Case Study: Online Bookstore API

Introduction

This document outlines the steps and functionalities of the Online Bookstore API project. The project includes user registration, login, and CRUD operations for authors, books, and categories. It also supports adding books to a shopping cart and sending purchase confirmation emails to users.

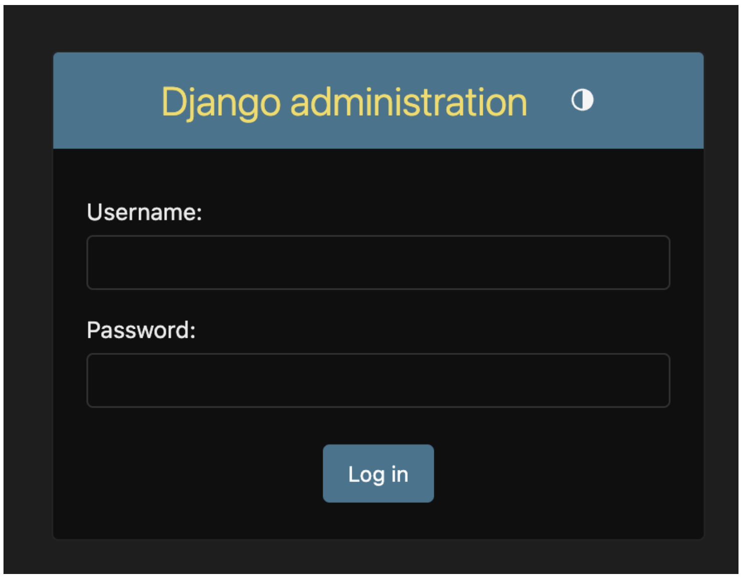
User Registration and Authentication

A superuser is created, and the credentials are used to log in. The email entered while creating the superuser will receive a purchase confirmation email.

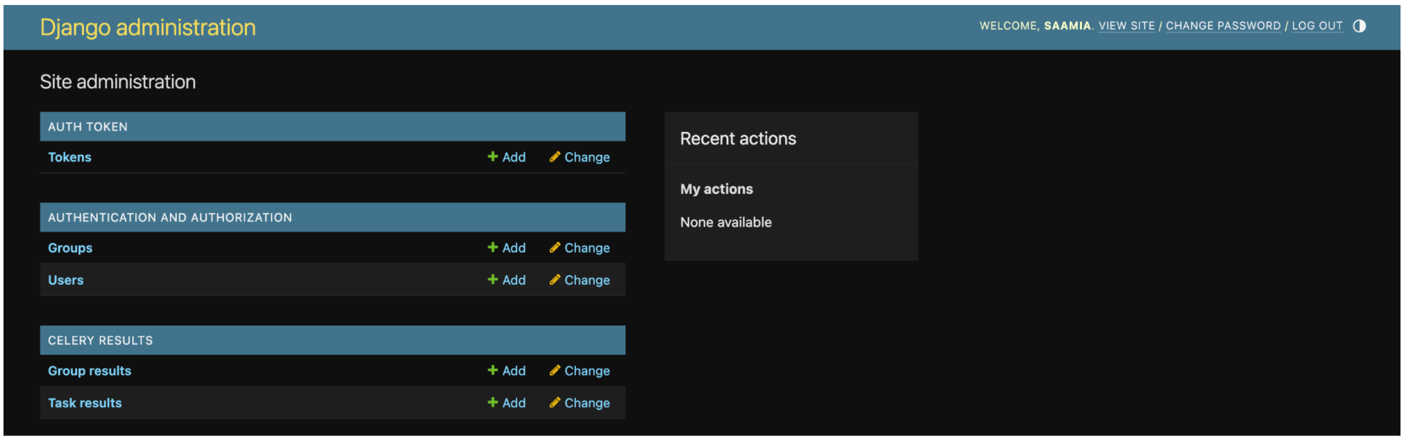
Login

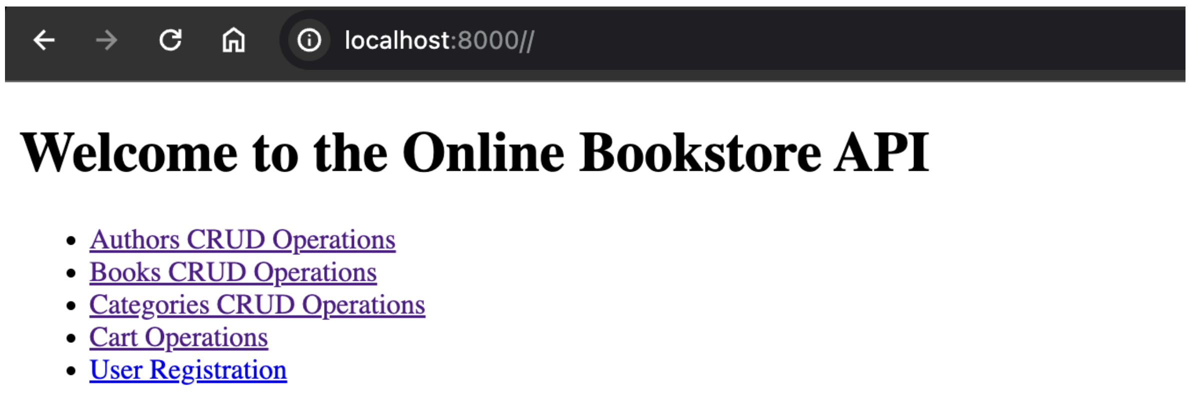
To access the Django admin interface and perform CRUD operations, log in at:

Login URL: http://localhost:8000/admin/



After logging in, you will be directed to the Django admin interface, where you can manage authors, books, and categories.





