Antonio C. Soldevillo 04/16/2025

BSIT-3R5

To-Do List Application Documentation

Project Title:

To-Do List App using FastAPI and React

This project is a full-stack To-Do List application built using **FastAPI** for the backend and **React (Vite)** for the frontend. It allows users to:

- Add, edit, delete tasks
- Mark tasks as completed
- Filter tasks by status (all, completed, pending)
- Toggle dark mode
- Persist data using a backend API and database

The backend exposes a RESTful API and is connected to a database using SQLAlchemy with SQLite. SQLite was used in this project because the free PostgreSQL instance on Render had already been used for a previous activity, and SQLite provided a simple and lightweight alternative suitable for development and small-scale deployment. The frontend interacts with this API using Axios and is styled for responsiveness and theme toggling.

FastAPI vs Django REST Framework (DRF)

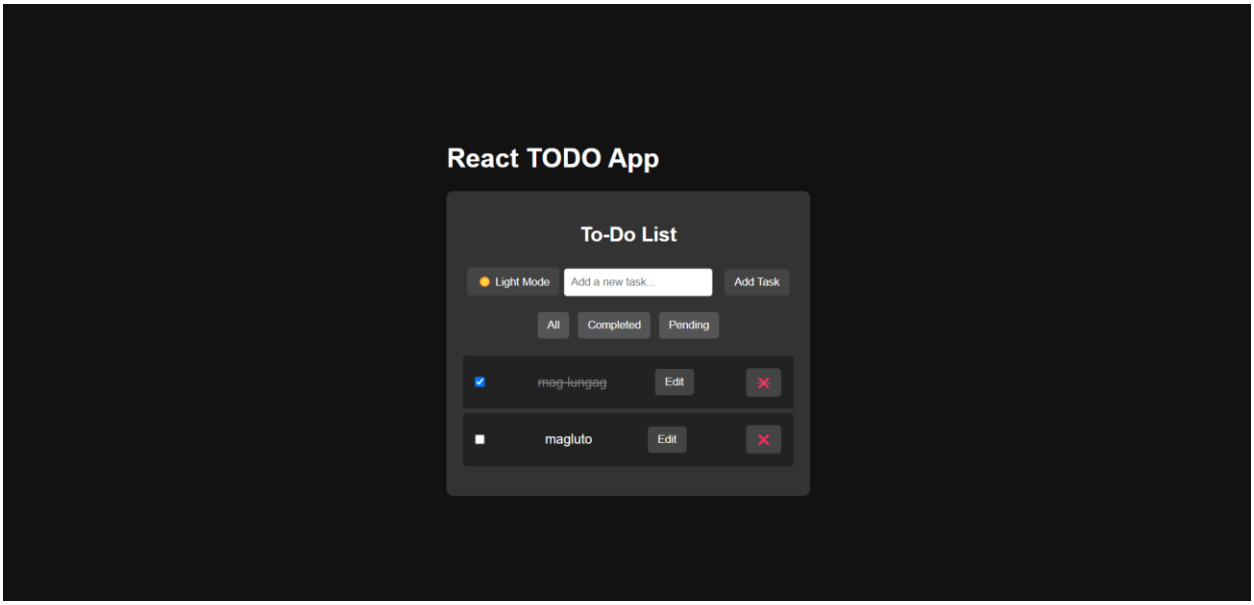
Feature	FastAPI	Django REST Framework (DRF)
Performance	Very fast (async supported)	Slower, sync by default
Simplicity	Lightweight and easy to set up	More boilerplate, heavier
Documentation	Auto Swagger and ReDoc out-of-box	Good docs but requires setup
Learning Curve	Easier for beginners in APIs	Requires understanding Django

