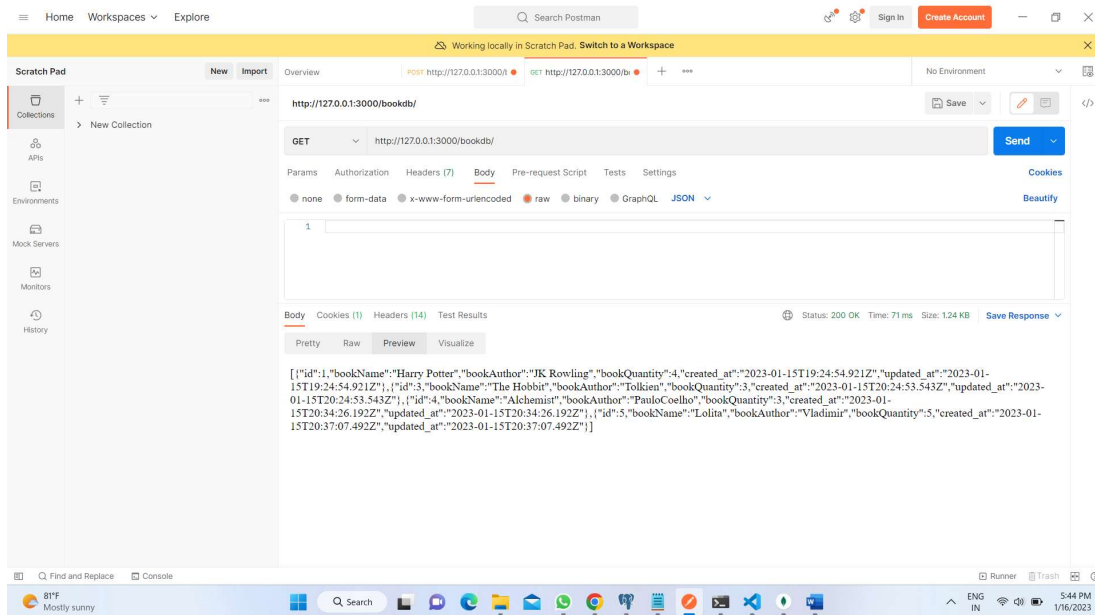
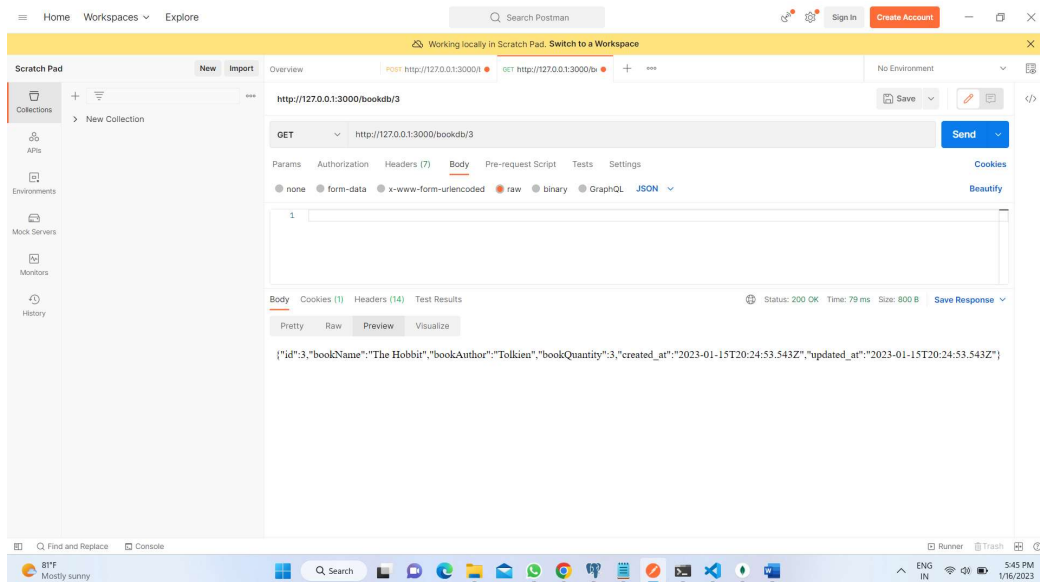


