

Exercise 1: Configuring a Basic Spring Application

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>

    <properties>
        <maven.compiler.source>24</maven.compiler.source>
        <maven.compiler.target>24</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <!-- Spring Core Context Dependency -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

Bookrepository.java:

```
package com.library.repository;

public class BookRepository {

    public String getBookInfo() {

        return "Spring in Action - Craig Walls";

    }

}
```

BookService.java:

```
package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    // Setter for Spring injection

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void displayBook() {

        System.out.println("Book: " + bookRepository.getBookInfo());

    }

}
```

MainApp.java:

```
package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {
```

```

public static void main(String[] args) {

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

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

    bookService.displayBook();

}
}

```

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">

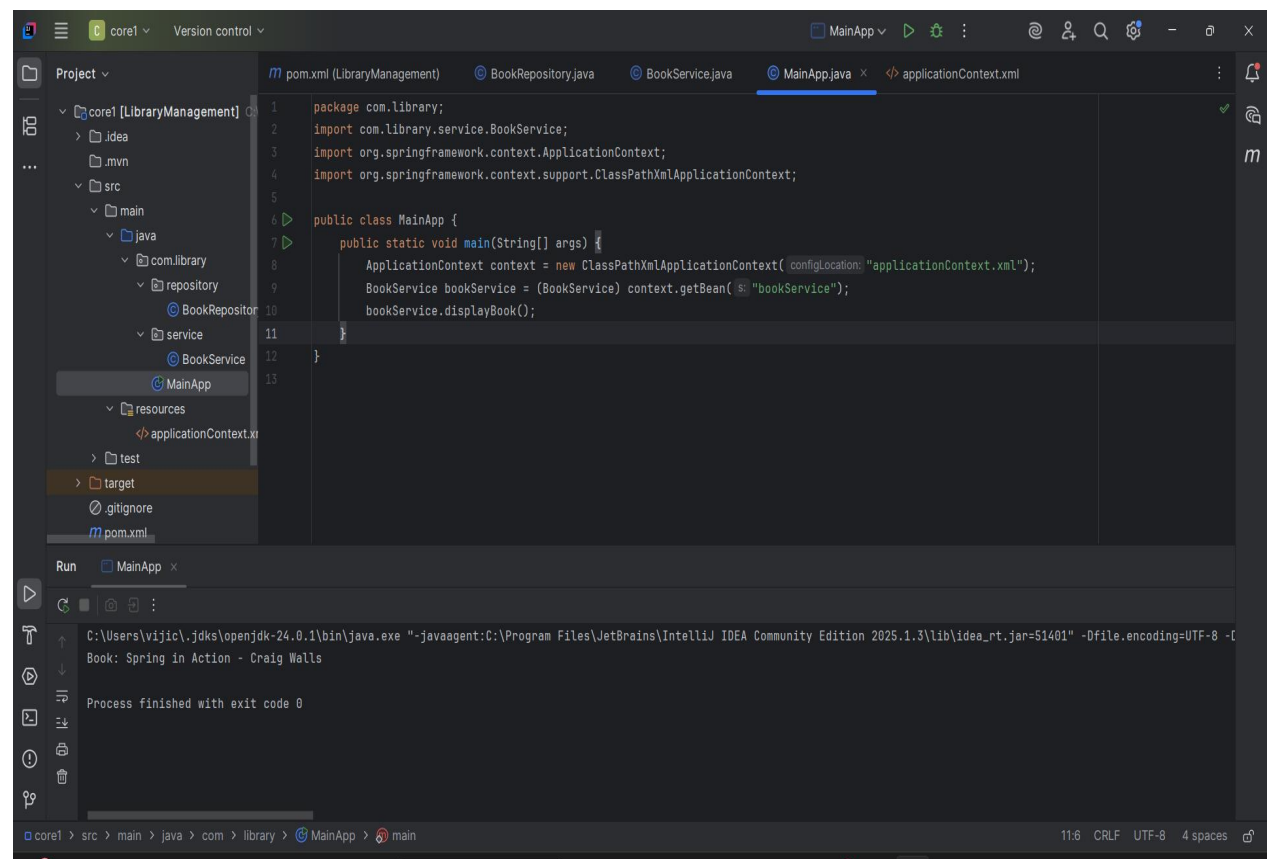
    <bean id="bookRepository" class="com.library.repository.BookRepository" />

    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository" />
    </bean>

</beans>

```

Output:



Exercise 2: Implementing Dependency Injection

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>
```

```

<properties>
  <maven.compiler.source>24</maven.compiler.source>
  <maven.compiler.target>24</maven.compiler.target>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>
<dependencies>
  <!-- Spring Core Context Dependency -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.34</version>
  </dependency>
</dependencies>
</project>

