<u>REPORT OF THE BOOKSHELF PROJECT</u>

Project Report: Building a REST API using Django and PostgreSQL

Introduction:

The objective of this project was to create a REST API using Django and PostgreSQL that provides CRUD functionality for a Bookshelf. The project also involved testing the API using Postman and implementing a simple front-end using Bootstrap and JavaScript.

Functionality:

The API provides the expected CRUD functionality for the Bookshelf. Users can perform GET, POST, PATCH, and DELETE operations on Books data. As well as provide Login, Signup and Logout functionality.

Code quality:

The code is well-organized and follows best practices in Django. Code structure and naming conventions are consistent throughout the project. The code is also maintainable and modular, with clear separation of concerns between the models, views, and serializers.

Database integration:

The project is correctly integrated with the PostgreSQL database, using the psycopg2 module to establish the database connection. The models.py file defines the schema for User and Book table, including the required fields and relationships

Front-end design:

The front-end is minimalistic but functional, using Bootstrap and CSS for styling. The design is intuitive, with a Card displaying the list of books and form inputs for adding and editing book records.

AJAX and JavaScript implementation:

One of the challenges faced during the project was implementing AJAX and JavaScript to handle client-side functionality for the front-end. This was overcome by using fetch and AJAX to make asynchronous calls to the Django views and serializers. The front-end JavaScript code is well-structured, with clear separation of concerns between the DOM manipulation, event listeners, and API requests.

Postman testing:

The API passes all tests in Postman, returning expected responses for all endpoints and methods. The API also correctly handles error cases and returns appropriate HTTP status codes and error messages.

Conclusion:

In conclusion, the project successfully created a REST API using Django and PostgreSQL, integrated it with a simple front-end, and tested it using Postman. The project also demonstrated the importance of clear code organization, maintainability, and adherence to best practices. However, the project faced challenges in using AJAX and JavaScript to handle client-side functionality for the front-end. These challenges were overcome by using fetch and AJAX to make asynchronous calls to the Django views and serializers. Overall, the project was a valuable learning experience in building REST APIs with Django and PostgreSQL, and in implementing AJAX and JavaScript to handle client-side functionality for the front-end.