Spring Core and Maven

## Exercise 1: Configuring a Basic Spring Application Step 1: Create a New Maven Project

1) Open Eclipse.  
2) Go to File > New > Project…  
3) In the wizard, expand Maven, and select Maven Project.

**Step 2: Add JUnit Dependency**

1) In **Project Explorer**, expand your project.  
2️)Double-click **pom.xml** to open it.  
3️)Inside <dependencies>, paste this block:

<dependencies>

<!-- Spring Core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.30</version>

</dependency>

</dependencies>

**Step 3: Create the XML Configuration**Create applicationContext.xml file under **src/test/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

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

<!-- BookRepository Bean -->

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

<!-- BookService Bean -->

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

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

</bean>

</beans>

**Step 4: Create Service ,Repository and Main Classes**  
**BookRepository.java**

**Code :**

package com.library.repository;

public class BookRepository {

public void saveBook(String title) {

System.out.println("Book '" + title + "' saved to the repository.");

}

**BookService.java**

**Code :**

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

System.out.println("Adding book: " + title);

bookRepository.saveBook(title);

}

}

**LibraryApp.java**

**Code :**

public class LibraryApp {

public static void main(String[] args) {

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

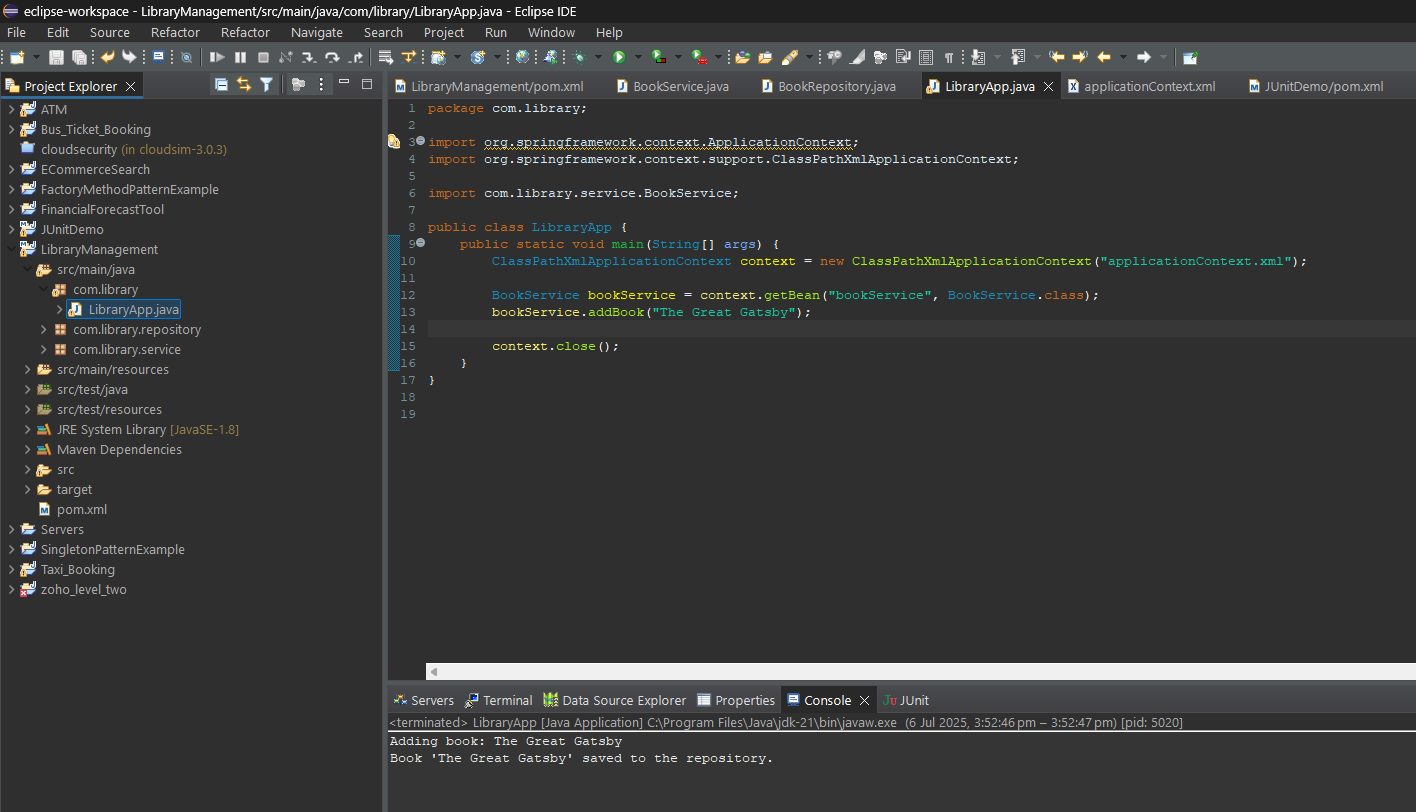
BookService bookService = context.getBean("bookService", BookService.class);

bookService.addBook("The Great Gatsby");

}

}

**RESULT :**



## Exercise 2: Implementing Dependency Injection Step 1: Create a New Maven Project

1) Open Eclipse.  
2) Go to File > New > Project…  
3) In the wizard, expand Maven, and select Maven Project.