```

Bookrepository.java:

```

package com.library.repository;

import java.util.Arrays;
import java.util.List;

public class BookRepository {
  public List<String> getAllBooks() {
    return Arrays.asList(
      "Spring in Action - Craig Walls",
      "Effective Java - Joshua Bloch",
      "Clean Code - Robert C. Martin"
    );
  }
}

```

BookService.java:

```
package com.library.service;

import com.library.repository.BookRepository;

import java.util.List;

public class BookService {

    private BookRepository bookRepository;

    // Setter for DI

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    // Method to display all books

    public void displayAllBooks() {

        List<String> books = bookRepository.getAllBooks();

        System.out.println("Available Books:");

        for (String book : books) {

            System.out.println("- " + book);

        }

    }

}
```

MainApp.java:

```
package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {

    public static void main(String[] args) {
```

```

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

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

    // Display the list of all books

    bookService.displayAllBooks();

}

}

```

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">

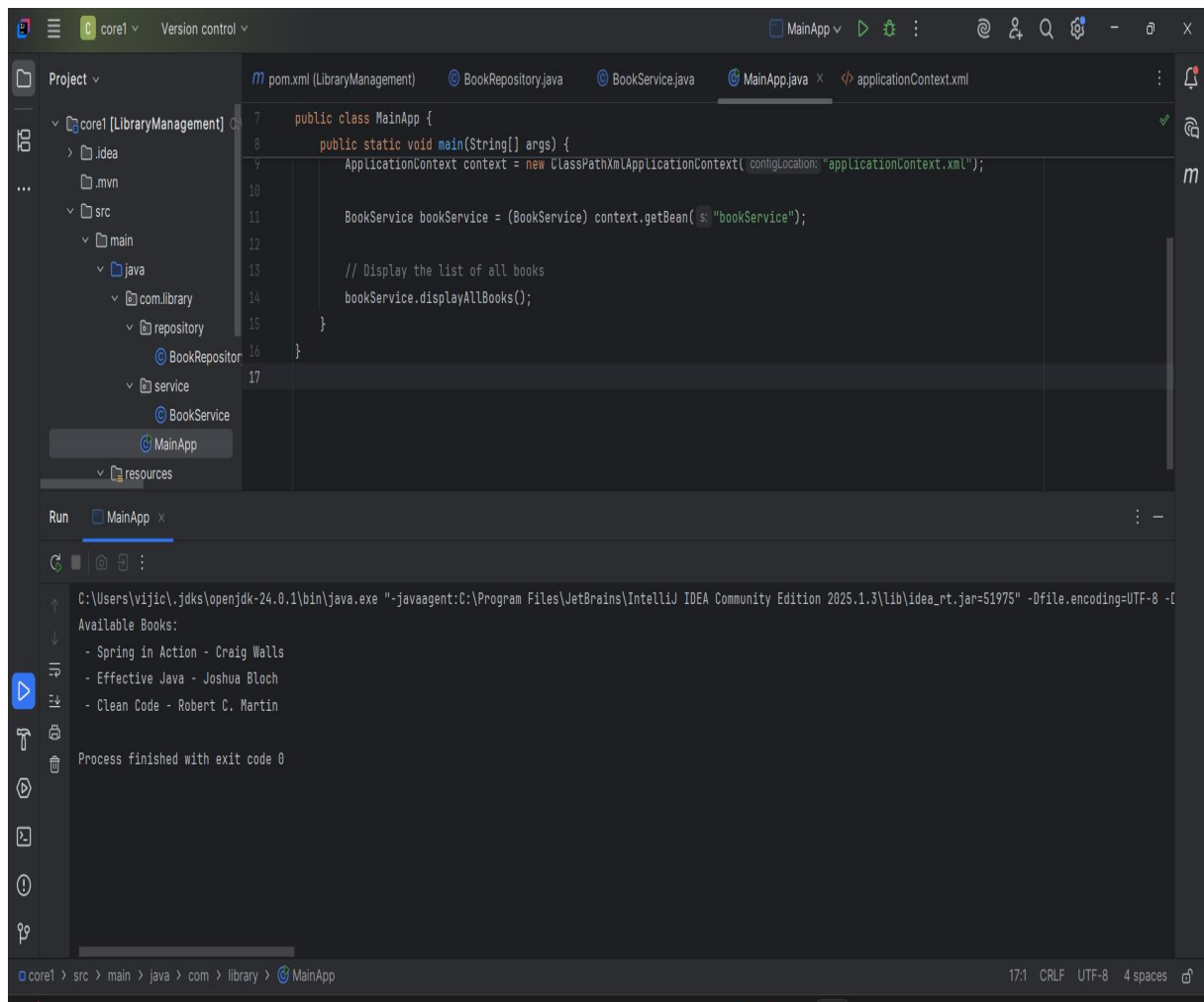
    <bean id="bookRepository" class="com.library.repository.BookRepository" />

