Week-3

Spring Maven | Spring Data JPA with Spring Boot, Hibernate

**Exercise 1: Configuring a Basic Spring Application**

**Scenario:**

Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

**Code:**

**Pom.xml**

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

</dependencies>

**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>">

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

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

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

</bean>

</beans>

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public void saveBook() {

System.out.println("BookRepository: Book saved to the database.");

}

}

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook() {

System.out.println("BookService: Adding book...");

bookRepository.saveBook();

}

}

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

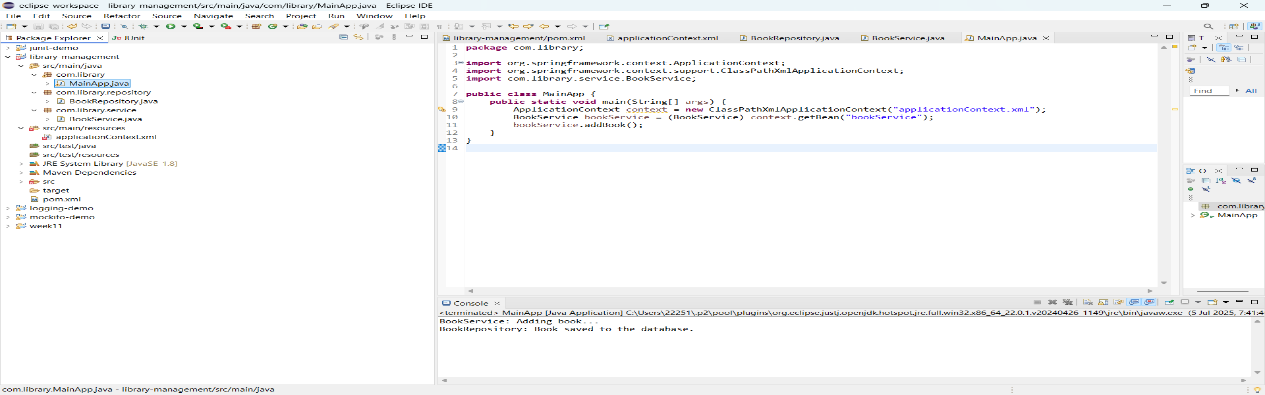
BookService bookService = (BookService) context.getBean("bookService");

bookService.addBook();

}

}

**Output**:



**Exercise 2: Implementing Dependency Injection**

**Scenario:**

In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

**Code:**

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

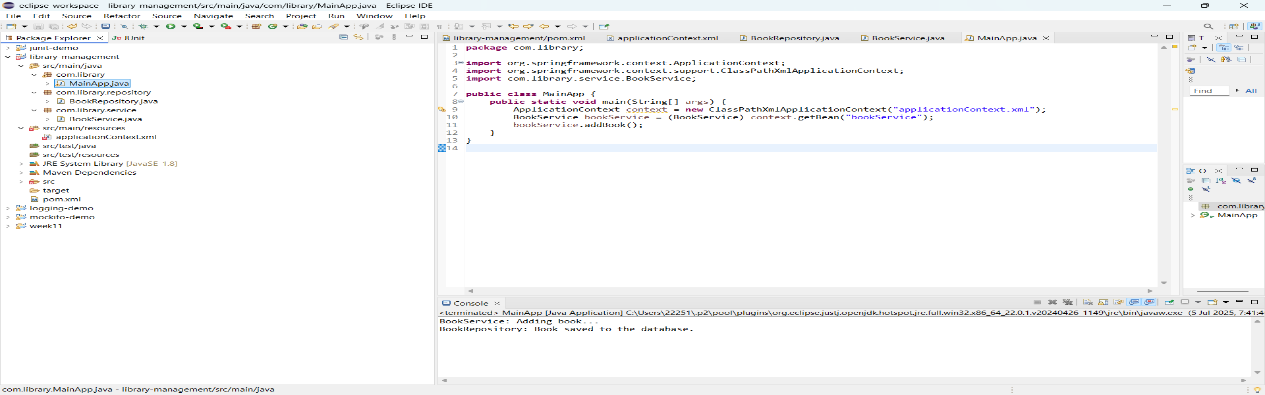
BookService bookService = (BookService) context.getBean("bookService");

bookService.addBook();

}

}

**Output**:



**Exercise 7: Implementing Constructor and Setter Injection**

**Scenario:**

The library management application requires both constructor and setter injection for better control over bean initialization.