# Book stock management API

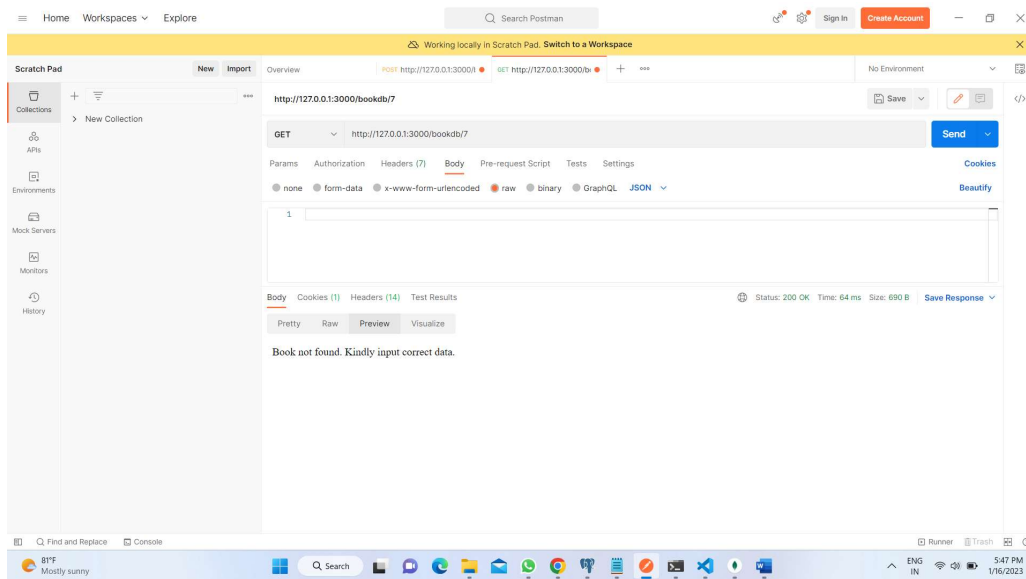
## List of all books



## Details of individual book – Valid id is provided

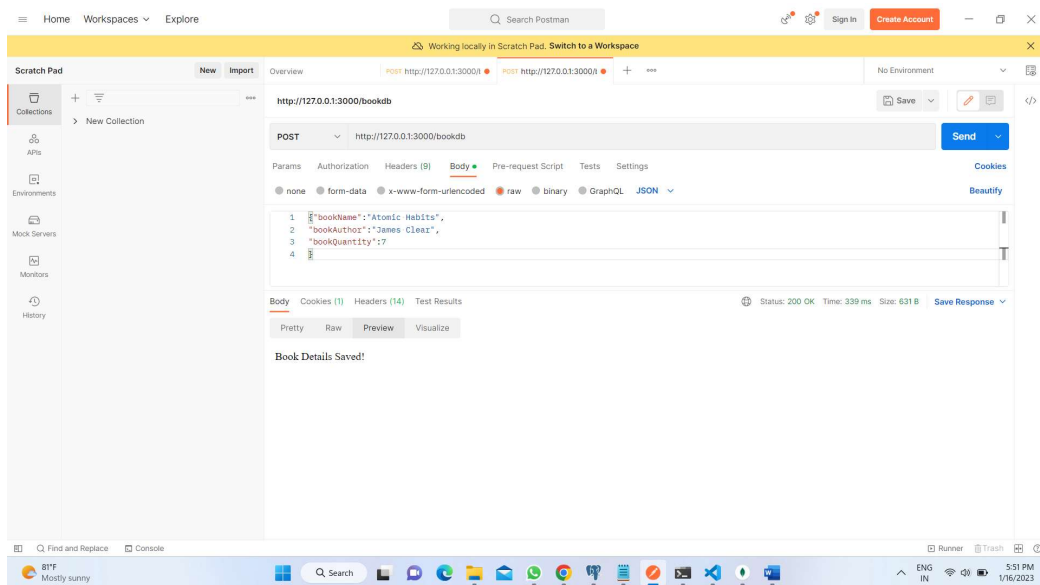


## Details of individual book – Invalid id is provided



The screenshot shows the Postman interface with a GET request to `http://127.0.0.1:3000/bookdb/7`. The request is saved, and the response is displayed. The status is 200 OK, and the body contains the message: "Book not found. Kindly input correct data."

