RESTful API for a Bookstore Management System API Documentation

Overview

This documentation outlines the RESTful API for the Bookstore Management System. The API allows users to add, retrieve, update, and delete book information. It's built using Python with the Flask framework and SQL Alchemy ORM, supporting basic and JWT authentication.

Table of Contents

Authentication

- **Login Endpoint:**

  - **URL:** `/auth/login`

  - **Method:** POST

  - **Body:**

    - `username` - String

    - `password` - String

  - **Response:** JWT Token for authenticated requests.

Endpoints

1. Adding a new book

Endpoint: /books

Method: POST

Request:

Body should contain JSON data with book details:

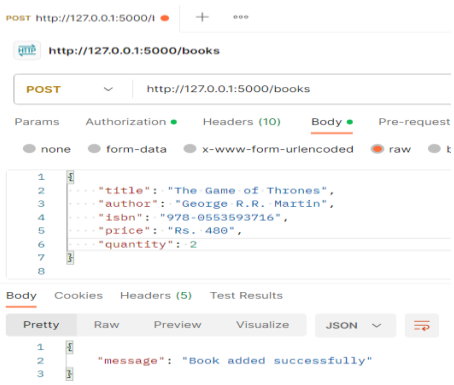
```
{
  "title": "The Life of John Doe",
  "author": "John Doe",
  "isbn": "123-4567890123",
  "price": "Rs. 123",
  "quantity": 10
}
```

Authorization: Basic [base64-encoded-username:password]

Response:

Successful response (HTTP status code 200):

```
{
  "message": "Book added successfully"
}
```



Error response (HTTP status code 500):

{

"message": "Failed to add book"

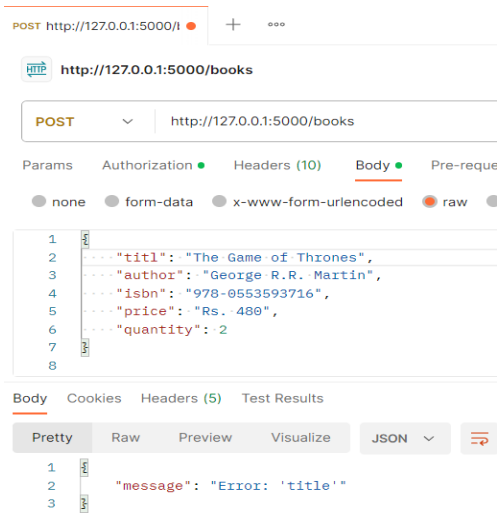
}

or

{

"message": "Error ‘{parameter}’"

}



2. Retrieving all books

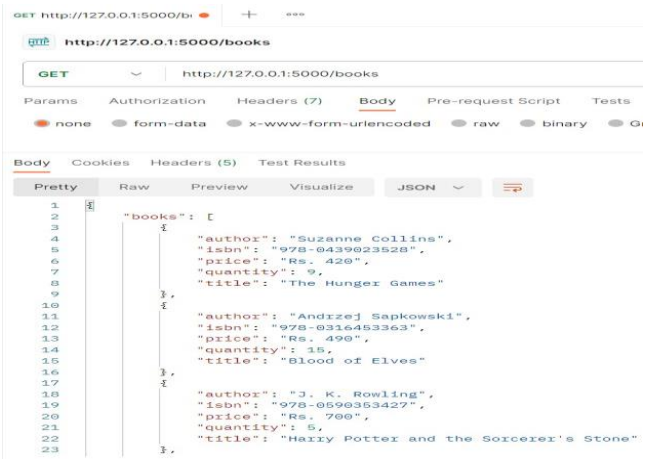
Endpoint: /books

Method: GET

Response:

Successful response (HTTP status code 200):

```
{
   "books": [
        {
          "title": "The Life of John Doe",

          "author": "John Doe",

          "isbn": "123-4567890123",

          "price": "Rs. 123",

          "quantity": 10

        },

        // ... other books
        ]
```