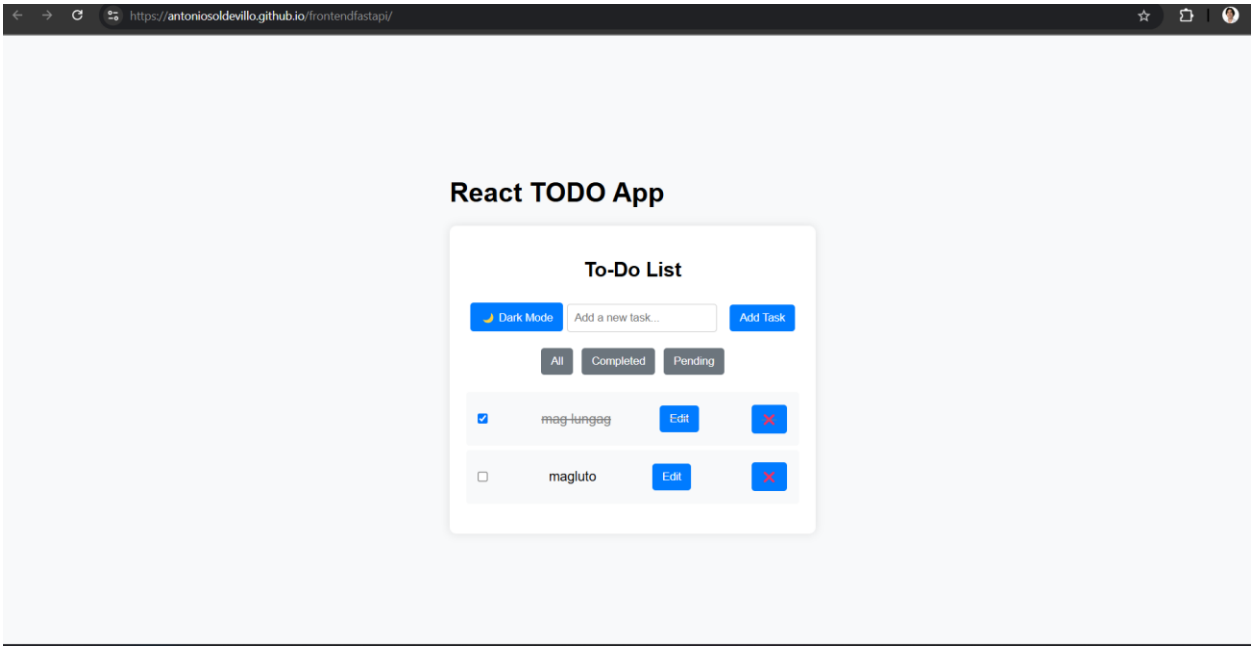
Use Case	Best for microservices, APIs	Best for full web backend stacks

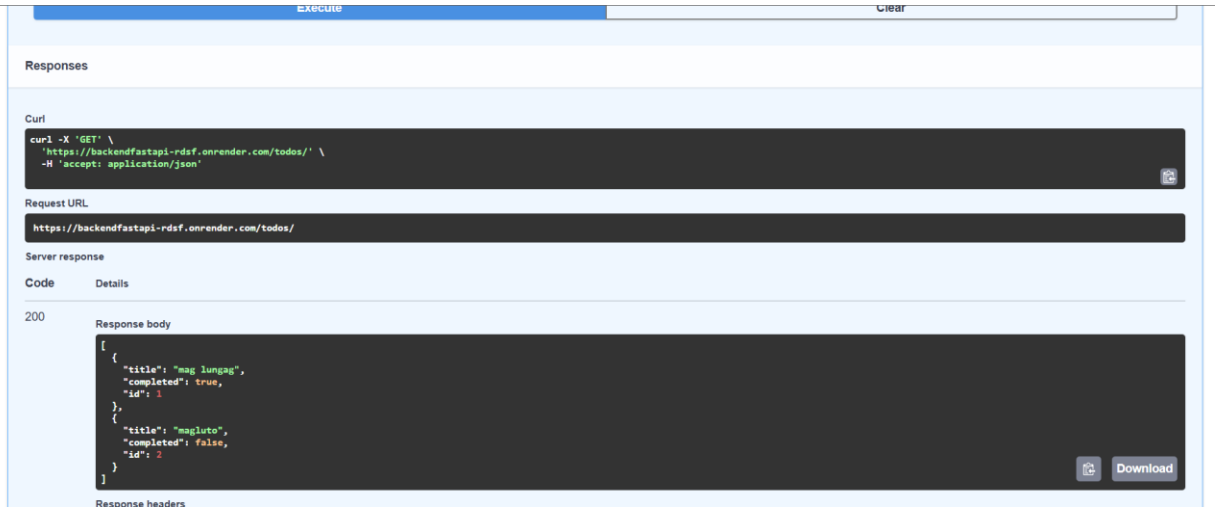
Technologies Used:

- Frontend: React + Vite
- Backend: FastAPI + SQLAlchemy
- Database: SQLite (Free Postgres instance on render had already been used for a previous activity)
- Deployment:
    - Backend on Render
    - Frontend on GitHub Pages

Screenshots:







Live Links:

Backend: https://backendfastapi-rdsf.onrender.com/docs

Frontend: https://antoniosoldevillo.github.io/frontendfastapi/

Challenges Faced:

- Django REST Framework (DRF): The main challenge encountered during the DRF implementation was related to deployment on Render. I initially had issues pushing the project to GitHub, which delayed the deployment process.

- FastAPI: With my prior experience in DRF, I found working with FastAPI to be much easier. There were no significant challenges in the FastAPI implementation, as the framework's simplicity and my familiarity with similar tools allowed me to quickly set up and deploy the backend.