**SPRING DATA JPA HANDSON**

**HANDS ON 1: SPRING DATA JPA - QUICK EXAMPLE**   
  
**Software Pre-requisites**

* MySQL Server 8.0
* MySQL Workbench 8
* Eclipse IDE for Enterprise Java Developers 2019-03 R
* Maven 3.6.2

**Code:**

**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.jpa.hibernate.ddl-auto=update

**schema.sql**

**CREATE** **TABLE** department (

    id **INT** **PRIMARY KEY**,

**name** **VARCHAR**(100)

);

**Department.java**

**package** **com.example**;

**import** **javax.persistence.Entity**;

**import** **javax.persistence.Id**;

@**Entity**

**public** **class** Department {

    @**Id**

**private** **int** id;

**private** **String** name;

*// Getters & 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;

    }

*// toString*

    @**Override**

**public** **String** toString() {

**return** "Department [id=" **+** id **+** ", name=" **+** name **+** "]";

    }

}

**DepartmentRepository.java**

**package** **com.example**;

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

**public** **interface** DepartmentRepository **extends** JpaRepository<**Department**, **Integer**> {

}

**OrmLearnApplication.java**

**package** **com.example**;

**import** **java.util.List**;

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

**import** **org.springframework.boot.CommandLineRunner**;

**import** **org.springframework.boot.SpringApplication**;

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

@**SpringBootApplication**

**public** **class** OrmLearnApplication **implements** CommandLineRunner {

    @**Autowired**

**private** **DepartmentRepository** departmentRepository;

**public** **static** **void** main(**String**[] args) {

        SpringApplication.run(OrmLearnApplication.class, args);

    }

    @**Override**

**public** **void** run(**String**... args) {

**Department** dept **=** **new** Department();

        dept.setId(1);

        dept.setName("HR");

        departmentRepository.save(dept);

**List**<**Department**> departments **=** departmentRepository.findAll();

        departments.forEach(System.out**::**println);

    }

}

**App.java  
import** **org.springframework.boot.SpringApplication**;

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

**import** **org.springframework.boot.autoconfigure.domain.EntityScan**;

**import** **org.springframework.boot.CommandLineRunner**;

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

**import** **java.util.List**;

@**SpringBootApplication**

@**EntityScan**(basePackages **=** ".")

**public** **class** App **implements** CommandLineRunner {

    @**Autowired**

**private** **DepartmentRepository** departmentRepository;

**public** **static** **void** main(**String**[] args) {

        SpringApplication.run(App.class, args);

    }

    @**Override**

**public** **void** run(**String**... args) {

**Department** dept **=** **new** Department();

        dept.setId(1);

        dept.setName("HR");

        departmentRepository.save(dept);

**List**<**Department**> departments **=** departmentRepository.findAll();

        departments.forEach(System.out**::**println);

