Exercise 1: Configuring a Basic Spring Application

pom.xml

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.34</version>
  </dependency>
</dependencies>
```

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>
```

BookRepository.java

```
package com.library.repository;
public class BookRepository {
  public String getBook() {
    return "Effective Java by Joshua Bloch";
  } }
```

BookService.java

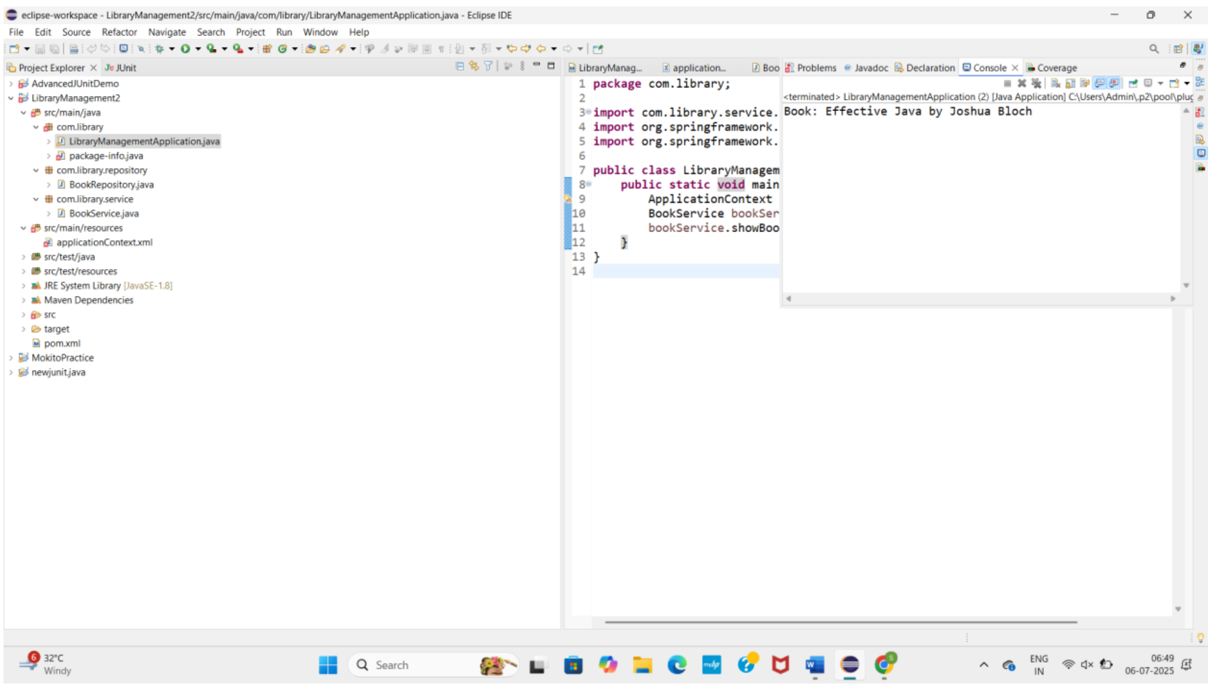
```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
  private BookRepository bookRepository;
```

```
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }
    public void showBook() {
        System.out.println("Book: " + bookRepository.getBook());
    }
}
```

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.showBook();
    } }
```

OUTPUT:



eclipse-workspace - LibraryManagement2/src/main/java/com/library/LibraryManagementApplication.java - Eclipse IDE

```
package com.library;

import com.library.service.
import org.springframework.
import org.springframework.

public class LibraryManagem
    public static void main
        ApplicationContext
        BookService bookSer
        bookService.showBoo
    }
}
```

```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\plug
Book: Effective Java by Joshua Bloch
```

Exercise 2: Implementing Dependency Injection

BookRepository.java

```
package com.library.repository;

public class BookRepository {

   public void fetchBooks() {

      System.out.println("Fetching books from database...");