    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository" />
    </bean>

</beans>

```

Output:



The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar shows the 'Run' button (a green play icon). The 'Project' view on the left shows the project structure: core1 [LibraryManagement] > src > main > java > com > library > repository > MainApp. The 'MainApp.java' file is open in the editor, showing the following code:

```
7 public class MainApp {  
8     public static void main(String[] args) {  
9         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
10  
11         BookService bookService = (BookService) context.getBean("bookService");  
12  
13         // Display the list of all books  
14         bookService.displayAllBooks();  
15     }  
16 }  
17
```

The 'Run' window at the bottom shows the execution output for 'MainApp'. The command executed is:

```
C:\Users\vijic\.jdk\openjdk-24.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_rt.jar=51975" -Dfile.encoding=UTF-8 -I
```

The output shows the list of available books:

```
Available Books:  
- Spring in Action - Craig Walls  
- Effective Java - Joshua Bloch  
- Clean Code - Robert C. Martin
```

The process finished with exit code 0.

The status bar at the bottom indicates the current file is 'MainApp' with a line number of 17, and the encoding is UTF-8 with 4 spaces.

Exercise 4: 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>
  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
    <!-- Spring Core + Context (for IoC and DI) -->
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context</artifactId>
      <version>5.3.34</version>
    </dependency>

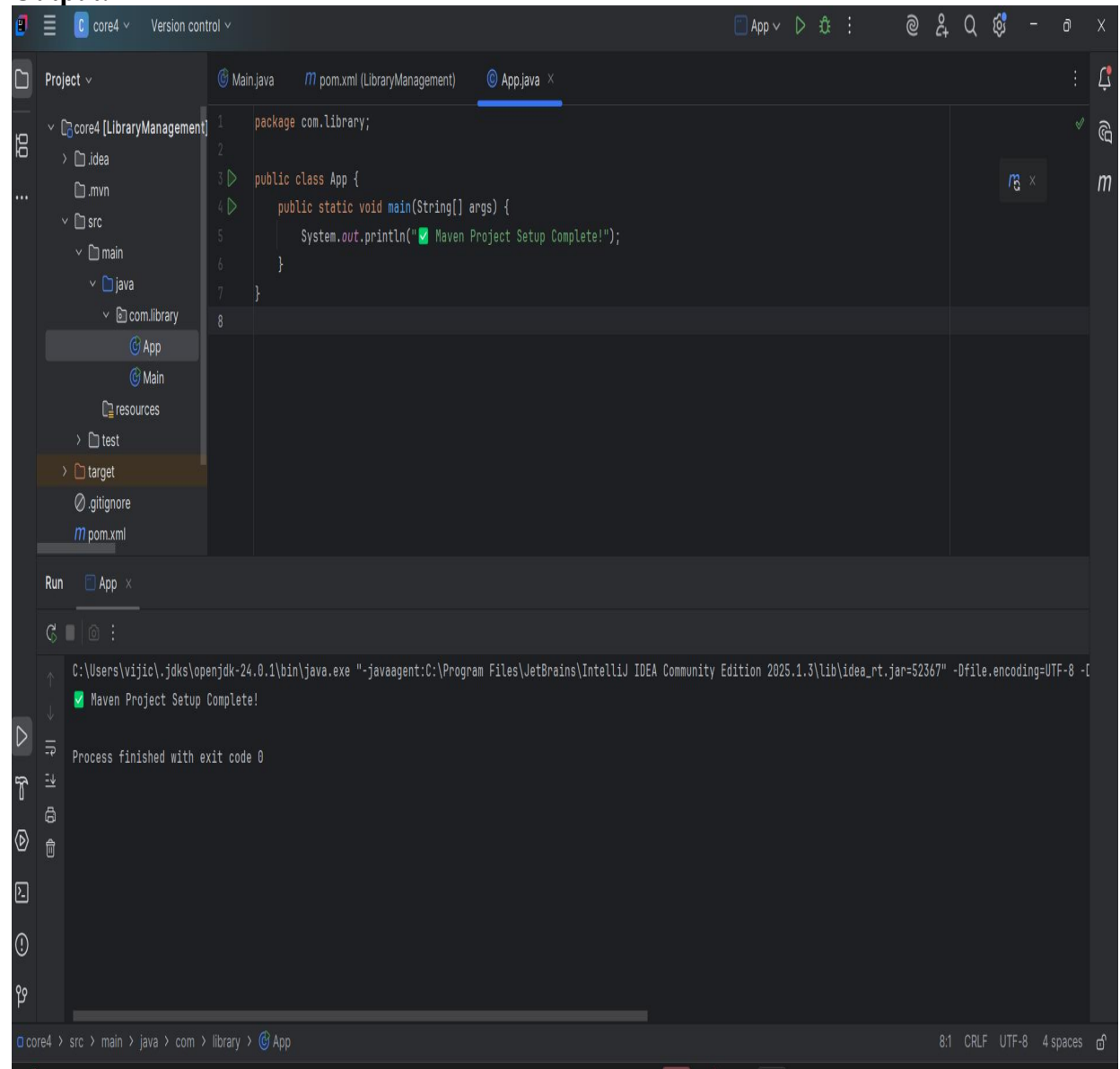
    <!-- Spring AOP -->
    <dependency>
      <groupId>org.springframework</groupId>
```

```
        <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>1.8</source>
                <target>1.8</target>
            </configuration>
        </plugin>
    </plugins>
</build>
</project>
```