3. Retrieving a specific book by ISBN

Endpoint: /books/{isbn}

Method: GET

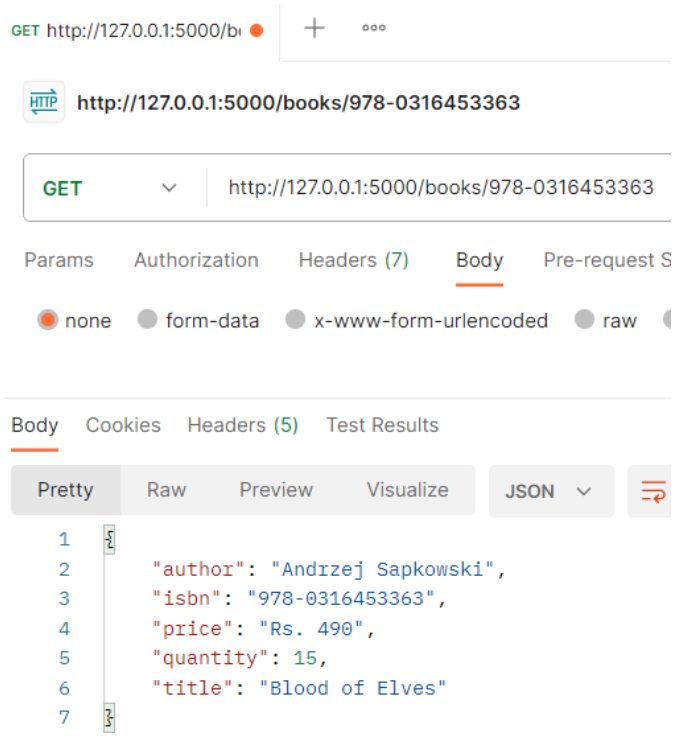
Path Parameters:

isbn: ISBN number of the book.

Response:

Successful response (HTTP status code 200):


```
{
  "title": "The Life of John Doe",
  "author": "John Doe",
  "isbn": "123-4567890123",
  "price": "Rs. 123",
  "quantity": 10
}
```

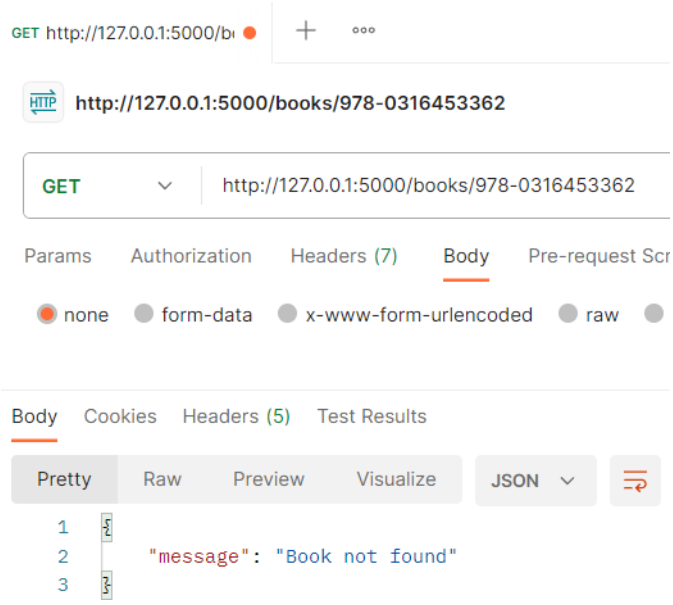



Error response (HTTP status code 404):

{

"message": "Book not found"

}



4. Updating book details

Endpoint: /books/{isbn}

Method: PUT

Path Parameters:

isbn: ISBN number of the book.

Request:

Body should contain JSON data with updated book details.

