Project1

DigitalBook

17.03.2024

Jingwen Li

C O N T E N T S

Tasks

Part .01

Weeks 1: Digitizing "Readers HavenInventory

> A local bookstore, "Readers Haven", has been operating traditionally for the past 30 years. The owner, Mrs. Smith, has recently seen a decline in sales due to the rise of online platforms. To compete, she wants to digitize her inventory. Currently, she maintains a handwritten ledger containing details of books, including their titles, authors, and prices. She envisions a digital system where she can easily input this data and retrieve a list of all books. Furthermore, she often gets requests for books by specific authors, so she wants a feature where she can filter books based on the author's name.

Create a project in a programming language of your choice that will implement functions to:

1. Add a book to the inventory
2. Retrieve the entire inventory
3. Filter and display books by a specific author

Enhancing Search and Sorting Capabilities

> After successfully digitizing her inventory, Mrs. Smith found it time-consuming to scroll through the entire list to find a specific book. She needs a feature where she can search for a book by its title. Additionally, during stock-taking or when she's trying to determine which books to put on sale, she wants to sort the inventory either based on the price (from highest to lowest or vice versa) or alphabetically by title.

Expand the existing system by:

- Implementing a search function that matches partial or full titles and returns relevant results
- Adding sorting functions that can rearrange the inventory based on price (both ascending and descending) or title

Enhancing Search and Sorting Capabilities

> "Readers Havenïs a dynamic bookstore. New stock arrives, old editions get replaced, and sometimes, certain books are no longer sold. Mrs. Smith finds it cumbersome to manually delete or modify entries. She needs a feature where she can easily update the details of a book, such as its price or edition. Additionally, if a book is no longer available, she wants to remove it from the digital inventory.

Suggested Approach:
To answer to the dynamic nature of the inventory, you should:

- implement a function to modify existing book details, ensuring that changes are reflected accurately
- design a deletion function that removes a book based on its title or other unique identifier

Key points: *Integrity of the data; checking there is no duplicate entries and ensure that deletions don't accidentally remove multiple items*

Weekly clinic: Efficient Bulk Operations

> Twice a year, "Readers Haven"holds a grand sale. During this time, Mrs. Smith introduces many new titles and removes some old ones. She finds it tedious to add or remove books one by one. She needs a feature where she can handle bulk additions and deletions, ensuring the system remains efficient and does not crash during these operations.

Suggested Approach:
To handle bulk operations efficiently, you should:

- Design a function to add multiple books at once, possibly by accepting a list of books
- Create a function to remove multiple books based on a list of titles or unique identifiers

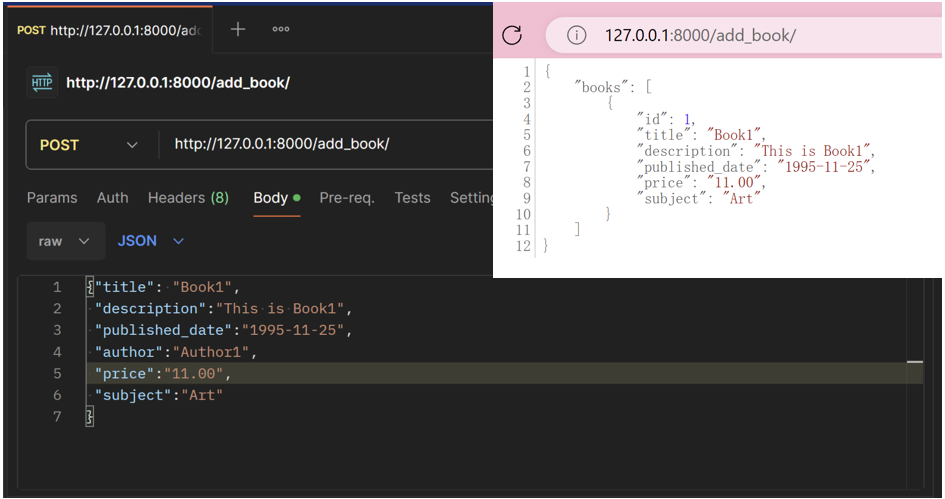
Key points: *Optimisation is key here - you should consider batch processing techniques and ensure that the system can handle large data inputs without significant lag or crashes*

