Exercise 1: Configuring a Basic Spring Application

Subject: Spring Framework

Title: LibraryManagement - A Basic Spring Backend Setup

Submitted by: Likhitha Sri Dasari

# Objective

To configure a basic Spring application using XML-based configuration for a library management system. The application consists of BookService and BookRepository beans, defined and managed using the Spring Framework.

# Step 1: Set Up a Spring Project

## Project Structure

LibraryManagement/  
├── src/  
│ └── main/  
│ ├── java/  
│ │ └── com/  
│ │ └── library/  
│ │ ├── service/  
│ │ │ └── BookService.java  
│ │ └── repository/  
│ │ └── BookRepository.java  
│ └── resources/  
│ └── applicationContext.xml  
├── pom.xml

## pom.xml Dependencies

<project xmlns="http://maven.apache.org/POM/4.0.0" ...>  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <dependencies>  
 <!-- Spring Core -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.29</version>  
 </dependency>  
 </dependencies>  
</project>

# Step 2: Configure the Application Context

applicationContext.xml (placed inside src/main/resources)

<?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">  
  
 <!-- Repository Bean -->  
 <bean id="bookRepository" class="com.library.repository.BookRepository"/>  
  
 <!-- Service Bean with Dependency Injection -->  
 <bean id="bookService" class="com.library.service.BookService">  
 <property name="bookRepository" ref="bookRepository"/>  
 </bean>  
</beans>

# Step 3: Define Service and Repository Classes

## BookRepository.java

package com.library.repository;  
  
public class BookRepository {  
 public void displayBookInfo() {  
 System.out.println("BookRepository: Fetching book data...");  
 }  
}

## BookService.java

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
 private BookRepository bookRepository;  
  
 // Setter-based Dependency Injection  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void showBookInfo() {  
 System.out.println("BookService: Calling repository...");  
 bookRepository.displayBookInfo();  
 }  
}

# Step 4: Run the Application

## MainApp.java

package com.library;  
  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class MainApp {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService bookService = (BookService) context.getBean("bookService");  
 bookService.showBookInfo();  
 }  
}

## Output

BookService: Calling repository...  
BookRepository: Fetching book data...

# Conclusion

This assignment demonstrates the fundamental setup of a Spring application using XML-based configuration. The BookService and BookRepository beans were successfully wired and tested using a simple Java application. This exercise lays the groundwork for more advanced Spring concepts such as annotations and Spring Boot.