# API Testing using Postman (Screenshots and summary)

**Project Name:** Practice Website

**Prepared By:** Reeti Thapa

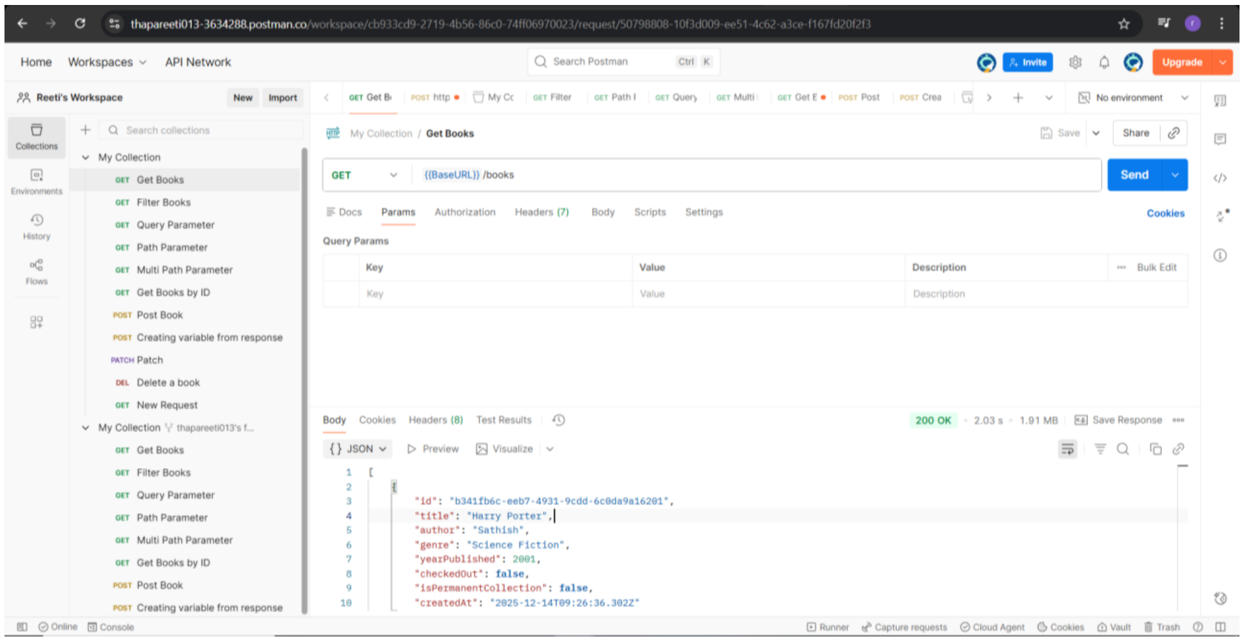
**Tools used:** Postman

**Description:**

This document contains screenshots and explanations of API testing performed using Postman. The APIs were tested using GET, POST, PATCH, and DELETE methods, including positive and negative scenarios.

**Note:** *This document contains selected API Testing using Postman screenshots as visual proof of my testing work. The full test case list with other details is available in* ***06_API_Testing-Postman.xlsx***