   }

}
```

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 display() {

      System.out.println("BookService is working.");

      bookRepository.fetchBooks();

   }

}
```

applicationContext.xml

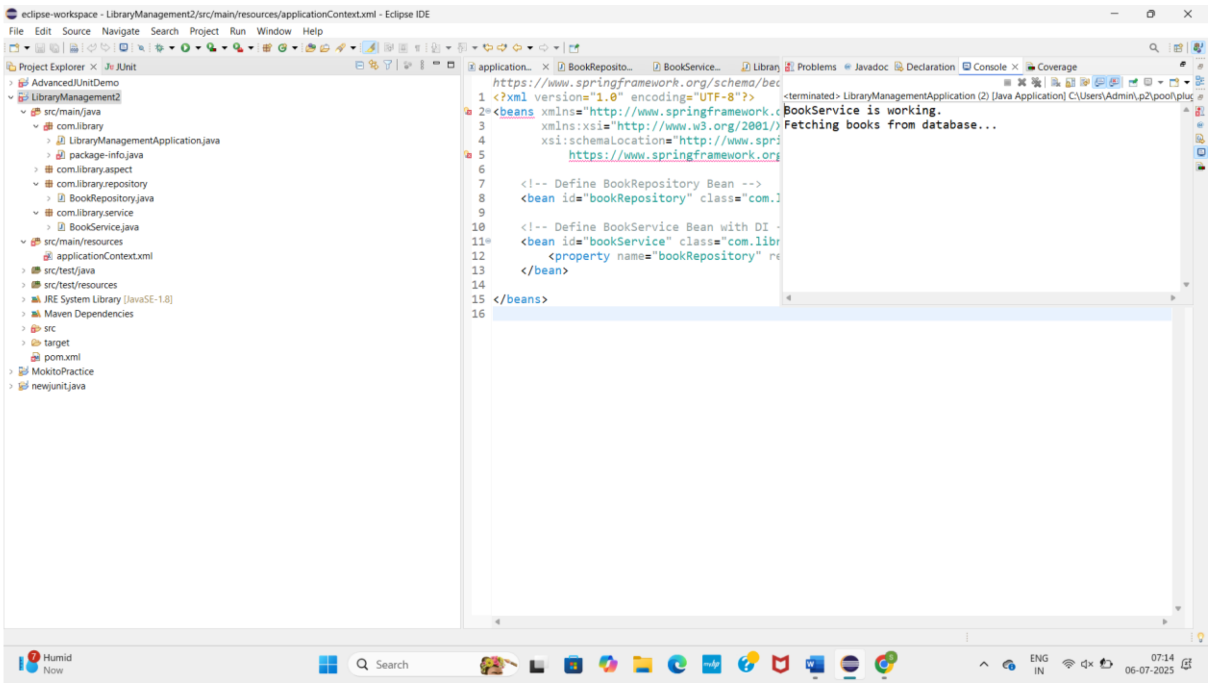
```
<?xml version="1.0" encoding="UTF-8"?>

<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

        https://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>
```

LibraryManagementApplication.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.showBook();

    }

}
```

OUTPUT:



```
https://www.springframework.org/schema/bec
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.c
       xmlns:xsi="http://www.w3.org/2001/>
       xsi:schemaLocation="http://www.spri
           https://www.springframework.org

    <!-- Define BookRepository Bean -->
    <bean id="bookRepository" class="com.l

    <!-- Define BookService Bean with DI -
    <bean id="bookService" class="com.libr
        <property name="bookRepository" re
    </bean>

</beans>
```

```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\plug
BookService is working.
Fetching books from database...
```

Exercise 3: Spring AOP Logging

LoggingAspect.java

```
package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;

public class LoggingAspect {

   @Around("execution(* com.library.service.BookService.*(..))")

   public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

      System.out.println("[LOG] Method " + joinPoint.getSignature().getName() + " started");

      long start = System.currentTimeMillis();

      Object result = joinPoint.proceed();

      long end = System.currentTimeMillis();

      System.out.println("[LOG] Method " + joinPoint.getSignature().getName() + " ended in " + (end -
start) + "ms");

      return result;

   }

}
```

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

      xsi:schemaLocation="

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

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

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

       https://www.springframework.org/schema/context/spring-context.xsd

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

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

      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

   <context:component-scan base-package="com.library"/>

   <aop:aspectj-autoproxy/>

   <bean id="loggingAspect" class="com.library.aspect.LoggingAspect"/>

</beans>
```