    }

}

**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>orm-learn</artifactId>

  <version>1.0.0</version>

  <parent>

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

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

    <version>2.7.18</version>

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

  </parent>

  <properties>

    <java.version>11</java.version>

  </properties>

  <dependencies>

    <dependency>

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

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

    </dependency>

    <dependency>

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

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

    </dependency>

    <dependency>

      <groupId>com.h2database</groupId>

      <artifactId>h2</artifactId>

      <scope>runtime</scope>

    </dependency>

  </dependencies>

  <build>

    <plugins>

      <plugin>

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

        <artifactId>spring-boot-maven-plugin</artifactId>

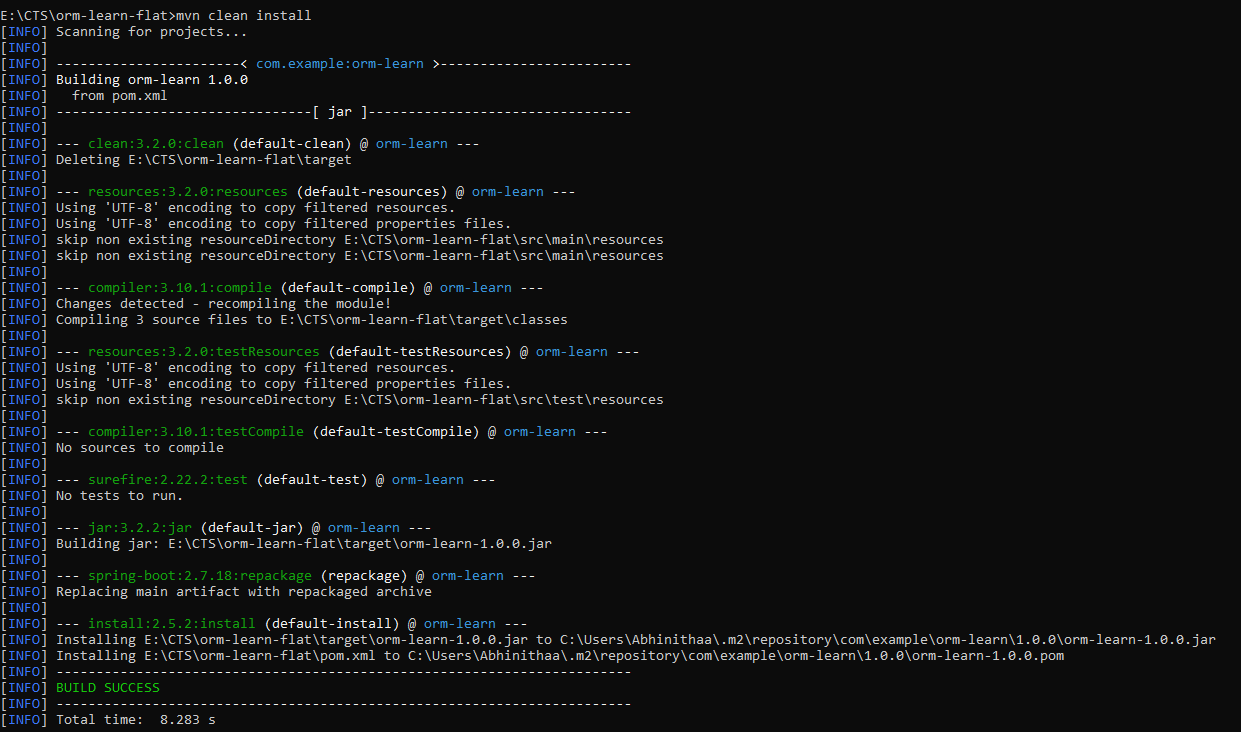
      </plugin>

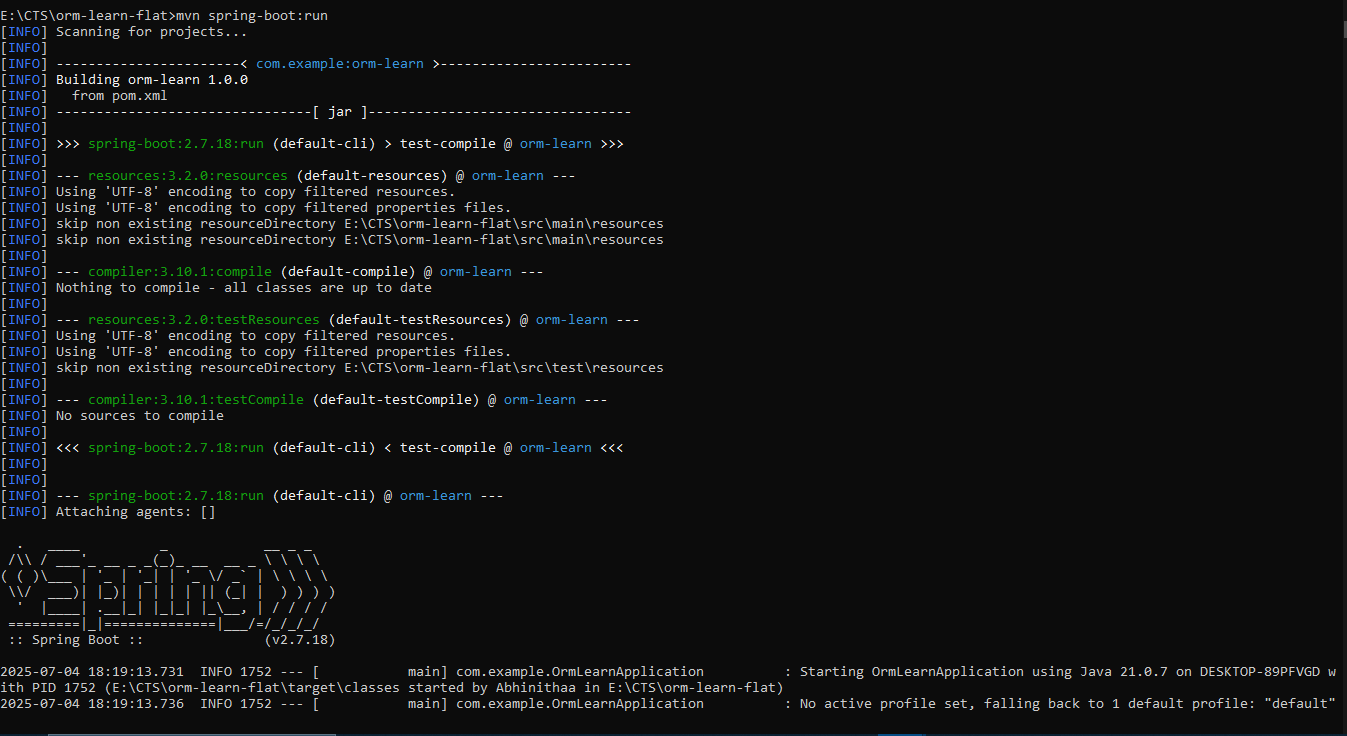
    </plugins>

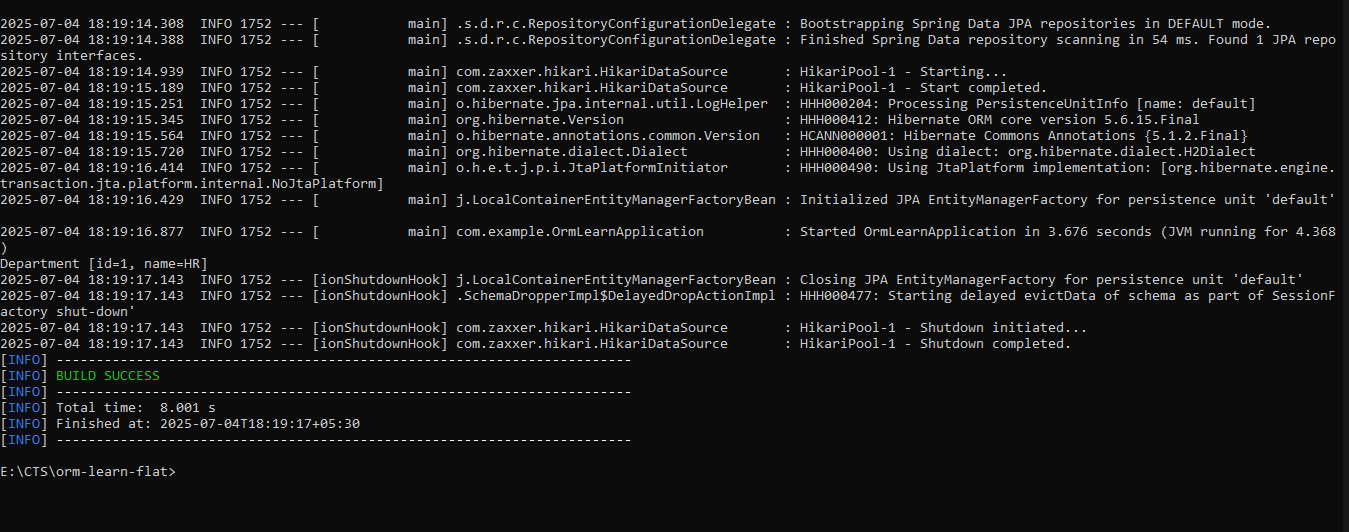
  </build>

</project>

**Output:**

****

****

****

**HANDSON 02: DIFFERENCE BETWEEN JPA, HIBERNATE AND SPRING DATA JPA**   
  
Java Persistence API (JPA)

* JSR 338 Specification for persisting, reading and managing data from Java objects
* Does not contain concrete implementation of the specification
* Hibernate is one of the implementation of JPA

