**Exercise 5: Online Bookstore - Customizing Response Status and Headers**

**Business Scenario:**

Customize the HTTP response status and headers for the book management endpoints.

**Instructions:**

1. **Response Status:**
   * Use **@ResponseStatus** to customize HTTP status codes for your endpoints.
2. **Custom Headers:**
   * Add custom headers to the response using **ResponseEntity**.

**Solution**

**BookController.java**

package com.example.bookstoreapi.controller;

import com.example.bookstoreapi.model.Book;

import org.springframework.http.HttpHeaders;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import java.util.ArrayList;

import java.util.List;

@RestController

@RequestMapping("/api/books")

public class BookController {

private List<Book> books = new ArrayList<>();

// GET: Retrieve all books with custom headers

@GetMapping

public ResponseEntity<List<Book>> getAllBooks() {

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "BookListHeader");

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

}

// GET: Retrieve a book by ID with custom headers

@GetMapping("/{id}")

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

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "SingleBookHeader");

Book book = books.stream().filter(b -> b.getId().equals(id)).findFirst().orElse(null);

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

}

// POST: Create a new book with custom headers

@PostMapping

public ResponseEntity<Book> createBook(@RequestBody Book book) {

books.add(book);

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "CreateBookHeader");

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

}

// PUT: Update an existing book with custom headers

@PutMapping("/{id}")

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

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "UpdateBookHeader");

for (int i = 0; i < books.size(); i++) {

Book book = books.get(i);

if (book.getId().equals(id)) {

books.set(i, updatedBook);

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

}

}

return new ResponseEntity<>(null, headers, HttpStatus.NOT\_FOUND);

}

// DELETE: Delete a book by ID with custom headers

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteBook(@PathVariable Long id) {

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "DeleteBookHeader");

books.removeIf(book -> book.getId().equals(id));

return new ResponseEntity<>(headers, HttpStatus.NO\_CONTENT);

}

}