Exercise 2: Implementing Dependency Injection

BookService.java :

```java
package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    // Setter Injection
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }

    public void addBook(String title) {
        bookRepository.saveBook(title);
    }
}
```

App.java

```java
package com.library;

import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
```

```java
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {
    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");

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

        bookService.addBook("Clean Code");

        ((ClassPathXmlApplicationContext) context).close();
    }
}
```

Output:

Project ∨

LibraryManagement ~/IdeaProjects/LibraryManagement
> 📁 .idea
  📁 .mvn
∨ 📁 src
  ∨ 📁 main
    ∨ 📁 java
      ∨ 📁 com.library
          📁 App

(LibraryManagement)   🔵 App.java ×   </> applicationContext.xml   🔵 BookService.java   🔵 BookRepository.java

```java
7    public class App {
8        public static void main(String[] args) {
              ((ClassPathXmlApplicationContext) context).close();
15       }
16    }
17    |
```

Run   📁 App ×

```
/opt/homebrew/opt/java/libexec/openjdk.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Conter
Book saved: Clean Code

Process finished with exit code 0
```

Performance

Show Results   ⊙ 00:00

Process Terminated

LibraryManagement > src > main > java > com > library > 🔵 App   17:1   LF   UTF-8   4 spaces