App.java:

```
package com.library;
public class App {
    public static void main(String[] args) {
        System.out.println("✔ Maven Project Setup Complete!");
    }
}
```

Output:



Exercise 5: Configuring the Spring IoC Container

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>core5</artifactId>

    <version>1.0-SNAPSHOT</version>

    <properties>

        <maven.compiler.source>24</maven.compiler.source>

        <maven.compiler.target>24</maven.compiler.target>

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

    </properties>

    <dependencies>

        <!-- Spring Context (Core + IoC container) -->

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-context</artifactId>

            <version>5.3.34</version>

        </dependency>

        <!-- Optional but useful for future exercises -->
```

```

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-beans</artifactId>
    <version>5.3.34</version>
</dependency>
</dependencies>
</project>

```

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 for BookRepository -->
    <bean id="bookRepository" class="com.library.repository.BookRepository" />
    <!-- Bean for BookService with dependency injection -->
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
</beans>

```

Bookrepository.java:

```

package com.library.repository;

public class BookRepository {
    public String getBookTitle() {
        return "The Great Gatsby";
    }
}

```

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 displayBookTitle() {

        String title = bookRepository.getBookTitle();

        System.out.println("Book Title: " + title);

    }

}
```

LibraryManagementApp.java:

```
package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryManagementApp {

    public static void main(String[] args) {

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

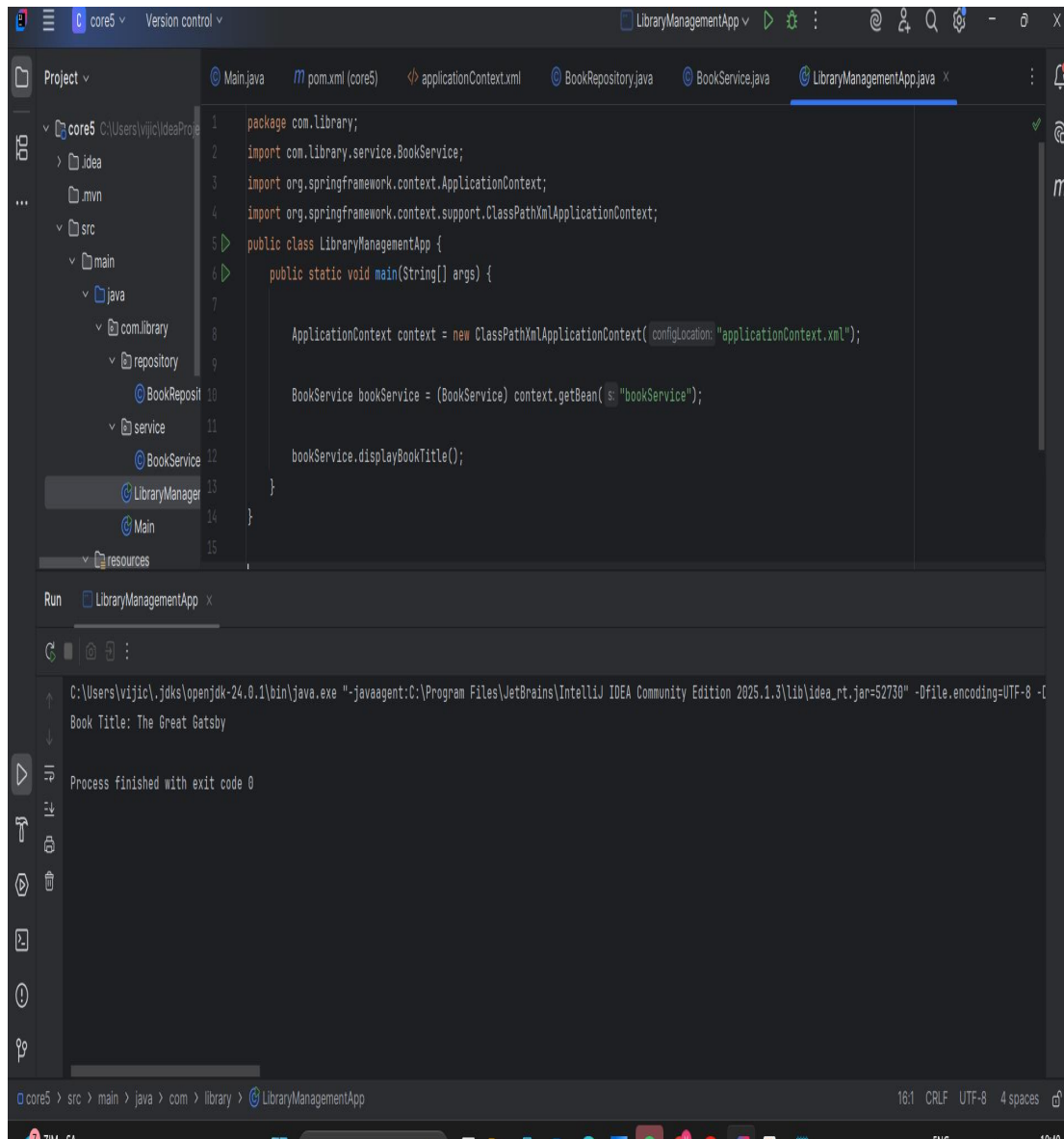
        BookService bookService = (BookService) context.getBean("bookService");

