Library Management System

A Spring Boot-based REST API for managing books, patrons, and borrowing records in a library.

- Features

1. Manage books (CRUD operations).
2. Manage patrons (CRUD operations).
3. Borrow and return books.
4. Input validation and error handling.

- Technologies Used

1. Spring Boot
2. Spring Data JPA
3. MySQL

- JUnit and Mockito for testing

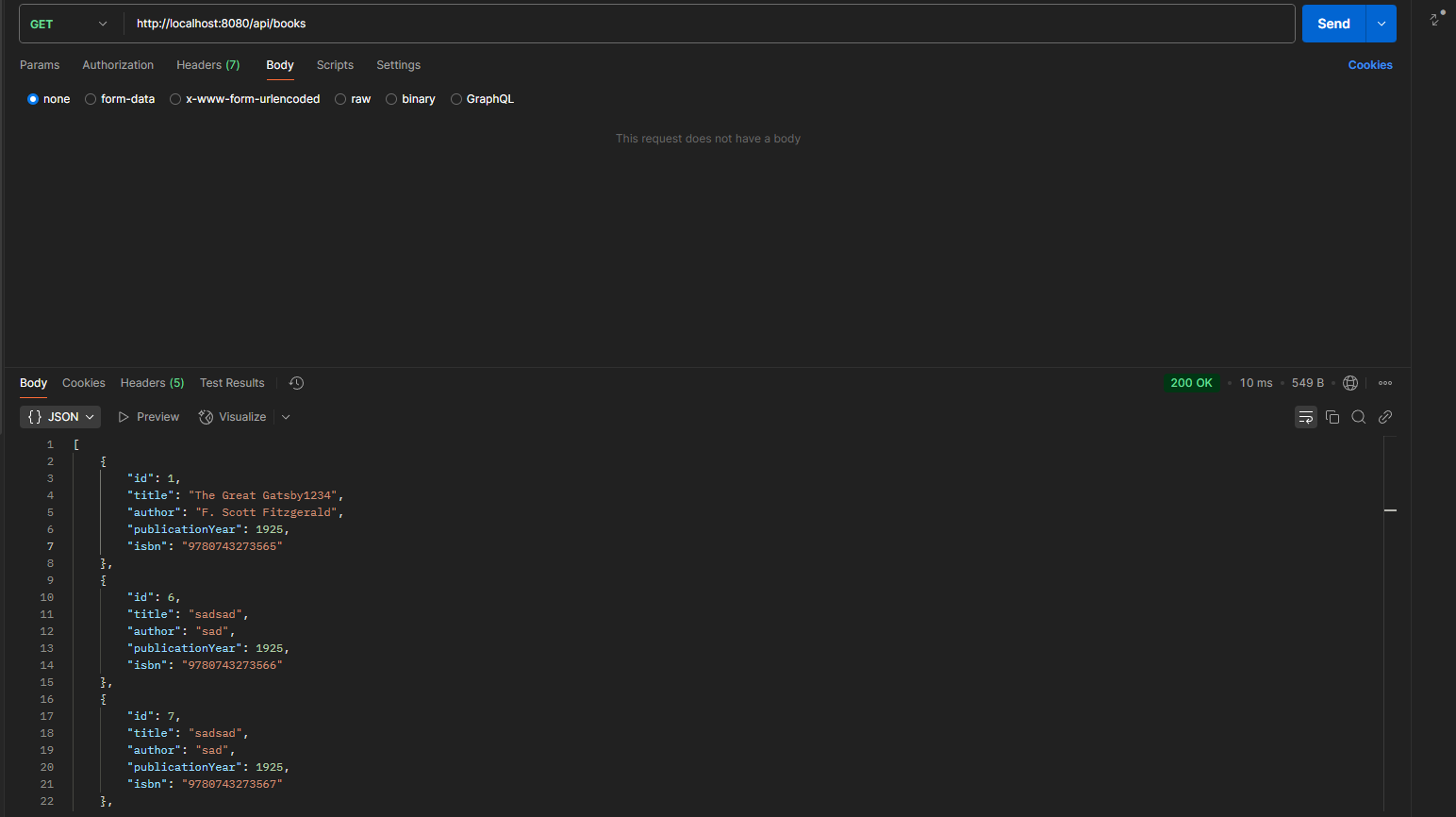
1. ## Prerequisites
2. Java 17
3. MySQL
4. Gradle

