**Exercise 2: Online Bookstore - Creating Basic REST Controllers**

**Business Scenario:**

**Implement RESTful endpoints to manage books.**

**Instructions:**

1. **Create Book Controller:**
   * **Define a BookController class with request mappings for /books.**
2. **Handle HTTP Methods:**
   * **Implement methods to handle GET, POST, PUT, and DELETE requests.**
3. **Return JSON Responses:**
   * **Ensure the controller returns JSON responses.**
   * **Define the Book entity with attributes like id, title, author, price, and isbn.**

**BOOKCONTROLLER**BookController.javapackage com.example.bookstoreapi.controller;

import com.example.bookstoreapi.model.Book;

import com.example.bookstoreapi.service.BookService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

import java.util.Optional;

@RestController

@RequestMapping("/books")

public class BookController {

@Autowired

private BookService bookService;

// GET /books - Retrieve all books

@GetMapping

public List<Book> getAllBooks() {

return bookService.getAllBooks();

}

// GET /books/{id} - Retrieve a book by its ID

@GetMapping("/{id}")

public Optional<Book> getBookById(@PathVariable Long id) {

return bookService.getBookById(id);

}

// POST /books - Create a new book

@PostMapping

public Book createBook(@RequestBody Book book) {

return bookService.createBook(book);

}

// PUT /books/{id} - Update an existing book by its ID

@PutMapping("/{id}")

public Book updateBook(@PathVariable Long id, @RequestBody Book updatedBook) {

return bookService.updateBook(id, updatedBook);

}

// DELETE /books/{id} - Delete a book by its ID

@DeleteMapping("/{id}")

public void deleteBook(@PathVariable Long id) {

bookService.deleteBook(id);

}

}