## Add a new book



The screenshot shows the Postman interface with a POST request to `http://127.0.0.1:3000/bookdb`. The request body is a JSON object with the following details:

```
{  "bookName": "Atomic Habits",  "bookAuthor": "James Clear",  "bookQuantity": 7}
```

The response status is 200 OK, and the body contains the message: "Book Details Saved!"

## Add same book again

Postman interface showing a POST request to `http://127.0.0.1:3000/bookdb`. The request body is a JSON object:

```
{  "bookName": "Atomic Habits",  "bookAuthor": "James Clear",  "bookQuantity": 7}
```

The response status is 200 OK. The response body is:

```
Book already exists, Kindly update quantity
```

## Update Book

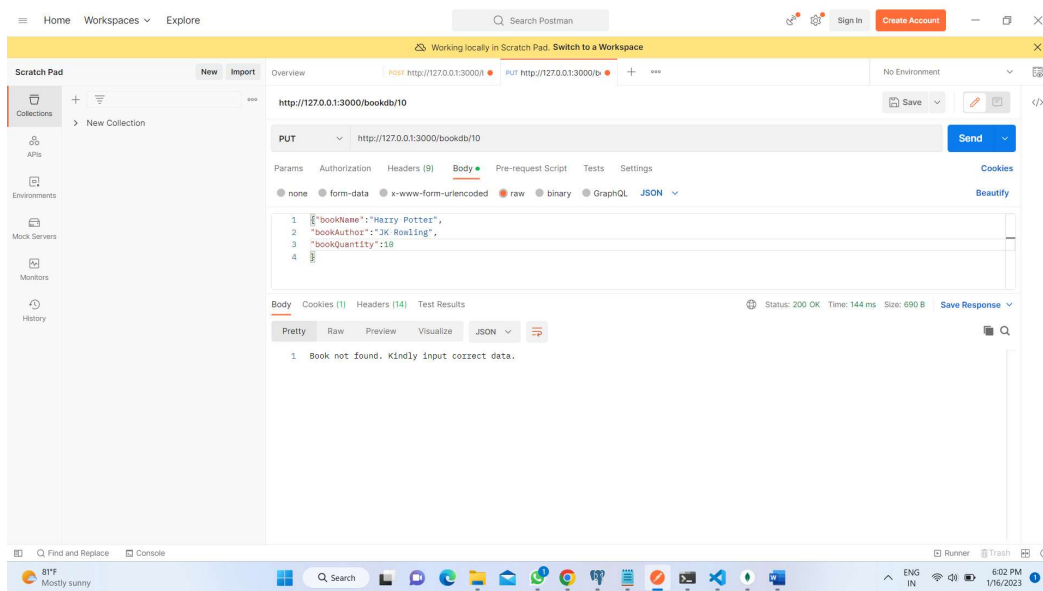
Postman interface showing a PUT request to `http://127.0.0.1:3000/bookdb/1`. The request body is a JSON object:

```
{  "bookName": "Harry Potter",  "bookAuthor": "JK Rowling",  "bookQuantity": 10}
```