Base URL : <http://localhost:8080/api>

\*\*\*\*\*\* Book management endpoints: \*\*\*\*\*\*

● GET /api/books: Retrieve a list of all books. (Done)

http://localhost:8080/api/books



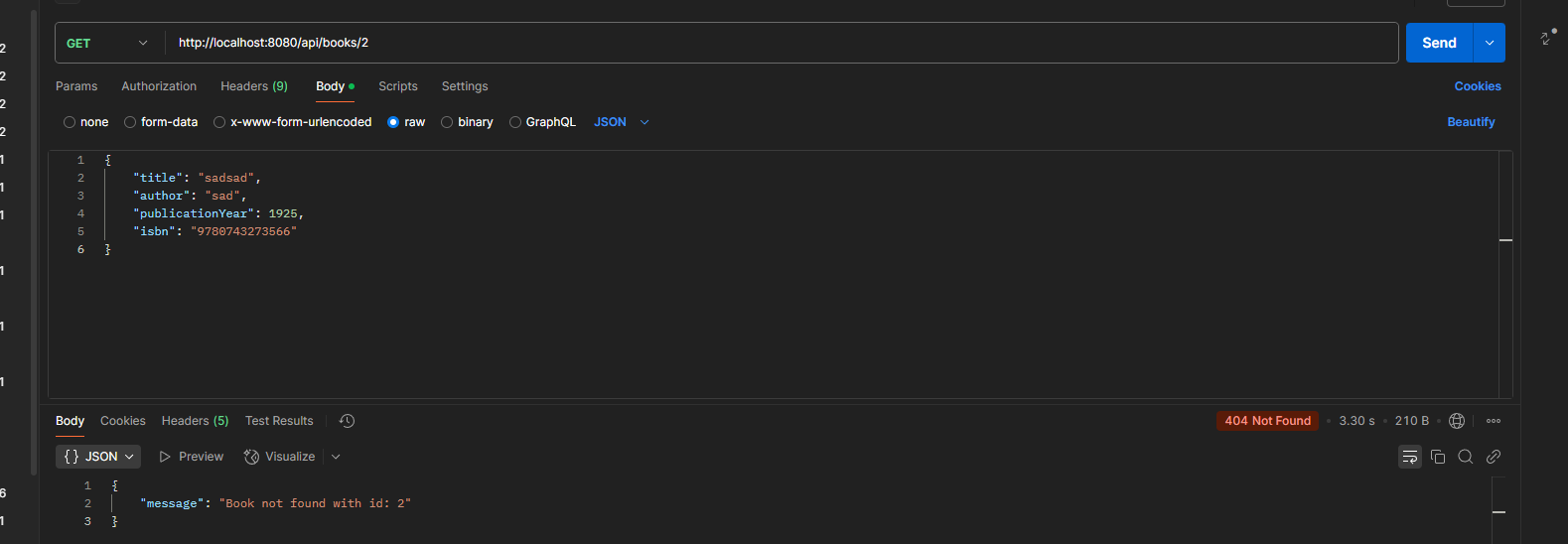
● GET /api/books/{id}: Retrieve details of a specific book by ID. (Done)

[http://localhost:8080/api/books/](http://localhost:8080/api/books/2)1 When Id Available

A screenshot of a computer

AI-generated content may be incorrect.

