**WEEK 3**

**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.repository;  
  
public class BookRepository {  
 public void saveBook(String title) {  
 System.out.println("Saved book: " + title);  
 }  
}

**BookService.java**

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
 private BookRepository bookRepository;  
  
 // Setter for Spring to inject dependency  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void addBook(String title) {  
 System.out.println("Adding book: " + title);  
 bookRepository.saveBook(title);  
 }  
}

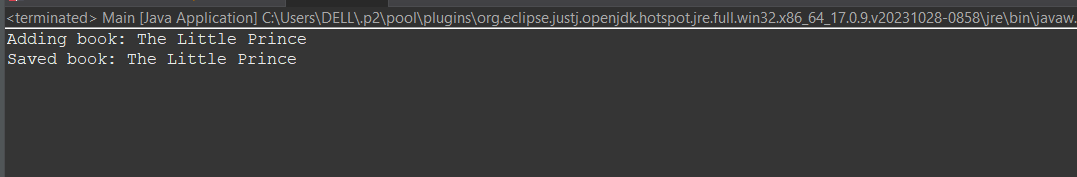
**Main.java**

package com.library1;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import com.library.service.BookService;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService bookService = context.getBean("bookService", BookService.class);  
 bookService.addBook("The Little Prince");  
 }  
}

**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>

**Output:**



**Exercise 2: Implementing Dependency Injection Using Spring’s IoC**

Scenario:  
In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's Inversion of Control (IoC) and Dependency Injection (DI).

**BookRepository.java**

package com.library.repository;  
  
public class BookRepository {  
 public void getBooks() {  
 System.out.println("Fetching book list from repository...");  
 }  
}

**BookService.java**

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
 private BookRepository bookRepository;  
  
 // Setter for Spring to inject dependency  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void displayBooks() {  
 bookRepository.getBooks();  
 }  
}

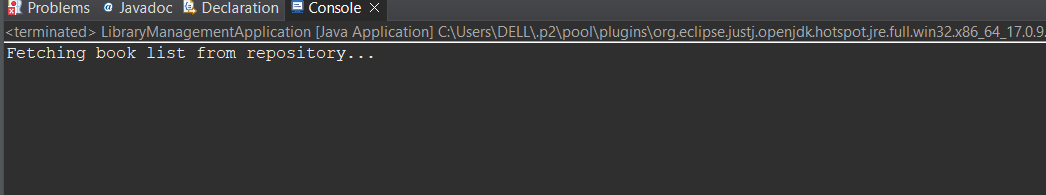
**LibraryManagementApplication.java**

package com.library;  
  
import com.library.service.BookService;  
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.displayBooks();  
 }  
}

**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>

**Expected 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.

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

• Open Eclipse and go to File > New > Maven Project.  
• Choose 'Create a simple project' if needed and click Next.  
• Enter the following details:  
 - Group Id: com.library  
 - Artifact Id: LibraryManagement  
• Click Finish.

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

Add the following dependencies inside the <dependencies> tag:

<dependencies>  
 <!-- Spring Context -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.34</version>  
 </dependency>  
  
 <!-- Spring AOP -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>5.3.34</version>  
 </dependency>  
  
 <!-- Spring WebMVC -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>5.3.34</version>  
 </dependency>  
</dependencies>

**Step 3: Configure Maven Compiler Plugin**

Add the following inside the <build> tag in pom.xml:

<build>  
 <plugins>  
 <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>