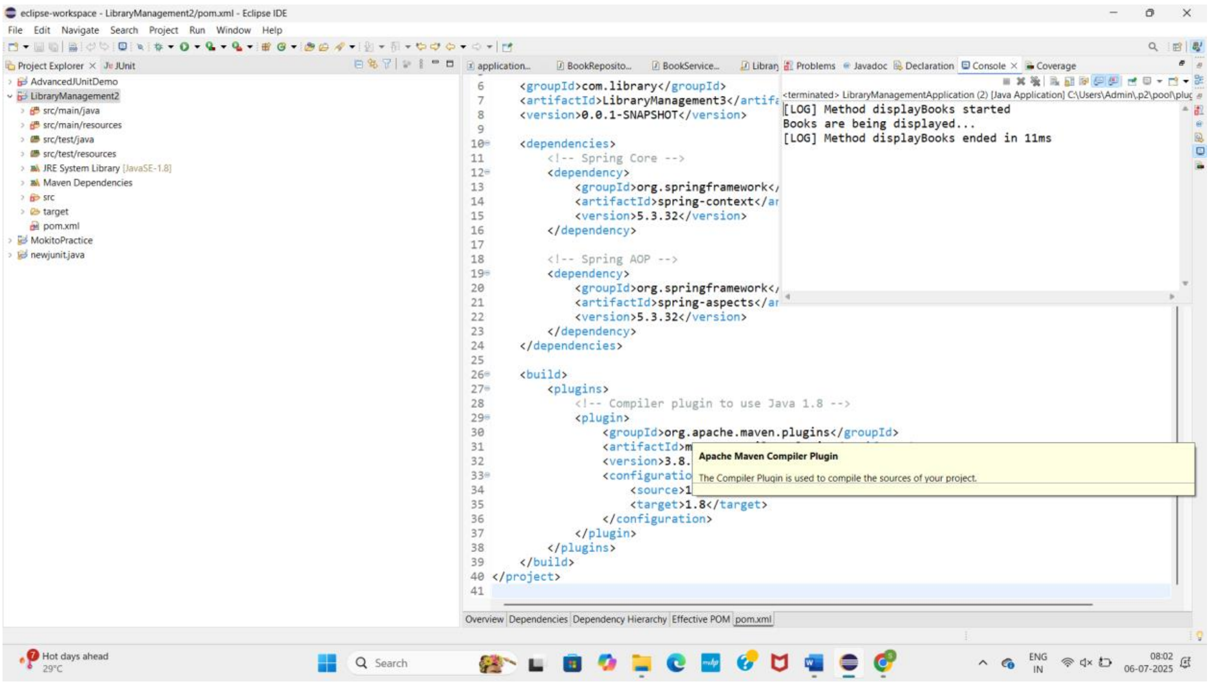
BookService.java

```java
package com.library.service;

import org.springframework.stereotype.Service;

public class BookService {

    public void displayBooks() {

        System.out.println("Books are being displayed...");

    }

}
```

LibraryManagementApplication.java

```java
package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

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

        bookService.displayBooks();

    } }
```

OUTPUT:



```xml
    <groupId>com.library</groupId>
    <artifactId>LibraryManagement3</artifa
    <version>0.0.1-SNAPSHOT</version>

    <dependencies>
        <!-- Spring Core -->
        <dependency>
            <groupId>org.springframework</
            <artifactId>spring-context</ar
            <version>5.3.32</version>
        </dependency>

        <!-- Spring AOP -->
        <dependency>
            <groupId>org.springframework</
            <artifactId>spring-aspects</ar
            <version>5.3.32</version>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- Compiler plugin to use Java 1.8 -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>m
                <version>3.8.
                <configuratio
                    <source>1
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

Apache Maven Compiler Plugin

The Compiler Plugin is used to compile the sources of your project.