**Code:**

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository,

private String libraryName;

public BookService(String libraryName) {

this.libraryName = libraryName;

}

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook() {

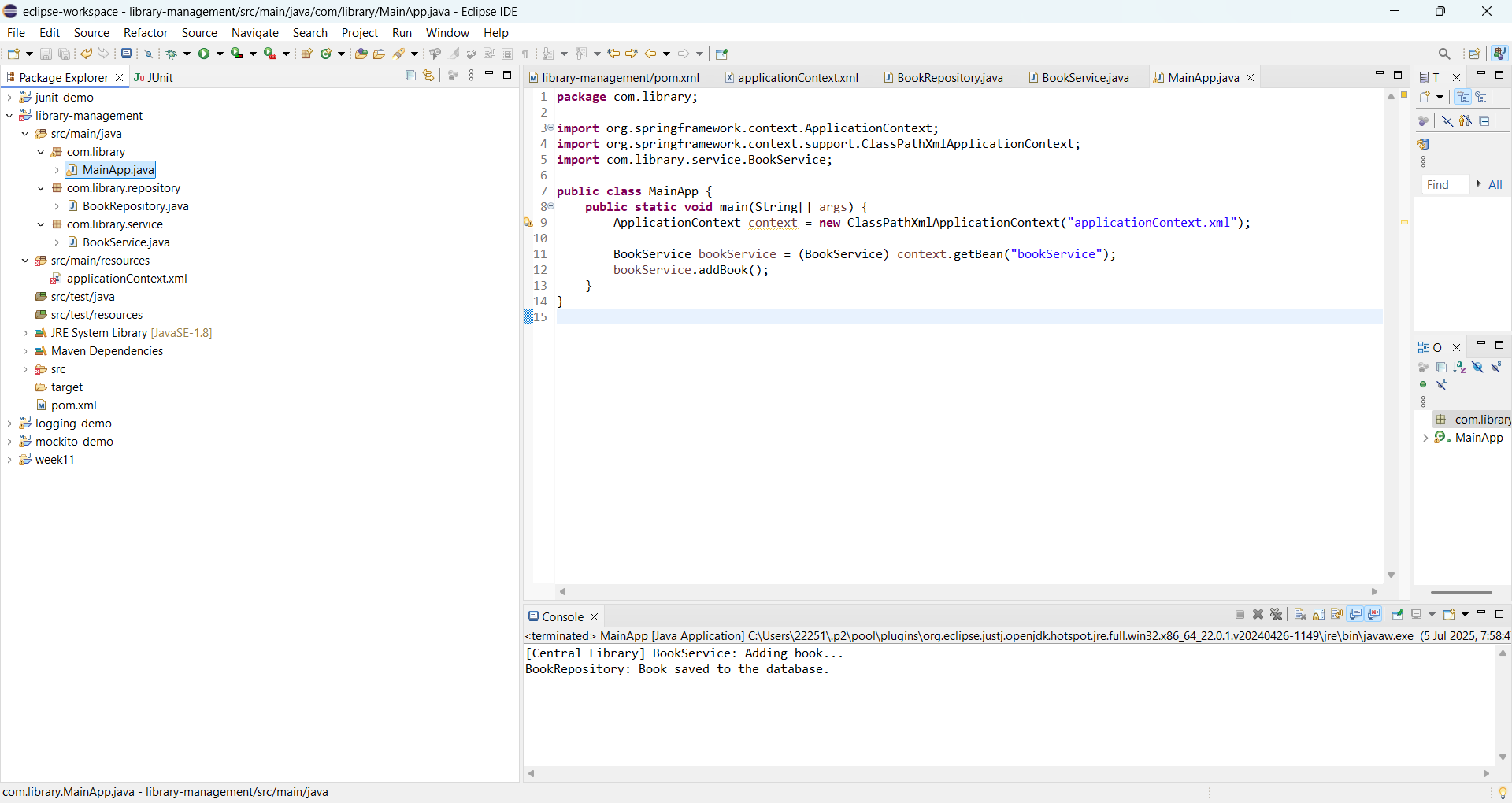
System.out.println("[" + libraryName + "] BookService: Adding book...");

bookRepository.saveBook();