Hibernate

* ORM Tool that implements JPA

Spring Data JPA

* Does not have JPA implementation, but reduces boiler plate code
* This is another level of abstraction over JPA implementation provider like Hibernate
* Manages transactions

**Code:**

**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>orm-learn</artifactId>

    <version>1.0.0</version>

    <packaging>jar</packaging>

    <properties>

        <java.version>17</java.version>

    </properties>

<dependencies>

    <dependency>

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

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

        <version>2.7.18</version>

    </dependency>

    <dependency>

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

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

        <version>2.7.18</version>

    </dependency>

    <dependency>

        <groupId>com.h2database</groupId>

        <artifactId>h2</artifactId>

        <version>2.1.214</version>

        <scope>runtime</scope>

    </dependency>

*<!-- ✅ Add these for Java 9+ -->*

    <dependency>

        <groupId>javax.xml.bind</groupId>

        <artifactId>jaxb-api</artifactId>

        <version>2.3.1</version>

    </dependency>

    <dependency>

        <groupId>org.glassfish.jaxb</groupId>

        <artifactId>jaxb-runtime</artifactId>

        <version>2.3.1</version>

    </dependency>

</dependencies>

<build>

  <plugins>

    <plugin>

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

      <artifactId>spring-boot-maven-plugin</artifactId>

      <version>2.7.18</version>

      <configuration>

        <mainClass>com.example.OrmLearnApplication</mainClass>  *<!-- use full package here -->*

      </configuration>

    </plugin>

  </plugins>

</build>

</project>

**Department.java**

**package** **com.example**;

**import** **javax.persistence.Entity**;

**import** **javax.persistence.Id**;

@**Entity**

**public** **class** Department {

    @**Id**

**private** **int** id;

**private** **String** name;

*// Getters & 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;

    }

*// toString*

    @**Override**

**public** **String** toString() {

**return** "Department [id=" **+** id **+** ", name=" **+** name **+** "]";

    }

}

**DepartmentRepository.java**

**package** **com.example**;

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

**public** **interface** DepartmentRepository **extends** JpaRepository<**Department**, **Integer**> {

}

**OrmLearningApplication.java**

**package** **com.example**;

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

**import** **org.springframework.boot.CommandLineRunner**;

**import** **org.springframework.boot.SpringApplication**;

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

@**SpringBootApplication**

**public** **class** OrmLearnApplication **implements** CommandLineRunner {

    @**Autowired**

**private** **StudentRepository** studentRepository;

**public** **static** **void** main(**String**[] args) {

        SpringApplication.run(OrmLearnApplication.class, args);

    }

    @**Override**

**public** **void** run(**String**... args) **throws** **Exception** {

        studentRepository.save(**new** Student(1, "Alice"));

        studentRepository.save(**new** Student(2, "Bob"));

        System.out.println("All Students:");