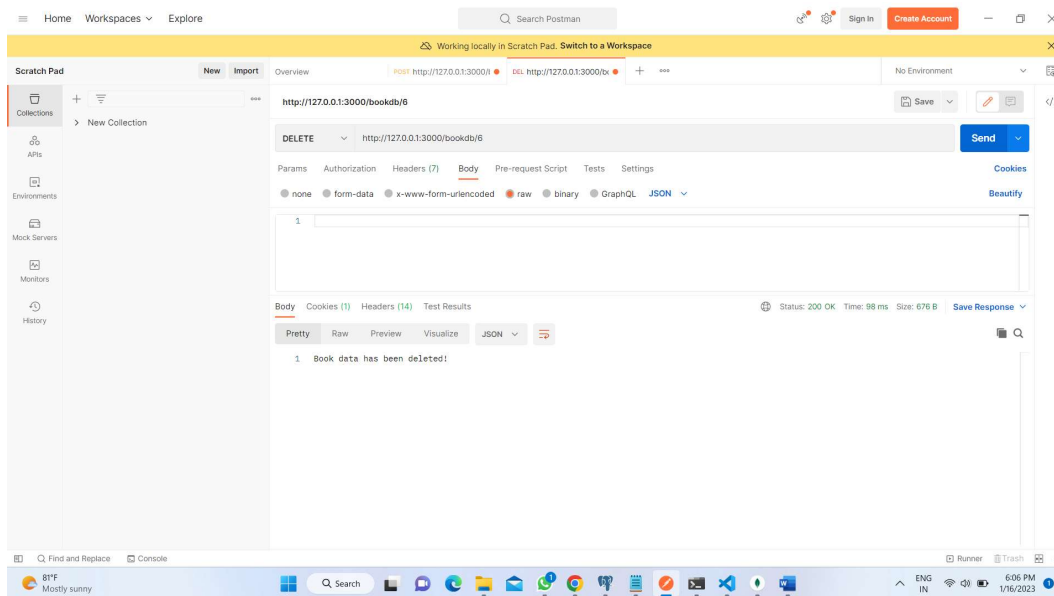
The response status is 200 OK. The response body is:

```
1. Book Data Updated Successfully!
```

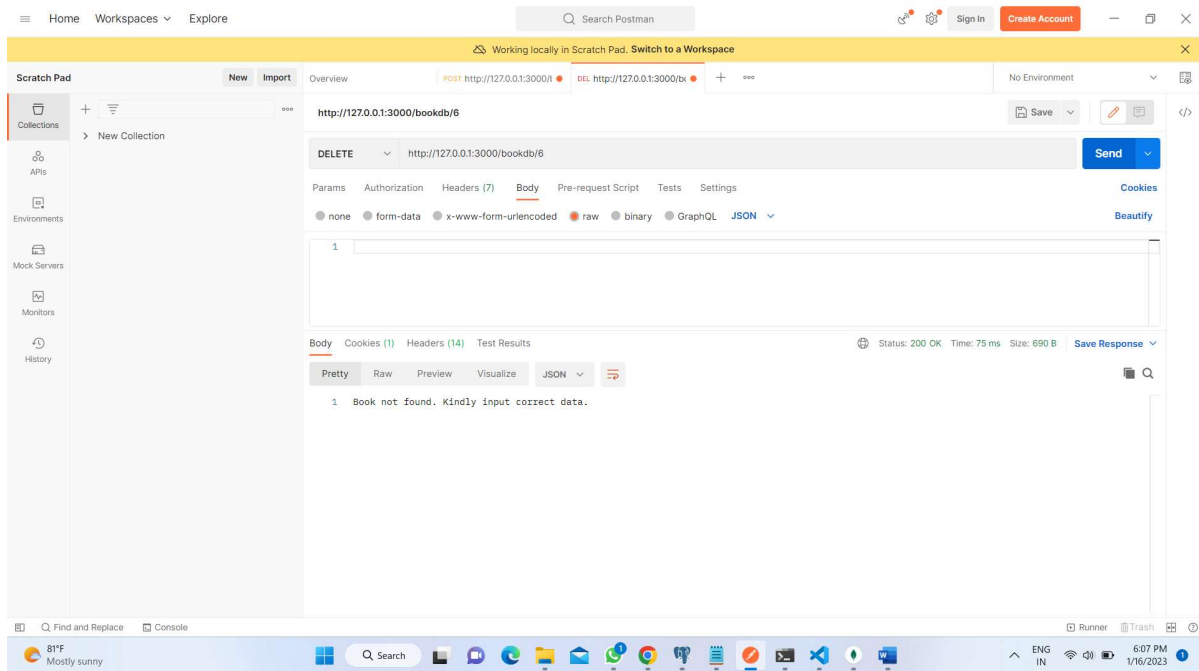
## Provide **Invalid** id while updating



