**WEEK-3**

**SUPER SET ID : 6389675**

**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.**

**BookRepository.java**

**package com.library.librarymanagement;**

**import org.springframework.stereotype.Repository;**

**@Repository**

**public class BookRepository {**

**public String getBookTitle() {**

**return "Spring Boot in Practice";**

**}**

**}**

**BookService.java:**

**package com.library.librarymanagement;**

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

**import org.springframework.stereotype.Service;**

**@Service**

**public class BookService {**

**private final BookRepository bookRepository;**

**@Autowired**

**public BookService(BookRepository bookRepository) {**

**this.bookRepository = bookRepository;**

**}**

**public void showBook() {**

**System.out.println("Book Title: " + bookRepository.getBookTitle());**

**}**

**}**

**LibrarymanagementApplication.java:**

**package com.library.librarymanagement;**

**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 LibrarymanagementApplication implements CommandLineRunner {**

**@Autowired**

**private BookService bookService;**

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

**SpringApplication.run(LibrarymanagementApplication.class, args);**

**}**

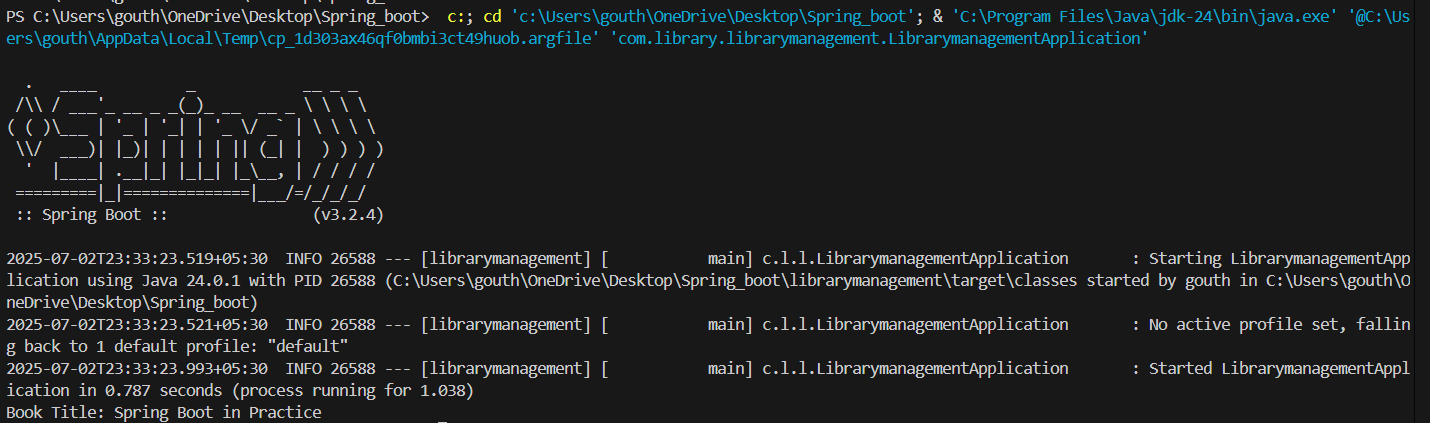
**@Override**

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

**bookService.showBook();  // This runs automatically after app starts**

**}**

**}**

**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.**

**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**

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

**<!-- Define BookRepository bean -->**

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

**<!-- Define BookService bean and inject BookRepository using setter -->**

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

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

**</bean>**

**</beans>**

**BookService.java:**

**package com.library.librarymanagement;**

**public class BookService {**

**private BookRepository bookRepository;**

**public void setBookRepository(BookRepository bookRepository) {**

**this.bookRepository = bookRepository;**

**}**

**public void showBook() {**

**System.out.println("Book Title: " + bookRepository.getBookTitle());**

**}**

**}**

**BookRepository.java:**

**package com.library.librarymanagement;**

**public class BookRepository {**

**public String getBookTitle() {**

**return "Spring in Action";**

**}**

**}**

**LibrarymanagementApplication.java:**

**package com.library.librarymanagement;**

**import org.springframework.context.ApplicationContext;**

**import org.springframework.context.support.ClassPathXmlApplicationContext;**

**public class LibrarymanagementApplication {**

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

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

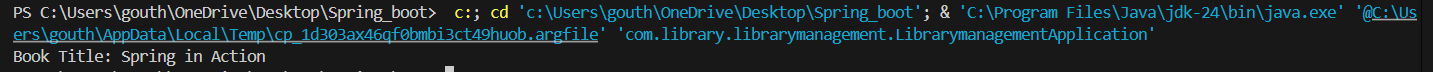
**BookService bookService = context.getBean("bookService", BookService.class);**