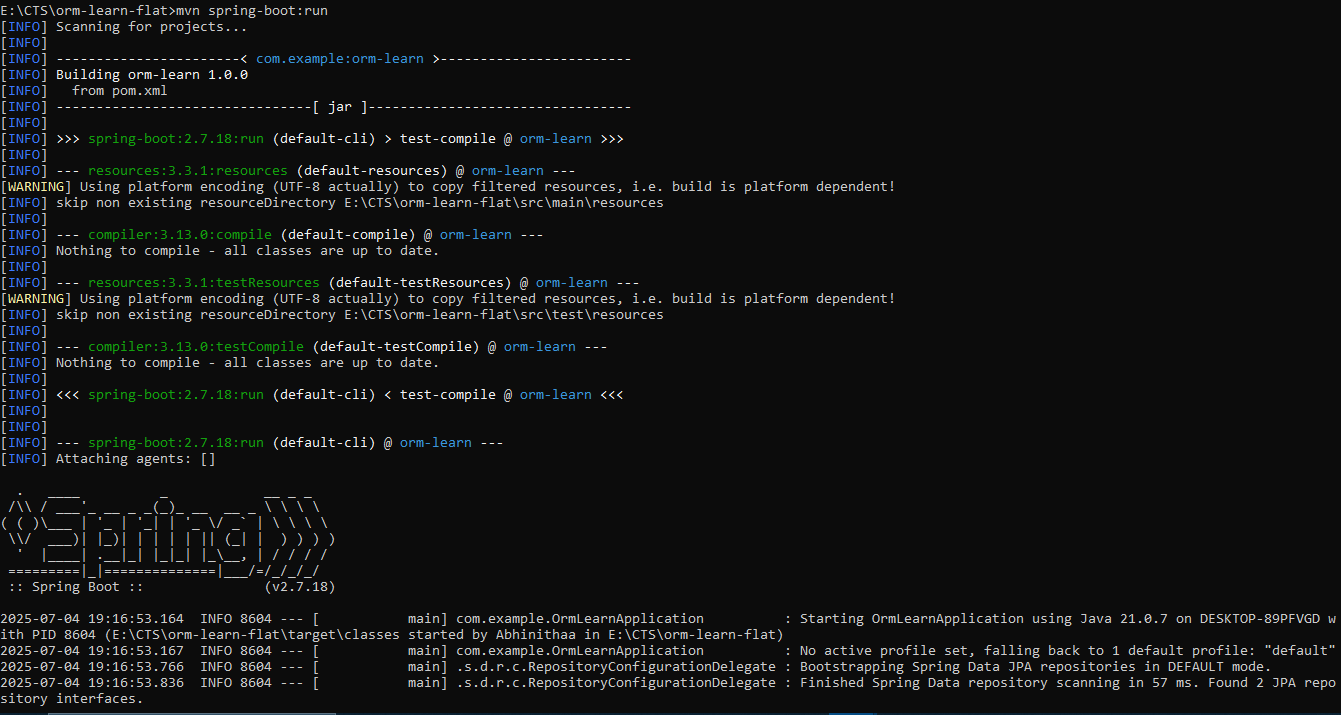
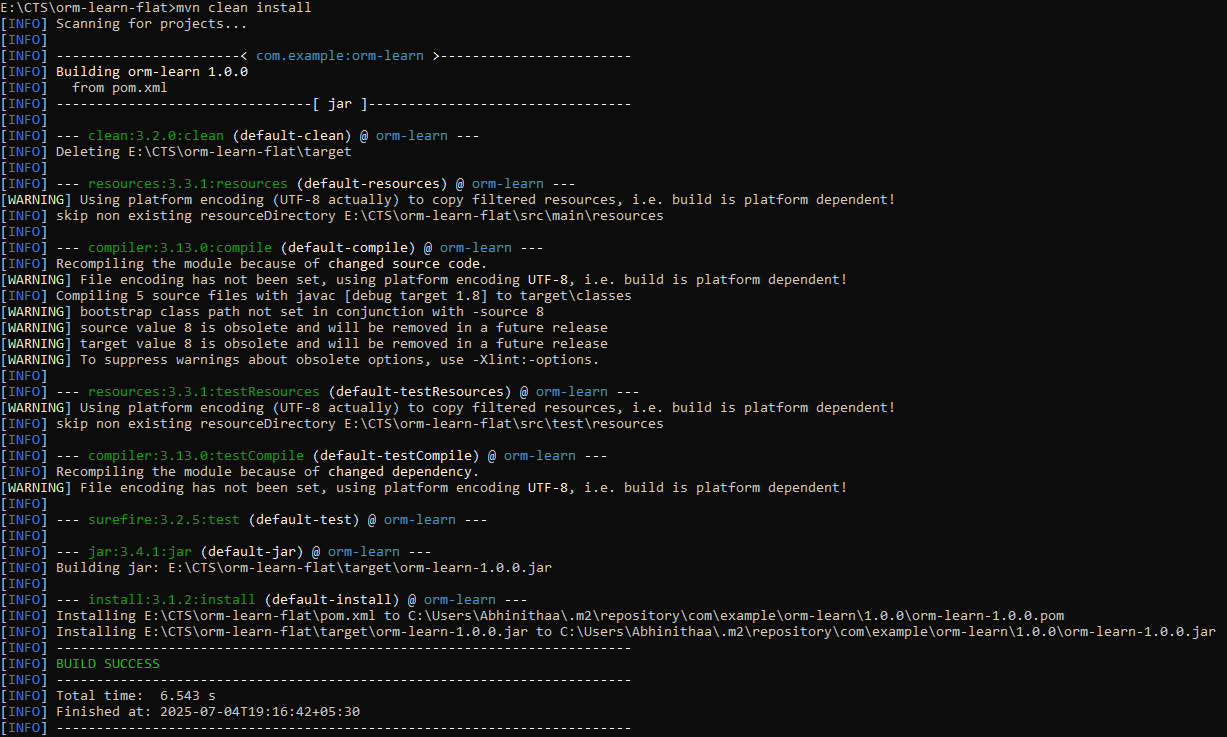
        studentRepository.findAll().forEach(System.out**::**println);