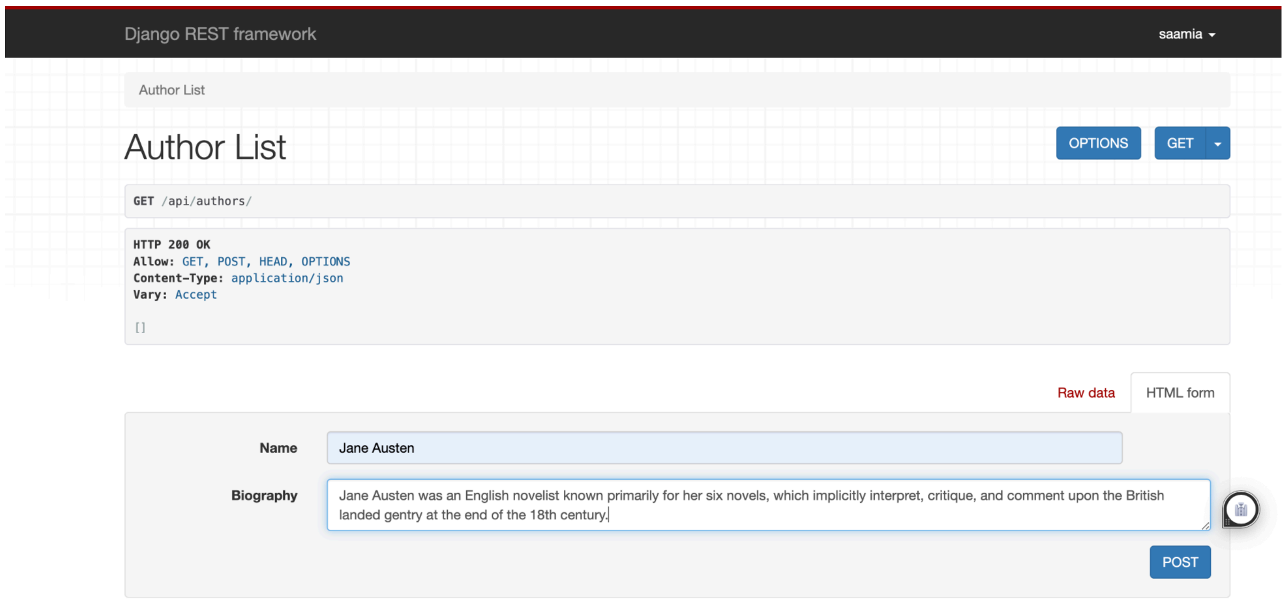
Welcome to the Online Bookstore API

- Authors CRUD Operations
- Books CRUD Operations
- Categories CRUD Operations
- Cart Operations
- User Registration

Author CRUD Operations

- Creation

To create a new author, use the following endpoint:
Endpoint: http://localhost:8000/api/authors/
Method: POST



Author List

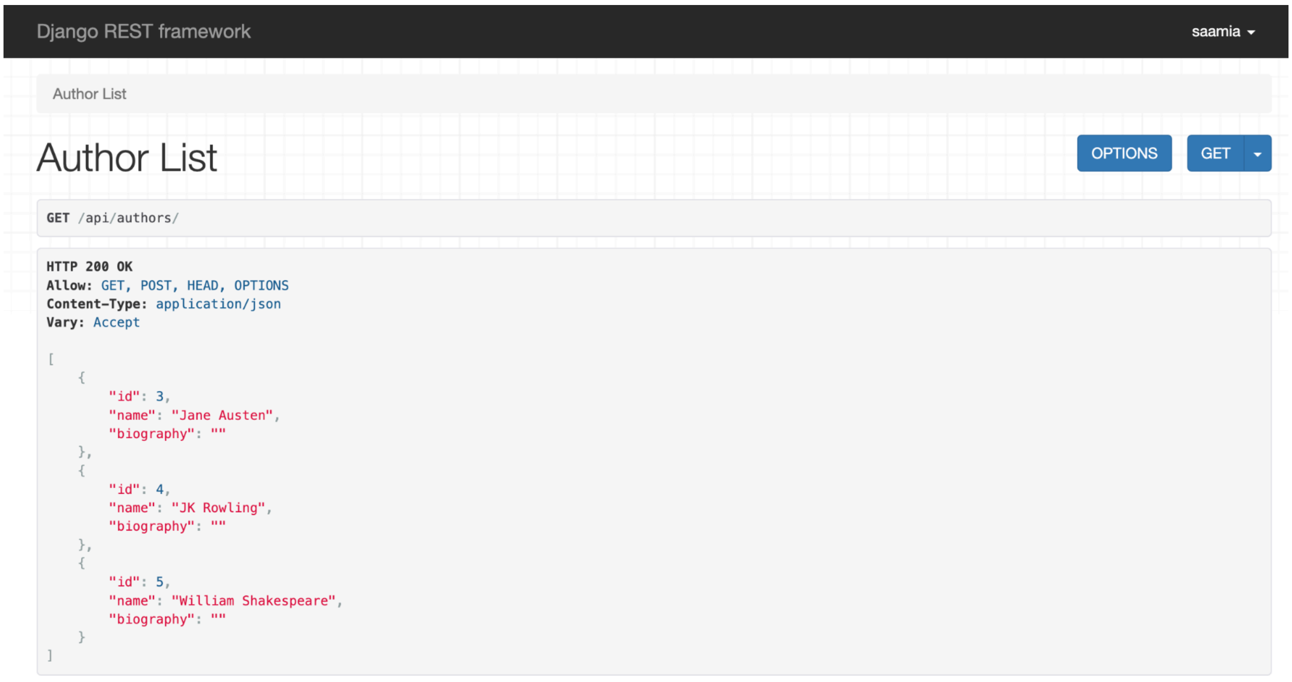
```
GET /api/authors/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[]
```

Name	Jane Austen
Biography	Jane Austen was an English novelist known primarily for her six novels, which implicitly interpret, critique, and comment upon the British landed gentry at the end of the 18th century.

- Read

To display all authors from the database, click on the GET button in the API interface.



Author List

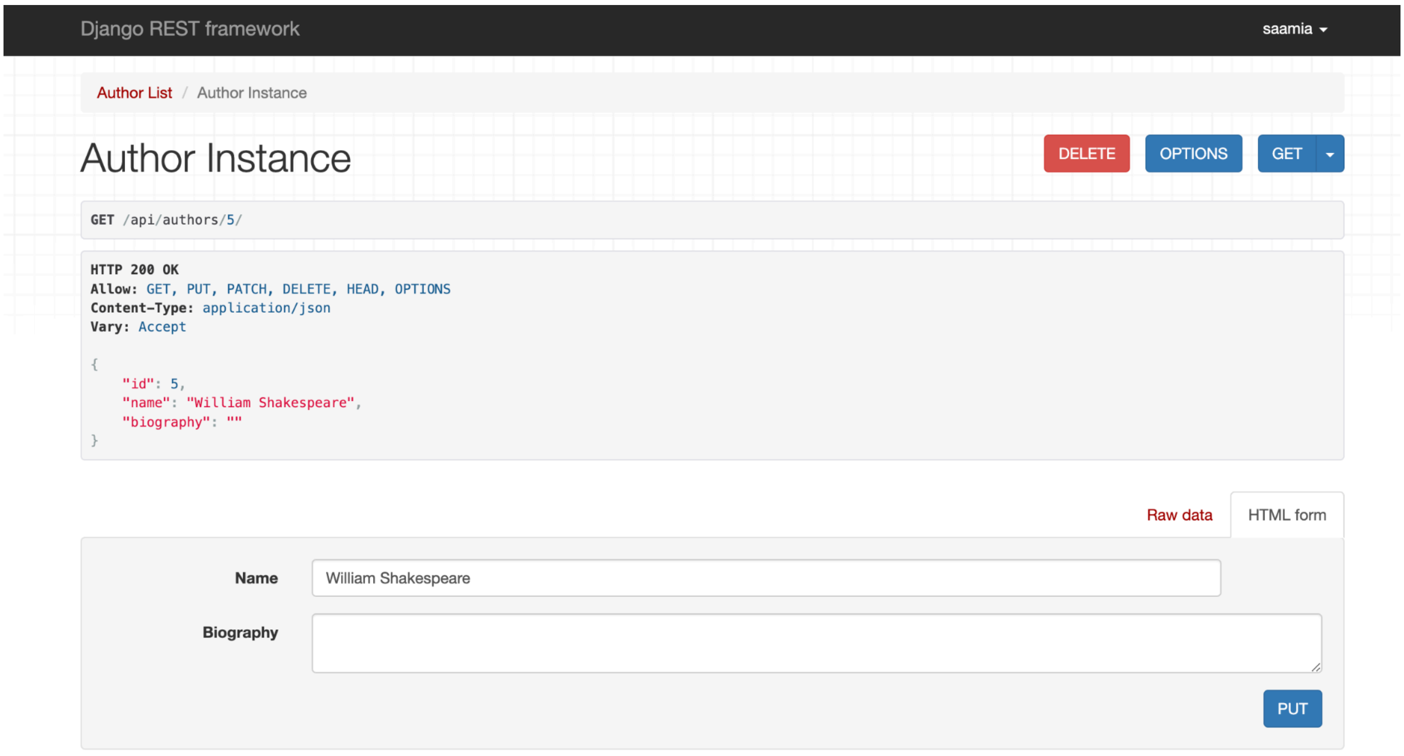
```
GET /api/authors/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 3,
        "name": "Jane Austen",
        "biography": ""
    },
    {
        "id": 4,
        "name": "JK Rowling",
        "biography": ""
    },
    {
        "id": 5,
        "name": "William Shakespeare",
        "biography": ""
    }
]
```