```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\pluc
[LOG] Method displayBooks started
Books are being displayed...
[LOG] Method displayBooks ended in 11ms
```

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
https://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <groupId>com.library</groupId>
   <artifactId>LibraryManagement3</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <dependencies>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring-context</artifactId>
         <version>5.3.32</version>
      </dependency>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring-aspects</artifactId>
         <version>5.3.32</version>
      </dependency>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring-webmvc</artifactId>
         <version>5.3.32</version>
      </dependency>
   </dependencies>
   <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>
</project>
```

Exercise 5: Configuring the Spring IoC Container

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
        https://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>
```

BookRepository.java

```
package com.library.repository;
public class BookRepository {
   public void fetchBooks() {
      System.out.println("📚 BookRepository: Fetching books from DB...");
   }}
```

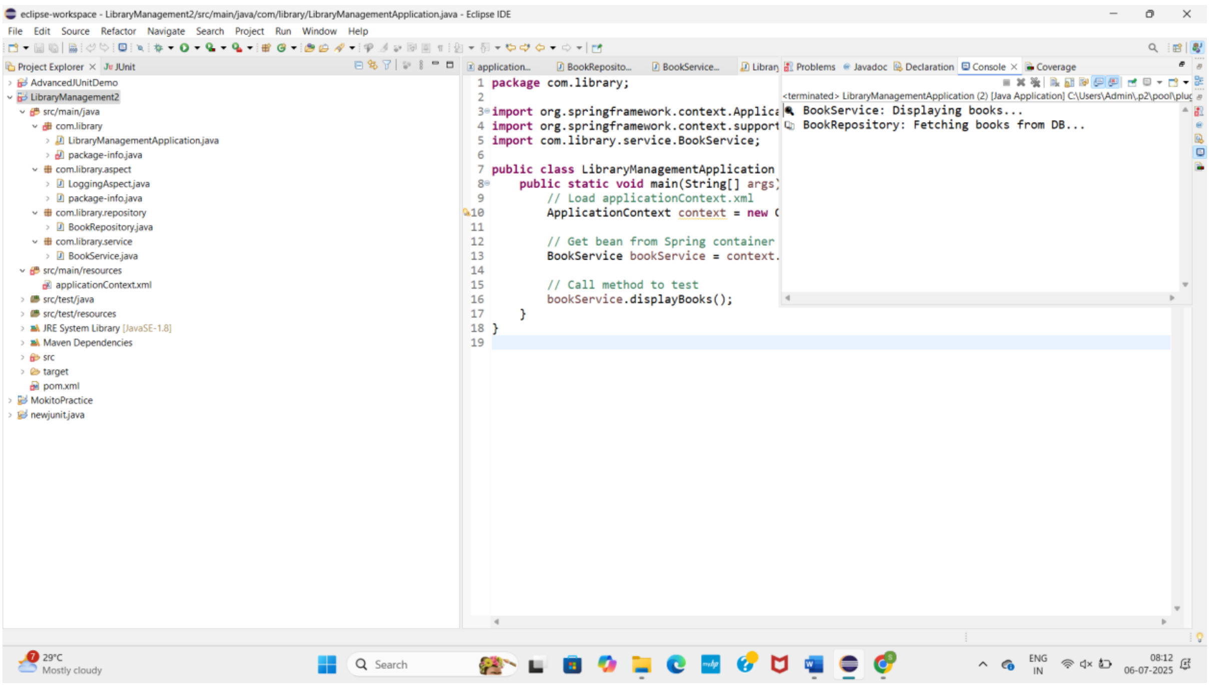
BookService.java

```
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 displayBooks() {

        System.out.println("🔍 BookService: Displaying books...");

        bookRepository.fetchBooks();

    }}
```

LibraryManagementApplication.java

```
package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

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

        bookService.displayBooks();

    }}
```

OUTPUT:



```
package com.library;

import org.springframework.context.Applica
import org.springframework.context.support
import com.library.service.BookService;

public class LibraryManagementApplication
    public static void main(String[] args)
        // Load applicationContext.xml
        ApplicationContext context = new C

        // Get bean from Spring container
        BookService bookService = context.