**bookService.showBook();**

**}**

**}**

**OUTPUT:**

**--------------------------------------------------------------------------------------------------------------------------------------**

**Exercise 4: Creating and Configuring a Maven Project**

**Scenario:**

**You need to set up a new Maven project for the library management application and add Spring dependencies.**

**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.library</groupId>**

**<artifactId>librarymanagement</artifactId>**

**<version>0.0.1-SNAPSHOT</version>**

**<packaging>jar</packaging>**

**<name>librarymanagement</name>**

**<description>Library Management Spring Boot App</description>**

**<parent>**

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

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

**<version>3.2.4</version> <!-- or latest compatible -->**

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

**</parent>**

**<properties>**

**<java.version>17</java.version> <!--- Use 17 or 21, not 24 -->**

**</properties>**

**<dependencies>**

**// Spring Boot Starter**

**<dependency>**

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

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

**</dependency>**

**//Spring Boot Testing (optional)**

**<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>**

**</plugin>**

**</plugins>**

**</build>**

**</project>**

**BookRepository.java:**

**package com.library.librarymanagement;**

**public class BookRepository {**

**public String getBookTitle() {**

**return "This is Exercise 3";**

**}**

**}**

**BookService.java:**

**package com.library.librarymanagement;**

**public class BookService {**

**private BookRepository bookRepository;**

**public void setBookRepository(BookRepository bookRepository) {**

**this.bookRepository = bookRepository;**

**}**

**public void showBook() {**

**System.out.println("Book Title: " + bookRepository.getBookTitle());**

**}**

**}**

**LibrarymanagementApplication.java:**

**package com.library.librarymanagement;**

**import org.springframework.context.ApplicationContext;**

**import org.springframework.context.support.ClassPathXmlApplicationContext;**

**public class LibrarymanagementApplication {**

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

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

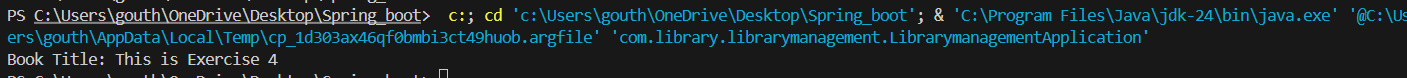
**BookService bookService = context.getBean("bookService", BookService.class);**

**bookService.showBook();**

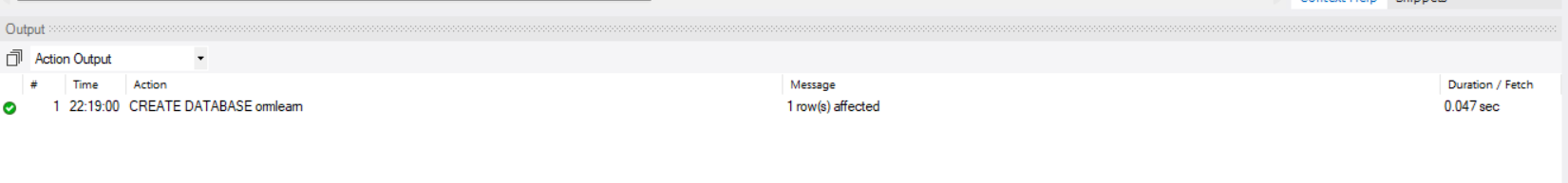
**}**

**}**

**OUPUT:**



**--------------------------------------------------------------------------------------------------------------------------------------**

1. **Spring Data JPA - Quick Example**
2. **application.properties:**
3. //Logging
4. logging.level.org.springframework=info
5. logging.level.com.cognizant=debug
6. logging.level.org.hibernate.SQL=trace
7. logging.level.org.hibernate.type.descriptor.sql=trace
8. logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n
9. // Database
10. spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
11. spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn
12. spring.datasource.username=root
13. spring.datasource.password=root
14. // Hibernate
15. spring.jpa.hibernate.ddl-auto=validate
16. spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
17. **Create table:**
18. create table country(
19. code varchar(2) primary key,
20. name varchar(50)
21. );
22. insert into country values ('IN', 'India');
23. insert into country values ('US', 'United States of America');