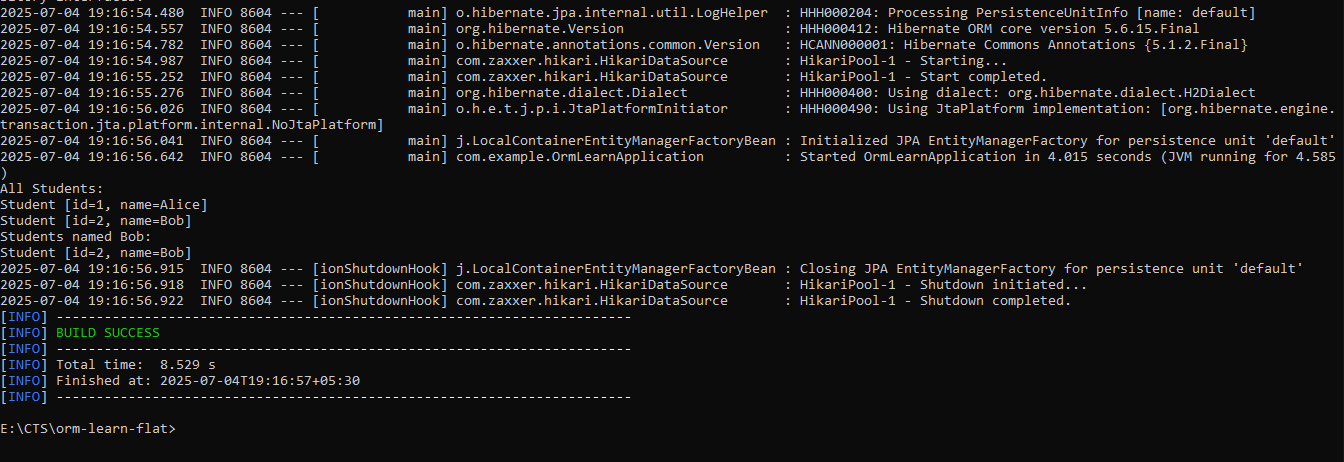
        System.out.println("Students named Bob:");

        studentRepository.findByName("Bob").forEach(System.out**::**println);

    }

}

**Output:**

****

**Implement services for managing Country**

**Country.java**

package com.example.demo;

import jakarta.persistence.\*;

@Entity

public class Country {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String code;

// Getters and Setters

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; }

}

**CountryController.java**

package com.example.demo;

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;

@PostMapping

public Country addCountry(@RequestBody Country country) {

return countryService.addCountry(country);

}

@GetMapping

public List<Country> getAllCountries() {

return countryService.getAllCountries();

}

@GetMapping("/{code}")

public Country getByCode(@PathVariable String code) {

return countryService.getCountryByCode(code);

}

// Find by exact name

@GetMapping("/name/{name}")

public Country getByName(@PathVariable String name) {

return countryService.getByName(name);

}

// Find all countries that contain a keyword (case-insensitive)

@GetMapping("/search")

public List<Country> search(@RequestParam String keyword) {

return countryService.searchCountriesByName(keyword);

}

// Find all countries starting with a prefix

@GetMapping("/starts")

public List<Country> startsWith(@RequestParam String prefix) {

return countryService.countriesStartingWith(prefix);

}

}

**CountryRepository.java**

package com.example.demo;

import java.util.List;

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

public interface CountryRepository extends JpaRepository<Country, Long> {

Country findByName(String name);

// Find by code

Country findByCode(String code);

// Find all countries where name contains something (case-insensitive)

List<Country> findByNameContainingIgnoreCase(String keyword);

