**Find a country based on country code** 

**STRUCTURE:**



**CountryController.java**

package com.example.library.controller;

import java.util.List;

import java.util.Optional;

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

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.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.example.library.entity.Country;

import com.example.library.service.CountryService;

@RestController

@RequestMapping("/countries")

public class CountryController {

    @Autowired

    private CountryService service;

    @GetMapping

    public List<Country> getAllCountries() {

        return service.getAllCountries();

    }

    @PostMapping

    public Country addCountry(@RequestBody Country country) {

        return service.addCountry(country);

    }

    @GetMapping("/code/{code}")

    public Optional<Country> getCountryByCode(@PathVariable String code) {

        return service.getCountryByCode(code);

    }

}

**Country.java**

package com.example.library.entity;

import jakarta.persistence.Column;

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;

    @Column(unique = true, nullable = false)

    private String code;

    public Country() {}

    public Country(String name, String code) {

        this.name = name;

        this.code = code;

    }

    public Long getId() { return id; }

    public void setId(Long id) { this.id = id; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getCode() { return code; }

    public void setCode(String code) { this.code = code; }

}

**CountryRepository.java**

package com.example.library.repository;

import java.util.Optional;

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

import com.example.library.entity.Country;

public interface CountryRepository extends JpaRepository<Country, Long> {

    Optional<Country> findByCode(String code);

}

**CountryService.java**

package com.example.library.service;

import java.util.List;

import java.util.Optional;

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);

    }

    public Optional<Country> getCountryByCode(String code) {

        return repository.findByCode(code);

    }

}

**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

**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>

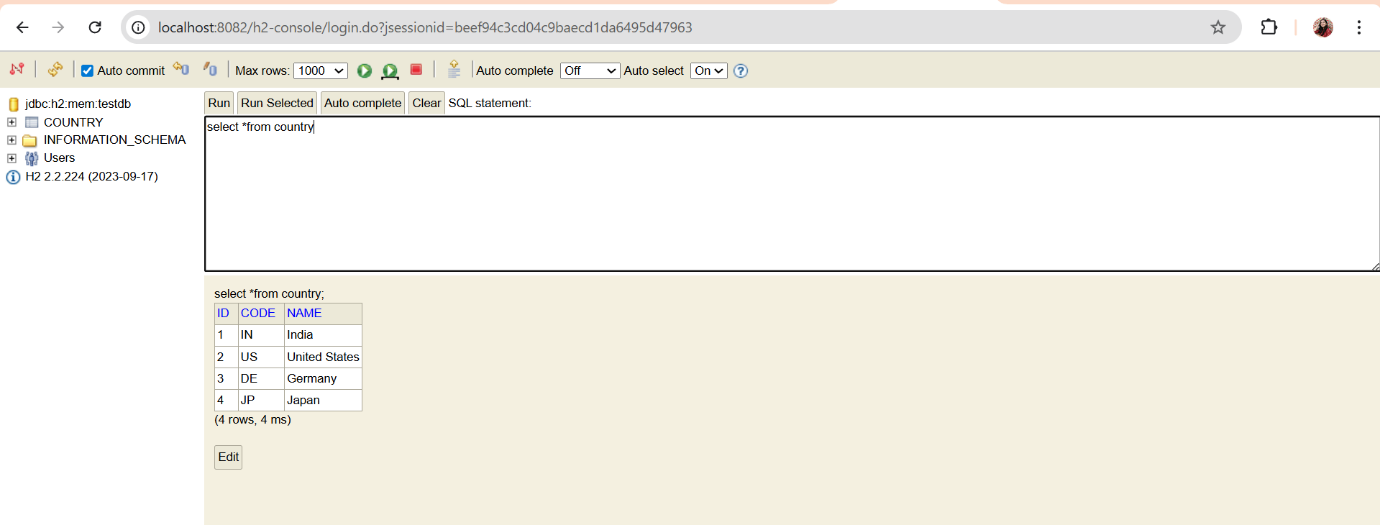
                <version>3.2.0</version>

            </plugin>

        </plugins>

    </build>

</project>

**OUTPUT:**

