 <dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <optional>true</optional>  
 </dependency>  
  
 <!-- Optional: For devtools and hot reload -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
  
 <!-- Testing (optional) -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <!-- Spring Boot Maven Plugin -->  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 <version>${spring.boot.version}</version>  
 </plugin>  
  
 <!-- Compiler Plugin -->  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.11.0</version>  
 <configuration>  
 <source>${java.version}</source>  
 <target>${java.version}</target>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**Output:**