        bookService.displayBookTitle();

    }

}
```

Output:



The screenshot displays an IDE window with the following components:

- Project Explorer:** Shows the project structure for 'core5'. The 'src/main/java/com/library' package is expanded, showing 'BookRepository', 'BookService', 'LibraryManager', and 'Main'.
- Code Editor:** Displays the code for 'LibraryManagementApp.java'. The code is as follows:

```
1 package com.library;
2 import com.library.service.BookService;
3 import org.springframework.context.ApplicationContext;
4 import org.springframework.context.support.ClassPathXmlApplicationContext;
5 public class LibraryManagementApp {
6     public static void main(String[] args) {
7
8         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
9
10        BookService bookService = (BookService) context.getBean("bookService");
11
12        bookService.displayBookTitle();
13    }
14 }
15
```
- Run Console:** Shows the execution of 'LibraryManagementApp'. The output is:

```
C:\Users\vijic\.jdk\openjdk-24.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_rt.jar=52730" -Dfile.encoding=UTF-8 -l
Book Title: The Great Gatsby

Process finished with exit code 0
```
- Status Bar:** Indicates the file is 'core5 > src > main > java > com > library > LibraryManagementApp' with a line length of 16:1, CRLF line endings, UTF-8 encoding, and 4 spaces.

Exercise 6: Configuring Beans with Annotations

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>

    <properties>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
        <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    </properties>
    <dependencies>
        <!-- Spring Core Dependency -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>

        <!-- Spring AOP (optional but commonly used) -->
```



```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-aop</artifactId>
  <version>5.3.34</version>
</dependency>
<!-- Spring Web MVC (optional if not using web features) -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-webmvc</artifactId>
  <version>5.3.34</version>
</dependency>
<!-- Logging dependency for Spring -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.36</version>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-simple</artifactId>
  <version>1.7.36</version>
</dependency>

</dependencies>

<build>
  <plugins>
```

```
<!-- Compiler plugin for Java 8 -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.10.1</version>
  <configuration>
    <source>1.8</source>
    <target>1.8</target>
  </configuration>
</plugin>
</plugins>
</build>
</project>
```

BookRepository.java:

```
package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository
public class BookRepository {

    public String getBookTitle() {
        return "The Alchemist";
    }

}
```

BookService.java:

```
package com.library.service;

import com.library.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service

public class BookService {

    private BookRepository bookRepository;

    @Autowired // Spring injects the bean automatically
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }

    public void displayBookTitle() {
        System.out.println("Book Title: " + bookRepository.getBookTitle());
    }

}
```

LibraryManagementApp.java:

```
package com.library;

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