## 1. Test Scenario: Get all books

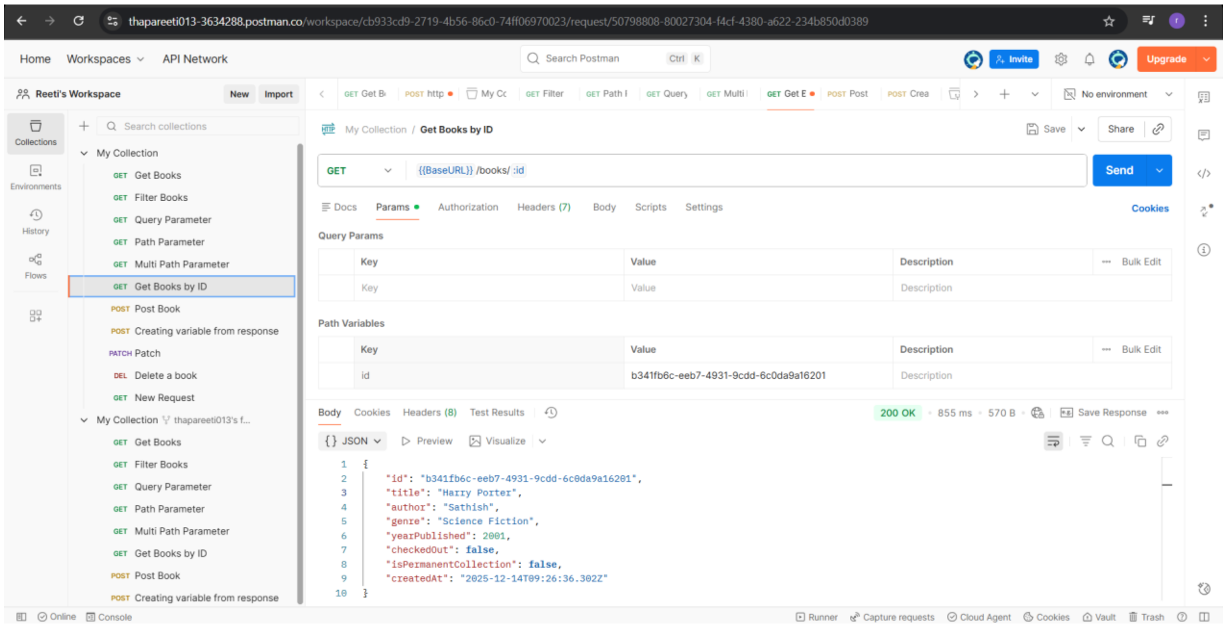

thapareeti013-3634288.postman.co/workspace/cb933cd9-2719-4b56-86c0-74ff06970023/request/50798808-10f3d009-ee51-4c62-a3ce-f167fd20f2f3
Reeti's Workspace
My Collection / Get Books
Send
200 OK · 2.03 s · 1.91 MB

GET {{BaseURL}} /books

```
[
    {
        "id": "b341fb6c-eeb7-4931-9cdd-6c0da9a16201",
        "title": "Harry Porter",
        "author": "Sathish",
        "genre": "Science Fiction",
        "yearPublished": 2001,
        "checkedOut": false,
        "isPermanentCollection": false,
        "createdAt": "2025-12-14T09:26:36.302Z"
```

**Execution Note:** GET Request returned 200 OK with list of books

## 2. Test Scenario: Get book by valid ID



GET {{BaseURL}} /books/ :id

Path Variables

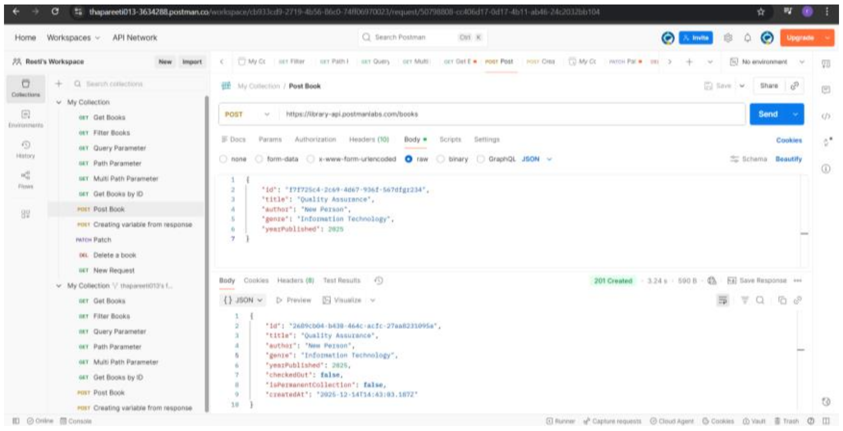
| Key | Value | Description |
|---|---|---|
| id | b341fb6c-eeb7-4931-9cdd-6c0da9a16201 | |

200 OK · 855 ms · 570 B

```
{
    "id": "b341fb6c-eeb7-4931-9cdd-6c0da9a16201",
    "title": "Harry Porter",
    "author": "Sathish",
    "genre": "Science Fiction",
    "yearPublished": 2001,
    "checkedOut": false,
    "isPermanentCollection": false,
    "createdAt": "2025-12-14T09:26:36.302Z"
}
```

