Jazel Jean Achas 5/22/2025

BSIT-3R5

TO-DO LIST APP DOCUMENTATION

Project Title: Todo 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)

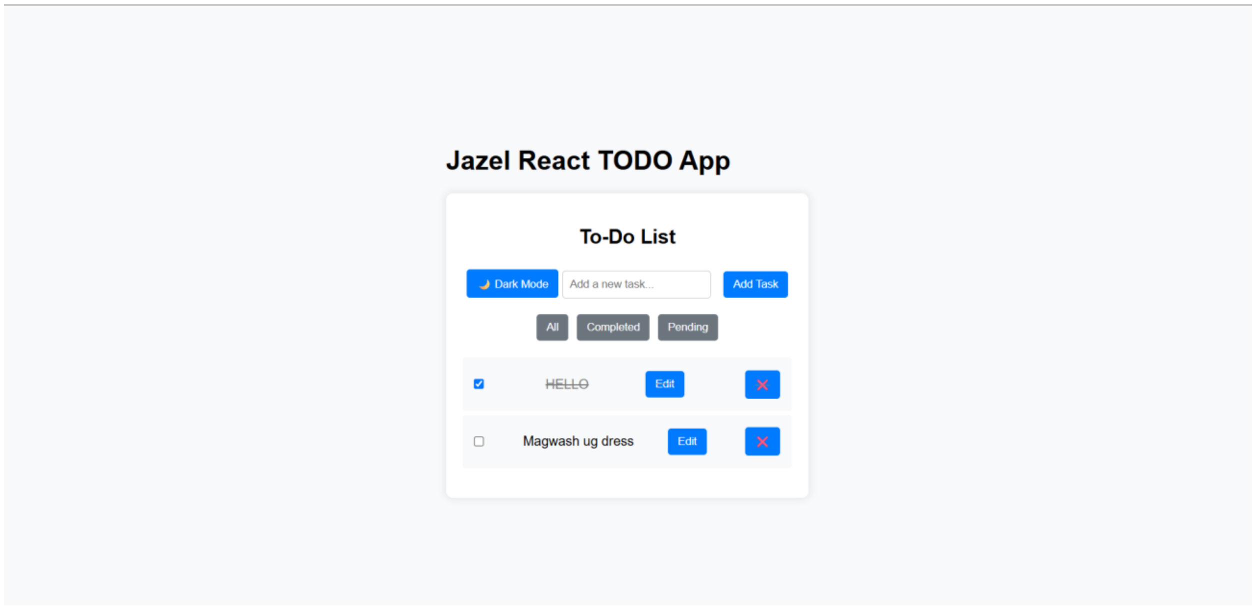
Feature	**FastAPI**	**Django REST Framework (DRF)**
Performance	Very fast (async supported)	Slower, sync by default
Simplicity	Lightweight and easy to set up	More boilerplate, heavier
Learning Curve	Requires understanding async concepts	Easier for beginners in APIs, if familiar with Django
Flexibility	Highly flexible, modular	Tied closely to Django's ecosystem
Best For	High-performance APIs, microservices	Full-featured web apps, rapid prototyping

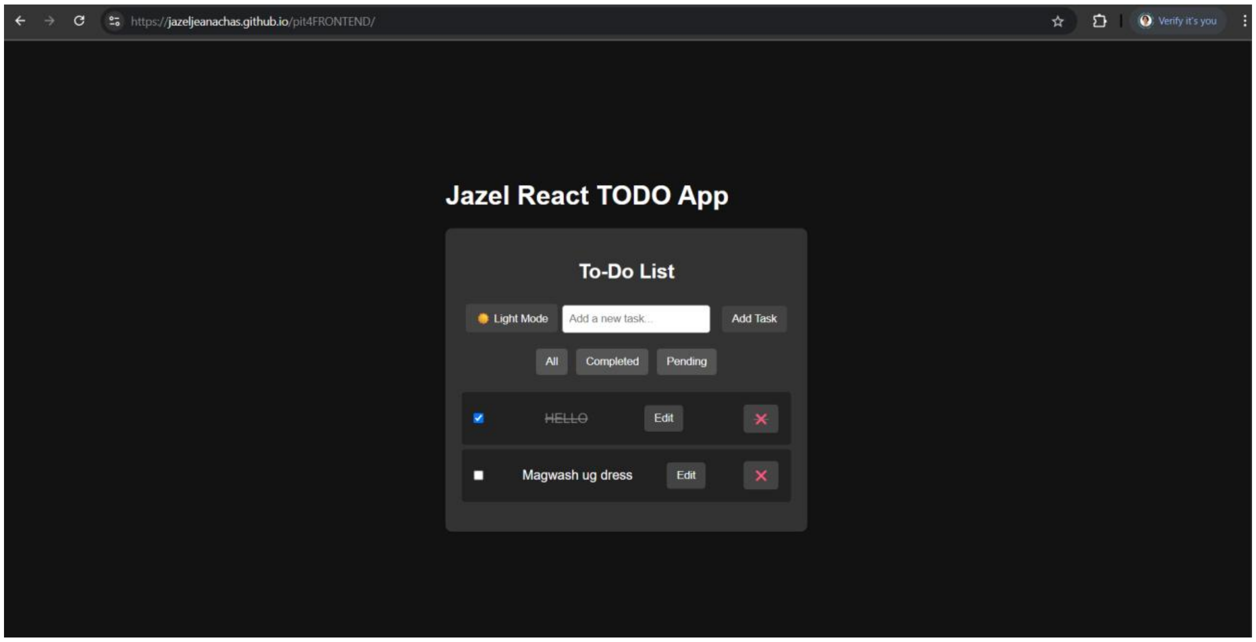
Technologies Used:

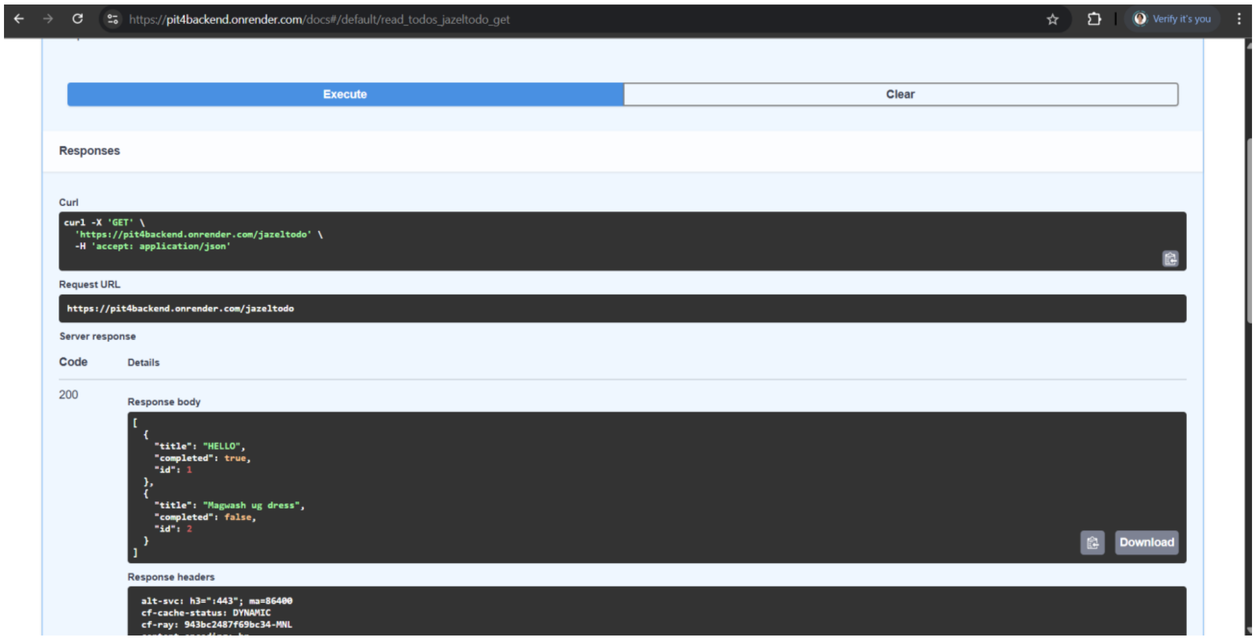
- Frontend: React + Vite
- Backend: FastAPI + SQLAlchemy
- Database: SQLite (Free Postgres instance on render had already been used for a previous activity)

- **Deployment:** Backend on Render, Frontend on GitHub Pages

Screenshots:







```
curl -X 'GET' \
  'https://pit4backend.onrender.com/jazeltodo' \
  -H 'accept: application/json'
```

```
https://pit4backend.onrender.com/jazeltodo
```

200

```
[
  {
    "title": "HELLO",
    "completed": true,
    "id": 1
  },
  {
    "title": "Magwash ug dress",
    "completed": false,
    "id": 2
  }
]
```

```
alt-svc: h3=":443"; ma=86400
cf-cache-status: DYNAMIC
cf-ray: 943bc2487f69bc34-MNL
```

Live Links:

Frontend: https://jazeljeanachas.github.io/pit4FRONTEND/
Backend: https://pit4backend.onrender.com/docs

Challenges Encountered:

- **Django REST Framework (DRF):** Deployment to Render was slow due to initial GitHub push issues, complex environment and database setup, managing multiple settings files, and unclear deployment errors.
- **FastAPI:** Though familiar with DRF, FastAPI required learning asynchronous code and project structure. Minor dependency and deployment issues on Render occurred, especially with Uvicorn setup and requirements, but overall it was easier to configure.