```

public class LibraryManagementApp {
    public static void main(String[] args) {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        BookService bookService = context.getBean(BookService.class);
        bookService.displayBookTitle();
    }
}

```

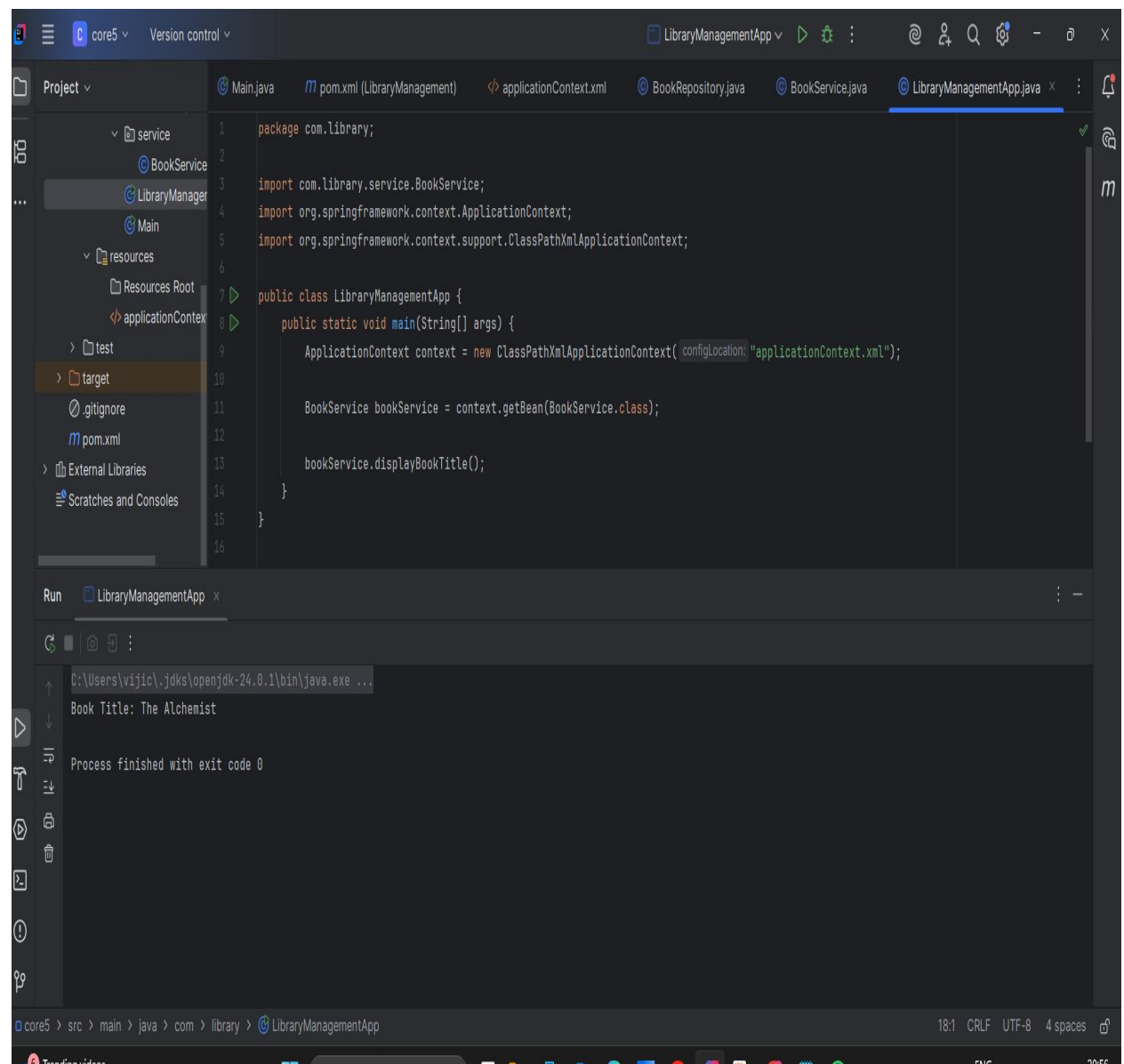
ApplicationContext.xml:

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    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
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-
context.xsd">
    <!-- Enable annotation scanning -->
    <context:component-scan base-package="com.library" />
</beans>

```

Output:



```
1 package com.library;
2
3 import com.library.service.BookService;
4 import org.springframework.context.ApplicationContext;
5 import org.springframework.context.support.ClassPathXmlApplicationContext;
6
7 public class LibraryManagementApp {
8     public static void main(String[] args) {
9         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
10
11         BookService bookService = context.getBean(BookService.class);
12
13         bookService.displayBookTitle();
14     }
15 }
16
```