Implemented functions

Part .02

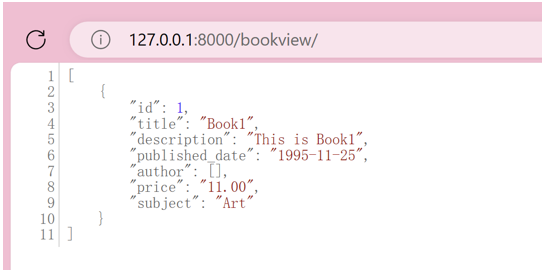
Implemented functions1-1 Add a book

Test by Postman



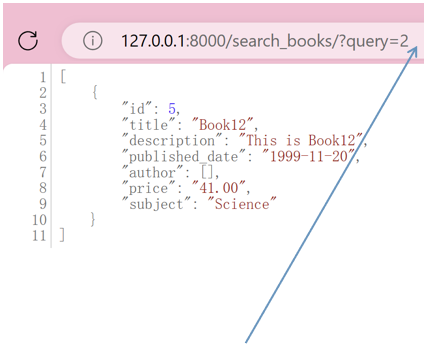
Implemented functions1-2 Retrieve entire inventory

Test by http://127.0.0.1:8000/bookview/

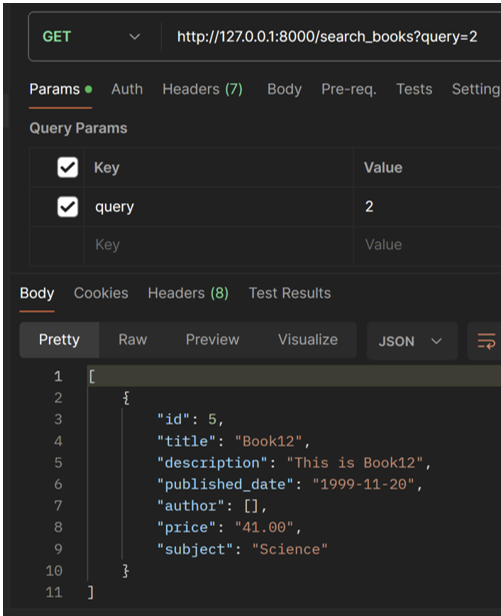


Implemented functions2-1Search

Test by **Postman** and **http://127.0.0.1:8000/search_books/?query=blabla**

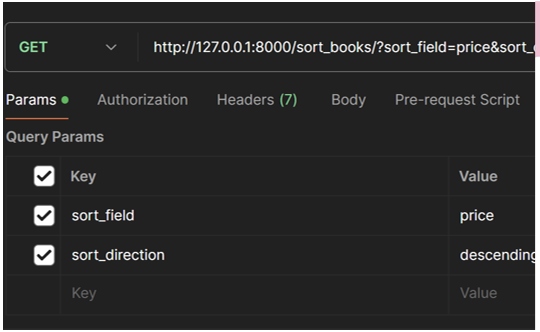


if no “?query=blabla”
this page will show all books



Implemented functions2-2 Sort

Test by **Postman** and **http://127.0.0.1:8000/sort_books/?sort_field=price&sort_direction=descending**

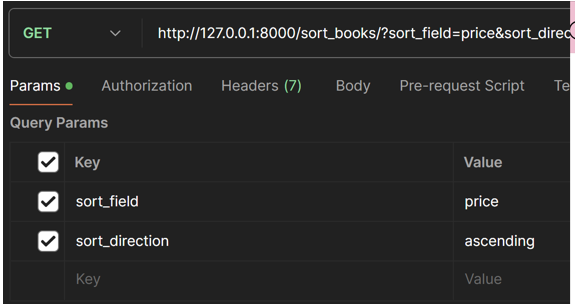


127.0.0.1:8000/sort_books/?sort_field=price&sort_direction=descending


```
[
    {
        "id": 5,
        "title": "Book12",
        "description": "This is Book12",
        "published_date": "1999-11-20",
        "author": [],
        "price": "41.00",
        "subject": "Science"
    },
    {
        "id": 6,
        "title": "Book2",
        "description": "This is Book2",
        "published_date": "1947-12-20",
        "author": [],
        "price": "21.00",
        "subject": "Maths"
    },
    {
        "id": 4,
        "title": "Book1",
        "description": "This is Book1",
        "published_date": "1995-11-25",
        "author": [],
        "price": "11.00",
        "subject": "Art"
    }
]
```

