**MANDATORY HANDS-ON**

1. **Spring Data JPA - Quick Example**

**application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn  
spring.datasource.username=root  
spring.datasource.password=qwerty0987  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
  
spring.jpa.show-sql=true  
spring.jpa.hibernate.ddl-auto=none  
  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

**Country.java**

package com.cognizant.orm\_learn.model;  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.Id;  
import jakarta.persistence.Table;  
  
@Entity  
@Table(name = "country")  
public class Country {  
  
 @Id  
 private String coCode;  
  
 private String coName;  
  
 // Getters and setters  
 public String getCoCode() {  
 return coCode;  
 }  
  
 public void setCoCode(String coCode) {  
 this.coCode = coCode;  
 }  
  
 public String getCoName() {  
 return coName;  
 }  
  
 public void setCoName(String coName) {  
 this.coName = coName;  
 }  
}

**CountryService.java**

package com.cognizant.orm\_learn.service;  
  
import java.util.List;  
  
import org.springframework.transaction.annotation.Transactional;  
  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import com.cognizant.orm\_learn.model.Country;  
import com.cognizant.orm\_learn.repository.CountryRepository;  
  
@Service  
public class CountryService {  
  
 @Autowired  
 private CountryRepository countryRepository;  
  
 @Transactional  
 public List<Country> getAllCountries() {  
 return countryRepository.findAll();  
 }  
}

**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 https://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>3.1.3</version>  
 <relativePath/>  
 </parent>  
  
 <groupId>com.cognizant</groupId>  
 <artifactId>orm-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>orm-learn</name>  
 <description>Demo project for Spring Data JPA and Hibernate</description>  
  
 <properties>  
 <java.version>17</java.version>  
 </properties>  
  
 <dependencies>  
 <!-- Spring Boot Starter for Spring Data JPA -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <!-- MySQL JDBC Driver -->  
 <dependency>  
 <groupId>com.mysql</groupId>  
 <artifactId>mysql-connector-j</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
  
 <!-- Spring Boot DevTools -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
  
 <!-- Spring Boot Starter Test -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**CountryRepository.java**

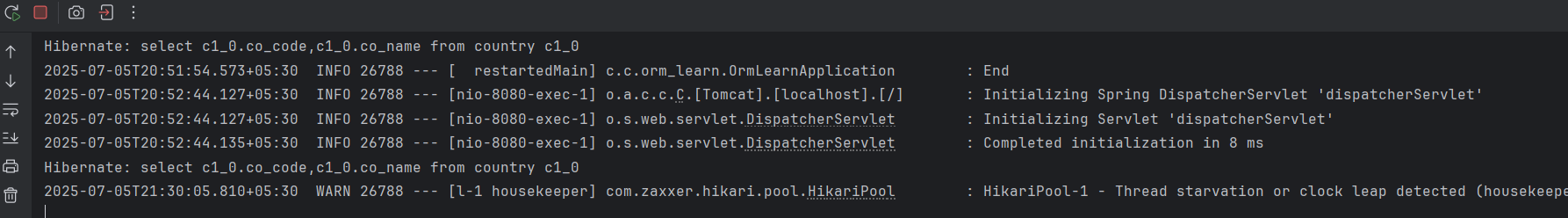
package com.cognizant.orm\_learn.repository;  
  
import com.cognizant.orm\_learn.model.Country;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface CountryRepository extends JpaRepository<Country, String> {  
}

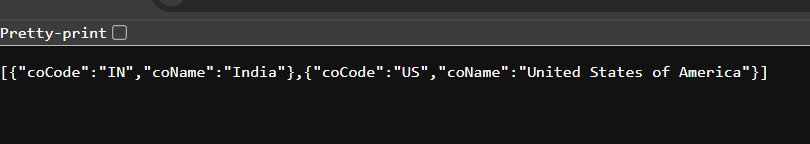
**CountryController.java**

package com.cognizant.orm\_learn.controller;  
  
import com.cognizant.orm\_learn.model.Country;  
import com.cognizant.orm\_learn.repository.CountryRepository;  
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 CountryRepository countryRepository;  
  
 @GetMapping  
 public List<Country> getAllCountries() {  
 return countryRepository.findAll();  
 }  
}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;  
  
import java.util.List;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
  
import com.cognizant.orm\_learn.model.Country;  
import com.cognizant.orm\_learn.service.CountryService;  
  
@SpringBootApplication  
public class OrmLearnApplication {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(OrmLearnApplication.class);  
 private static CountryService *countryService*;  
  
 public static void main(String[] args) {  
 ApplicationContext context = SpringApplication.*run*(OrmLearnApplication.class, args);  
 *LOGGER*.info("Inside main");  
  
 *countryService* = context.getBean(CountryService.class);  
 *testGetAllCountries*();  
 }  
  
 private static void testGetAllCountries() {  
 *LOGGER*.info("Start");  
 List<Country> countries = *countryService*.getAllCountries();  
 *LOGGER*.debug("countries={}", countries);  
 *LOGGER*.info("End");  
 }  
}





1. **Difference between JPA, Hibernate and Spring Data JPA**

**Difference**

| **Feature** | **JPA** | **Hibernate** | **Spring Data JPA** |
| --- | --- | --- | --- |
| **Type** | It's just a specification  (a set of rules or guidelines). | A real tool (implementation) that follows JPA rules. | A higher-level framework built on top of JPA and Hibernate. |
| **Does it Provide Code?** | No – just interfaces and annotations. | Yes – it provides full working code and utilities. | Yes – reduces boilerplate by auto-generating repository code. |
| **Transaction Handling** | Doesn't handle transactions by itself. | Yes – handles transactions. | Yes – Spring manages transactions automatically. |
| **Repository Pattern** | Not built-in – we have to write our own implementation. | Possible – but we must create custom implementations. | Built-in – just extend JpaRepository and it's ready to use. |

**DemoApplication.java**

package com.exmaple.demo;  
  
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 DemoApplication implements CommandLineRunner {  
  
 @Autowired  
 private EmployeeService employeeService;  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(DemoApplication.class, args);  
 }  
  
 @Override  
 public void run(String... args) {  
 Employee emp = new Employee();  
 emp.setName("Alice");  
 employeeService.addEmployee(emp);  
 System.*out*.println("Employee Saved via Spring Data JPA");  
 }  
}

**Employee.java**

package com.exmaple.demo;  
  
import jakarta.persistence.\*;  
  
@Entity  
@Table(name = "employee")  
public class Employee {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Integer id;  
  
 private String name;  
  
 // Getters and setters  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
}

**EmployeeRepository.java**

package com.exmaple.demo;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  
}

**EmployeeService.java**

package com.exmaple.demo;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 @Transactional  
 public void addEmployee(Employee employee) {  
 employeeRepository.save(employee);  
 }  
}

**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.jpa.hibernate.ddl-auto=update  
spring.h2.console.enabled=true

**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>3.2.5</version>  
 <relativePath/>  
 </parent>  
  
 <groupId>com.example</groupId>  
 <artifactId>employee-management</artifactId>  
 <version>1.0.0</version>  
 <packaging>jar</packaging>  
  
 <name>employee-management</name>  
 <description>Spring Boot Employee Management Project</description>  
  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
</project>