**Step 2: Add JUnit Dependency**

1) In **Project Explorer**, expand your project.  
2️)Double-click **pom.xml** to open it.  
3️)Inside <dependencies>, paste this block:

<dependencies>

<!-- Spring Core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.30</version>

</dependency>

</dependencies>

**Step 3: Create the XML Configuration**Create applicationContext.xml file under **src/test/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

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

<!-- BookRepository Bean -->

<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>

**Step 4: Create Service ,Repository and Main Classes**  
**BookRepository.java**

**Code :**

package com.library.repository;

public class BookRepository {

public void saveBook(String title) {

System.out.println("Book '" + title + "' saved to the repository.");

}

**BookService.java**

**Code :**

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

System.out.println("Adding book: " + title);

bookRepository.saveBook(title);

}

}

**LibraryApp.java**

**Code :**

public class LibraryApp {

public static void main(String[] args) {

try (ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml")) {

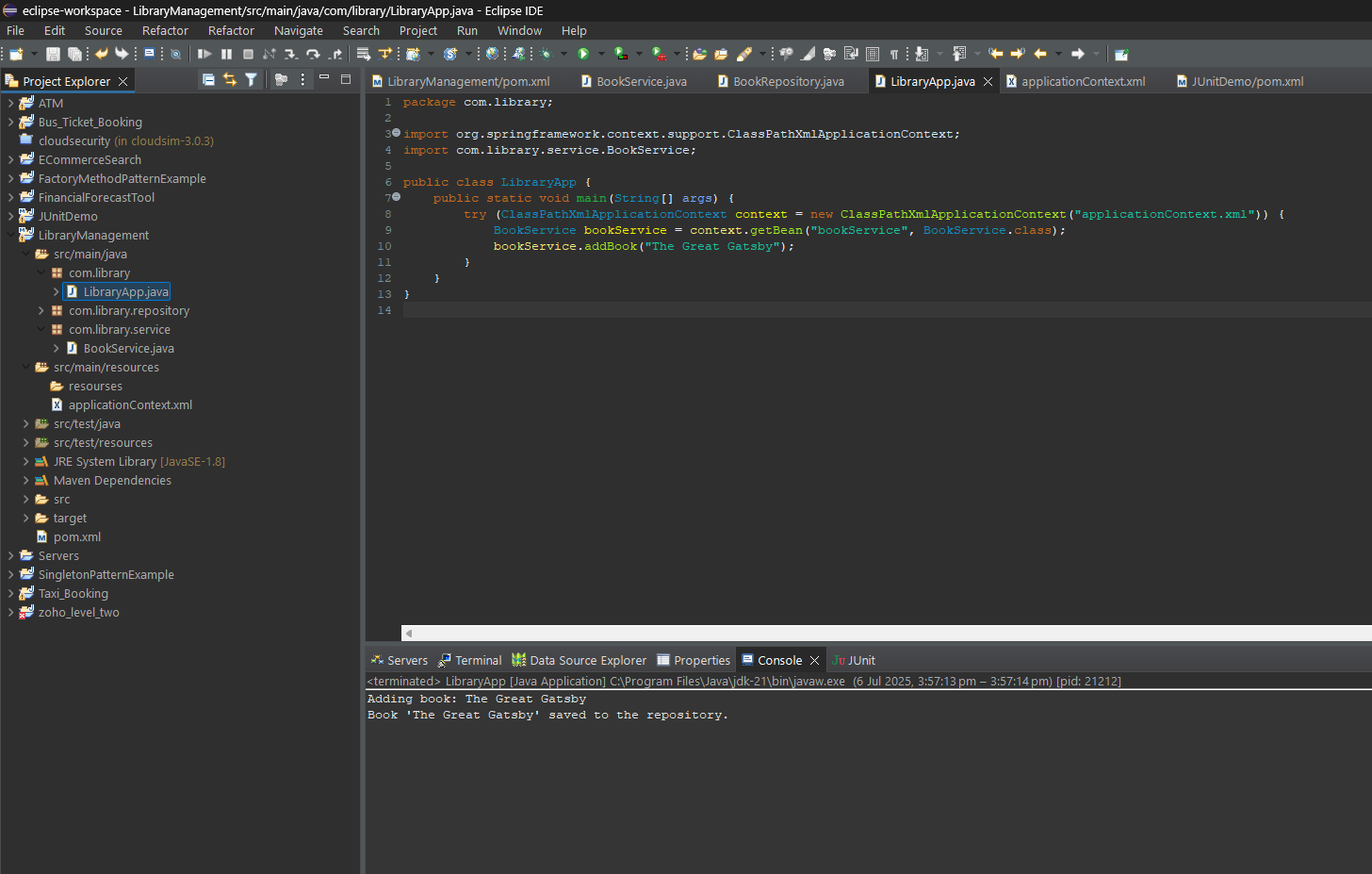
BookService bookService = context.getBean("bookService", BookService.class);

bookService.addBook("The Great Gatsby");

}

}

}**RESULT :**



## Exercise 4: Creating and Configuring a Maven Project

## Step 1: Create a New Maven Project

1) Open Eclipse.  
2) Go to File > New > Project…  
3) In the wizard, expand Maven, and select Maven Project.

**Step 2: Add Spring Dependencies**

1) In **Project Explorer**, expand your project.  
2️)Double-click **pom.xml** to open it.

<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

https://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>

<dependencies>

<!-- Spring Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.30</version>

</dependency>

<!-- Spring AOP -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.30</version>

</dependency>

<!-- Spring WebMVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.30</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>  
  
**Step 3: Update the Maven Project**1) Right-click your project  
2) Maven > Update Project...  
3) Check Force Update of Snapshots/Releases

**RESULT (Verify Dependencies) :**