// Find all countries starting with a letter

List<Country> findByNameStartingWith(String prefix);

}

**CountryService.java**

package com.example.demo;

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

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

public Country addCountry(Country country) {

return countryRepository.save(country);

}

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

public Country getCountryByCode(String code) {

return countryRepository.findByCode(code);

}

public List<Country> searchCountriesByName(String keyword) {

return countryRepository.findByNameContainingIgnoreCase(keyword);

}

public List<Country> countriesStartingWith(String prefix) {

return countryRepository.findByNameStartingWith(prefix);

}

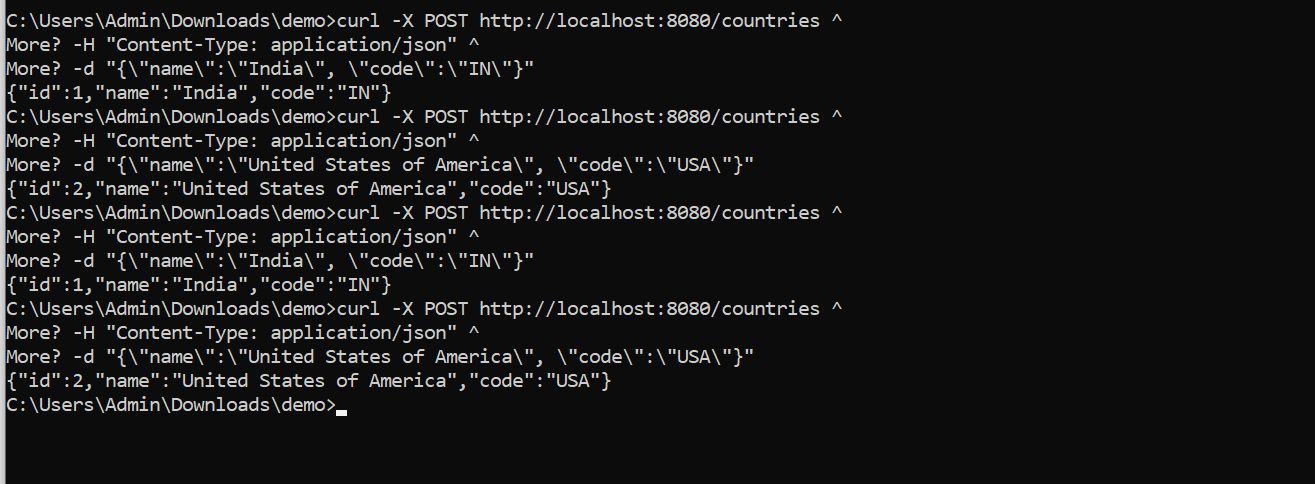
public Country getByName(String name) {

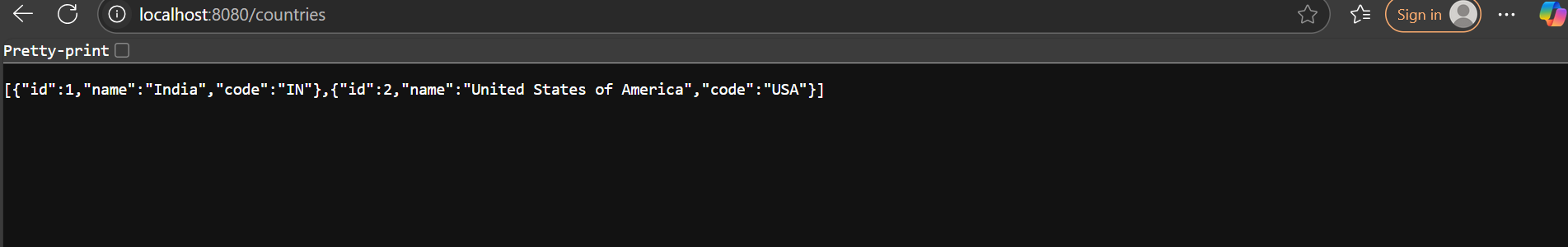
return countryRepository.findByName(name);