        // Call method to test
        bookService.displayBooks();
    }
}
```

```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\pluc
🔍 BookService: Displaying books...
📚 BookRepository: Fetching books from DB...
```

Exercise 6: Configuring Beans with Annotations

applicationContext.xml

```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">
  <context:component-scan base-package="com.library" />
</beans>
```

BookRepository.java

```java
package com.library.repository;
import org.springframework.stereotype.Repository;
public class BookRepository {
  public void printBooks() {
    System.out.println("📚 Book list: 'Spring in Action', 'Head First Java', 'Clean Code'");
  }
}
```

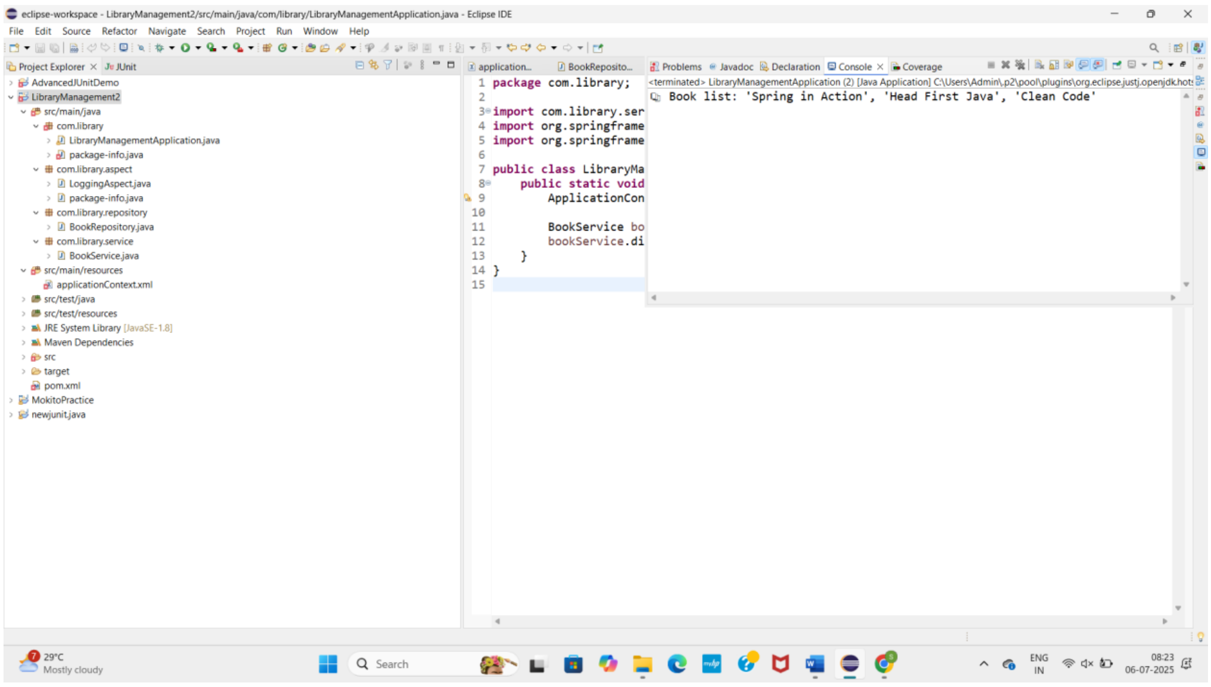
BookService.java

```java
package com.library.service;
import com.library.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
public class BookService {
  private BookRepository bookRepository;
  public void displayBooks() {
    bookRepository.printBooks();
  }
}
```

LibraryManagementApplication.java

```java
package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

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

        bookService.displayBooks();

    }

}
```

OUTPUT:



```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hot
Book list: 'Spring in Action', 'Head First Java', 'Clean Code'
```

Exercise 7: Constructor and Setter Injection in Spring (XML-based)

BookRepository.java

```java
package com.library.repository;

public class BookRepository {

    public void fetchBooks() {

        System.out.println("🗄 Fetching books from database...");
    }
}
```

BookService.java

```java
package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    public BookService(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

        System.out.println("🔧 Constructor Injection: BookRepository injected!");

    }

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

        System.out.println("□ Setter Injection: BookRepository injected!");

    }

    public void displayBooks() {

        bookRepository.fetchBooks();