- **Update and Delete**

To update or delete an author, use the following endpoint:

Endpoint: http://localhost:8000/api/authors/{id_of_the_author}/
Method: PUT for update, DELETE for delete



Author Instance

```
GET /api/authors/5/
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 5,
    "name": "William Shakespeare",
    "biography": ""
}
```

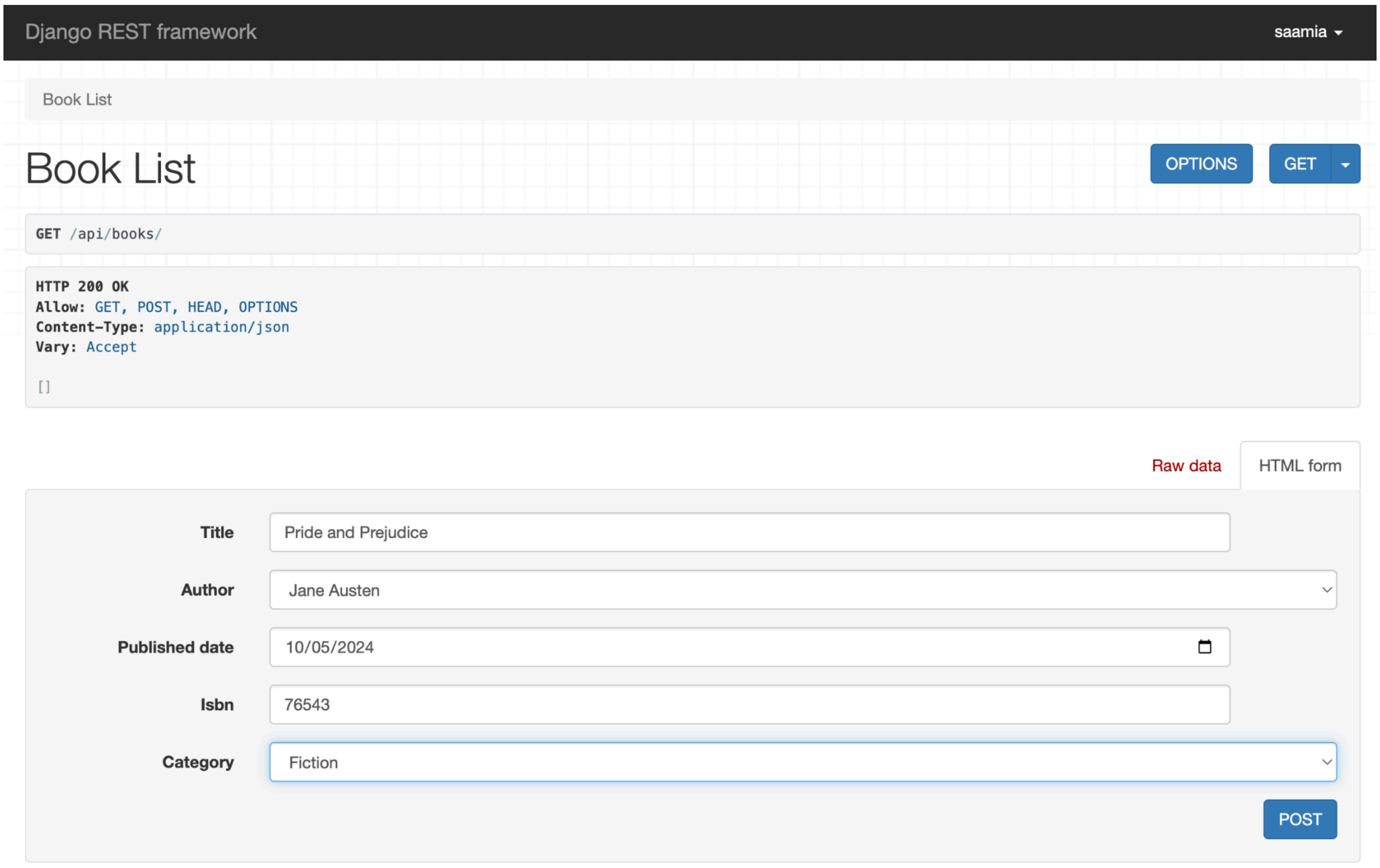
Name	William Shakespeare
Biography	

Book CRUD Operations

- **Creation**

To create a new book, use the following endpoint:

Endpoint: http://localhost:8000/api/books/
Method: POST



Book List

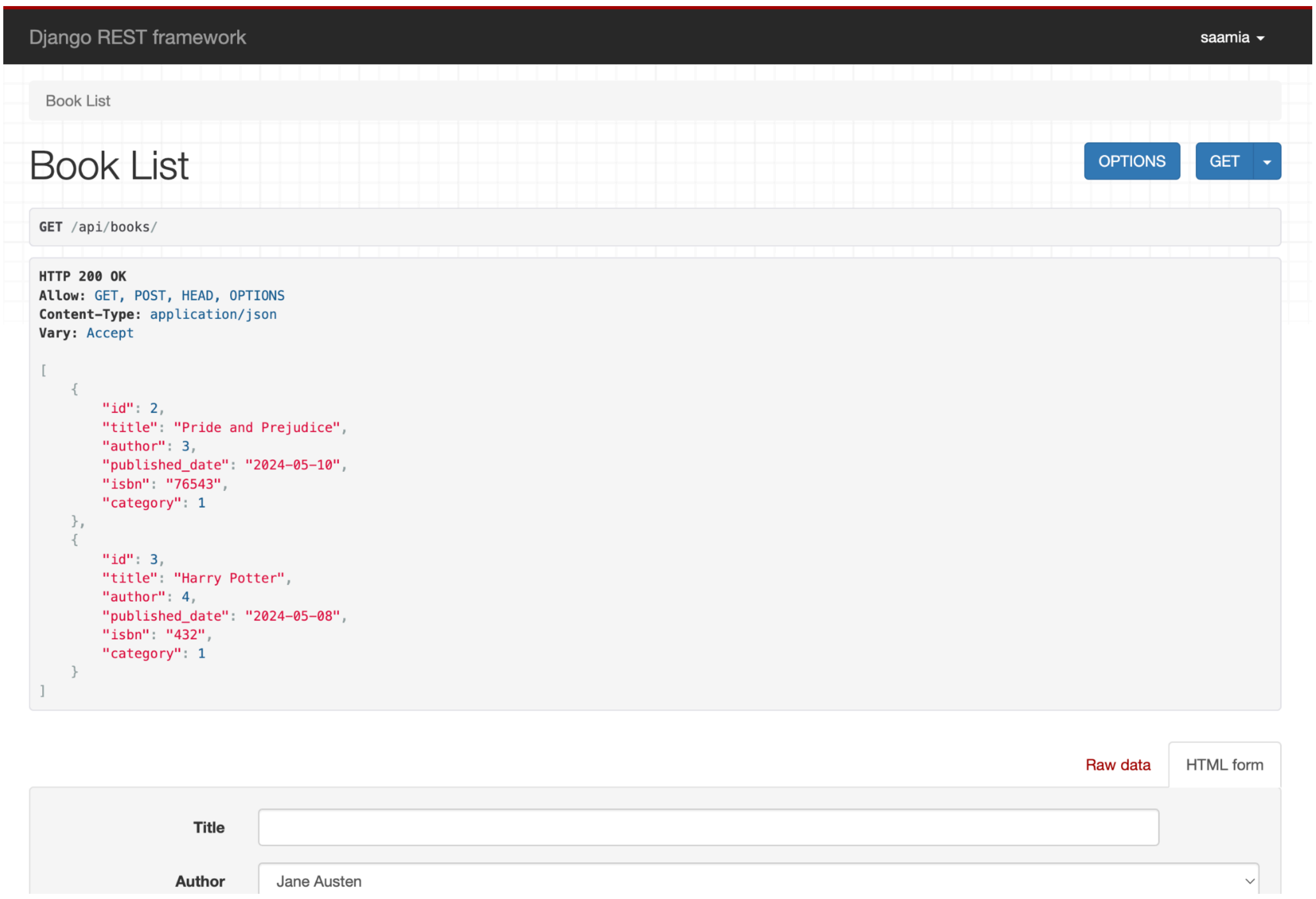
```
GET /api/books/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[]
```

Title	Pride and Prejudice
Author	Jane Austen
Published date	10/05/2024
Isbn	76543
Category	Fiction

- **Read**

To display all books from the database, click on the GET button in the API interface.



Book List

```
GET /api/books/
```

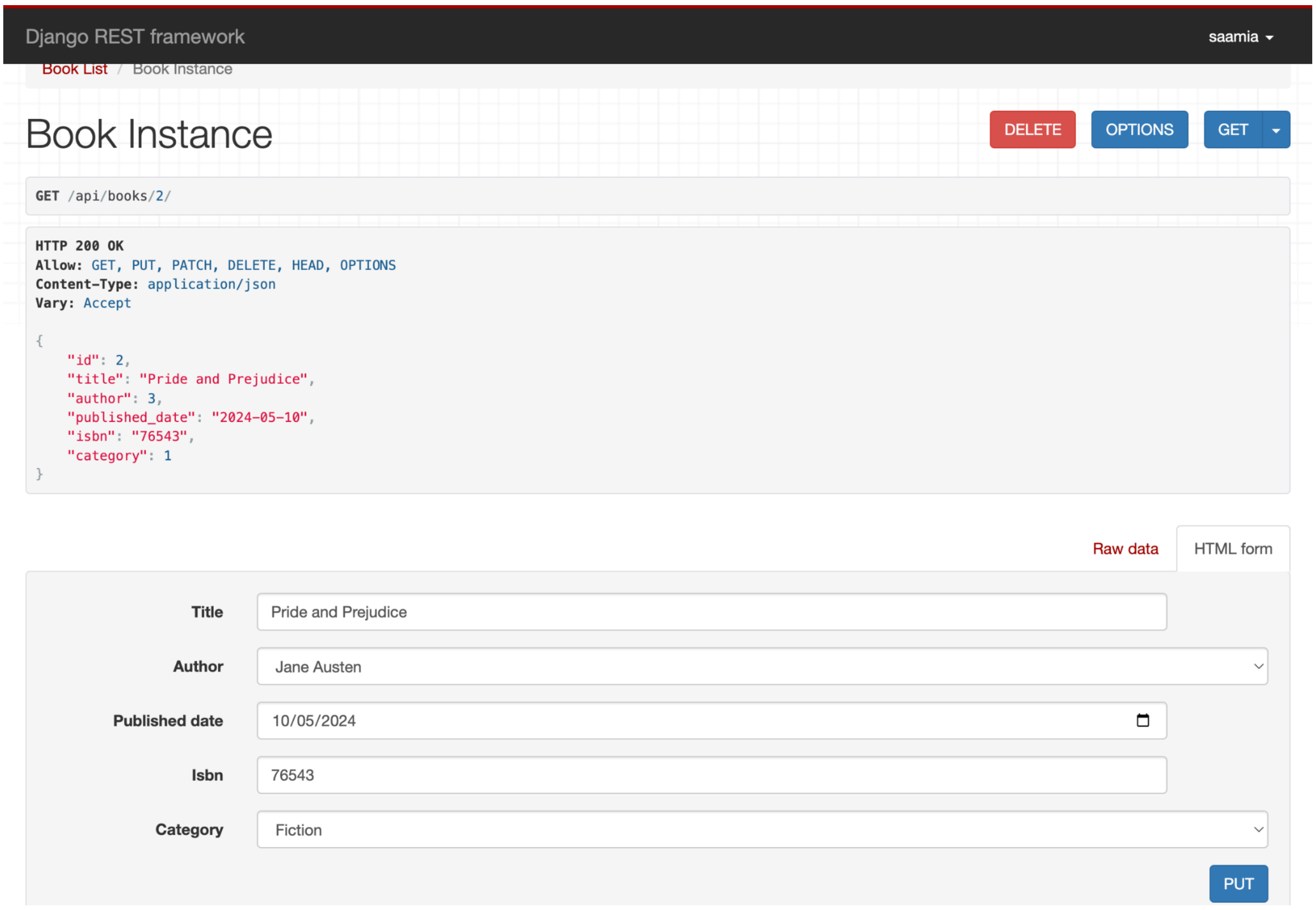
```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 2,
        "title": "Pride and Prejudice",
        "author": 3,
        "published_date": "2024-05-10",
        "isbn": "76543",
        "category": 1
    },
    {
        "id": 3,
        "title": "Harry Potter",
        "author": 4,
        "published_date": "2024-05-08",
        "isbn": "432",
        "category": 1
    }
]
```