}

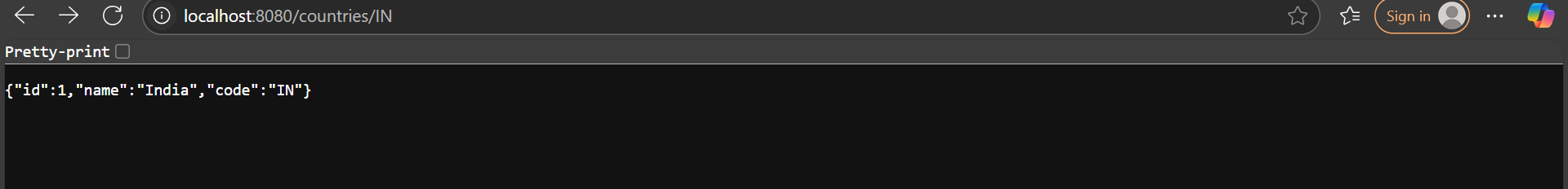
}

**Add a new country**

****

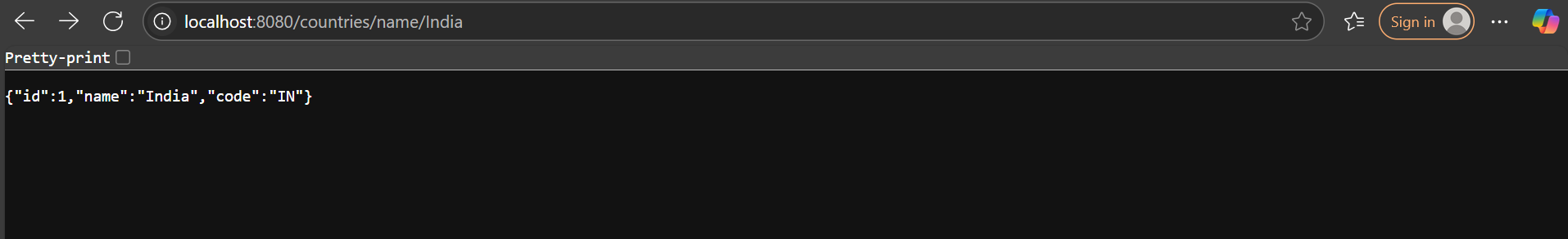
****

**Find a country based on country code**

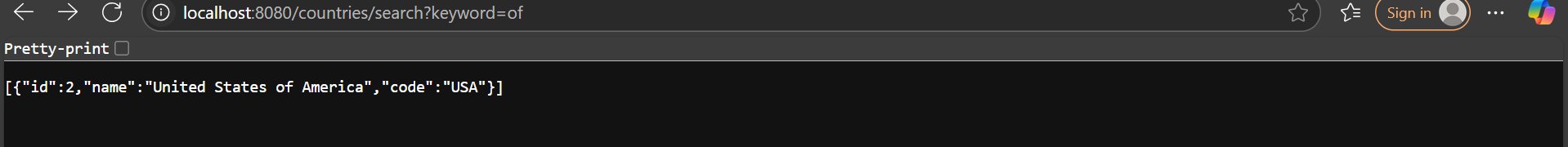
****

**Demonstrate implementation of Query Methods feature of Spring Data JPA**

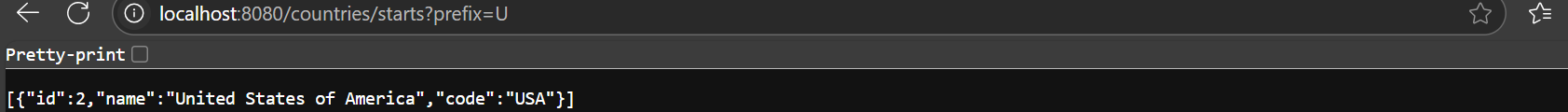
**Byname**

****

**By keyword**



**By prefix**



**Demonstrate implementation of O/R Mapping**

package com.example.demo;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity // Maps this class to a table

public class Country {

@Id // Marks this field as primary key

@GeneratedValue(strategy = GenerationType.IDENTITY) // Auto-increment

private Long id;

private String name; // Will map to column 'name'

private String code; // Will map to column 'code'

// Constructors, Getters, Setters

public Country() {

}

public Country(String name, String code) {

this.name = name;

this.code = code;

}

public Long getId() {

return 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;

}

}