    }

}
```

applicationContext.xml

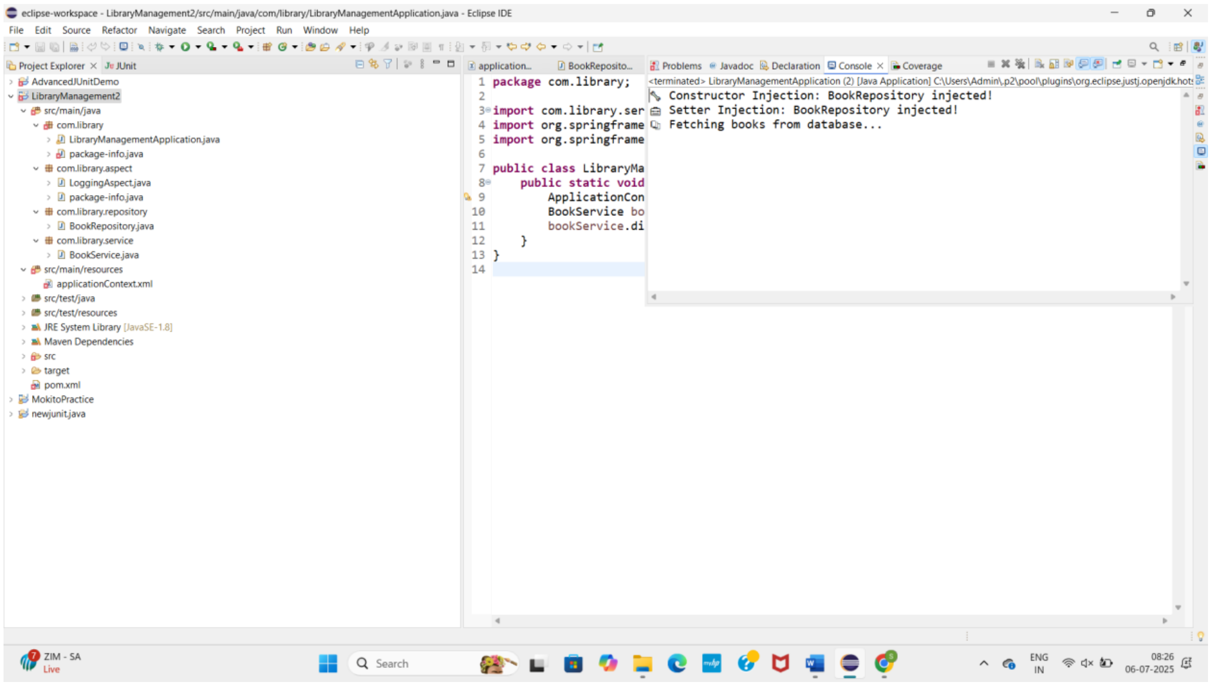
```xml
<?xml version="1.0" encoding="UTF-8"?>

<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

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

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

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

        <constructor-arg ref="bookRepository" />

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

    </bean>
```

```
</beans>
```

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();

    }

}
```

OUTPUT:



eclipse-workspace - LibraryManagement2/src/main/java/com/library/LibraryManagementApplication.java - Eclipse IDE

```
package com.library;

import com.library.ser
import org.springframe
import org.springframe

public class LibraryMa
    public static void
        ApplicationCon
        BookService bo
        bookService.di
    }
}
```

```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hots
Constructor Injection: BookRepository injected!
Setter Injection: BookRepository injected!
Fetching books from database...
```

Exercise 8: Implementing Basic AOP with Spring (XML-Based)

LoggingAspect.java

```
package com.library.aspect;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;

public class LoggingAspect {

   @Before("execution(* com.library.service.BookService.displayBooks(..))")

   public void logBefore() {

      System.out.println("📋 [LOG - BEFORE] About to display books...");

   }

   @After("execution(* com.library.service.BookService.displayBooks(..))")

   public void logAfter() {

      System.out.println("📋 [LOG - AFTER] Finished displaying books.");

   }

}
```