Title	
Author	Jane Austen

- **Update and Delete**

To update or delete a book, use the following endpoint:

Endpoint: http://localhost:8000/api/books/{id_of_the_book}/
Method: PUT for update, DELETE for delete



Book Instance

```
GET /api/books/2/
```

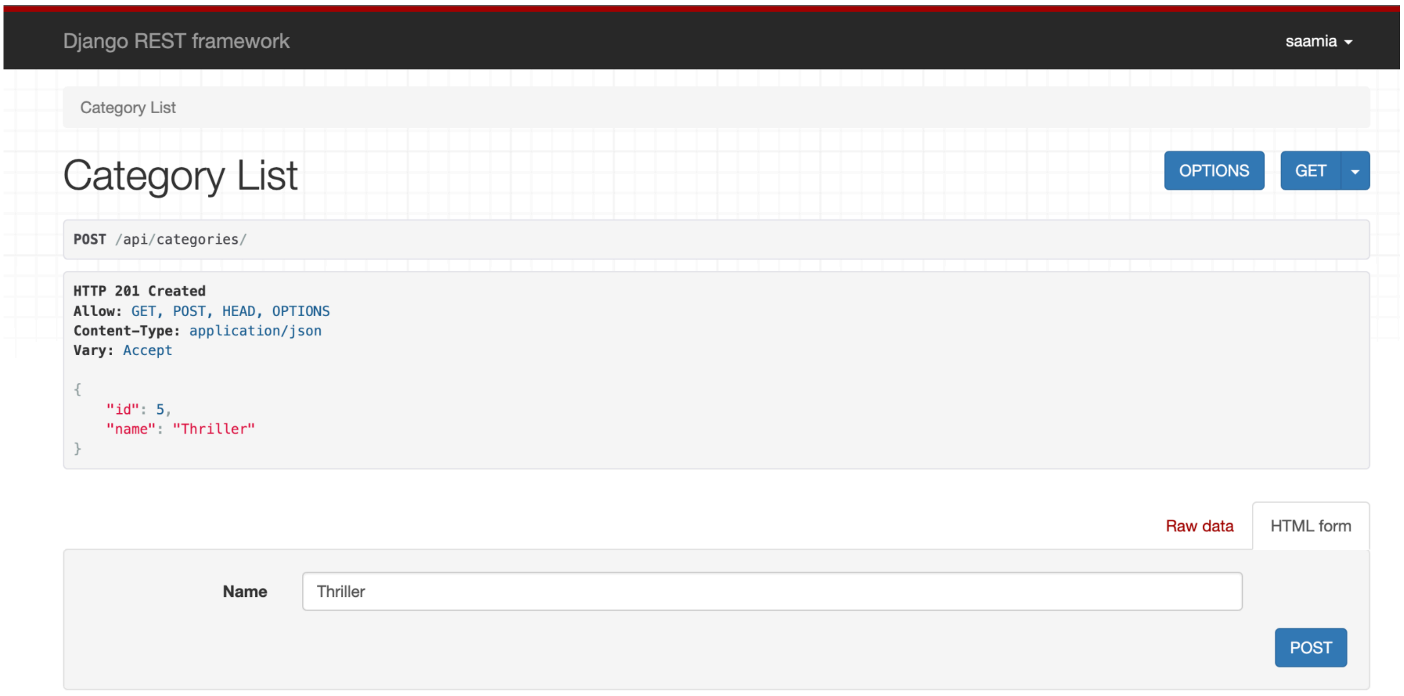
```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 2,
    "title": "Pride and Prejudice",
    "author": 3,
    "published_date": "2024-05-10",
    "isbn": "76543",
    "category": 1
}
```

Title	Pride and Prejudice
Author	Jane Austen
Published date	10/05/2024
Isbn	76543
Category	Fiction

Category CRUD Operations

- Creation

To create a new category, use the following endpoint:
Endpoint: http://localhost:8000/api/categories/
Method: POST



Category List

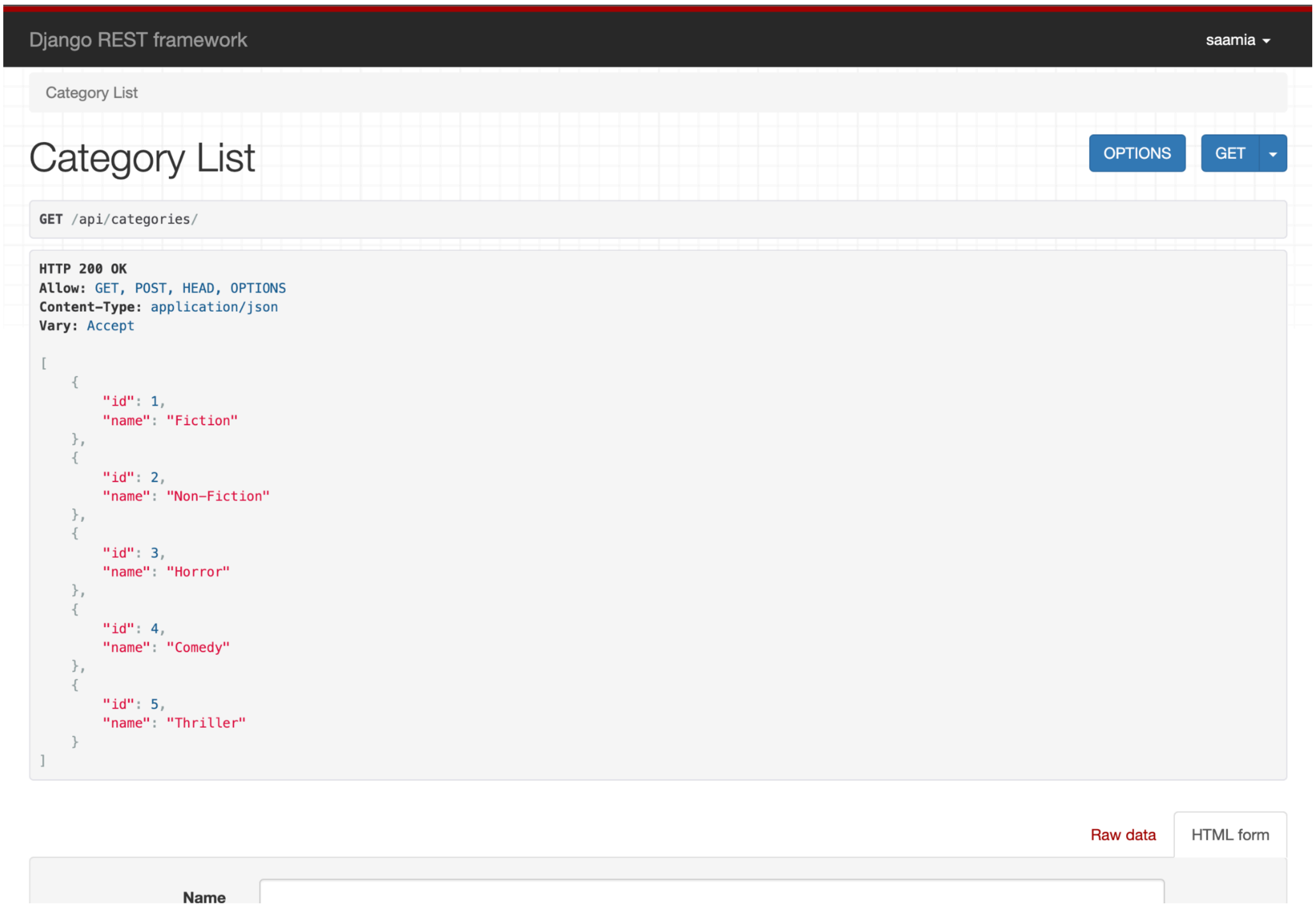
```
POST /api/categories/
```

```
HTTP 201 Created
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 5,
    "name": "Thriller"
}
```