**Execution Note:** GET request returned 200 OK with correct book details for the given ID

## 3. Test scenario: Create book with valid data



POST https://library-api.postmanlabs.com/books

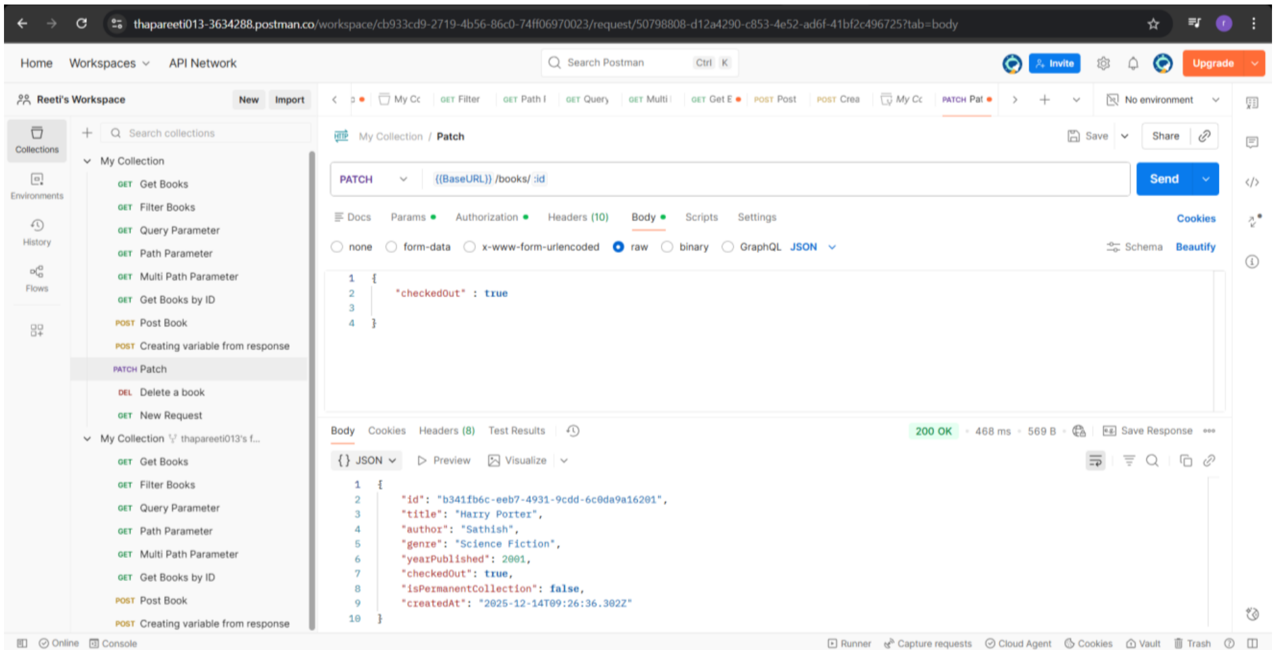
```
{
    "id": "f7f725c4-2c69-4d67-936f-567dfgr234",
    "title": "Quality Assurance",
    "author": "New Person",
    "genre": "Information Technology",
    "yearPublished": 2025
}
```

201 Created · 3.24 s · 590 B

```
{
    "id": "2689cb04-b438-464c-acfc-27aa8231095a",
    "title": "Quality Assurance",
    "author": "New Person",
    "genre": "Information Technology",
    "yearPublished": 2025,
    "checkedOut": false,
    "isPermanentCollection": false,
    "createdAt": "2025-12-14T14:43:03.187Z"
}
```

**Execution Note:** POST request executed successfully. API returned 201 Created with a generated book ID.

## 4. Test Scenario: Update book using PATCH



thapareeti013-3634288.postman.co/workspace/cb933cd9-2719-4b56-86c0-74ff06970023/request/50798808-d12a4290-c853-4e52-ad6f-41bf2c496725?tab=body

