BookController.java

package com.example.bookstoreapi.controllers;

import com.example.bookstoreapi.models.Book;

import com.example.bookstoreapi.services.BookService;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import java.util.List;

@RestController

@RequestMapping("/books")

public class BookController {

private final BookService bookService;

@Autowired

public BookController(BookService bookService) {

this.bookService = bookService;

}

// ... existing endpoints ...

@GetMapping("/{id}")

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

Book book = bookService.getBookById(id);

return new ResponseEntity<>(book, HttpStatus.OK);

}

@GetMapping

public ResponseEntity<List<Book>> getBooksByFilter(@RequestParam(required = false) String title,

@RequestParam(required = false) String author) {

List<Book> books = bookService.getBooksByFilter(title, author);

return new ResponseEntity<>(books, HttpStatus.OK);

}

}

BookService.java

package com.example.bookstoreapi.services;

import com.example.bookstoreapi.models.Book;

import com.example.bookstoreapi.repositories.BookRepository;

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

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class BookService {

private final BookRepository bookRepository;

@Autowired

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

// ... existing methods ...

public Book getBookById(Long id) {

return bookRepository.findById(id).orElse(null);

}

public List<Book> getBooksByFilter(String title, String author) {

if (title != null && author != null) {

return bookRepository.findByTitleAndAuthor(title, author);

} else if (title != null) {

return bookRepository.findByTitle(title);

} else if (author != null) {

return bookRepository.findByAuthor(author);

} else {

return bookRepository.findAll();

}

}

}

BookRepository.java

package com.example.bookstoreapi.repositories;

import com.example.bookstoreapi.models.Book;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface BookRepository extends JpaRepository<Book, Long> {

List<Book> findByTitle(String title);

List<Book> findByAuthor(String author);

@Query("SELECT b FROM Book b WHERE b.title = :title AND b.author = :author")

List<Book> findByTitleAndAuthor(@Param("title") String title, @Param("author") String author);

}

Here's what's new:

Path Variables: The getBookById endpoint uses a path variable id to fetch a book by its ID.

Query Parameters: The getBooksByFilter endpoint uses query parameters title and author to filter books. The BookService class handles the filtering logic, and the BookRepository class provides the necessary methods to query the database.