When Id is not Available



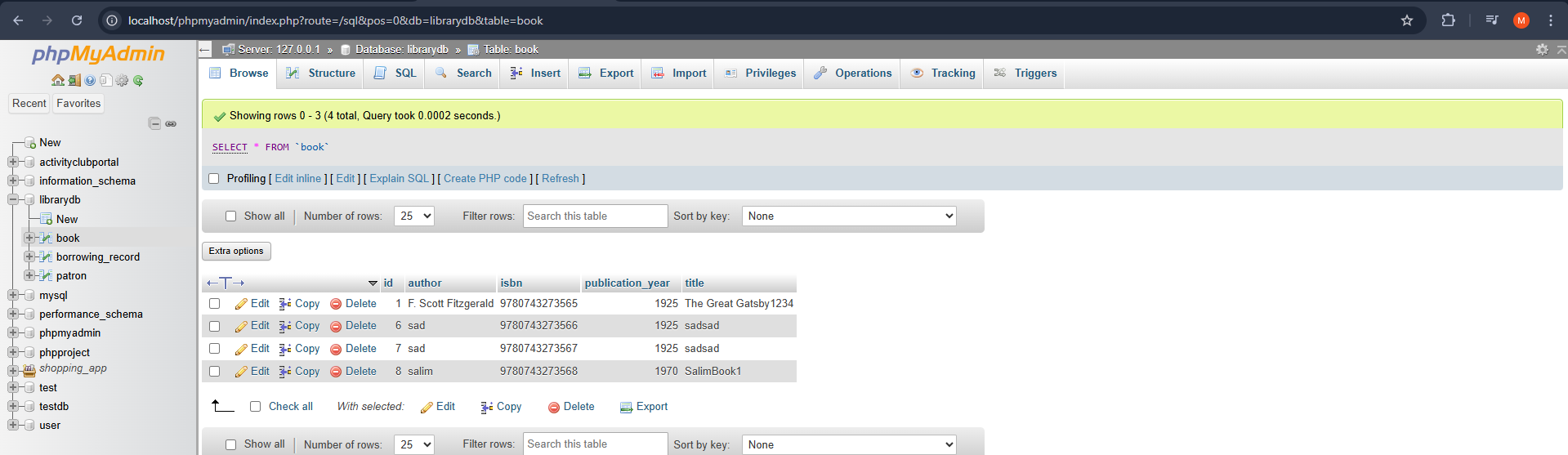
● POST /api/books: Add a new book to the library. (Done)

[http://localhost:8080/api/books](http://localhost:8080/api/books/1)

{

    "title": "SalimBook1"  
 "author": "salim",  
 "publicationYear": 1970,   
 "isbn": " 9780743273568”

