**LIBRARY MANAGEMENT SYSTEM**

This documentation contains details on entities, APIs and other important aspects implement for the Library management system.

**Tools Used:**

Java 17

Maven

Spring Boot

H2 Database

Github

Postman (for testing)

**Entities & APIs:**

1. **Book :** This entity contains attributes for books. The attributes used for this are:

Title, Author, Publication, Year, ISBN

APIs:

* POST (To insert new records to db): Since it’s a h2 database, we have to first insert records to proceed with other APIs.

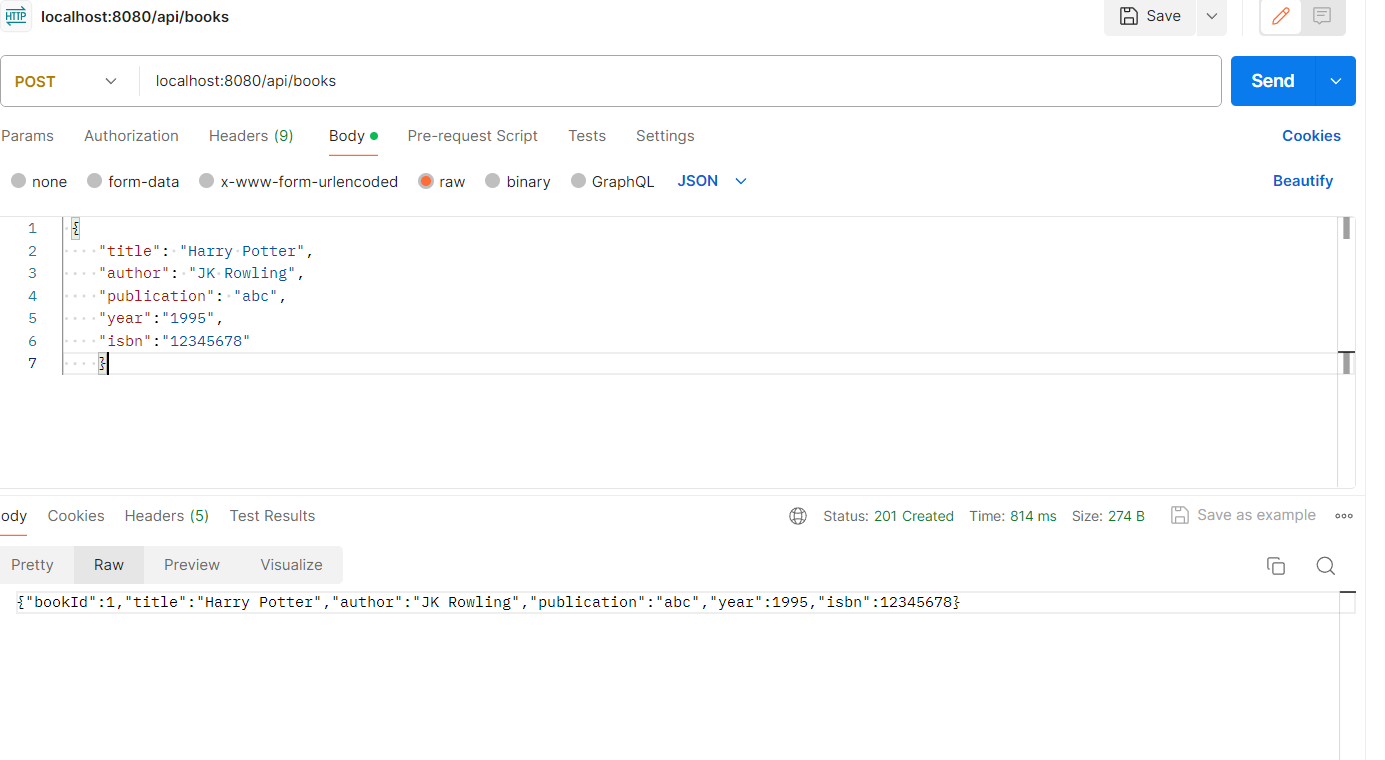
Request Body:

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Validation |
| bookId | Long | This is autogenerated and hence no validation |
| title | String | Not Blank |
| author | String | Not Blank |
| publication | String | Not Blank |
| ISBN | LOng | Not Blank |
| year | int | Should be exactly 4 digits long |

Response Body:

201 created

Sample:



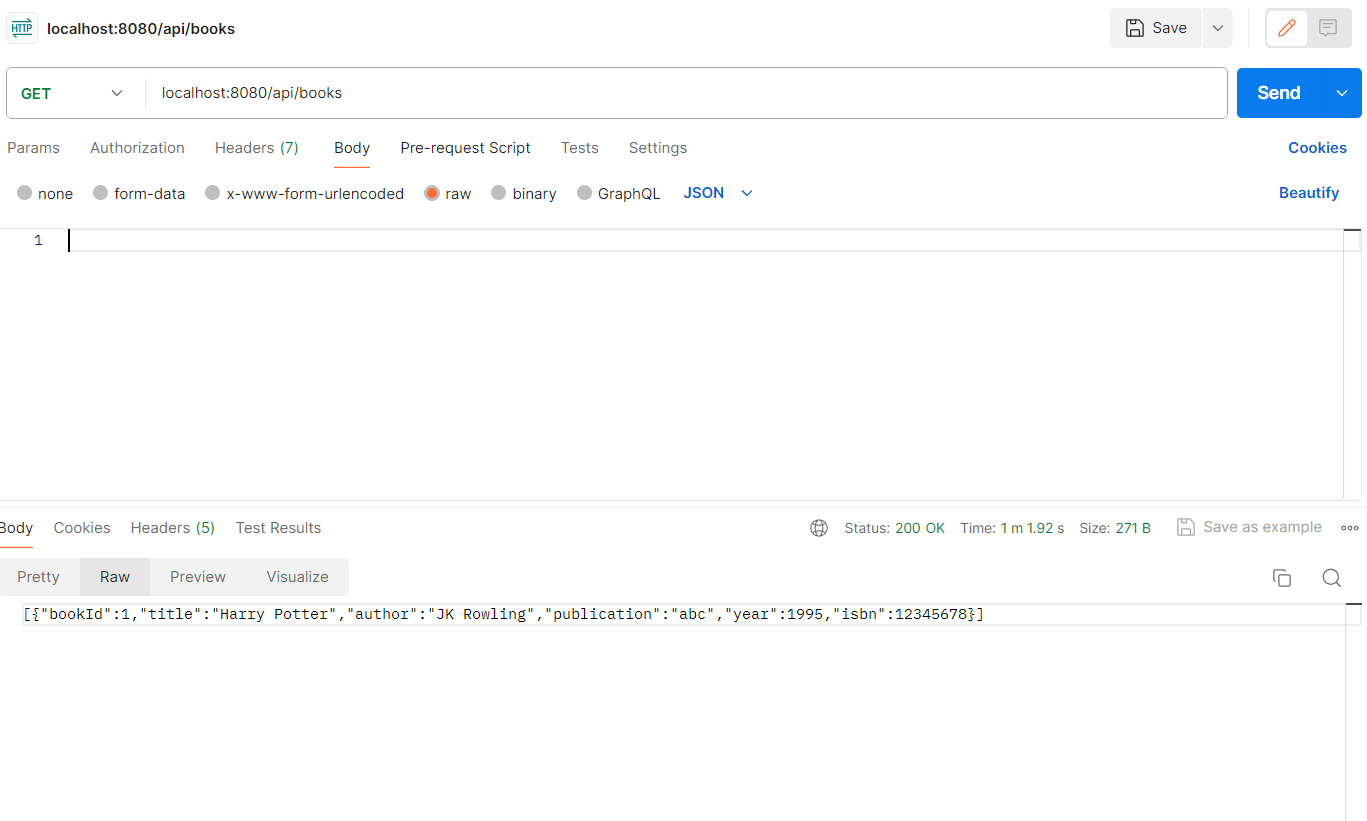
* GET(To Fectch records from db):

