Nicole C. Dacillo 04/21/2025

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.

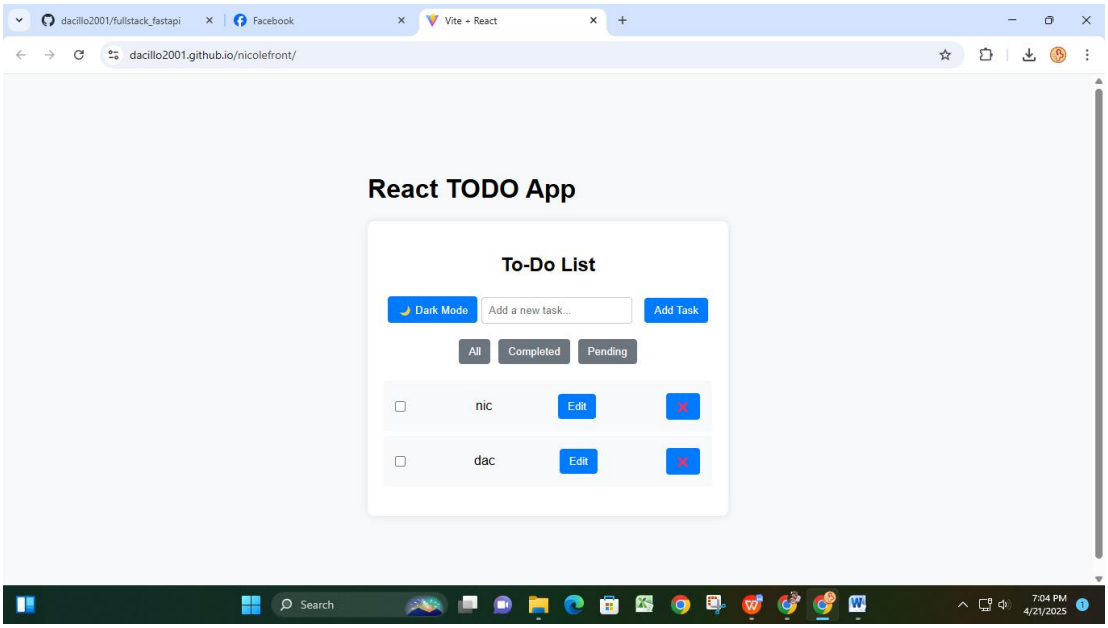
FastAPI vs Django REST Framework (DRF)

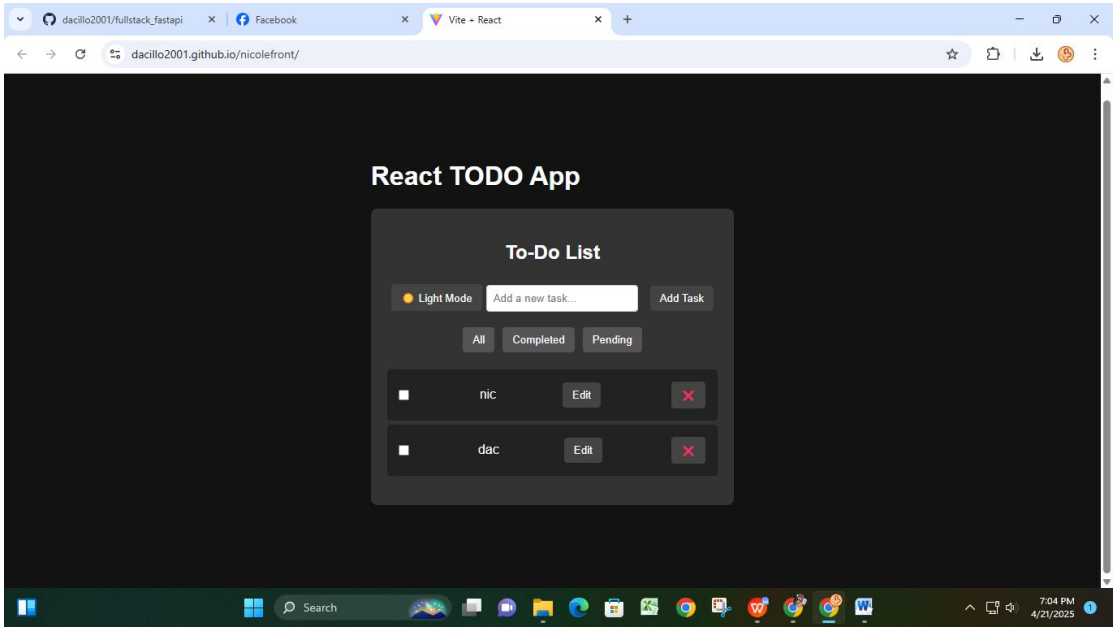
Feature	FastAPI	Django REST Framework (DRF)
Speed	Very Fast (async support)	Slower due to Django overhead
Ideal Use Case	Microservices, lightweight APIs	Full-featured web apps
Documentation	Auto Swagger and ReDoc out-of-box	Good docs but requires setup
Learning Curve	Easier for beginners in APIs	Requires understanding Django
Use Case	Best for microservices, APIs	Best for full web backend stacks