}

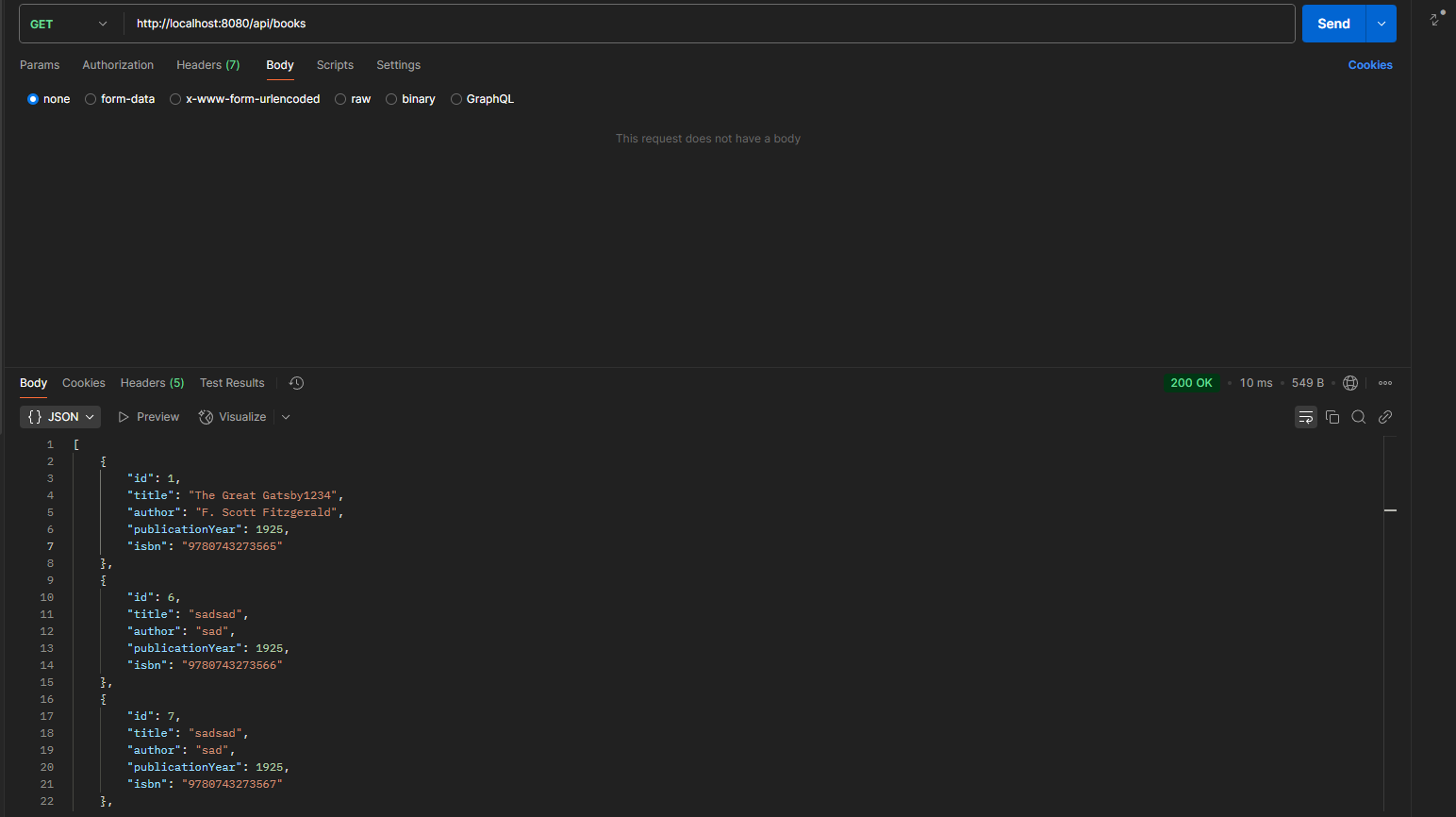


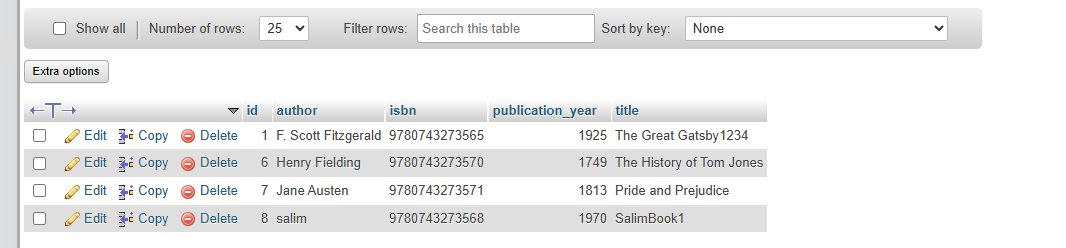
● PUT /api/books/{id}: Update an existing book's information.(Done)  
http://localhost:8080/api/books/7

UPDATE BOOK WITH ID : 7  
{

    "title": "Pride and Prejudice",  
 "author": "Jane Austen",  
 "publicationYear": 1813,  
 "isbn": "9780743273571"

}



After Updating the record   


Case when you are trying to update a book but no such id entered is defined under table book, the api will return a message : Book not found with id : {passed id}