Implemented functions2-2 Sort

Test by **Postman** and **http://127.0.0.1:8000/sort_books/?sort_field=price&sort_direction=ascending**



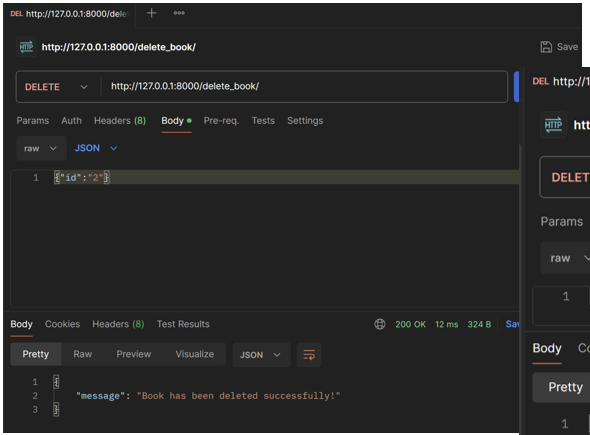
127.0.0.1:8000/sort_books/?sort_field=price&sort_direction=ascending

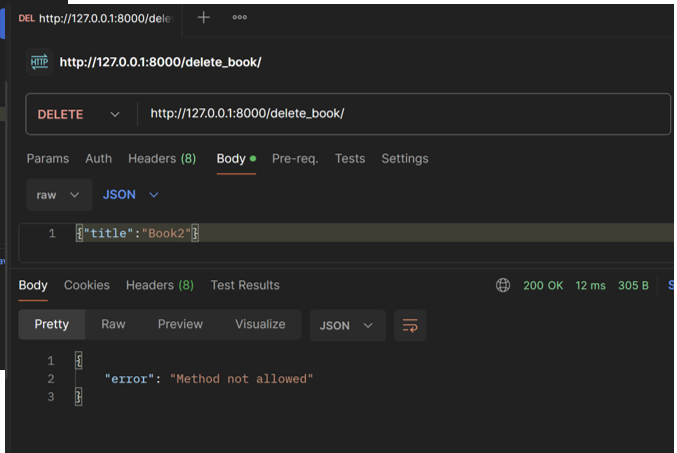

```
[
    {
        "id": 4,
        "title": "Book1",
        "description": "This is Book1",
        "published_date": "1995-11-25",
        "author": [],
        "price": "11.00",
        "subject": "Art"
    },
    {
        "id": 6,
        "title": "Book2",
        "description": "This is Book2",
        "published_date": "1947-12-20",
        "author": [],
        "price": "21.00",
        "subject": "Maths"
    },
    {
        "id": 5,
        "title": "Book12",
        "description": "This is Book12",
        "published_date": "1999-11-20",
        "author": [],
        "price": "41.00",
        "subject": "Science"
    }
]
```

Implemented functions3-2 Delete a book

Test by Postman

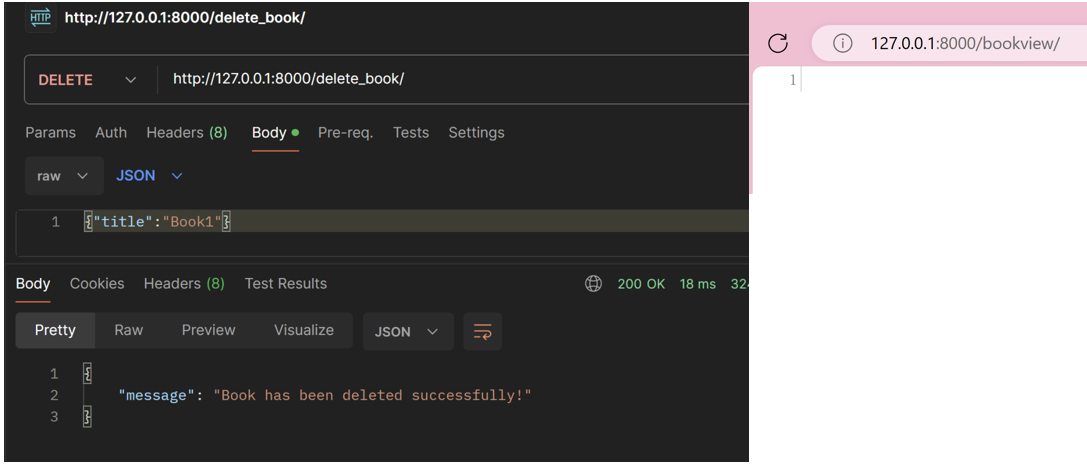
1 Enter an non-existent id of a book
2 Enter an non-existent title of a book