URL : localhost:8080/api/books

Response : 200 code with data

Incase of no data will result in 404 error.

Sample:

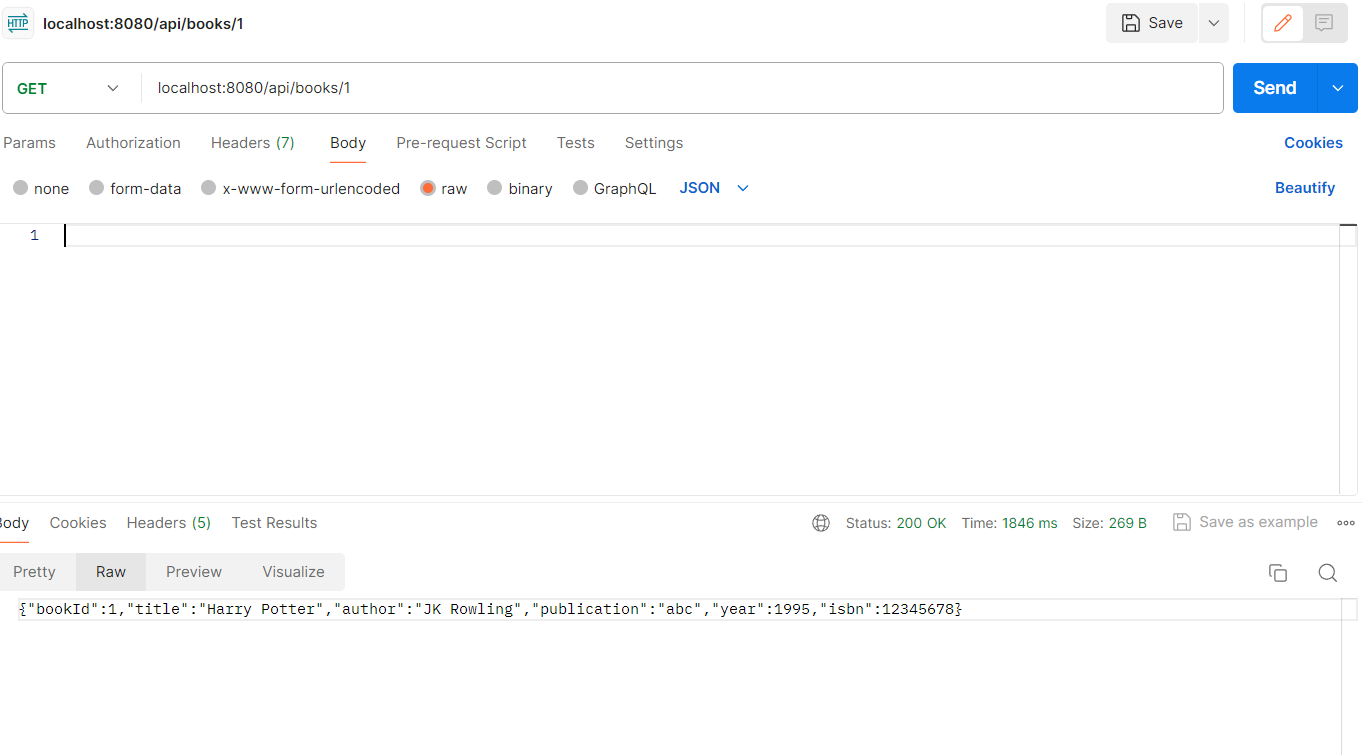


* GET based on ID passed:

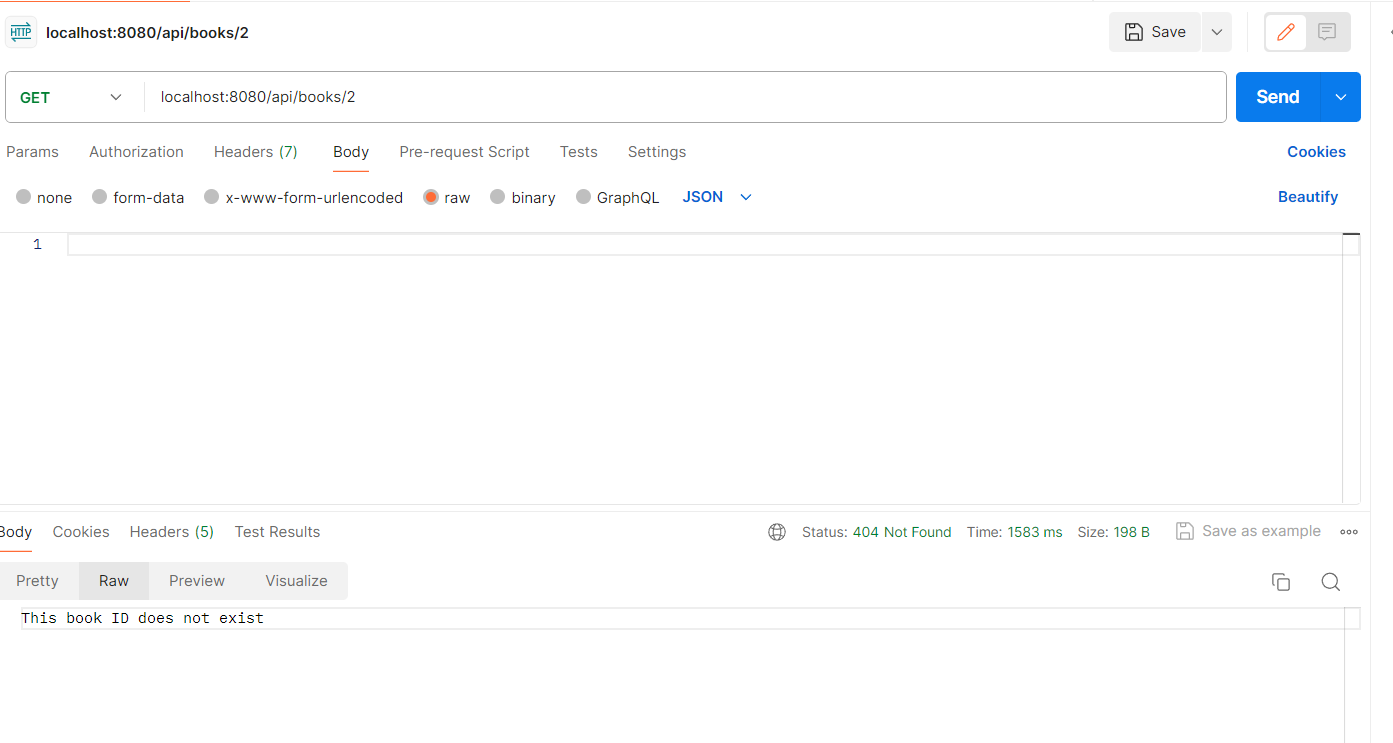
Request : localhost:8080/api/books/{id}

Response : 200 status code with data

Incase no resource is found for this ID will result in resource not found exception

Sample:  
  


When data with given ID is not found:



* PUT (To update entries in DB):

Request : localhost:8080/api/books/{id}

Request body :  {

    "title": " ",

    "author": "",

    "publication": "",

    "year":"",

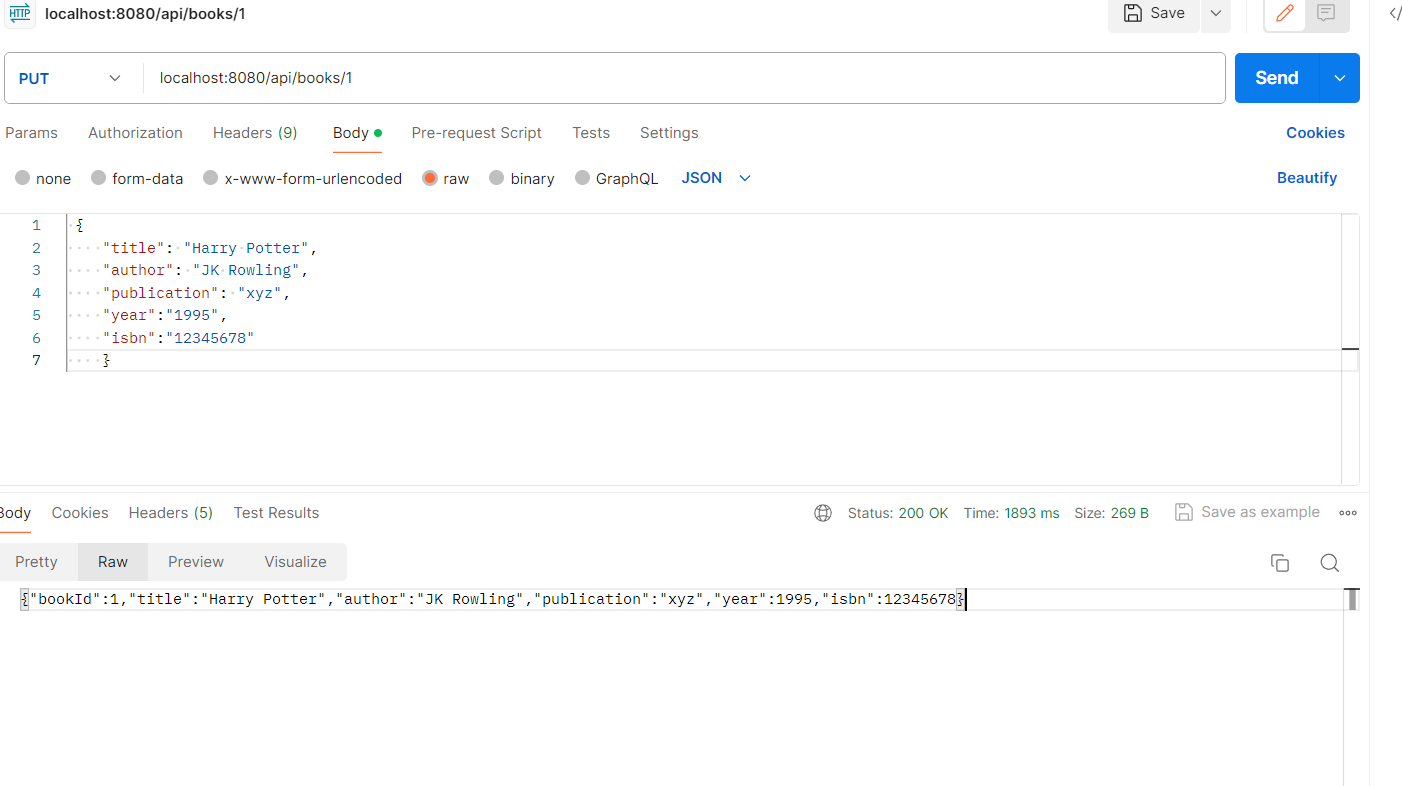
    "isbn":""

    }

Validations specified above is applicable here as well.

Response : 200 status code with updated data.

Incase no resource is found for this ID will result in resource not found exception

