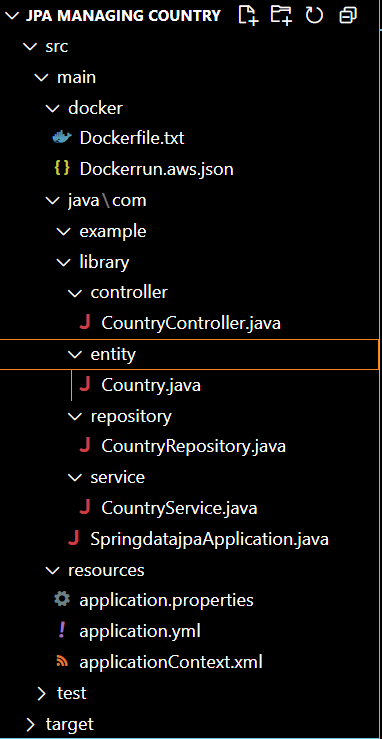
**Implement services for managing Country**  
An application requires for features to be implemented with regards to country. These features needs to be supported by implementing them as service using Spring Data JPA.

* Find a country based on country code
* Add new country
* Update country
* Delete country
* Find list of countries matching a partial country name

**STRUCTURE:**



**CountryController.java**

package com.example.library.controller;

import com.example.library.entity.Country;

import com.example.library.service.CountryService;

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

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/countries")

public class CountryController {

    @Autowired

    private CountryService countryService;

    // Get all countries

    @GetMapping

    public List<Country> getAllCountries() {

        return countryService.getAllCountries();

    }

    // Add a new country

    @PostMapping

    public Country addCountry(@RequestBody Country country) {

        return countryService.addCountry(country);

    }

}

**Country.java**

package com.example.library.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class Country {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long id;

    private String name;

    private String capital;

    public Country() {

    }

    public Country(String name, String capital) {

        this.name = name;

        this.capital = capital;

    }

    public Long getId() {

        return id;

    }

    public String getName() {

        return name;

    }

    public String getCapital() {

        return capital;

    }

    public void setId(Long id) {

        this.id = id;

    }

    public void setName(String name) {

        this.name = name;

    }

    public void setCapital(String capital) {

        this.capital = capital;

    }

}

**CountryRepository.java**

package com.example.library.repository;

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

import com.example.library.entity.Country;

public interface CountryRepository extends JpaRepository<Country, Long> {

}

**CountryService.java**

package com.example.library.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.example.library.entity.Country;

import com.example.library.repository.CountryRepository;

@Service

public class CountryService {

    @Autowired

    private CountryRepository repository;

    public List<Country> getAllCountries() {

        return repository.findAll();

    }

    public Country addCountry(Country country) {

        return repository.save(country);

    }

}

**SpringdatajpaApplication.java**

package com.example.library;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringdatajpaApplication {

    public static void main(String[] args) {

        SpringApplication.run(SpringdatajpaApplication.class, args);

    }

}

**Application.properties**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

# Automatically create tables from entities

spring.jpa.hibernate.ddl-auto=update

server.port=8082

**applicationContext.xml**

<?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

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

    <!-- BookRepository bean -->

    <bean id="bookRepository" class="com.example.library.repository.BookRepository" />

    <!-- BookService bean with constructor and setter injection -->

    <bean id="bookService" class="com.example.library.service.BookService">

        <!-- Constructor injection -->

        <constructor-arg value="Library Book Service" />

        <!-- Setter injection -->

        <property name="bookRepository" ref="bookRepository" />

    </bean>

</beans>

**Pom.xml**

<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>

    <groupId>com.example</groupId>

    <artifactId>springdatajpa</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <packaging>jar</packaging>

    <name>SpringDataJpa</name>

    <description>Spring Boot project for managing Country with H2 and JPA</description>

    <parent>

        <groupId>org.springframework.boot</groupId>

        <artifactId>spring-boot-starter-parent</artifactId>

        <version>3.2.0</version> <!-- Adjust to match your Spring Boot version -->

        <relativePath/> <!-- lookup parent from repository -->

    </parent>

    <properties>

        <java.version>17</java.version> <!-- Use 17 if you're using Java 17 -->

    </properties>

    <dependencies>

        <!-- Spring Boot Web Starter -->

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-web</artifactId>

        </dependency>

        <!-- Spring Boot JPA Starter -->

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-data-jpa</artifactId>

        </dependency>

        <!-- H2 Database -->

        <dependency>

            <groupId>com.h2database</groupId>

            <artifactId>h2</artifactId>

            <scope>runtime</scope>

        </dependency>

        <!-- Optional: DevTools for hot reload -->

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-devtools</artifactId>

            <scope>runtime</scope>

        </dependency>

        <!-- Optional: Testing -->

        <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>3.2.0</version>

            </plugin>

        </plugins>

    </build>

</project>

**OUTPUT:**