Implemented functions3-2 Delete a book

Test by Postman-- delete successfully

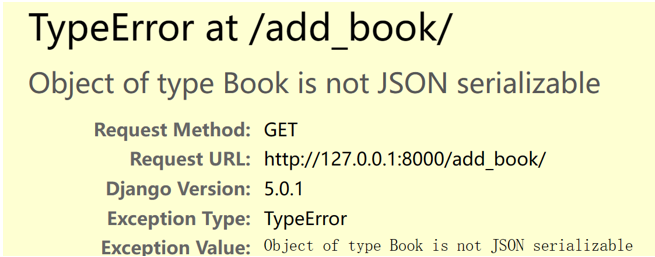


Solved Problems

Part .03

Solved Problems1-The book object cannot be serialized

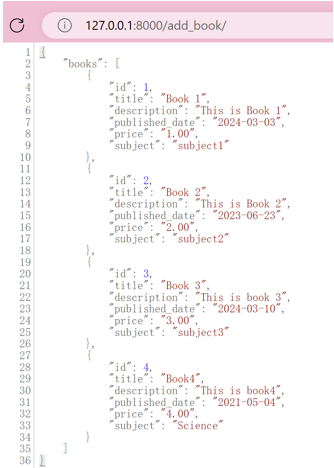
error display



Solution

Using values() method to take the fields of the Book object In the GET of the add_book view function, and convert them to a list of dictionaries.

Solved Problems1-The book object cannot be serialized



Solved Problems2-failure of adding a book by postman

error display--cannot directly add many-to-many field

```
  File "D:\Users\ljw99\anaconda3\Lib\site-packages\django\db\models\query.py", line 675, in create
    obj = self.model(**kwargs)
          ^^^^^^^^^^^^^^^^^^^^
  File "D:\Users\ljw99\anaconda3\Lib\site-packages\django\db\models\base.py", line 564, in __init__
    _setattr(self, prop, value)
  File "D:\Users\ljw99\anaconda3\Lib\site-packages\django\db\models\fields\related_descriptors.py", line 658, in __set__
    raise TypeError(
TypeError: Direct assignment to the forward side of a many-to-many set is prohibited. Use author.set() instead.
```

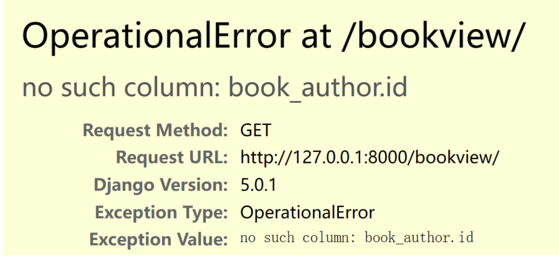
Solution

Create a new instance of AuthorSerializer, which serializes the authors field of the Book object into JSON-formatted data

```
class BookSerializer(serializers.ModelSerializer):
    author = AuthorSerializer(many=True)
    class Meta:
        model = Book
        fields = ['id','title','description','published_date','author','price','subject']
```

Solved Problems3-failure of creating new book instance because a table did not exist

error display



Solution

Delete all existing database contents and rebuild

Step1:
delete db.sqlite3
Step2:
delete all filed under migrations except __init.py__
Step3:
run command
python manage.py makemigrations
python manage.py migrate
Step4:
delete all data in tables
by command
python manage.py flush -> yes

Solved Problems4-failure of deleting a book

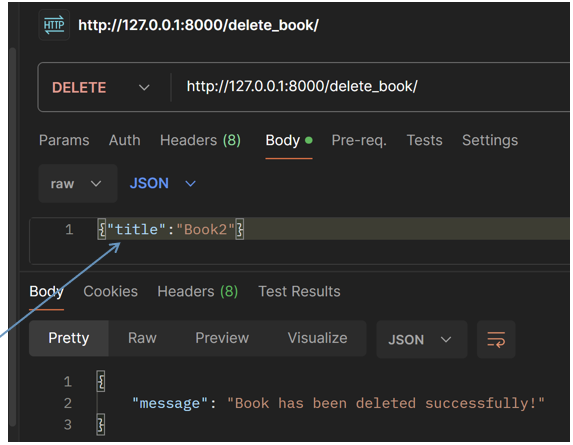
failure display

```
"GET /delete_book/ HTTP/1.1" 200 31
```