## Delete Book



## Delete book – provide invalid id

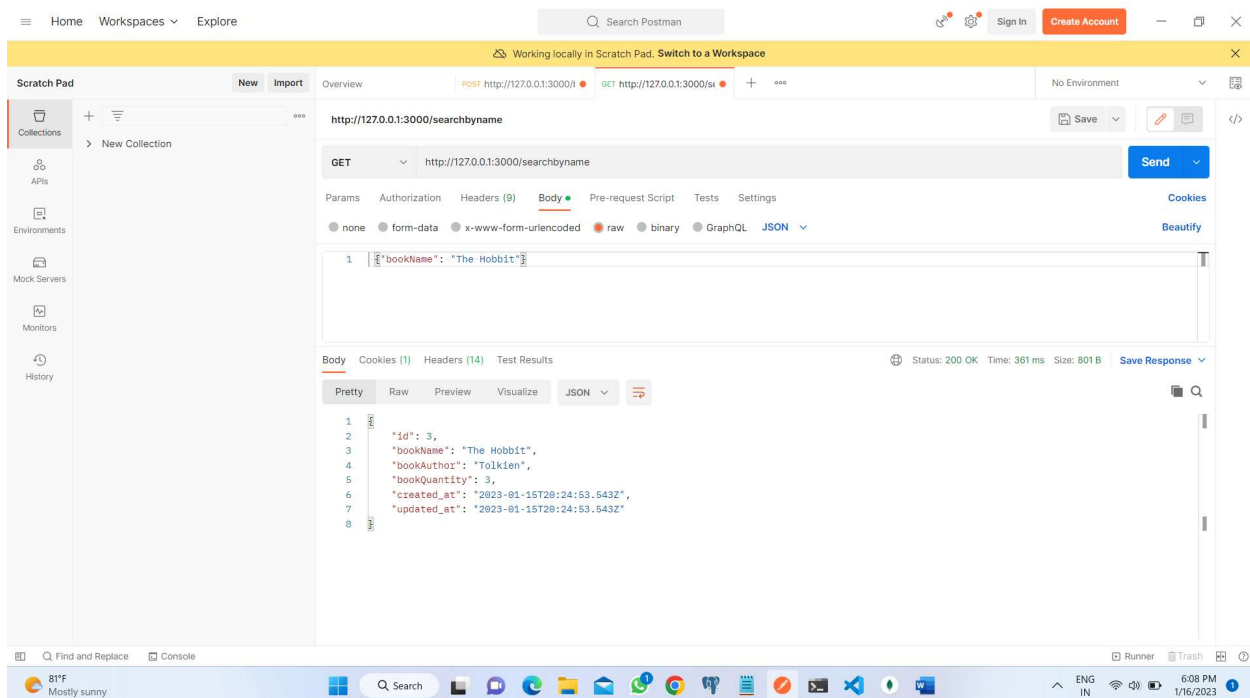


The screenshot shows the Postman interface with a DELETE request to `http://127.0.0.1:3000/bookdb/6`. The request is in the "Body" tab, and the response is displayed in the "Body" tab. The response status is 200 OK, and the message is "Book not found. Kindly input correct data."