applicationContext.xml

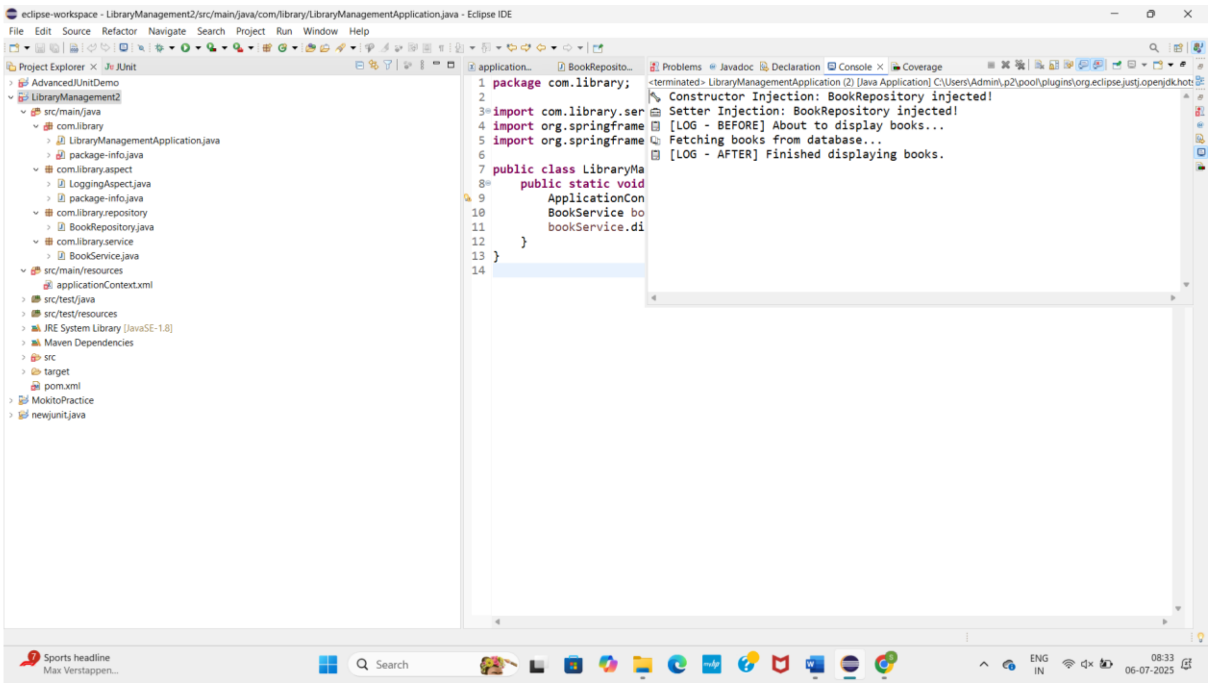
```
<?xml version="1.0" encoding="UTF-8"?>

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

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

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

      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

        http://www.springframework.org/schema/context
https://www.springframework.org/schema/context/spring-context.xsd

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

   <context:component-scan base-package="com.library" />

   <aop:aspectj-autoproxy />

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

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

      <constructor-arg ref="bookRepository"/>

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

   </bean>

   <bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />
```

```
</beans>
```

LibraryManagementApplication.java

```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();

    }

}
```

OUTPUT:



```
<terminated> LibraryManagementApplication (2) [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hot
Constructor Injection: BookRepository injected!
Setter Injection: BookRepository injected!
[LOG - BEFORE] About to display books...
Fetching books from database...
[LOG - AFTER] Finished displaying books.
```

Exercise 9: Creating a Spring Boot Application

application.properties

```
spring.datasource.url=jdbc:h2:mem:librarydb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa
spring.datasource.password=
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.h2.console.enabled=true
spring.jpa.hibernate.ddl-auto=update
```

Book Entity

```
package com.library.model;
import jakarta.persistence.*;
public class Book {
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   private String title;
   private String author;
   public Book() {}
   public Book(String title, String author) {
      this.title = title;
      this.author = author;
   }
   public Long getId() { return id; }
   public void setId(Long id) { this.id = id; }
   public String getTitle() { return title; }
   public void setTitle(String title) { this.title = title; }

   public String getAuthor() { return author; }
   public void setAuthor(String author) { this.author = author; }
}
```

Book Repository

```
package com.library.repository;
import com.library.model.Book;
import org.springframework.data.jpa.repository.JpaRepository;
public interface BookRepository extends JpaRepository<Book, Long> {}
```