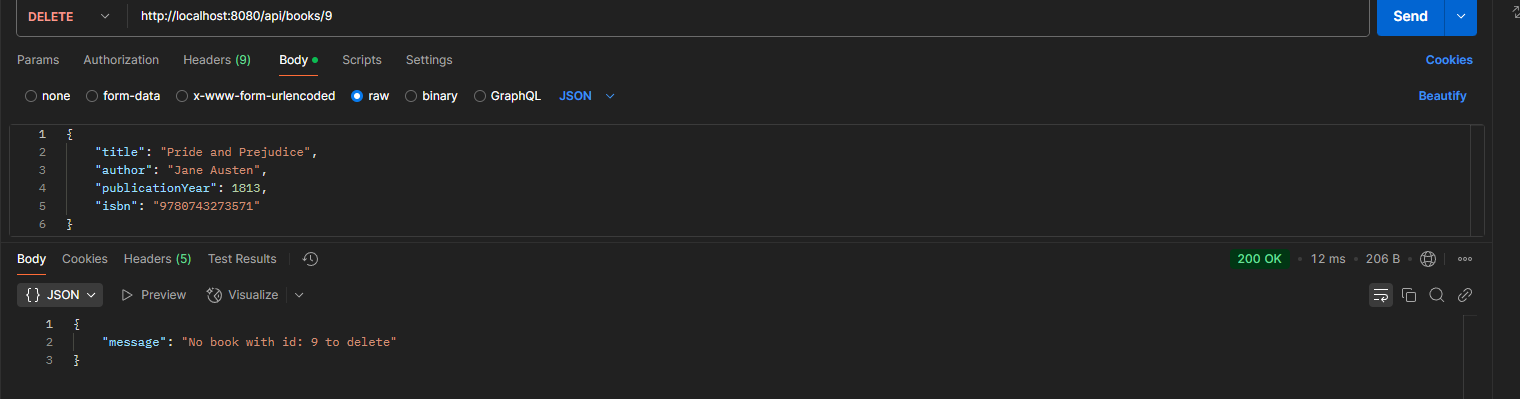
A screen shot of a computer

AI-generated content may be incorrect.

● DELETE /api/books/{id}: Remove a book from the library. (Done)

[http://localhost:8080/api/books/{id}](http://localhost:8080/api/books/%7bid%7d)

lets delete the first book created, <http://localhost:8080/api/books/1>  
book now is deleted   
A screenshot of a computer

AI-generated content may be incorrect.  
try to delete a book with id not exist in table case id : 9   


lets delete 6,7

A black screen with white text

AI-generated content may be incorrect.

\*\*\*\*\*\* Patron management endpoints: \*\*\*\*\*\*

● GET /api/patrons: Retrieve a list of all patrons. (Done)

When no data found

A black screen with blue text

AI-generated content may be incorrect.

When patrons exist

A black screen with white text

AI-generated content may be incorrect.

● GET /api/patrons/{id}: Retrieve details of a specific patron by ID. (Done)

A screenshot of a computer

AI-generated content may be incorrect.

When no id found in db   
A black screen with a blue number

AI-generated content may be incorrect.

● POST /api/patrons: Add a new patron to the system. (Done)

A screenshot of a computer

AI-generated content may be incorrect.

● PUT /api/patrons/{id}: Update an existing patron's information. (Done)

Change name to Salim Tyson when we call put api

A black screen with a blue and white number

AI-generated content may be incorrect.

When id not exist   
A screenshot of a video

AI-generated content may be incorrect.

● DELETE /api/patrons/{id}: Remove a patron from the system. (Done)

A screenshot of a computer

AI-generated content may be incorrect.

Lets delete patron with id 1

A black background with blue lights

AI-generated content may be incorrect.

Lets get all partons again

A screenshot of a computer

AI-generated content may be incorrect.