Request: `DELETE http://127.0.0.1:3000/bookdb/6`

Response: `1 Book not found. Kindly input correct data.`

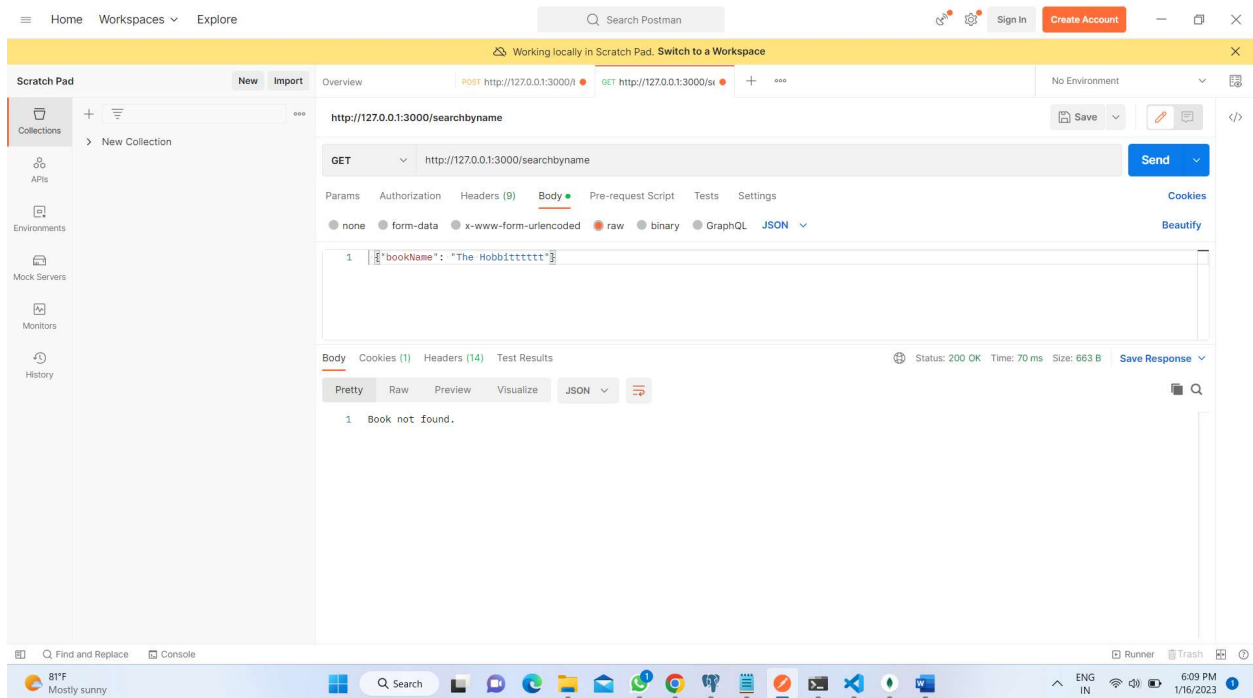
## Search by Name :



The screenshot shows the Postman interface with a GET request to `http://127.0.0.1:3000/searchbyname`. The request is in the "Body" tab, and the response is displayed in the "Body" tab. The response status is 200 OK, and the JSON object contains the following details:

```
{
  "id": 3,
  "bookName": "The Hobbit",
  "bookAuthor": "Tolkien",
  "bookQuantity": 3,
  "created_at": "2023-01-15T20:24:53.543Z",
  "updated_at": "2023-01-15T20:24:53.543Z"
}
```