**// Country.java:**

1. package com.cognizant.ormlearn.model;
2. import javax.persistence.\*;
3. @Entity
4. @Table(name = "country")
5. public class Country {
6. @Id
7. @Column(name = "code")
8. private String code;
9. @Column(name = "name")
10. private String name;
11. // Getters & Setters
12. public String getCode() { return code; }
13. public void setCode(String code) { this.code = code; }
14. public String getName() { return name; }
15. public void setName(String name) { this.name = name; }
16. @Override
17. public String toString() {
18. return "Country [code=" + code + ", name=" + name + "]";
19. }
20. }

**// CountryRepository.java:**

1. package com.cognizant.ormlearn.repository;
2. import org.springframework.data.jpa.repository.JpaRepository;
3. import org.springframework.stereotype.Repository;
4. import com.cognizant.ormlearn.model.Country;
5. @Repository
6. public interface CountryRepository extends JpaRepository<Country, String> {
7. }

**// CountryService.java:**

1. package com.cognizant.ormlearn.service;
2. import java.util.List;
3. import org.springframework.beans.factory.annotation.Autowired;
4. import org.springframework.stereotype.Service;
5. import org.springframework.transaction.annotation.Transactional;
6. import com.cognizant.ormlearn.model.Country;
7. import com.cognizant.ormlearn.repository.CountryRepository;
8. @Service
9. public class CountryService {
10. @Autowired
11. private CountryRepository countryRepository;
12. @Transactional
13. public List<Country> getAllCountries() {
14. return countryRepository.findAll();
15. }
16. }

**//OrmLearnApplication.java:**

1. package com.cognizant.ormlearn;
2. import java.util.List;
3. import org.slf4j.Logger;
4. import org.slf4j.LoggerFactory;
5. import org.springframework.boot.SpringApplication;
6. import org.springframework.boot.autoconfigure.SpringBootApplication;
7. import org.springframework.context.ApplicationContext;
8. import com.cognizant.ormlearn.model.Country;
9. import com.cognizant.ormlearn.service.CountryService;
10. @SpringBootApplication
11. public class OrmLearnApplication {
12. private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);
13. private static CountryService countryService;
14. public static void main(String[] args) {
15. ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);
16. countryService = context.getBean(CountryService.class);
17. testGetAllCountries();
18. }
19. private static void testGetAllCountries() {
20. LOGGER.info("Start");
21. List<Country> countries = countryService.getAllCountries();
22. LOGGER.debug("countries={}", countries);
23. LOGGER.info("End");
24. }
25. }

**OUTPUT:**



**--------------------------------------------------------------------------------------------------------------------------------------**

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

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0" ...>

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>orm-learn</name>

<description>Demo project for Spring Data JPA and Hibernate</description>

<parent>

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

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

<version>3.2.0</version>

</parent>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**// application.properties:**

// Logging

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=debug

logging.level.org.hibernate.type.descriptor.sql=trace

// DB Connection

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

**// Employee.java:**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

@Entity

@Table(name = "employee")

public class Employee {

@Id

private int id;

@Column(name = "name")

private String name;

@Column(name = "salary")

private double salary;

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; }

public double getSalary() { return salary; }

public void setSalary(double salary) { this.salary = salary; }

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + "]";

}

}

**// EmployeeRepository.java:**

package com.cognizant.ormlearn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Employee;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**// EmployeeService.java:**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.repository.EmployeeRepository;

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 emp) {

employeeRepository.save(emp);

}

}

**// OrmLearnApplication.java:**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.service.EmployeeService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class OrmLearnApplication {

private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);

private static EmployeeService employeeService;

public static void main(String[] args) {

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

employeeService = context.getBean(EmployeeService.class);

LOGGER.info("Inside main");

testAddEmployee();

}

private static void testAddEmployee() {

LOGGER.info("Start addEmployee");

Employee emp = new Employee();

emp.setId(1);

emp.setName("Siddu");

emp.setSalary(50000);

employeeService.addEmployee(emp);

LOGGER.info("End addEmployee");

}

}

**// SQL Table:**

CREATE DATABASE IF NOT EXISTS ormlearn;

USE ormlearn;

CREATE TABLE employee (

id INT PRIMARY KEY,

name VARCHAR(100),

salary DOUBLE

);

**OUPUT:**