}

}

**Output:**

****

**Spring Data JPA with Spring Boot, Hibernate**

**Exercise:Spring Data JPA - Quick Example**

**Code:**

***OrmLearnApp.java***

package com.cognizant.orm\_learn;

import java.util.List;

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 {

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

CountryService service = context.getBean(CountryService.class);

List<Country> countries = service.getAllCountries();

System.out.println("Country List:");

for (Country country : countries) {

System.out.println(country.getName());

}

}

}

**CountryService.java**

package com.cognizant.orm\_learn.service;

import java.util.Arrays;

import java.util.List;

import org.springframework.stereotype.Service;

import com.cognizant.orm\_learn.model.Country;

@Service

public class CountryService {

public List<Country> getAllCountries() {

return Arrays.asList(

new Country("India"),

new Country("USA"),

new Country("Germany"),

new Country("Japan")

);

}}

**Country.java**

package com.cognizant.orm\_learn.model;

public class Country {

private String name;

public Country() {}

public Country(String name) {

this.name = name;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**Output:**

**Country List:**

India

USA

Germany

Japan

**Exercise-2 Difference between JPA, Hibernate and Spring Data JPA**

**Code:**

CREATE TABLE country (

code VARCHAR(2) PRIMARY KEY,

name VARCHAR(100)

);

INSERT INTO country (code, name) VALUES ('IN', 'India'), ('US', 'United States'), ('JP', 'Japan');

**CountryServiceImpl.java**

package com.cognizant.ormlearn.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

@Service

public class CountryServiceImpl implements CountryService {

@Autowired

private CountryRepository countryRepository;

@Override

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**CountryService.java**

package com.cognizant.ormlearn.service;

import java.util.List;

import com.cognizant.ormlearn.model.Country;

public interface CountryService {

List<Country> getAllCountries();

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

import java.util.List;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

@SpringBootApplication(scanBasePackages = "com.cognizant.ormlearn")

public class OrmLearnApplication {

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

CountryService service = context.getBean(CountryService.class);

List<Country> countries = service.getAllCountries();

for (Country country : countries) {

System.out.println(country.getName());

}

}

}

**Output:**

India

United States

Japan

**Exercise-3:**

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

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

import java.util.List;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

@SpringBootApplication(scanBasePackages = "com.cognizant.ormlearn")

public class OrmLearnApplication {

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

CountryService service = context.getBean(CountryService.class);

System.out.println("=== Get country by name ===");

System.out.println(service.getCountryByName("India"));

System.out.println("\n=== Countries containing 'an' ===");

service.getCountriesByPartialName("an").forEach(System.out::println);

System.out.println("\n=== Countries sorted by name ===");

service.getCountriesSortedByName().forEach(System.out::println);

}

}

**CountryServiceImpl.java**

package com.cognizant.ormlearn.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

@Service

public class CountryServiceImpl implements CountryService {

@Autowired

private CountryRepository countryRepository;

@Override

public Country getCountryByName(String name) {

return countryRepository.findByName(name);

}

@Override

public List<Country> getCountriesByPartialName(String keyword) {

return countryRepository.findByNameContaining(keyword);

}

@Override

public List<Country> getCountriesSortedByName() {

return countryRepository.findByNameOrderByNameAsc();

}

}

**CountryService.java**

package com.cognizant.ormlearn.service;

import java.util.List;

import com.cognizant.ormlearn.model.Country;

public interface CountryService {

Country getCountryByName(String name);

List<Country> getCountriesByPartialName(String keyword);

List<Country> getCountriesSortedByName();

}

**CountryRepository.java**

package com.cognizant.ormlearn.repository;

import java.util.List;

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

import com.cognizant.ormlearn.model.Country;

public interface CountryRepository extends JpaRepository<Country, String> {

Country findByName(String name);

List<Country> findByNameContaining(String keyword);

List<Country> findByNameStartingWith(String prefix);

List<Country> findByNameEndingWith(String suffix);

List<Country> findByNameOrderByNameAsc();

}

**Country.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "country")

public class Country {

@Id

private String code;

private String name;

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

**Output:**

=== Get country by name ===

Country [code=IN, name=India]

=== Countries containing ===

Country [code=JP, name=Japan]

Country [code=FR, name=France]

=== Countries sorted by name ===

Country [code=CN, name=China]

Country [code=FR, name=France]

Country [code=IN, name=India]

Country [code=JP, name=Japan]

Country [code=US, name=United States]