My Collection / Patch

PATCH {{BaseURL}} /books/ :id

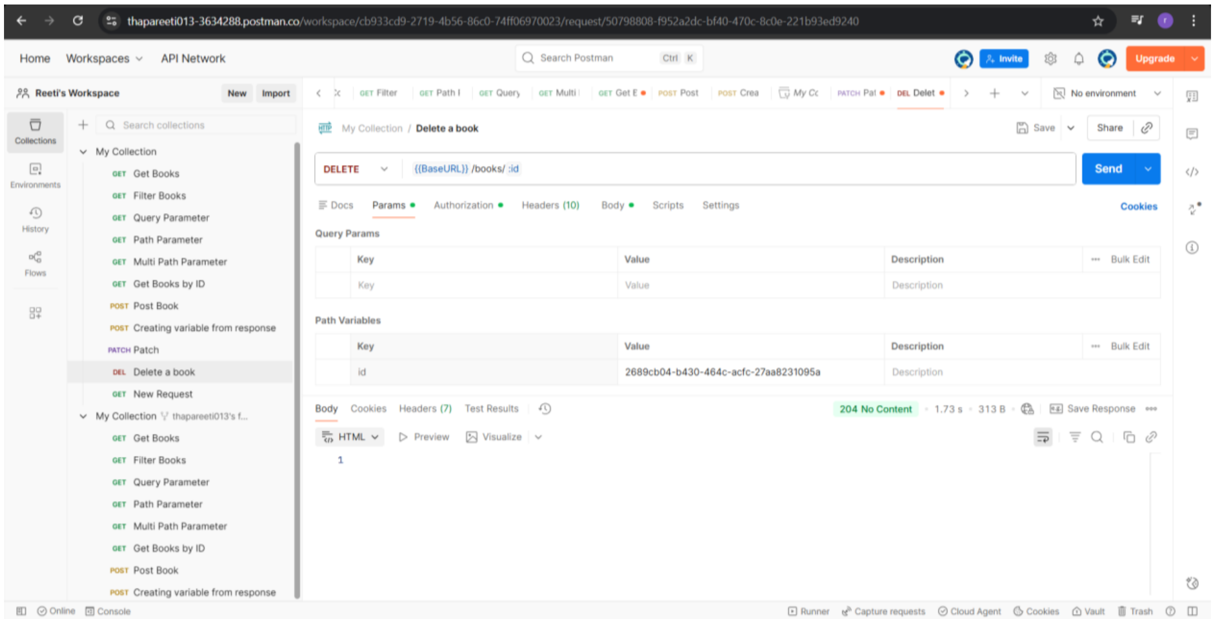
```
{
    "checkedOut" : true
}
```

200 OK · 468 ms · 569 B

```
{
    "id": "b341fb6c-eeb7-4931-9cdd-6c0da9a16201",
    "title": "Harry Porter",
    "author": "Sathish",
    "genre": "Science Fiction",
    "yearPublished": 2001,
    "checkedOut": true,
    "isPermanentCollection": false,
    "createdAt": "2025-12-14T09:26:36.302Z"
}
```

**Execution Note:** Book details updated successfully. API returned 200 OK

## 5. Test Scenario: Delete book with valid ID



thapareeti013-3634288.postman.co/workspace/cb933cd9-2719-4b56-86c0-74ff06970023/request/50798808-f952a2dc-bf40-470c-8c0e-221b93ed9240

My Collection / Delete a book

DELETE {{BaseURL}} /books/ :id

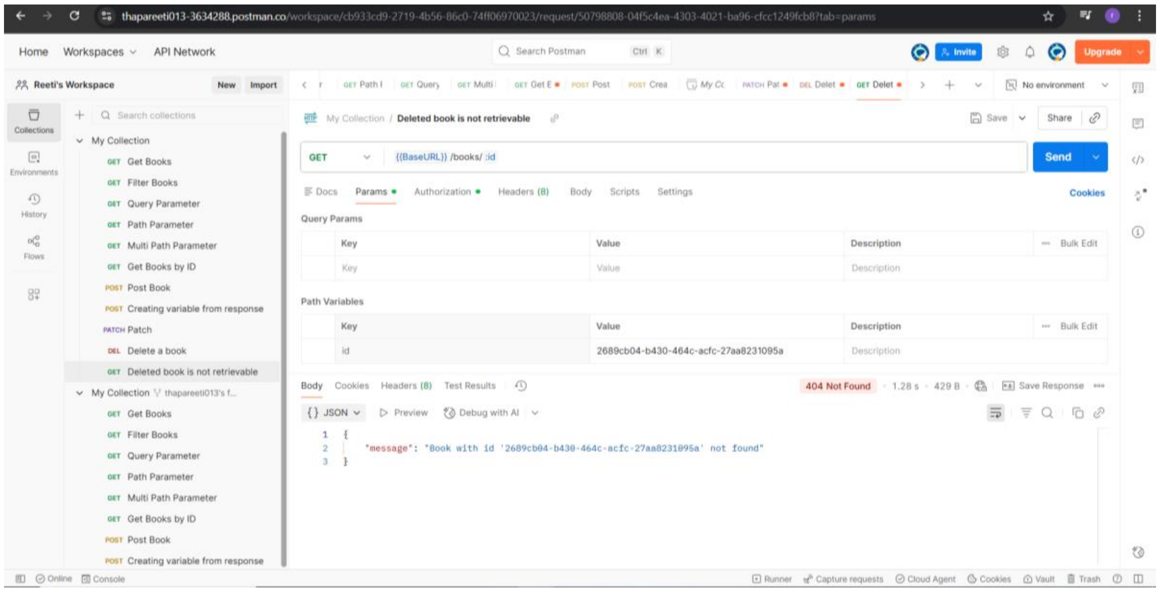
Path Variables

| Key | Value | Description |
|---|---|---|
| id | 2689cb04-b430-464c-acfc-27aa8231095a | |