{

"title": "The Life of John Doe",

"author": "John Doe",

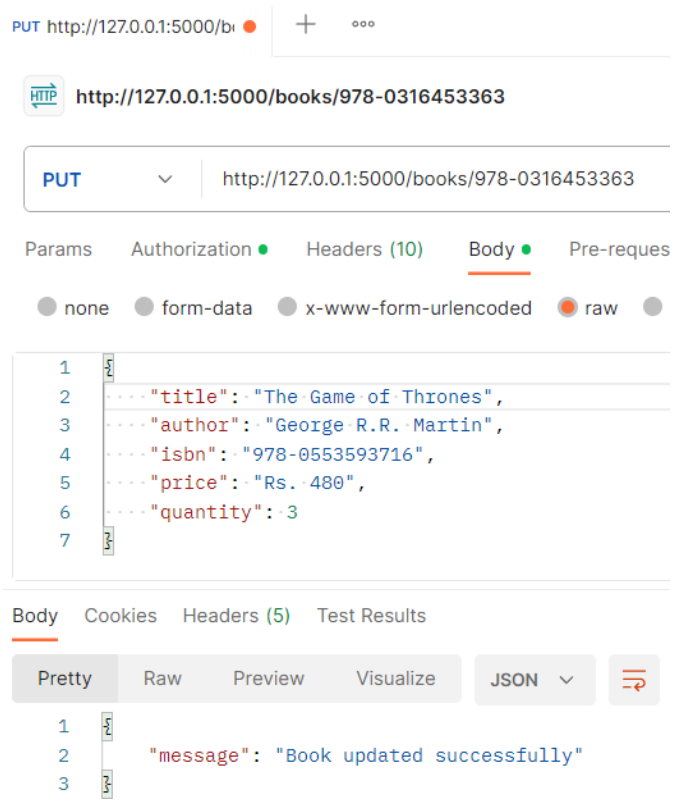
```
  "isbn": "123-4567890123",
  "price": "Rs. 123",
  "quantity": 5
}
```


Authorization: Basic [base64-encoded-username:password]

Response:

Successful response (HTTP status code 200):


```
{
  "message": "Book updated successfully"
}
```

5. Deleting a book

Endpoint: /books/{isbn}

Method: DELETE

Path Parameters:

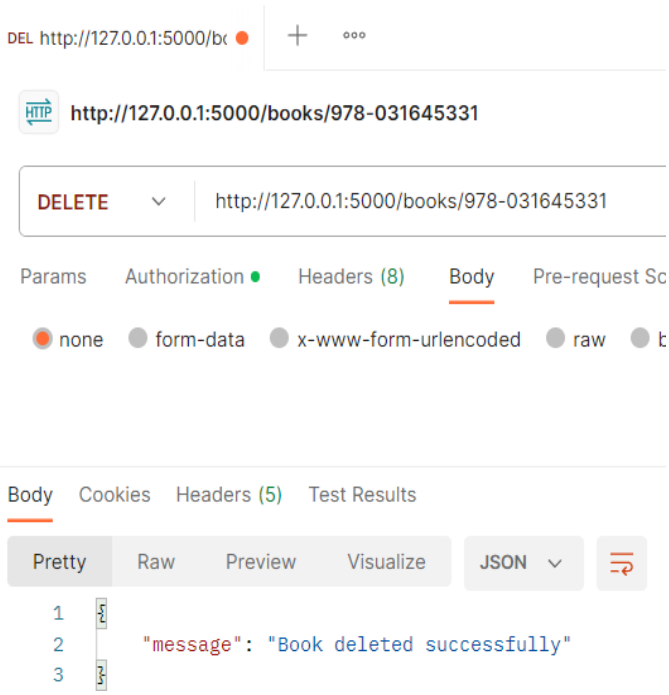
isbn: ISBN number of the book.

Authorization: Basic [base64-encoded-username:password]

Response:

Successful response (HTTP status code 200):

```
{
  "message": "Book deleted successfully"
}
```



Sample Data Creation

To create sample data, you can use the following example requests:

- **Add Books:**
  - **Endpoint:** `/books`
  - **Method:** POST
  - **Body:**
    ```json
    {
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "isbn": "1234567890",
      "price": 10.99,
      "quantity": 5
    }
    ```
    ```json
    {
      "title": "1984",
      "author": "George Orwell",
      "isbn": "1234567891",
      "price": 8.99,
      "quantity": 10
    }
    ```

Using the API

To use the API, follow these steps:

1. **Start the API Server:** Run the Flask application.
2. **Login:** Use the login endpoint to obtain a JWT token.
3. **Make Requests:** Use the provided endpoints to manage bookstore data.

Error Handling

All endpoints will return appropriate HTTP status codes for successful operations as well as for various errors (e.g., `400 Bad Request`, `401 Unauthorized`, `404 Not Found`, `500 Internal Server Error`).