Run LibraryManagementApp

```
C:\Users\vijic\jdk\openjdk-24.0.1\bin\java.exe ...
Book Title: The Alchemist

Process finished with exit code 0
```

core5 > src > main > java > com > library > LibraryManagementApp 18:1 CRLF UTF-8 4 spaces

Exercise 7: Implementing Constructor and Setter Injection

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>LibraryManagementApp</artifactId>
    <version>1.0-SNAPSHOT</version>
    <properties>
        <java.version>17</java.version> <!-- Use 17 or lower -->
    </properties>
    <dependencies>
        <!-- Spring Context for core container features -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
        <!-- Optional: Logging support -->
        <dependency>
            <groupId>commons-logging</groupId>
            <artifactId>commons-logging</artifactId>
```

```
        <version>1.2</version>
    </dependency>
</dependencies>

<build>

    <plugins>

        <!-- Compiler Plugin -->

        <plugin>

            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.10.1</version>
            <configuration>
                <source>${java.version}</source>
                <target>${java.version}</target>
            </configuration>
        </plugin>
    </plugins>
</build>
</project>
```

BookRepository.java:

```
package com.library.repository;

public class BookRepository {

    public String getBookTitle() {
        return "Spring in Action";
    }
}
```

```
}
```

BookService.java:

```
package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    // Constructor
    public BookService(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

        System.out.println("Constructor Injection used.");

    }

    // Setter
    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

        System.out.println("Setter Injection used.");

    }

    public void displayBookTitle() {

        System.out.println("Book Title: " + bookRepository.getBookTitle());

    }

}
```


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">

    <!-- BookRepository Bean -->

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

    <!-- BookService Bean using constructor injection -->

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

        <constructor-arg ref="bookRepository"/>

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

    </bean>

</beans>
```

LibraryManagementApp.java:

```
package com.library;

import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryManagementApp {
```

```

public static void main(String[] args) {

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

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

    bookService.displayBookTitle();

}
}

```

Output:

The screenshot displays the IntelliJ IDEA IDE with a project named 'core5'. The project structure on the left shows a package 'com.library' containing 'BookRepository', 'BookService', and 'LibraryManagementApp'. The main editor shows the 'LibraryManagementApp.java' file with the following code:

```

1 package com.library;
2
3 import com.library.service.BookService;
4 import org.springframework.context.ApplicationContext;
5 import org.springframework.context.support.ClassPathXmlApplicationContext;
6
7 public class LibraryManagementApp {
8     public static void main(String[] args) {
9         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
10
11         BookService bookService = (BookService) context.getBean("bookService");
12         bookService.displayBookTitle();
13     }
14 }

```

The Run tab at the bottom shows the execution output:

```

C:\Users\vijic\.jdk\openjdk-24.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_rt.jar=53724" -Dfile.encoding=UTF-8 -I
Constructor Injection used.
Setter Injection used.
Book Title: Spring in Action
Process finished with exit code 0

```

The status bar at the bottom indicates the file encoding is UTF-8 and the line length is 15:1.

Exercise 8: Implementing Basic AOP with Spring

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>
    <dependencies>
        <!-- Spring Context (IoC container) -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
        <!-- Spring AOP for aspect-oriented programming -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
            <version>5.3.34</version>
        </dependency>
        <!-- AspectJ for weaving AOP proxies -->
        <dependency>
            <groupId>org.aspectj</groupId>
```

```
        <artifactId>aspectjweaver</artifactId>
        <version>1.9.21.1</version>
    </dependency>
    <!-- Spring Core (dependency of context and aop) -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>5.3.34</version>
    </dependency>
</dependencies>
<build>
    <plugins>
        <!-- Maven Compiler Plugin for Java 1.8 -->
        <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>
```

LibraryManagementApp.Java:

```
package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryManagementApp {

    public static void main(String[] args) {

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

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

        bookService.displayBookTitle();

    }

}
```

applicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

    xmlns:aop="http://www.springframework.org/schema/aop"

    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

        http://www.springframework.org/schema/aop

        http://www.springframework.org/schema/aop/spring-aop.xsd">

    <!-- Enable AOP Proxying -->

    <aop:config>
```

```

    <aop:aspect ref="loggingAspect">
        <aop:pointcut id="allMethods" expression="execution(*
com.library.service.*.*(..))" />
        <aop:before method="beforeAdvice" pointcut-ref="allMethods" />
        <aop:after method="afterAdvice" pointcut-ref="allMethods" />
    </aop:aspect>
</aop:config>

<!-- Aspect Bean -->
<bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />
<!-- Repository and Service -->
<bean id="bookRepository" class="com.library.repository.BookRepository" />
<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;

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

    public void displayBookTitle() {
        String title = bookRepository.findBookTitle();
    }
}