Name	Thriller

- **Read**

To display all categories from the database, click on the GET button in the API interface.

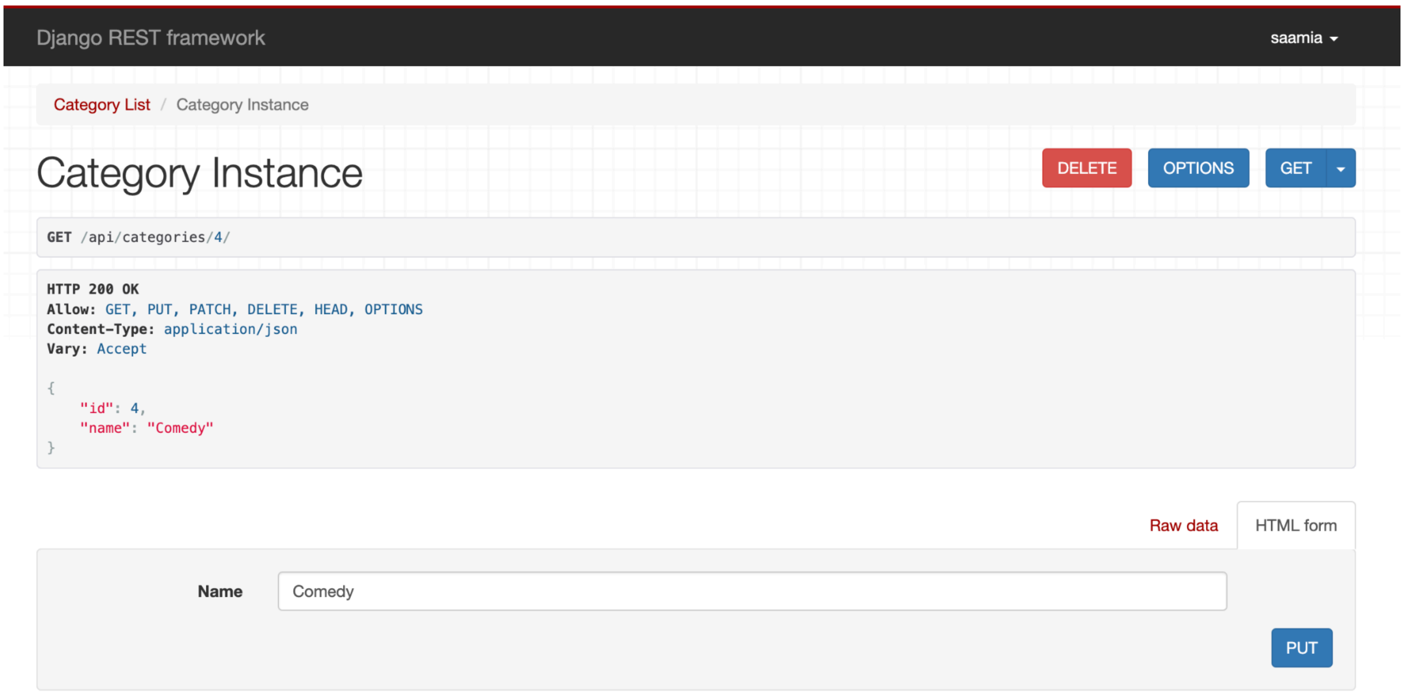


```
GET /api/categories/

HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "name": "Fiction"
    },
    {
        "id": 2,
        "name": "Non-Fiction"
    },
    {
        "id": 3,
        "name": "Horror"
    },
    {
        "id": 4,
        "name": "Comedy"
    },
    {
        "id": 5,
        "name": "Thriller"
    }
]
```

- **Update and Delete**

To update or delete a category, use the following endpoint:

Endpoint: http://localhost:8000/api/categories/{id_of_the_category}/
Method: PUT for update, DELETE for delete



Category Instance

```
GET /api/categories/4/
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 4,
    "name": "Comedy"
}
```

