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

**1. Customizing HTTP Response Status Codes**

We can use @ResponseStatus to customize the HTTP status codes for our endpoints. By default, Spring Boot uses common status codes like 200 OK for successful GET and 201 Created for POST requests, but we can override this behavior.

For example:

* **201 Created** for a successful creation of a book.
* **404 Not Found** if a book is not found.

Example: Custom Status with @ResponseStatus

import org.springframework.http.HttpStatus;

import org.springframework.web.server.ResponseStatusException;

// Method to create a book (returns 201 Created)

@PostMapping

@ResponseStatus(HttpStatus.CREATED)

public Book createBook(@RequestBody Book newBook) {

bookList.add(newBook);

return newBook;

}

// Method to get a book by ID (throws 404 if not found)

@GetMapping("/{id}")

public Book getBookById(@PathVariable Long id) {

return bookList.stream()

.filter(book -> book.getId().equals(id))

.findFirst()

.orElseThrow(() -> new ResponseStatusException(HttpStatus.NOT\_FOUND, "Book not found"));

}

* The **POST** method returns a 201 Created status when a book is successfully added.
* The **GET** method throws a 404 Not Found status if the book does not exist.

**2. Adding Custom Headers Using ResponseEntity**

To have more control over the response (including headers), we can use ResponseEntity. It allows us to set headers, status codes, and the response body.

**Example: Using ResponseEntity to Add Custom Headers**

import org.springframework.http.HttpHeaders;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

// Method to return all books with custom headers

@GetMapping

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

HttpHeaders headers = new HttpHeaders();

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

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

}

// Method to delete a book with custom status and headers

@DeleteMapping("/{id}")

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

boolean removed = bookList.removeIf(book -> book.getId().equals(id));

if (removed) {

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "Book Deleted");

return new ResponseEntity<>("Book deleted successfully", headers, HttpStatus.OK);

} else {

throw new ResponseStatusException(HttpStatus.NOT\_FOUND, "Book not found");

}

}

* **Custom Headers:** You can add headers to the response using HttpHeaders and set them in ResponseEntity.
* **ResponseEntity:** This class allows you to return not just the body but also headers and the status code.

**Full Example BookController with Customized Status and Headers**

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 org.springframework.web.server.ResponseStatusException;

import java.util.ArrayList;

import java.util.List;

@RestController

@RequestMapping("/books")

public class BookController {

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

// Create a new book (Response: 201 Created)

@PostMapping

@ResponseStatus(HttpStatus.CREATED)

public Book createBook(@RequestBody Book newBook) {

bookList.add(newBook);

return newBook;

}

// Get a book by ID (Response: 200 OK or 404 Not Found)

@GetMapping("/{id}")

public Book getBookById(@PathVariable Long id) {

return bookList.stream()

.filter(book -> book.getId().equals(id))

.findFirst()

.orElseThrow(() -> new ResponseStatusException(HttpStatus.NOT\_FOUND, "Book not found"));

}

// Get all books with custom headers (Response: 200 OK)

@GetMapping

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

HttpHeaders headers = new HttpHeaders();

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

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

}

// Delete a book (Response: 200 OK or 404 Not Found with custom headers)

@DeleteMapping("/{id}")

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

boolean removed = bookList.removeIf(book -> book.getId().equals(id));

if (removed) {

HttpHeaders headers = new HttpHeaders();

headers.add("Custom-Header", "Book Deleted");

return new ResponseEntity<>("Book deleted successfully", headers, HttpStatus.OK);

} else {

throw new ResponseStatusException(HttpStatus.NOT\_FOUND, "Book not found");

}

}

}