**Spring Core and Maven – Exercise**

**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.**

**Steps:**

**1. Set Up a Spring Project:**

**o Create a Maven project named LibraryManagement.**

**o Add Spring Core dependencies in the pom.xml file.**

**2. Configure the Application Context:**

**o Create an XML configuration file named applicationContext.xml in the src/main/resources directory.**

**o Define beans for BookService and BookRepository in the XML file.**

**3. Define Service and Repository Classes:**

**o Create a package com.library.service and add a class BookService.**

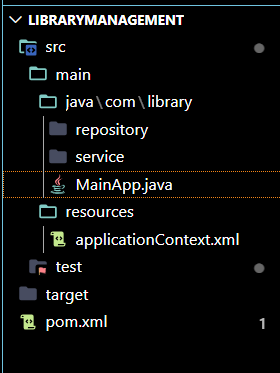
**o Create a package com.library.repository and add a class BookRepository.**

**4. Run the Application:**

**o Create a main class to load the Spring context and test the configuration.**

**ANS:**

**STEP 1:**

****

MAVEN PROJECT CREATED

DEPENDENCY ADD:

<dependencies>

<!-- Spring Core Dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

</dependencies>

**STEP 2:**

**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>

**Step 3:**

**BookRepository.java:**

package com.library.repository;

public class BookRepository {

public void saveBook(String title) {

System.out.println("BookRepository: Saving a book " + title);

}

}

**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(String title) {

System.out.println("BookService: Adding a new book now " + title);

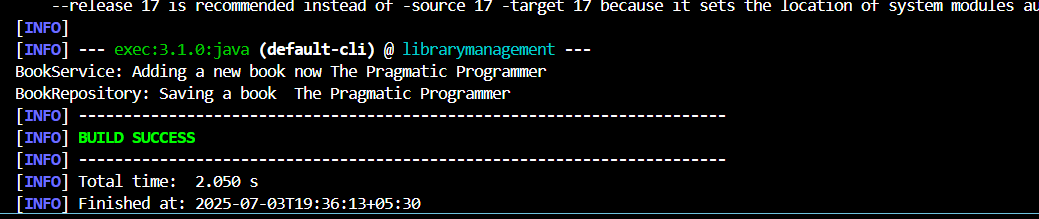
bookRepository.saveBook(title);

}

}

**Step 4:**

**RUN:**

****

**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.**

**Steps:**

**1. Modify the XML Configuration:**

**o Update applicationContext.xml to wire BookRepository into BookService.**

**2. Update the BookService Class:**

**o Ensure that BookService class has a setter method for BookRepository.**

**3. Test the Configuration:**

**o Run the LibraryManagementApplication main class to verify the dependency injection**.

**ANS:**

**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" />

<!-- BookService Bean with Dependency Injection -->

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

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

</bean>

</beans>

**BookService.java:**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Setter for Dependency Injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

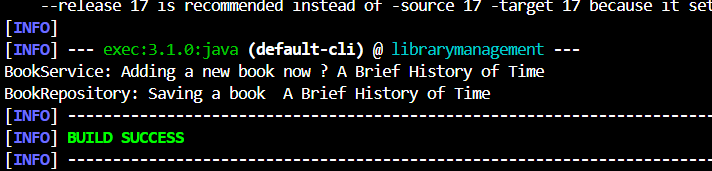
System.out.println("BookService: Adding a new book now " + title);

bookRepository.saveBook(title);

}

}

**RUN:**

****

**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.**

**Steps:**

**1. Create a New Maven Project:**

**o Create a new Maven project named LibraryManagement.**

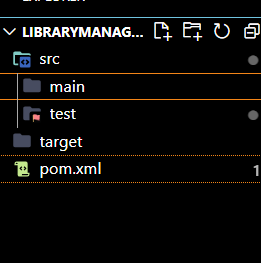
**2. Add Spring Dependencies in pom.xml:**

**o Include dependencies for Spring Context, Spring AOP, and Spring WebMVC.**

**3. Configure Maven Plugins:**

**o Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.**

**ANS:**

**STEP 1:**

**NEW MAVEN PROJECT FOR LIBRARYMANAGEMENT CREATED**

**STEP 2 and 3:**

o Include dependencies for Spring Context, Spring AOP, and Spring WebMVC.

3. Configure Maven Plugins:

o Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.

**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>

<groupId>com.library</groupId>

<artifactId>librarymanagement</artifactId>

<version>1.0-SNAPSHOT</version>

<name>Library Management</name>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

<dependencies>

<!-- Spring Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

<!-- Spring AOP -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.33</version>

</dependency>

<!-- Spring WebMVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.33</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Compiler Plugin -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

</project>