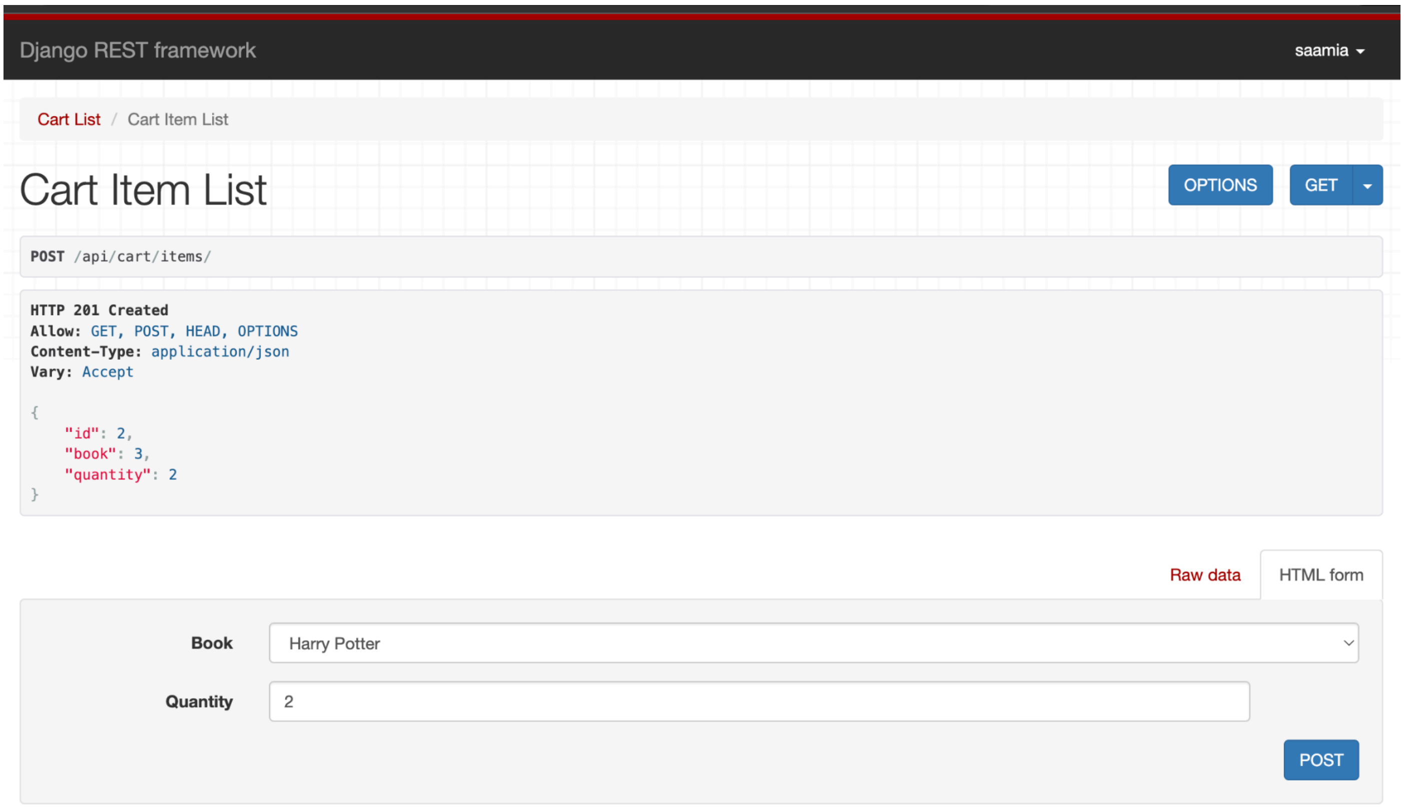
Name	Comedy

Shopping Cart Operations

Add to Cart

To add a book to the shopping cart, use the following endpoint:

- Endpoint: http://localhost:8000/api/cart/items/
- Method: POST



Cart Item List

```
POST /api/cart/items/
```

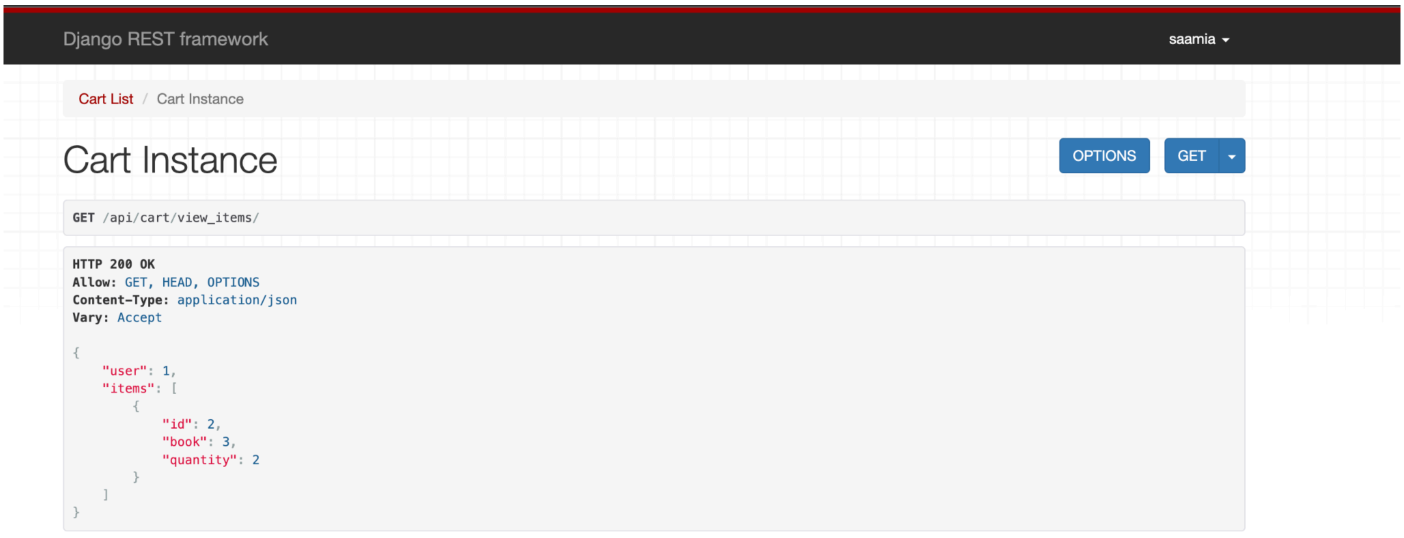
```
HTTP 201 Created
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 2,
    "book": 3,
    "quantity": 2
}
```

Book	Harry Potter
Quantity	2

View Cart Items

To view items in the shopping cart, use the following endpoint:

Endpoint: http://localhost:8000/api/cart/view_items/
Method: GET

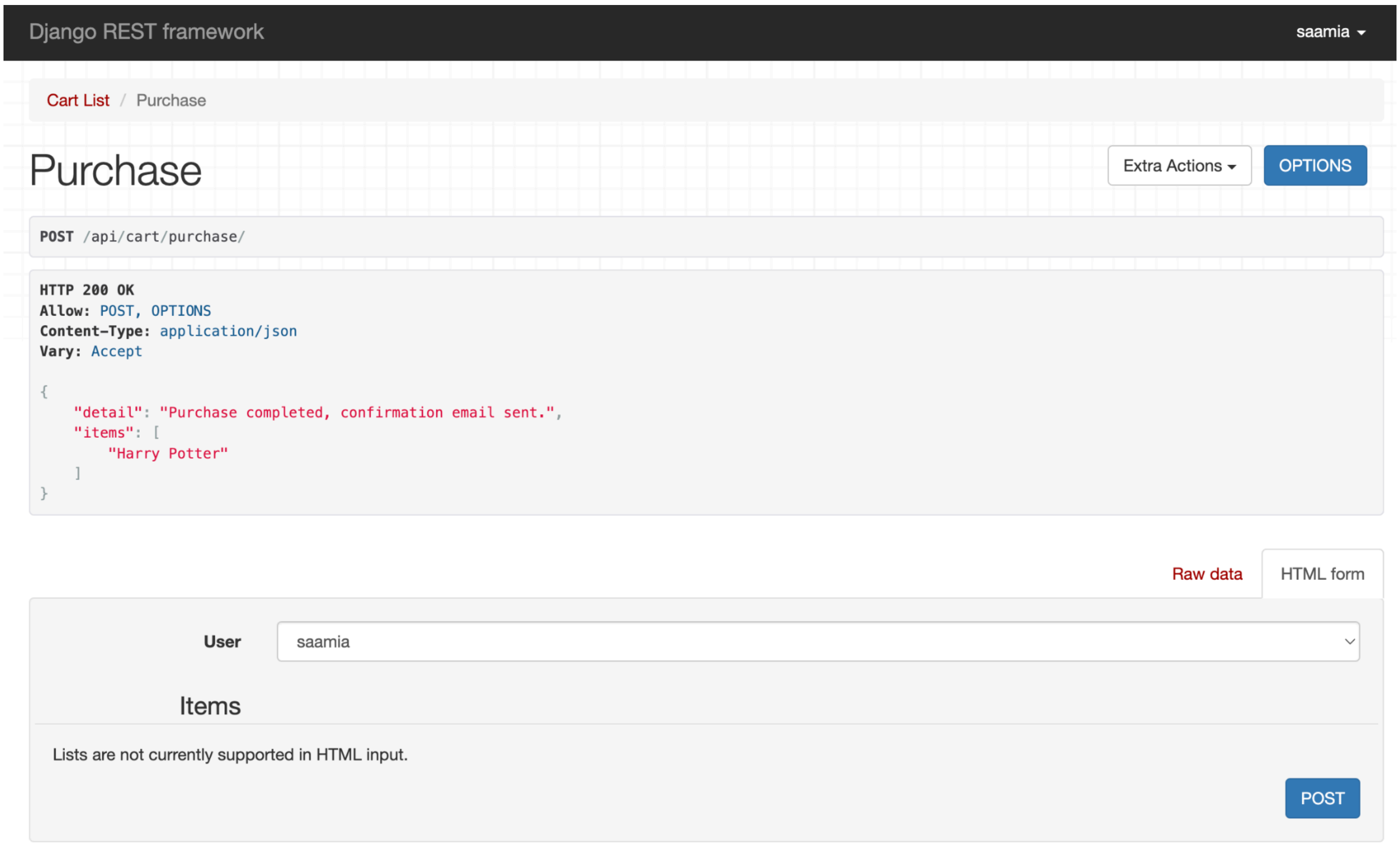


```
GET /api/cart/view_items/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "user": 1,
    "items": [
        {
            "id": 2,
            "book": 3,
            "quantity": 2
        }
    ]
}
```

Purchase
To purchase the books in the cart, use the following endpoint:

Endpoint: http://localhost:8000/api/cart/purchase/
Method: POST



```
POST /api/cart/purchase/
```

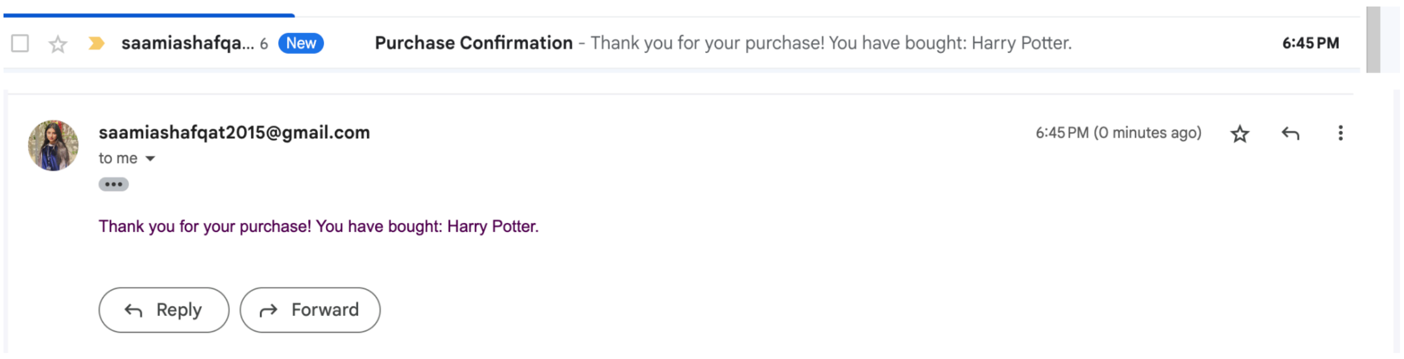
```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Purchase completed, confirmation email sent.",
    "items": [
        "Harry Potter"
    ]
}
```

User	saamia

Items

Lists are not currently supported in HTML input.

Upon successful purchase, an email is sent to the user's email address with the details of the purchase.


saamiashafqa... 6 New **Purchase Confirmation** - Thank you for your purchase! You have bought: Harry Potter. 6:45 PM

saamiashafqat2015@gmail.com
to me
6:45 PM (0 minutes ago)

Thank you for your purchase! You have bought: Harry Potter.

Running the Application Using Docker

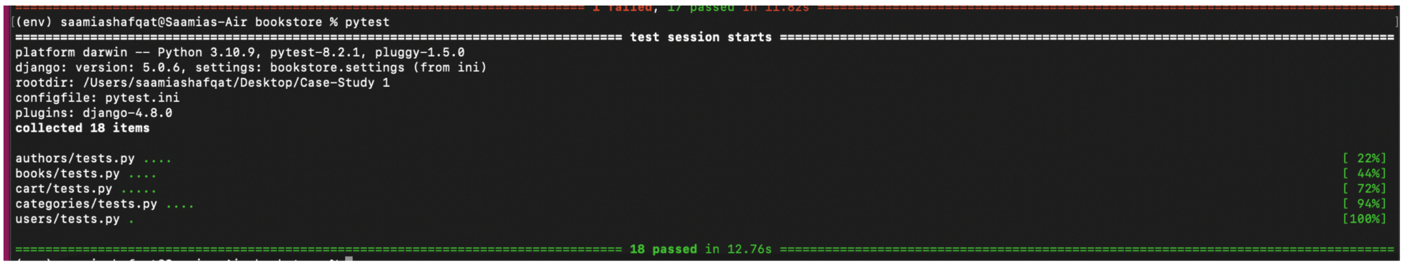
To run the application using Docker, follow these steps:

cd ~/desktop/Case-Study 1/bookstore

docker-compose up -d

docker-compose run web python manage.py migrate

Unit Testing


```
(env) saamiashafqat@Saamias-Air bookstore % pytest
================================================================================ test session starts ================================================================================
platform darwin -- Python 3.10.9, pytest-8.2.1, pluggy-1.5.0
django: version: 5.0.6, settings: bookstore.settings (from ini)
rootdir: /Users/saamiashafqat/Desktop/Case-Study 1
configfile: pytest.ini
plugins: django-4.8.0
collected 18 items

authors/tests.py ....                                                                                                                                                          [ 22%]
books/tests.py ....                                                                                                                                                            [ 44%]
cart/tests.py .....                                                                                                                                                            [ 72%]
categories/tests.py ....                                                                                                                                                       [ 94%]
users/tests.py .                                                                                                                                                               [100%]

================================================================================ 18 passed in 12.76s ================================================================================
```