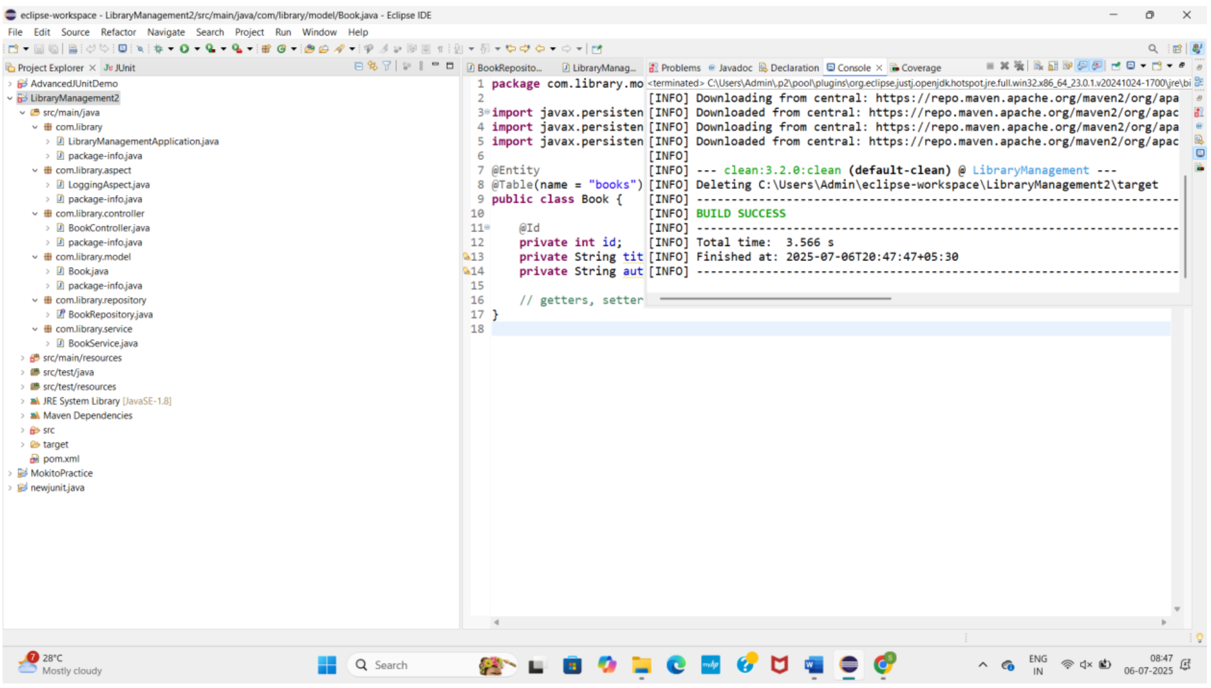
Book Controller

```
package com.library.controller;
import com.library.model.Book;
import com.library.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;
@RequestMapping("/books")
public class BookController {
   private BookRepository bookRepository;
   public List<Book> getAllBooks() {
      return bookRepository.findAll();
   }
   public Book createBook(@RequestBody Book book) {
      return bookRepository.save(book);
   }
   public Book getBookById(@PathVariable Long id) {
      return bookRepository.findById(id).orElse(null);
   }
   public Book updateBook(@PathVariable Long id, @RequestBody Book bookDetails) {
      Book book = bookRepository.findById(id).orElse(null);
      if (book != null) {
         book.setTitle(bookDetails.getTitle());
         book.setAuthor(bookDetails.getAuthor());
         return bookRepository.save(book);
      }
      return null;
   }
   public void deleteBook(@PathVariable Long id) {
      bookRepository.deleteById(id);
   }
}
```

Main Class

```
package com.library;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class LibraryManagementApplication {

    public static void main(String[] args) {

        SpringApplication.run(LibraryManagementApplication.class, args);

    }

}
```

OUTPUT:



```
package com.library.mo

import javax.persisten
import javax.persisten
import javax.persisten

@Entity
@Table(name = "books")
public class Book {

    @Id
    private int id;
    private String tit
    private String aut

    // getters, setter
}
```

```
<terminated> C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.1.v20241024-1700\jre\bi
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apa
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apac
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apa
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apac
[INFO]
[INFO] --- clean:3.2.0:clean (default-clean) @ LibraryManagement ---
[INFO] Deleting C:\Users\Admin\eclipse-workspace\LibraryManagement2\target
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.566 s
[INFO] Finished at: 2025-07-06T20:47:47+05:30
[INFO] ------------------------------------------------------------------------
```