127.0.0.1:8000/delete_book/

```
{
    "error": "Method not allowed"
}
```

a book title which actually doesn't exist



Solved Problems4-failure of deleting a book

Solution

Step1：
The browser only supports GET and POST requests.
Postman is required for delete requests.
Step2：
In Django's ORM, the delete() method does not throw an error, even if no matching object is found.
When Book.objects.filter(title=data['title']).delete() is called, Django tries to find all matching books and delete them. If no matching books are found, the delete() method will still execute successfully, but nothing will be deleted.

If wanting to return an error message when no matching book is found, it is better to GET the book first and then delete it.

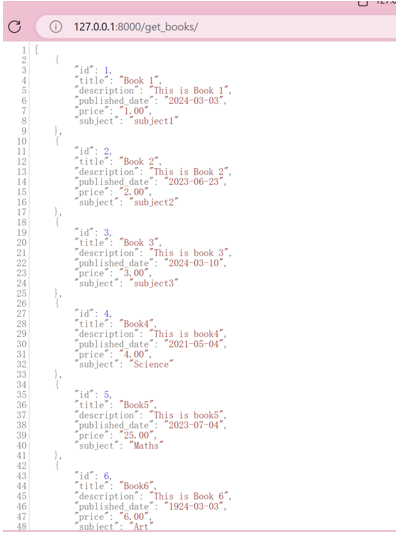
Test by Postman

Step1:
select "DELETE "from the Request method drop-down menu in Postman
Step2:
enter URL in the URL bar,
http://localhost:8000/delete_book/
Step3:
select “Body”,select”raw” and “JSON”
Step4:
enter JSON data in the text box,
{"id": 1} or {"title": "Book1"}
Step5:
click SEND to send request

Unsolved Problems

Part .04

Unsolved Problems1-failure of showing Author of a book
failure of passing content of an author



Unsolved Problems2-failure of sorting Books by title

127.0.0.1:8000/sort_books/?sort_field=title&sort_direction=descending

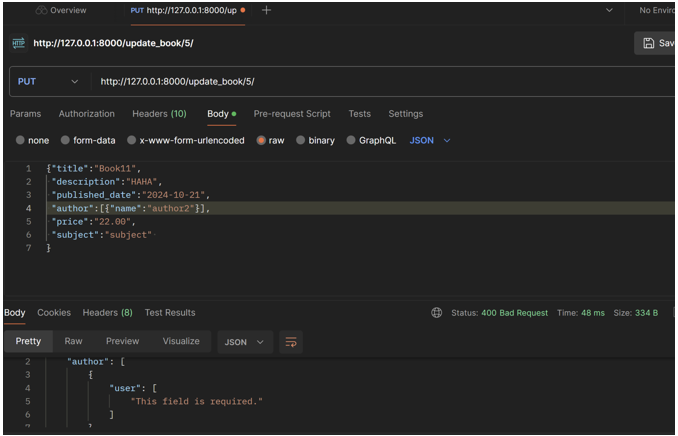

```
[
    {
        "id": 6,
        "title": "Book2",
        "description": "This is Book2",
        "published_date": "1947-12-20",
        "author": [],
        "price": "21.00",
        "subject": "Maths"
    },
    {
        "id": 5,
        "title": "Book12",
        "description": "This is Book12",
        "published_date": "1999-11-20",
        "author": [],
        "price": "41.00",
        "subject": "Science"
    },
    {
        "id": 4,
        "title": "Book1",
        "description": "This is Book1",
        "published_date": "1995-11-25",
        "author": [],
        "price": "11.00",
        "subject": "Art"
    }
]
```

Unsolved Problems3-failure of updating a book by details

ParseError at /update_book/5/

JSON parse error - Expecting value: line 1 column 1 (char 0)

Request Method:	GET
Request URL:	http://127.0.0.1:8000/update_book/5/
Django Version:	5.0.1



Unsolved Problems4,5-failure of adding/deleting multiple books

```
    raise TypeError(
TypeError: In order to allow non-dict objects to be serialized set the safe parameter to False.
```

```
ValueError: The view book.views.delete_multiple_books didn't return an HttpResponse object. It returned None instead.
```

Thank you !