**Exercise 1: Configuring a Basic Spring Application**

**Scenario:**

Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

**Steps:**

1. **Set Up a Spring Project:**
   * Create a Maven project named **LibraryManagement**.
   * Add Spring Core dependencies in the **pom.xml** file.
2. **Configure the Application Context:**
   * Create an XML configuration file named **applicationContext.xml** in the **src/main/resources** directory.
   * Define beans for **BookService** and **BookRepository** in the XML file.
3. **Define Service and Repository Classes:**
   * Create a package **com.library.service** and add a class **BookService**.
   * Create a package **com.library.repository** and add a class **BookRepository**.
4. **Run the Application:**
   * Create a main class to load the Spring context and test the configuration.

**Solution:**

**Application.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) {  
 System.*out*.println("Starting Library Management Application...");  
  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService bookService = (BookService) context.getBean("bookService");  
  
 System.*out*.println("\n=== Testing Library Management System ===");  
 bookService.displayAllBooks();  
  
 System.*out*.println("\n=== Adding new books ===");  
 bookService.addBook("Design Patterns");  
 bookService.addBook("Effective Java");  
  
 System.*out*.println("\n=== After adding books ===");  
 bookService.displayAllBooks();  
  
 System.*out*.println("\n=== Searching for books ===");  
 bookService.searchBook("Java");  
 bookService.searchBook("Python");  
  
 System.*out*.println("\n=== Removing a book ===");  
 bookService.removeBook("Clean Code");  
  
 System.*out*.println("\n=== Final book list ===");  
 bookService.displayAllBooks();  
   
 ((ClassPathXmlApplicationContext) context).close();  
  
 System.*out*.println("\nApplication completed successfully!");  
 }  
}

**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>  
 <packaging>jar</packaging>  
  
 <name>LibraryManagement</name>  
 <description>Spring-based Library Management System</description>  
  
 <properties>  
 <maven.compiler.source>11</maven.compiler.source>  
 <maven.compiler.target>11</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <spring.version>5.3.21</spring.version>  
 </properties>  
  
 <dependencies>  
 <!-- Spring Core -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-core</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- Spring Context -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- Spring Beans -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-beans</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- JUnit for testing -->  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>11</source>  
 <target>11</target>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>  
</project>

**BookRepository,java:**

package com.library.repository;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class BookRepository {  
 private List<String> books;  
  
 public BookRepository() {  
 this.books = new ArrayList<>();  
 // Initialize with some sample books  
 books.add("Spring in Action");  
 books.add("Java: The Complete Reference");  
 books.add("Clean Code");  
 System.*out*.println("BookRepository initialized with sample books");  
 }  
  
 public void addBook(String bookTitle) {  
 books.add(bookTitle);  
 System.*out*.println("Book added: " + bookTitle);  
 }  
  
 public List<String> getAllBooks() {  
 return new ArrayList<>(books);  
 }  
  
 public boolean removeBook(String bookTitle) {  
 boolean removed = books.remove(bookTitle);  
 if (removed) {  
 System.*out*.println("Book removed: " + bookTitle);  
 } else {  
 System.*out*.println("Book not found: " + bookTitle);  
 }  
 return removed;  
 }  
  
 public int getTotalBooks() {  
 return books.size();  
 }  
  
 public void displayBooks() {  
 System.*out*.println("Books in repository:");  
 for (int i = 0; i < books.size(); i++) {  
 System.*out*.println((i + 1) + ". " + books.get(i));  
 }  
 }  
}

**BookService.java:**

package com.library.service;  
  
import com.library.repository.BookRepository;  
import java.util.List;  
  
public class BookService {  
 private BookRepository bookRepository;  
  
  
 public BookService() {  
 System.*out*.println("BookService initialized");  
 }  
  
  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 System.*out*.println("BookRepository injected into BookService");  
 }  
  
 public void addBook(String bookTitle) {  
 if (bookTitle != null && !bookTitle.trim().isEmpty()) {  
 bookRepository.addBook(bookTitle);  
 } else {  
 System.*out*.println("Invalid book title");  
 }  
 }  
  
 public void removeBook(String bookTitle) {  
 bookRepository.removeBook(bookTitle);  
 }  
  
 public List<String> getAllBooks() {  
 return bookRepository.getAllBooks();  
 }  
  
 public void displayAllBooks() {  
 System.*out*.println("=== Library Books ===");  
 bookRepository.displayBooks();  
 System.*out*.println("Total books: " + bookRepository.getTotalBooks());  
 }  
  
 public void searchBook(String bookTitle) {  
 List<String> books = bookRepository.getAllBooks();  
 boolean found = false;  
  
 for (String book : books) {  
 if (book.toLowerCase().contains(bookTitle.toLowerCase())) {  
 System.*out*.println("Found book: " + book);  
 found = true;  
 }  
 }  
  
 if (!found) {  
 System.*out*.println("No books found matching: " + bookTitle);  
 }  
 }  
}

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

A black screen with white text

AI-generated content may be incorrect.**Output:**

A black rectangle with white dots

AI-generated content may be incorrect.