**DESIGN AND IMPLEMENT A RESTFUL API**

1. You can download the provided code (Extract Zip folder) OR clone my code from Github Repo. Link: <https://github.com/Mehfooz201/Backend-Task---CRUD-RestAPI>
2. Instructions on how to set up and run the application on a local environment.

**Clone the Repository:**

* Begin by cloning the project's GitHub repository to your local machine. You can use the following command, replacing <repository-url> with your actual GitHub repository URL:

git clone <repository-url>

**Create a Virtual Environment (Optional):**

* It's a good practice to create a virtual environment for the project. You can create one using venv or virtualenv. For example, using venv:

python -m venv venv

**Activate the Virtual Environment:**

* Activate the virtual environment to isolate project dependencies. The activation command depends on your operating system. For example, on Windows:

venv\Scripts\activate

**Install Dependencies:**

* Navigate to the project directory and install the required Python dependencies using pip. The requirements.txt file contains a list of all the necessary packages.

pip install -r requirements.txt

**Database Setup:**

* If your application uses a database, you may need to create a database, apply migrations, and populate initial data. Use Django management commands for this:

python manage.py migrate

python manage.py loaddata initial\_data.json # If you have initial data to load

**Run the Application:**

* Start your Django application using the following command:

python manage.py runserver

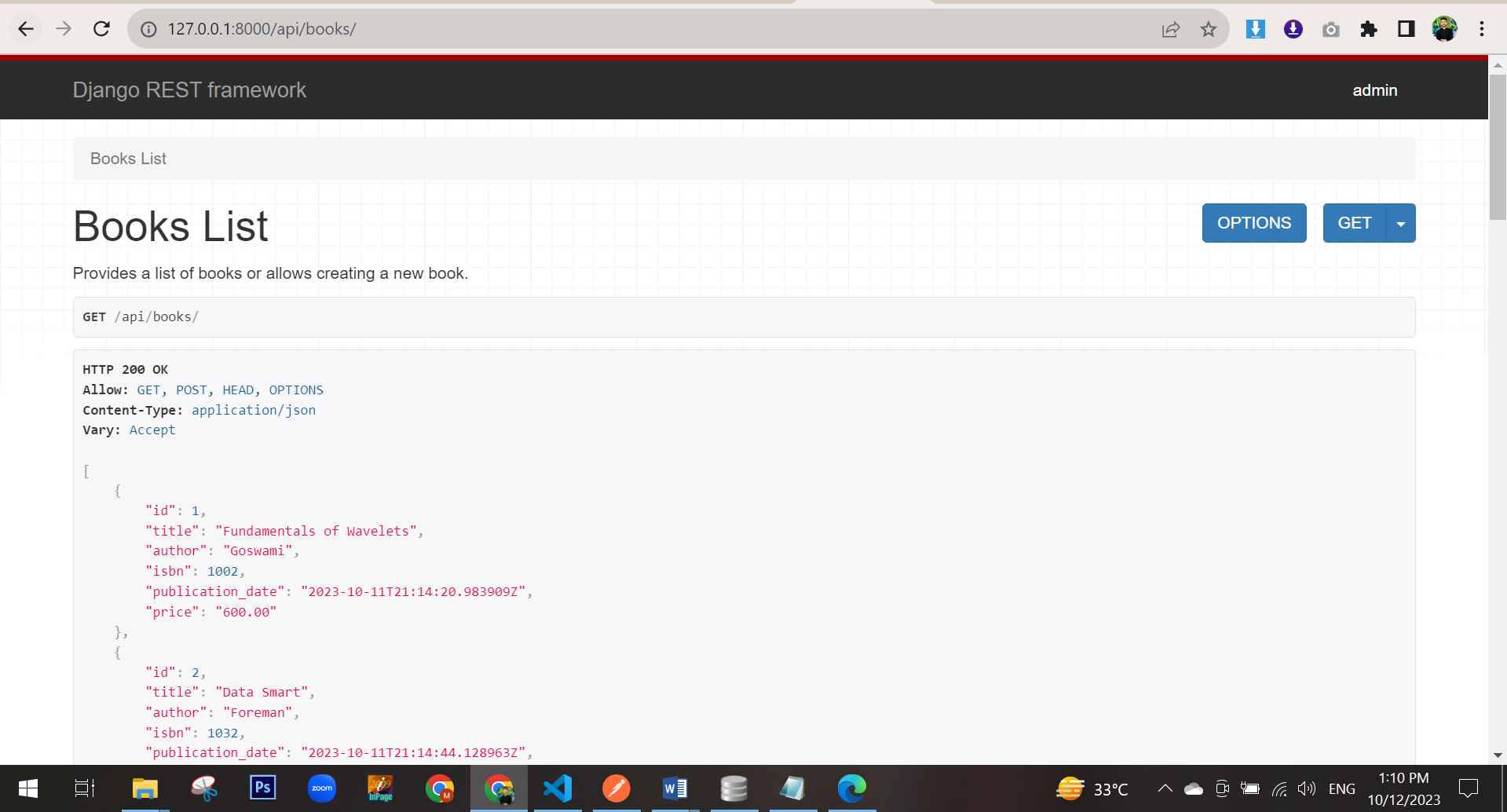
**Access the Application:**

* Open a web browser and navigate to http://localhost:8000/ or the URL provided by the runserver command. Your application should be up and running.

**Routes:** [**http://localhost:8000/**](http://localhost:8000/)

**API Books Route:** [**http://127.0.0.1:8000/api/books/**](http://127.0.0.1:8000/api/books/)

**API Specific Book route :** [**http://127.0.0.1:8000/api/books/2**](http://127.0.0.1:8000/api/books/2)



-------------------------------------------------

1. A short document explaining your design choices, including the choice of programming language, framework, and database.

**PURPOSE:**

* The primary goal of this project is to create simple API endpoints for managing book data, including a list of books and individual book details. The API is designed to facilitate easy access and manipulation of book information.

**PROGRAMMING LANGUAGE: PYTHON**

* The choice of the programming language is Python. Python is renowned for its simplicity, readability, and a vast ecosystem of libraries. It's an ideal language for web development, and its clean syntax makes it easy to understand.

**FRAMEWORK: DJANGO AND DJANGO REST FRAMEWORK**

* The web framework chosen is Django, along with Django Rest Framework (DRF). These frameworks were selected for several compelling reasons:
* Built-in Features: Django offers a plethora of built-in features, such as user authentication and database ORM, significantly speeding up development.
* Security: Django provides robust security measures to protect against common web vulnerabilities.
* Scalability: The framework is designed to handle high-traffic applications and offers scalability options.
* Community and Documentation: Django has a thriving community and extensive documentation, ensuring that solutions and resources are readily available.

**DATABASE: SQLITE3**

* The database selected is SQLite3. The choice of SQLite3 was made due to its lightweight nature and ease of understanding. It's well-suited for a project of this scale and simplifies data management.

**API DESIGN:**

* The API design follows RESTful principles. It consists of well-structured endpoints for listing books and retrieving details of individual books. The design aims for simplicity and user-friendliness. There are lots of features for making API’s, but I have utilized simple way using Classed Based View for making an API’s.

**TESTING AND VERIFICATION:**

* The API endpoints have been thoroughly tested using Postman, a widely-used API testing tool. This ensures that the API functions as intended and provides the expected responses.