204 No Content · 1.73 s · 313 B

**Execution Note:** Book deleted successfully with 204 No Content response.

## 6. Test Scenario: Verify deleted book is not retrievable



thapareeti013-3634288.postman.co/workspace/cb933cd9-2719-4b56-86c0-74ff06970023/request/50798808-04f5c4ea-4303-4021-ba96-cfcc1249fcb8?tab=params

My Collection / Deleted book is not retrievable

GET {{BaseURL}} /books/ :id

Path Variables

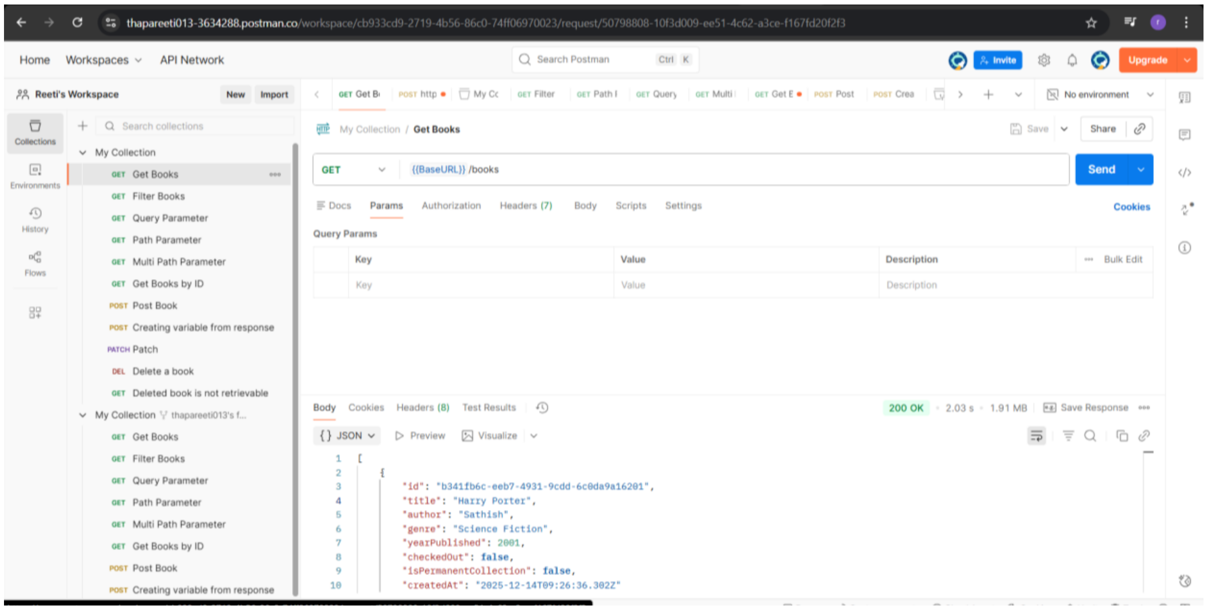
| Key | Value | Description |
|---|---|---|
| id | 2689cb04-b430-464c-acfc-27aa8231095a | |

404 Not Found · 1.28 s · 429 B

```
{
    "message": "Book with id '2689cb04-b430-464c-acfc-27aa8231095a' not found"
}
```

**Execution Note:** GET request returned 404 Not Found, confirming successful deletion.

## 7. Test Scenario: Verify BaseURl variable works



thapareeti013-3634288.postman.co/workspace/cb933cd9-2719-4b56-86c0-74ff06970023/request/50798808-10f3d009-ee51-4c62-a3ce-f167fd20f2f3

My Collection / Get Books

GET {{BaseURL}} /books

200 OK · 2.03 s · 1.91 MB

```
[
    {
        "id": "b341fb6c-eeb7-4931-9cdd-6c0da9a16201",
        "title": "Harry Porter",
        "author": "Sathish",
        "genre": "Science Fiction",
        "yearPublished": 2001,
        "checkedOut": false,
        "isPermanentCollection": false,
        "createdAt": "2025-12-14T09:26:36.302Z"
```

**Execution Note:** Request executed successfully using variable for BaseURL.

## Conclusion:

The API endpoints were tested successfully using Postman for positive and negative scenarios. Core CRUD operations (GET, POST, PATCH, DELETE) were validated, and responses were verified using appropriate status codes. This practice improved my understanding of API behavior and manual API testing.