

Exercise 3: Creating and Configuring a Maven Project

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>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <name>LibraryManagement</name>
  <properties>
    <!-- Java version -->
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
    <!-- Encoding -->
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>
  <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 Web MVC -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-webmvc</artifactId>
  <version>5.3.34</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>${maven.compiler.source}</source>
        <target>${maven.compiler.target}</target>
      </configuration>
    </plugin>
  </plugins>
</build>
</project>
```

</build>

</project>

3. Define Service and Repository Classes

com/library/repository/BookRepository.java

```
package com.library.repository;
public class BookRepository {

    public void saveBook(String title) {

        System.out.println("BookRepository: Saving book titled \"\" + title + \"\" to
        the database.");

    }
}
```

com/library/service/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 save(String bookName) {

        System.out.println("Adding book...");

        bookRepository.save(bookName);

    }
}
```

4.Run the Application

```
com/library/LibraryApp.java
package com.library;
import com.library.service.BookService;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryApp {

    public static void main(String[]

        args) { Load Spring context

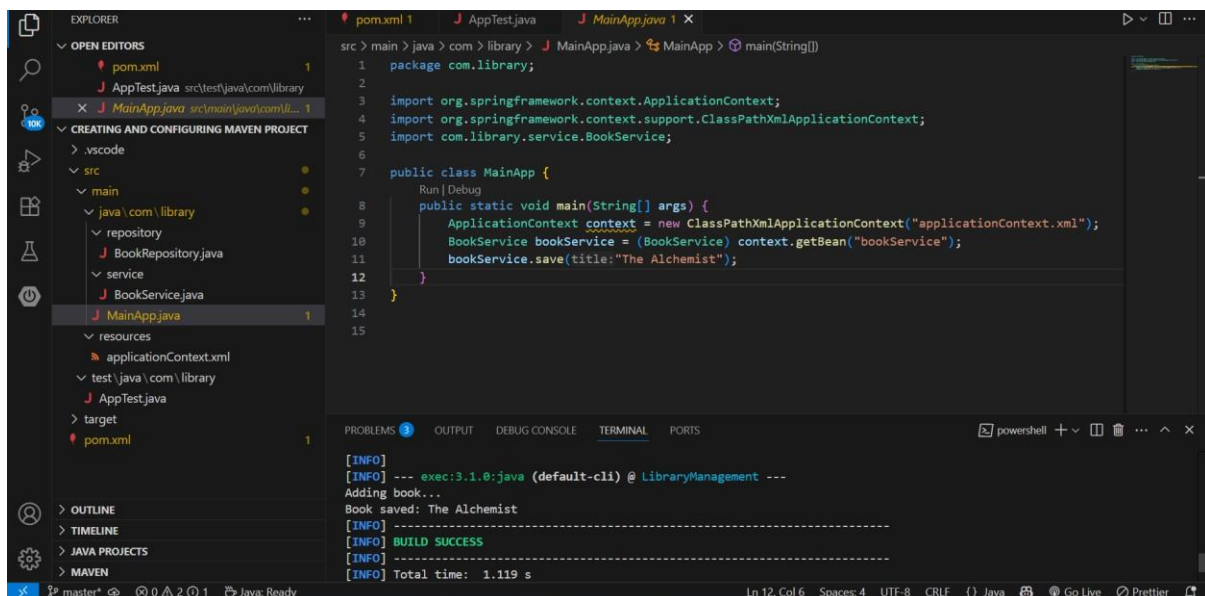
        from XML

        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        BookService bookService = context.getBean("bookService",
        BookService.class);
        bookService.addBook("Effective Java");

    }

}
```

Output

The screenshot shows an IDE with a project named 'LibraryManagement'. The Explorer view on the left shows the project structure with files like pom.xml, AppTest.java, MainApp.java, and applicationContext.xml. The Editor view shows the code for MainApp.java, which is a Java class that uses Spring's ApplicationContext to load a BookService and add a book. The Terminal view at the bottom shows the output of running the application, which includes the command 'exec:3.1.0:java (default-cli) @ LibraryManagement' and the output 'Adding book...' and 'Book saved: The Alchemist'. The terminal also shows the build status 'BUILD SUCCESS' and the total time '1.119 s'.

Compile:

mvn clean compile