## Search by Name: Invalid Name

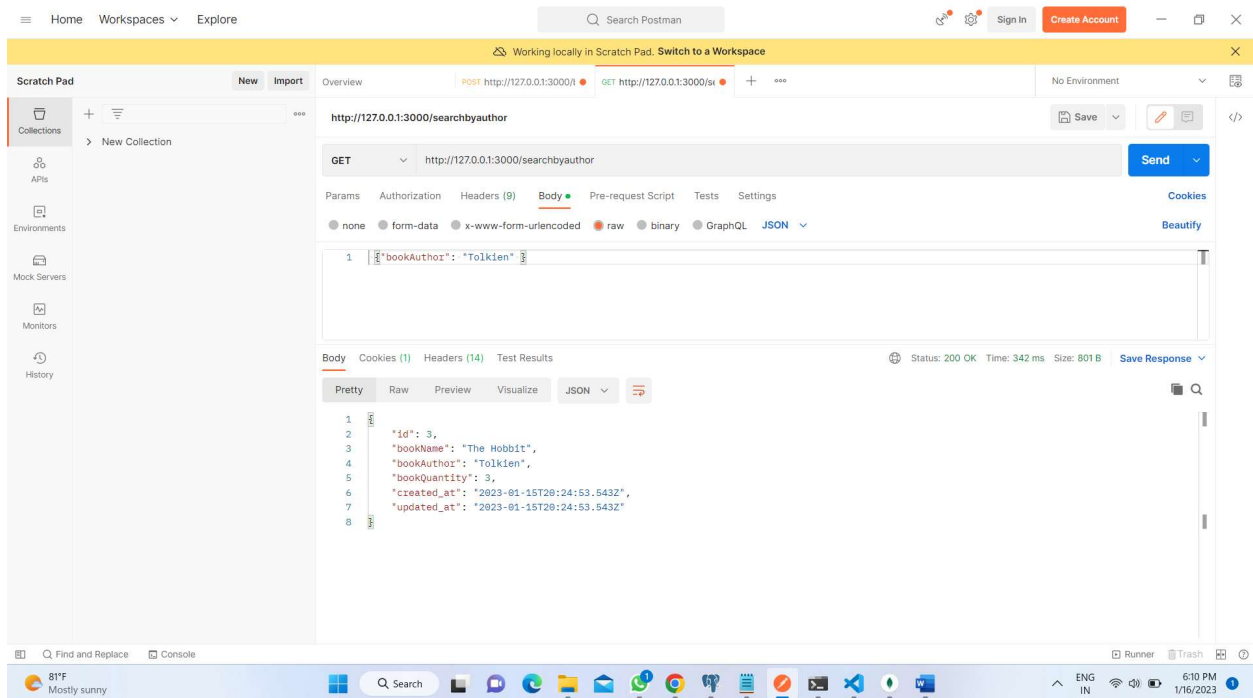


The screenshot shows the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', and 'Explore'. A search bar is present. The main workspace is titled 'Working locally in Scratch Pad. Switch to a Workspace'. The left sidebar shows 'Scratch Pad' with 'Collections' and 'New Collection'. The main panel displays a GET request to 'http://127.0.0.1:3000/searchbyname'. The request body is a JSON object: 

```
{ "bookName": "The Hobbitttttt" }
```

. The response status is '200 OK' with a time of '70 ms' and size of '663 B'. The response body is 'Book not found.'.

## Search By Author:



The screenshot shows the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', and 'Explore'. A search bar is present. The main workspace is titled 'Working locally in Scratch Pad. Switch to a Workspace'. The left sidebar shows 'Scratch Pad' with 'Collections' and 'New Collection'. The main panel displays a GET request to 'http://127.0.0.1:3000/searchbyauthor'. The request body is a JSON object: 

```
{ "bookAuthor": "Tolkien" }
```

. The response status is '200 OK' with a time of '342 ms' and size of '801 B'. The response body is a JSON object: 

```
{ "id": 3, "bookName": "The Hobbit", "bookAuthor": "Tolkien", "bookQuantity": 3, "created_at": "2023-01-15T20:24:53.543Z", "updated_at": "2023-01-15T20:24:53.543Z" }
```

.

## Search By Author: Invalid Author

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'Home', 'Workspaces', and 'Explore'. A search bar labeled 'Search Postman' is present. Below this, a yellow banner indicates 'Working locally in Scratch Pad. Switch to a Workspace'. The main interface is divided into a left sidebar and a main workspace. The sidebar has a 'Scratch Pad' section with 'New' and 'Import' buttons, and a 'Collections' section with a 'New Collection' button. The main workspace shows a REST client request for 'http://127.0.0.1:3000/searchbyauthor'. The request is a GET method. The 'Body' tab is selected, showing a JSON body: 

```
{ "bookAuthor": "Tolkiennnnn" }
```

. The response status is 200 OK, with a time of 114 ms and a size of 665 B. The response body is displayed in the 'Body' tab, showing the message: 

```
1 Author not found.
```

. The bottom of the screen shows a Windows taskbar with various application icons and a system tray displaying the date and time as 6:10 PM on 1/16/2023.

Home Workspaces Explore Search Postman

Working locally in Scratch Pad. Switch to a Workspace

Scratch Pad New Import Overview POST http://127.0.0.1:3000/ GET http://127.0.0.1:3000/searchbyauthor No Environment

GET http://127.0.0.1:3000/searchbyauthor Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 { "bookAuthor": "Tolkiennnnn" }
```

Body Cookies (1) Headers (14) Test Results Status: 200 OK Time: 114 ms Size: 665 B Save Response

Pretty Raw Preview Visualize JSON

```
1 Author not found.
```

Find and Replace Console Runner Trash 81°F Mostly sunny Search ENG IN 6:10 PM 1/16/2023