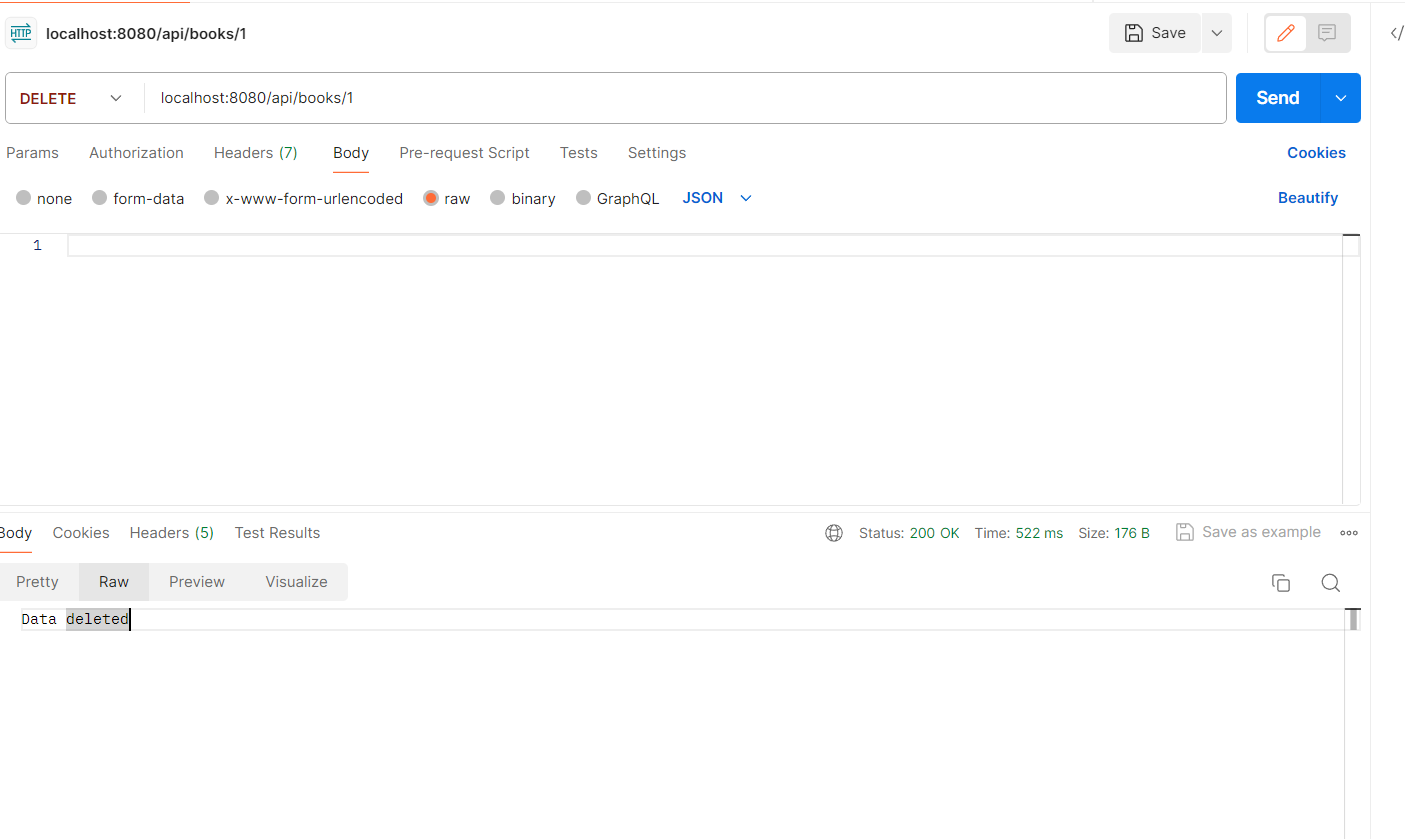


DELETE (To delete entries from DB based on ID passed):

Request: localhost:8080/api/books/{id}

Response : 200 status code with message “Data deleted”

Sample:



1. **Patron** : This entity contains attributes for Patrons.

The attributes are: patronId , name, address, phone number, email

**APIs:**

* POST :

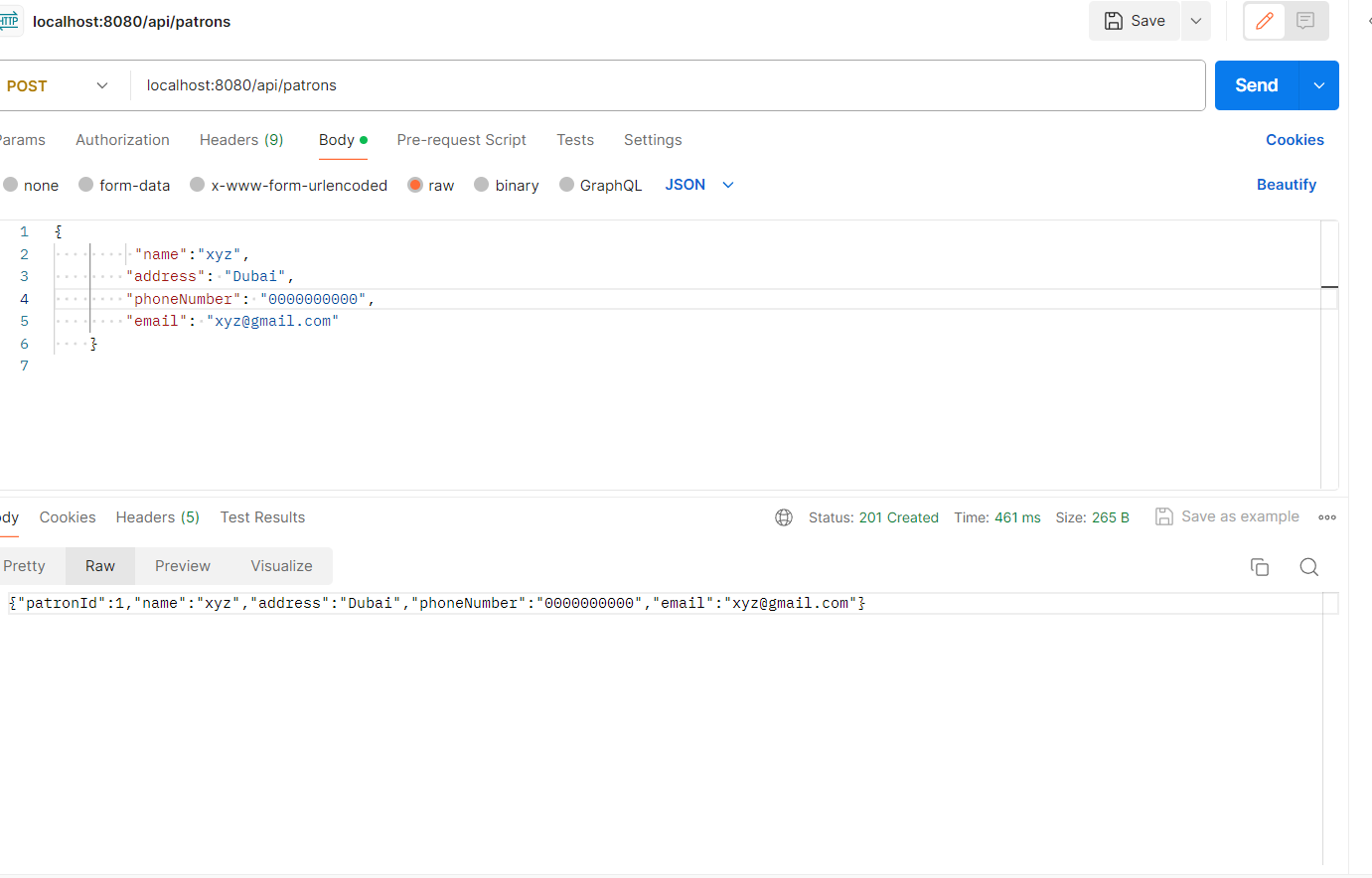
Request URL: localhost:8080/api/patrons

Request Body:

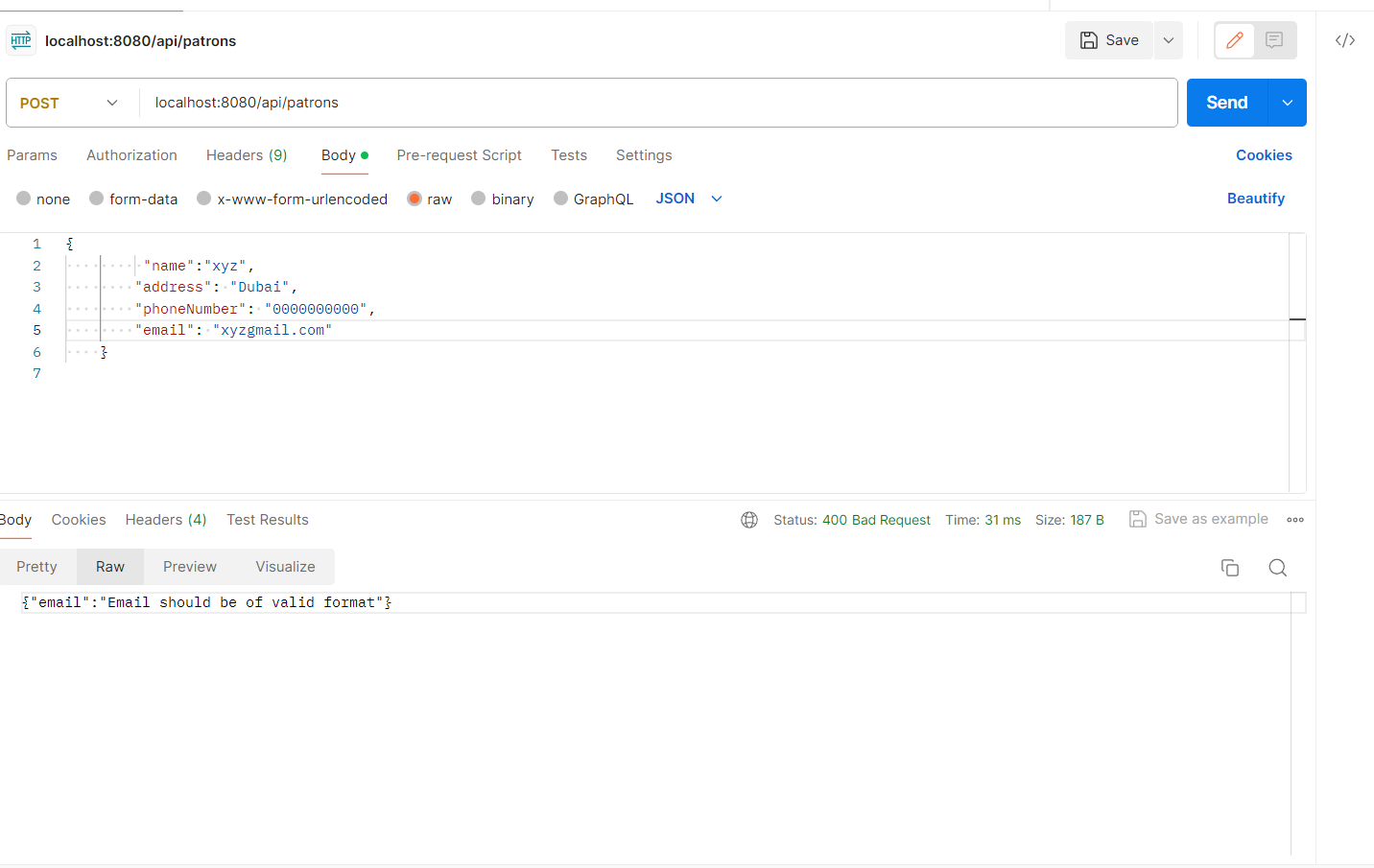
|  |  |  |
| --- | --- | --- |
| Attribute | Type | Validation |
| patronId | Long | This is autogenerated and hence no validation |
| name | String | Not Blank |
| address | String | Optional |
| phoneNumber | String | Number should have at least 10 or less than 15 digits |
| email | String | Should be of valid email format |

Response : 200 code with data saved

Sample:

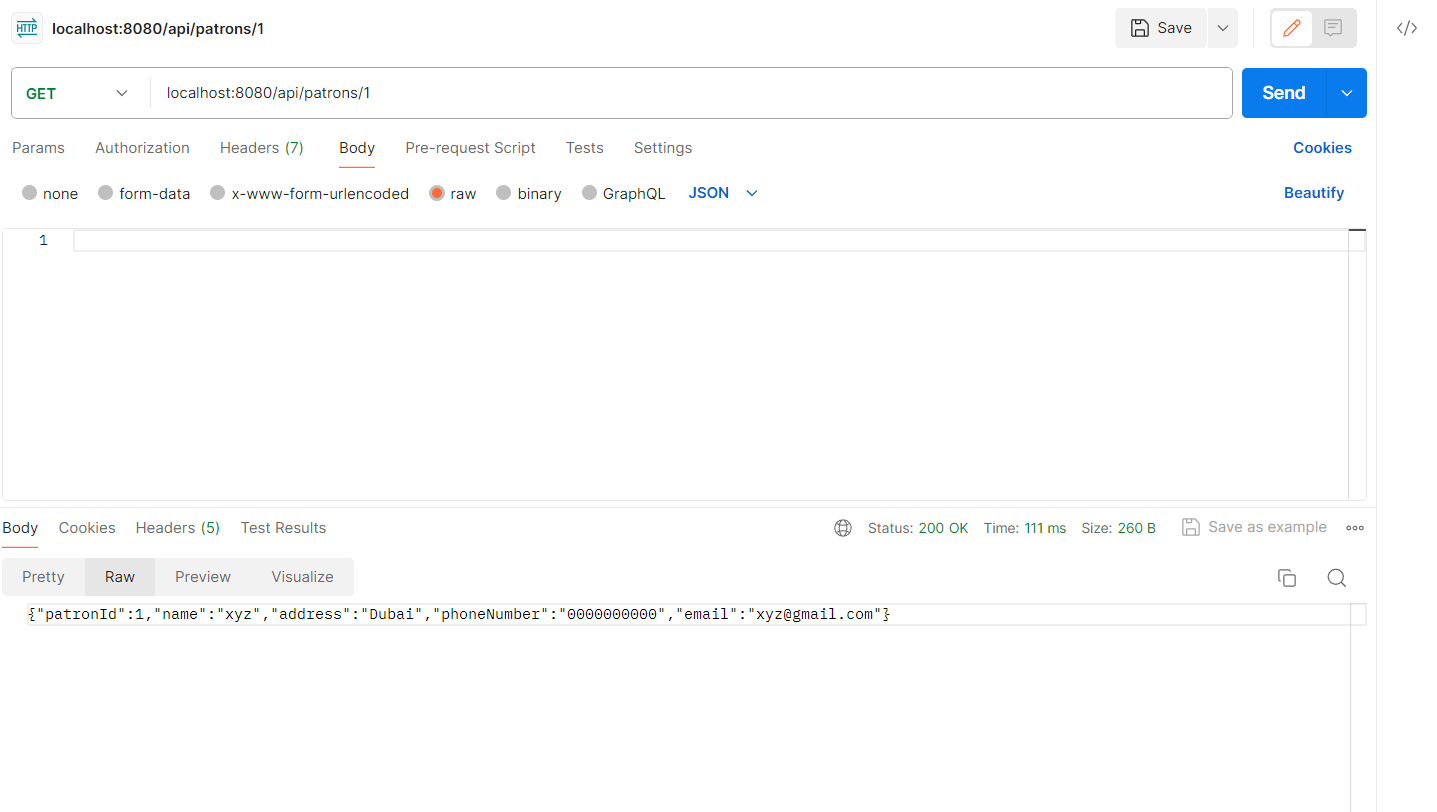


Sample for validation error in email:



GET : Fetch data with respect to ID passed

Request URL : localhost:8080/api/patrons/{id}

Response : 200 status code with data  
  


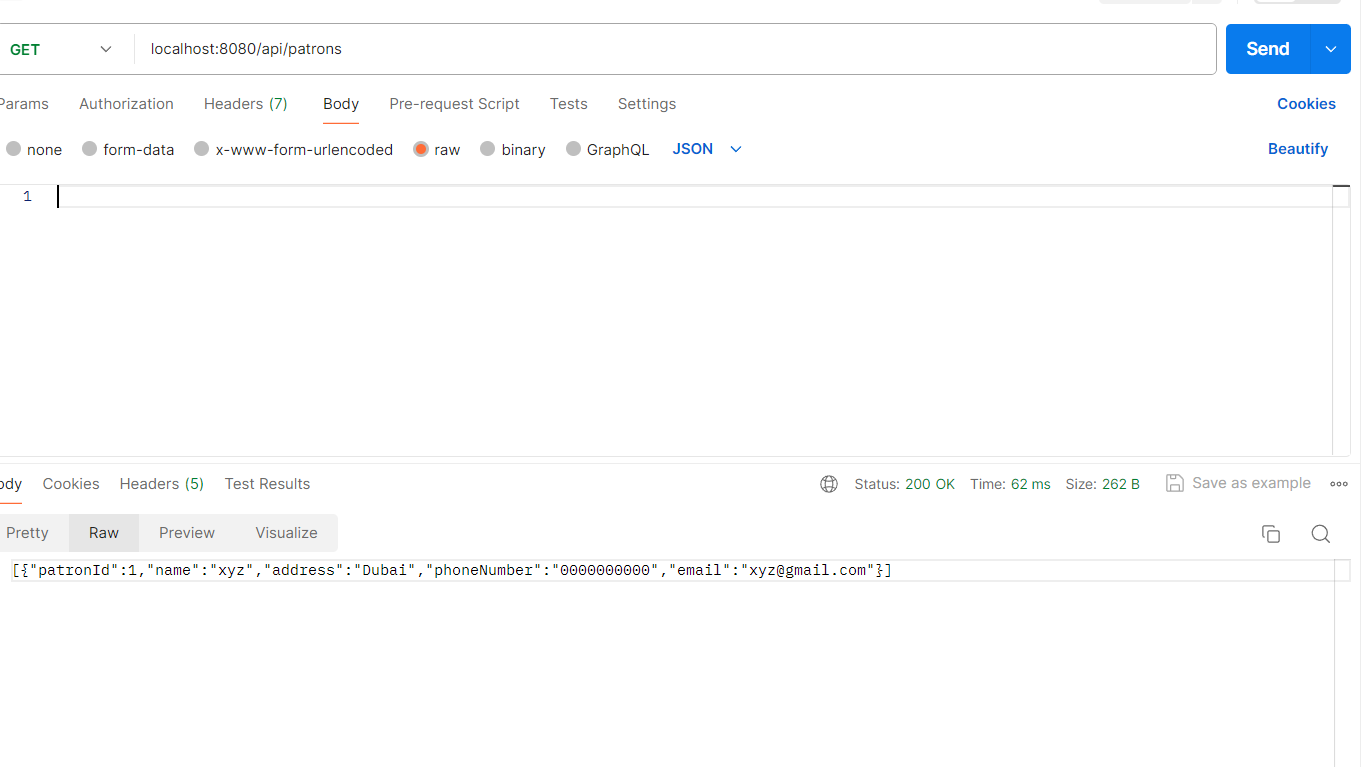
Returns resource not found error when there is no data in db.

GET : Fetch data with respect to ID passed

Request URL: localhost:8080/api/patrons

Response : 200 status with data fetched

Sample:

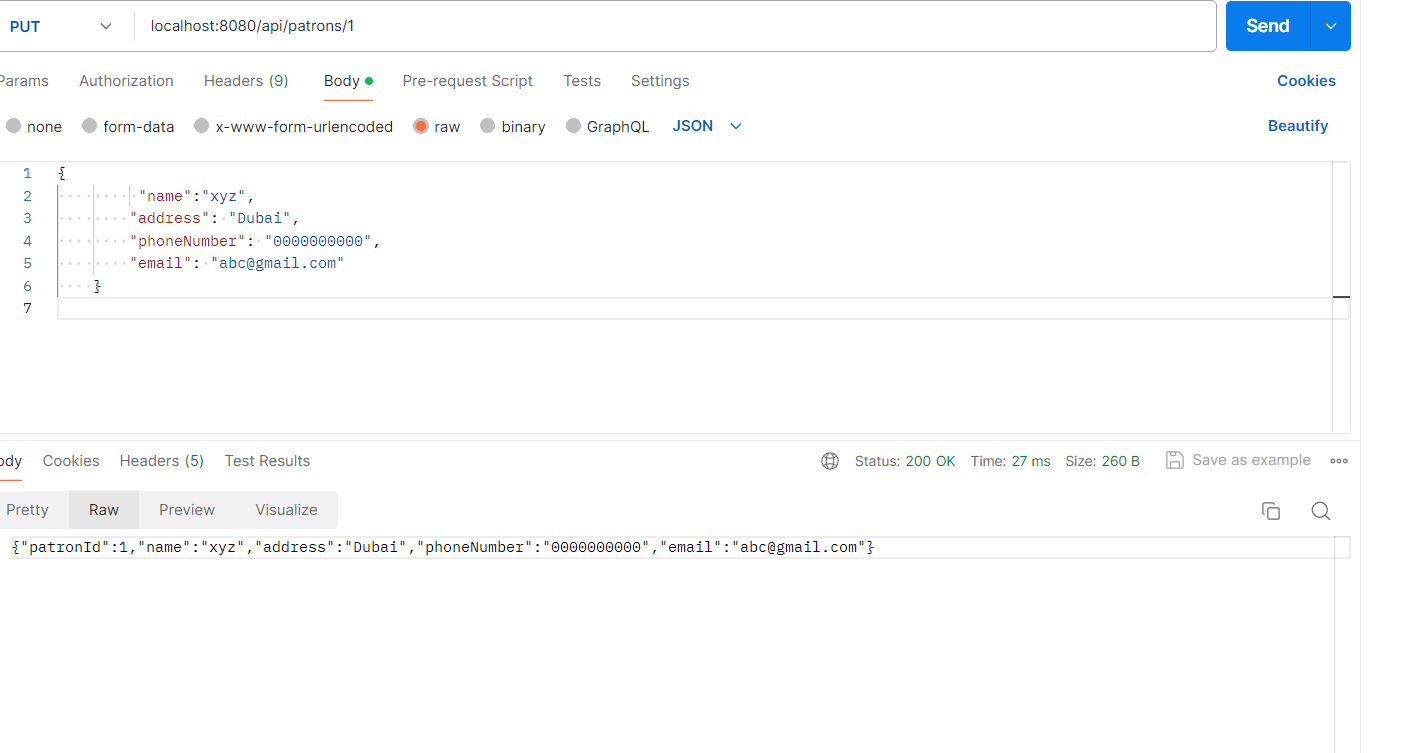


PUT : To update data in db

Request URL : localhost:8080/api/patrons/{id}

Response : 200 status code with updated data

Sample:



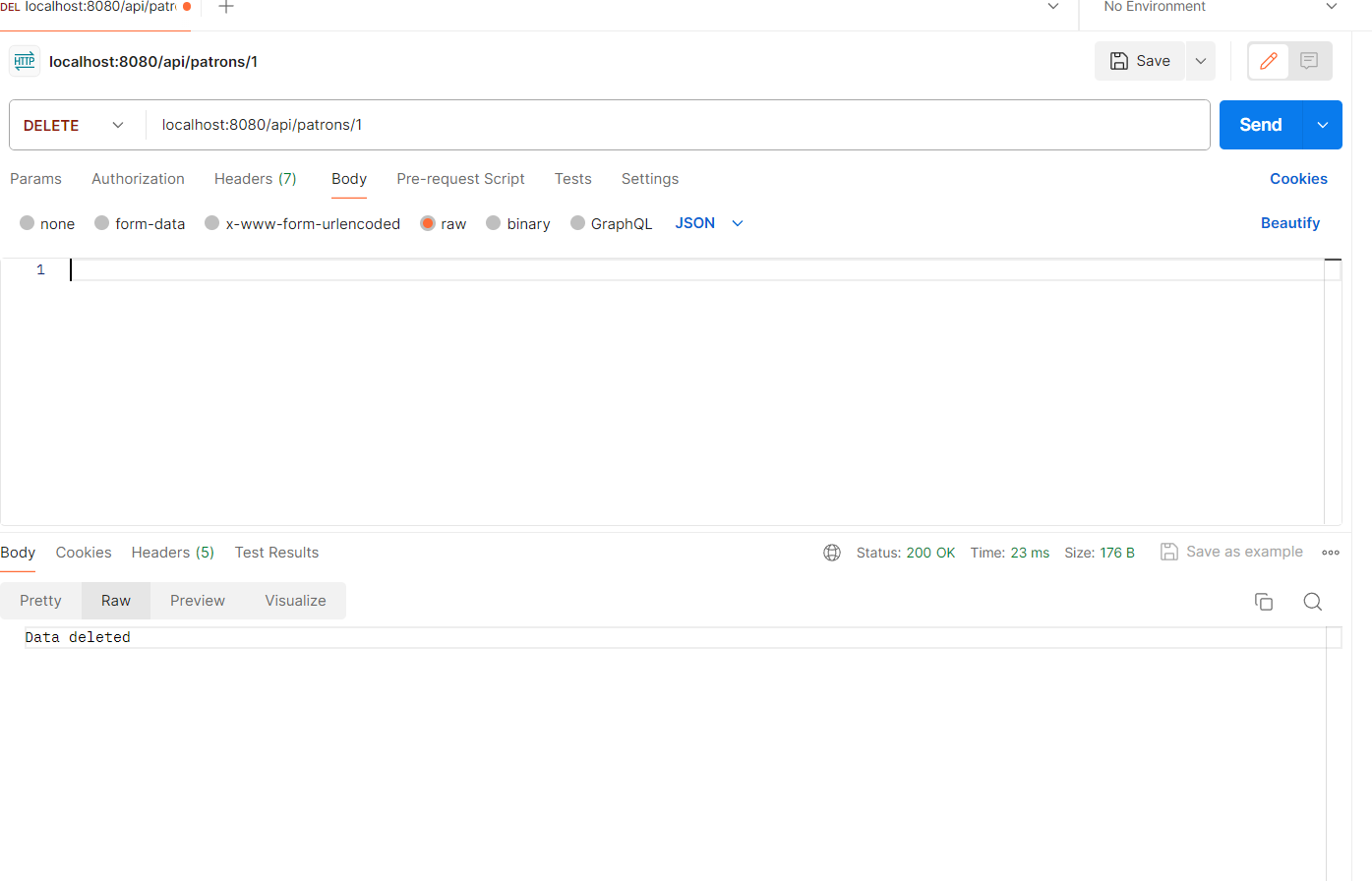
In case of no data, it will return resource not found exception .

DELETE: To data record based on Id passed

Request URL: localhost:8080/api/patrons/{id}

Response : 200 status code with message “Data deleted”

Sample:



1. **Borrowing Entity**: This entity contains attributes for borrowing enitiy.

The attributes are:

Book, patron, borrowedDate, returnDate, bookId and patronId

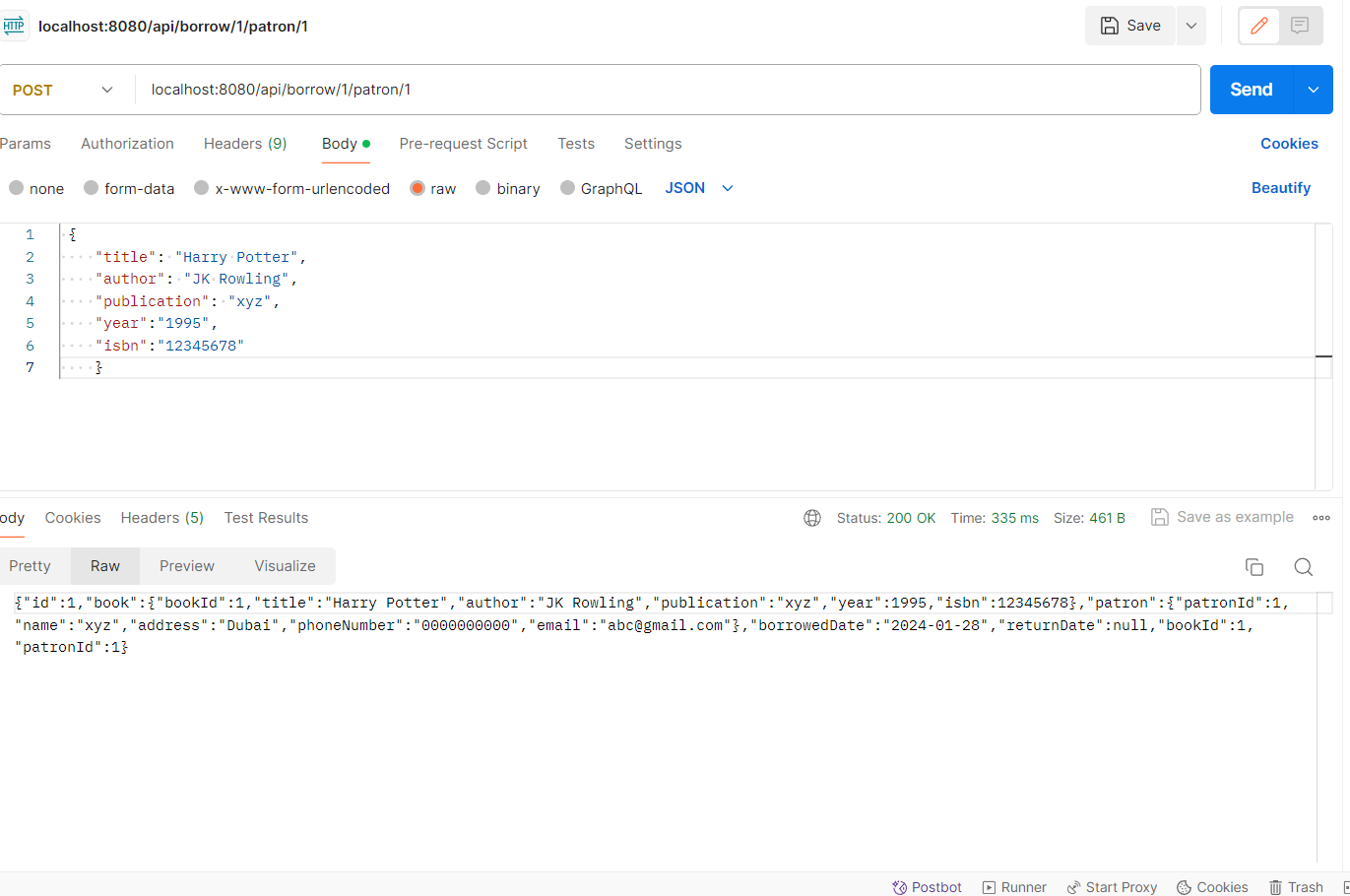
APIs:

POST : To update entries table with borrowed date

Request URL: localhost:8080/api/borrow/{bookId}/patron/{patronId}

Response : 200 status code with borrowed date updated.Here returned date will be null

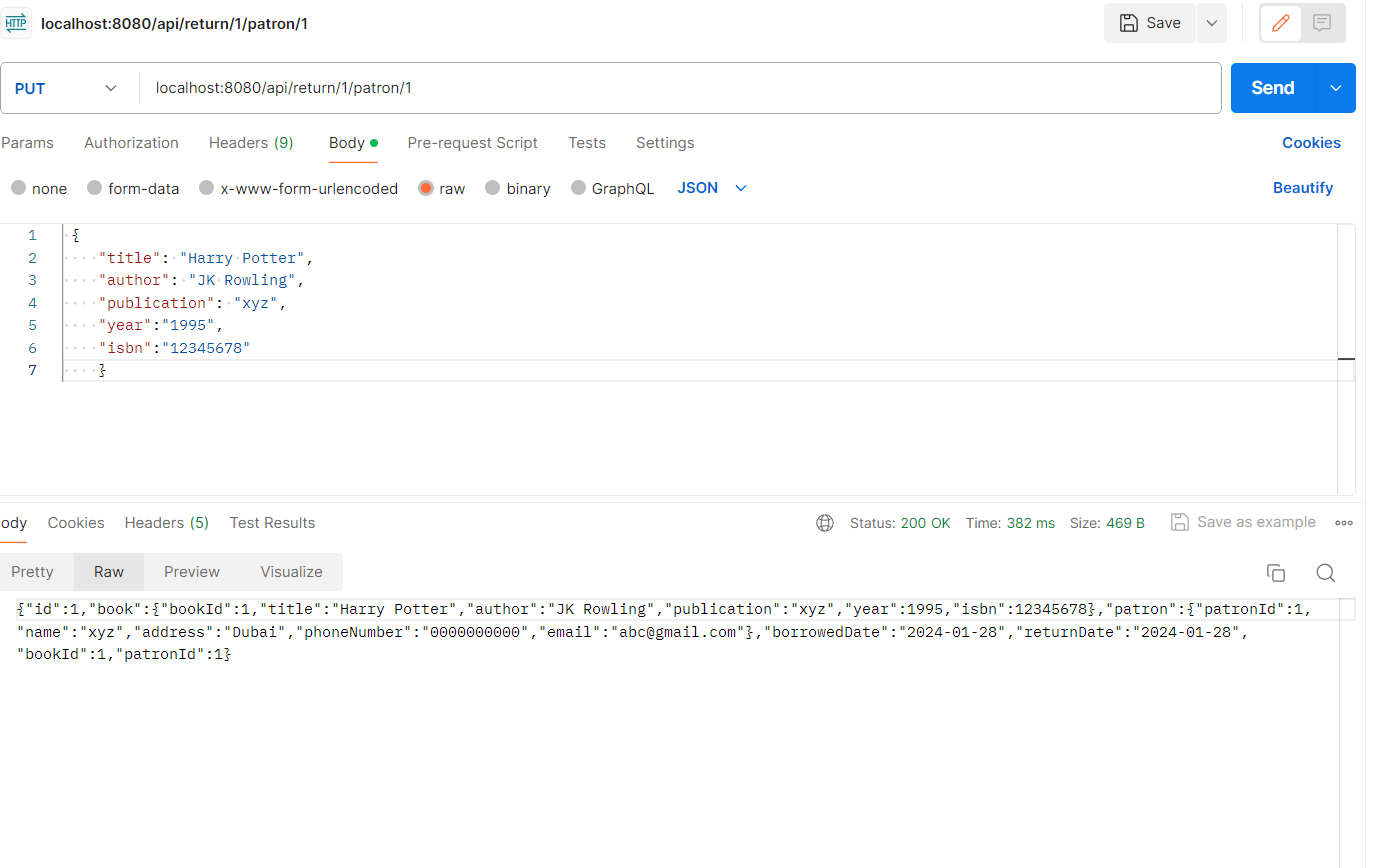
Here the existence of patronId and bookId in DB is checked before posting the entries



* PUT : This service is called when borrowed item is returned. The returned date is updated in this call

Request URL : localhost:8080/api/return/{bookId}/patron/{patronID}

Response : 200 status code with Updated entries



**Other Facts:**

* Logging mechanism is available and the log file is available in the folder :

./logs/app

* Caching mechanism is enabled