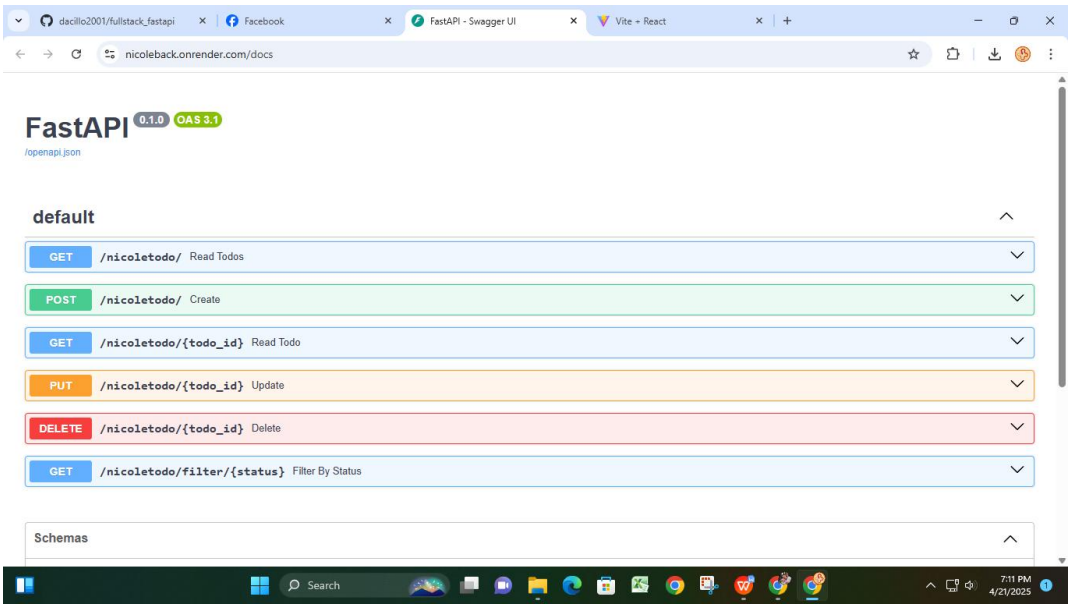
Technologies Used:

- 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://nicoleback.onrender.com/docs

Frontend: https://dacillo2001.github.io/nicolefront/

Challenges Faced:

- Django REST Framework (DRF): One of the key hurdles I faced while working with DRF was during the deployment phase, specifically when using Render. Before I could even begin deployment, I encountered difficulties uploading the project to GitHub, which caused some delays in the overall process.

- FastAPI:Transitioning to FastAPI was a much smoother experience. Thanks to my background with DRF and similar frameworks, I was able to adapt to FastAPI's structure quickly. Its straightforward design made backend development and deployment efficient, with no major obstacles encountered along the way.