Lets delete patron not exist (No Id Found)

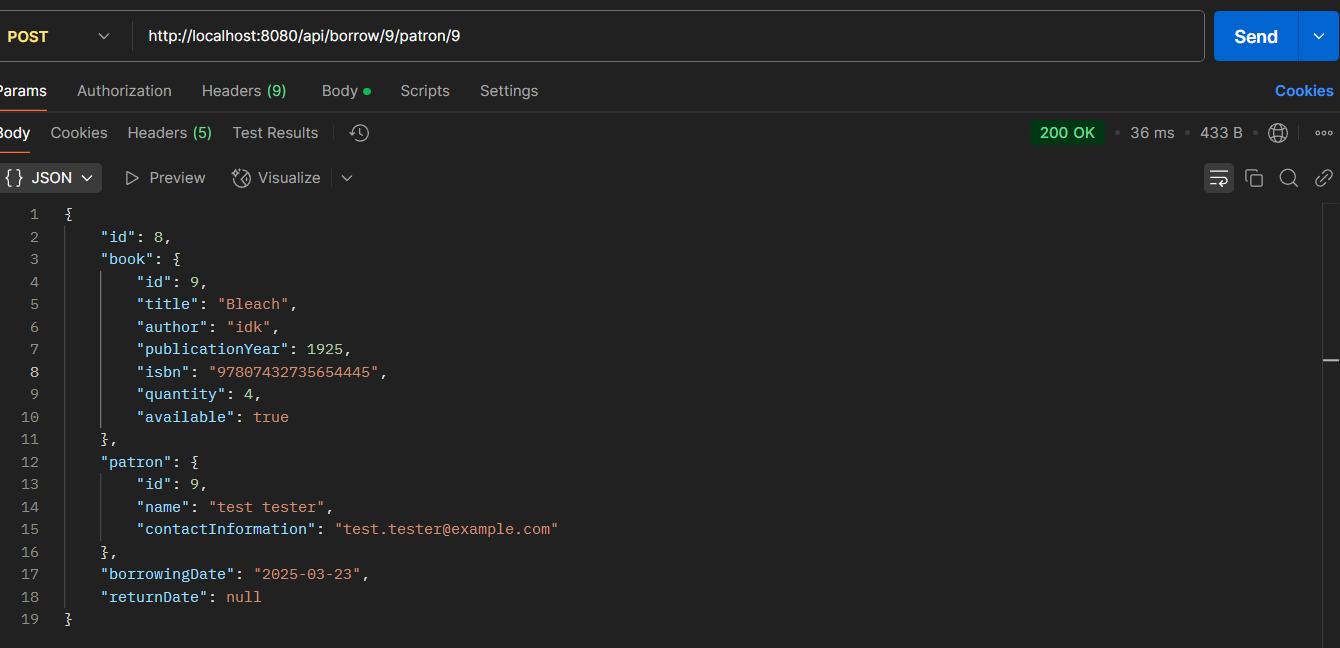
A screenshot of a video player

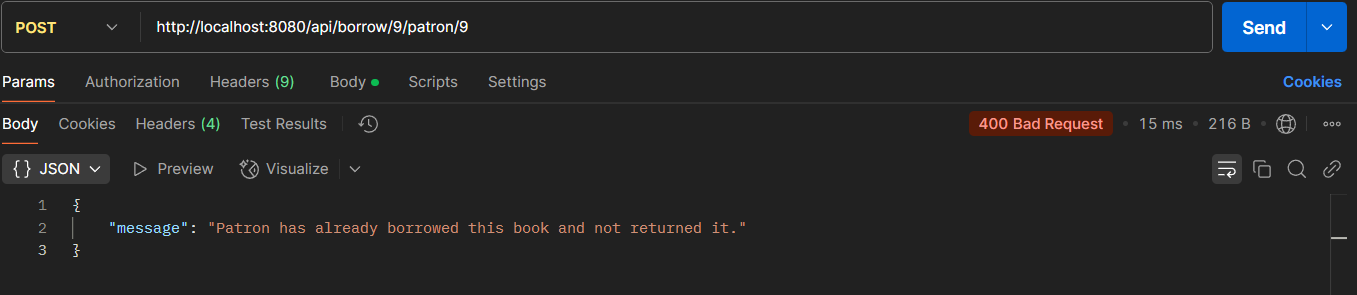
AI-generated content may be incorrect.