```

```
        System.out.println("Book Title: " + title);
    }
}
```

BookRepository.java:

```
package com.library.repository;

public class BookRepository {

    public String findBookTitle() {

        return "Effective Java";

    }

}
```

LoggingAspect.java:

```
package com.library.aspect;

import org.aspectj.lang.JoinPoint;

public class LoggingAspect {

    public void beforeAdvice(JoinPoint joinPoint) {

        System.out.println("[LOG - BEFORE] Method called: " +
            joinPoint.getSignature().getName());

    }

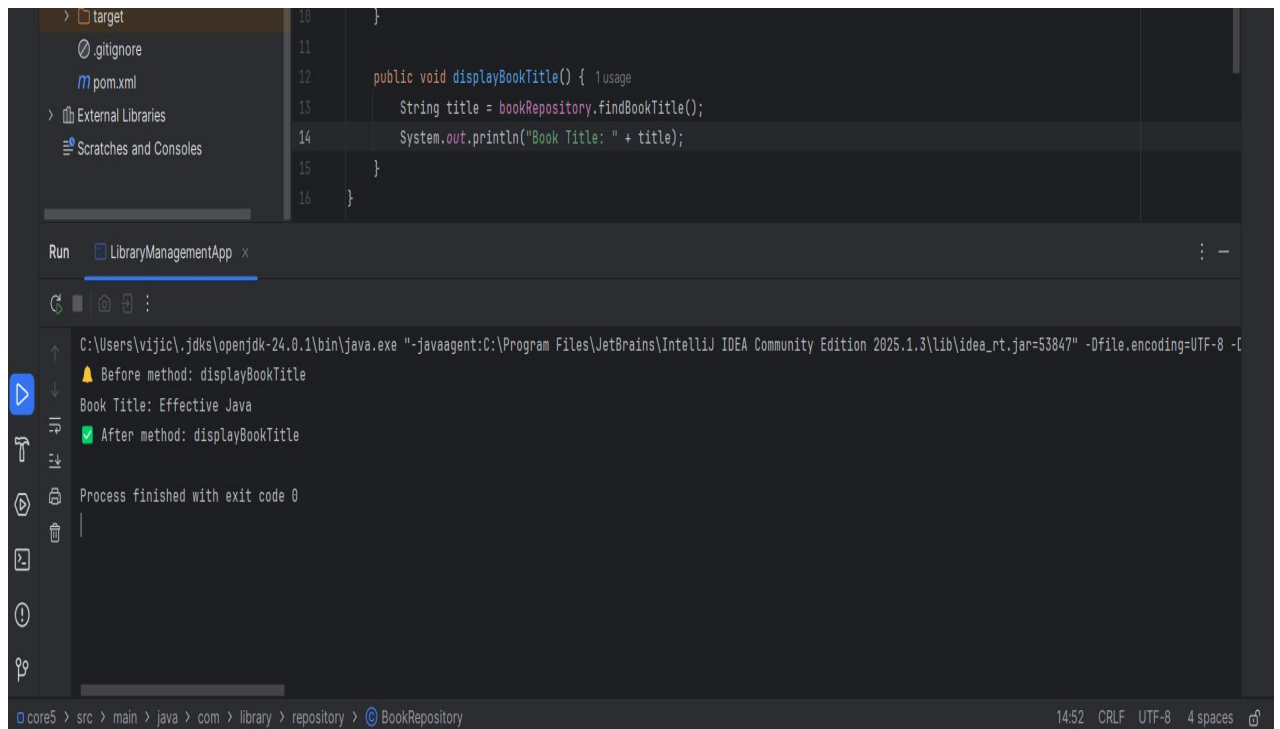
    public void afterAdvice(JoinPoint joinPoint) {

        System.out.println("[LOG - AFTER] Method finished: " +
            joinPoint.getSignature().getName());

    }

}
```

Output:



The screenshot displays the IntelliJ IDEA interface. The top pane shows a Java file with the following code:

```
10 }  
11  
12 public void displayBookTitle() { 1 usage  
13     String title = bookRepository.findBookTitle();  
14     System.out.println("Book Title: " + title);  
15 }  
16 }
```

The bottom pane shows the Run configuration for 'LibraryManagementApp' and the execution output. The output includes the following lines:

```
C:\Users\vijic\jdk\openjdk-24.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_rt.jar=53847" -Dfile.encoding=UTF-8 -I  
⚠ Before method: displayBookTitle  
Book Title: Effective Java  
✅ After method: displayBookTitle  
Process finished with exit code 0
```

The status bar at the bottom indicates the current file is 'BookRepository' in the 'repository' package, with a timestamp of 14:52 and encoding of UTF-8.