\*\*\*\*\*\* Borrowing Record management endpoints: \*\*\*\*\*\*

Method: POST

URL: /api/borrow/{bookId}/patron/{patronId}

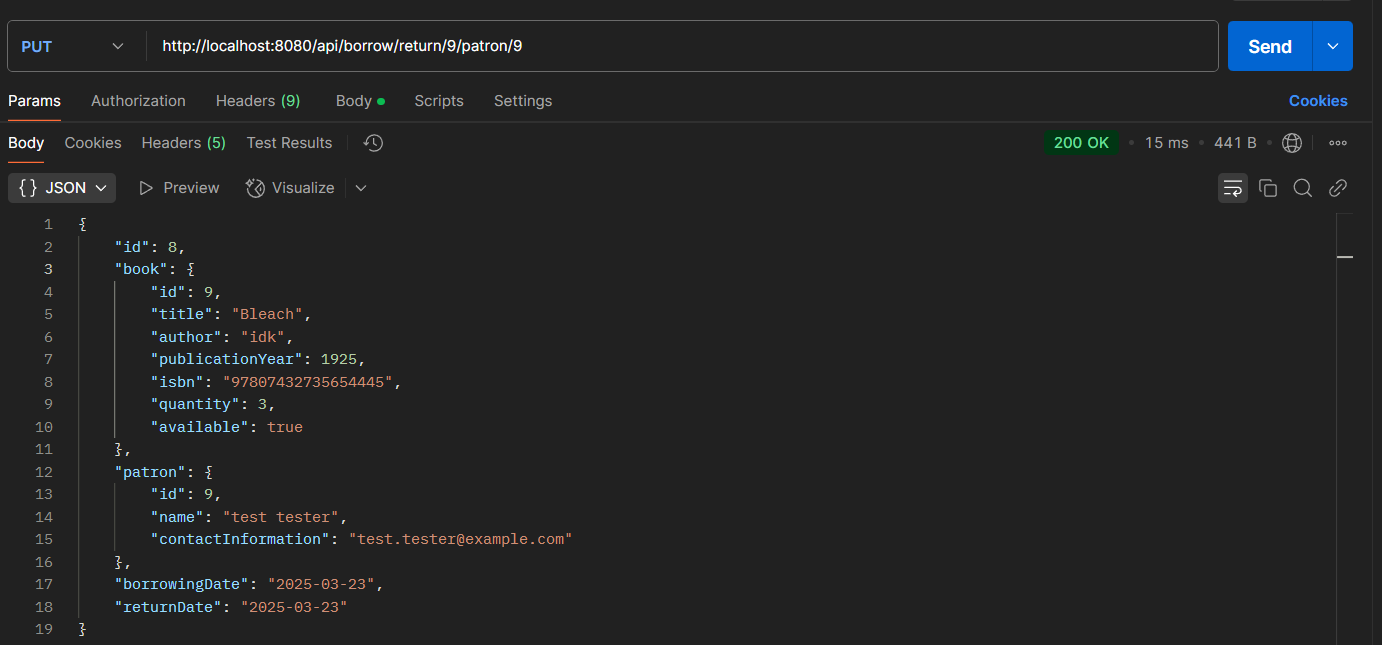


In case if the patron wants to borrow the same book twice :   


Method: PUT

URL: /api/borrow/return/{bookId}/patron/{patronId}

The updated borrowing record with the return date:

  
in case if patron return a book that he is not borrowing:  
