Rex Belli Isidore B. Lumantao BSIT-3R5

Todo List Application Documentation

Project Title: To-Do List App with FastAPI and React

This full-stack To-Do List application is built with FastAPI for the backend and React (Vite) for the frontend. It offers users the ability to:

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

The backend features a RESTful API and is integrated with a database using Postgresql. Postgresql was chosen for this project as it is a free instance on Render, Postgresql provides a lightweight, easy-to-use alternative that fits the development and small-scale deployment needs. The frontend communicates with the API using Axios and is designed for responsiveness with a theme-toggle feature.

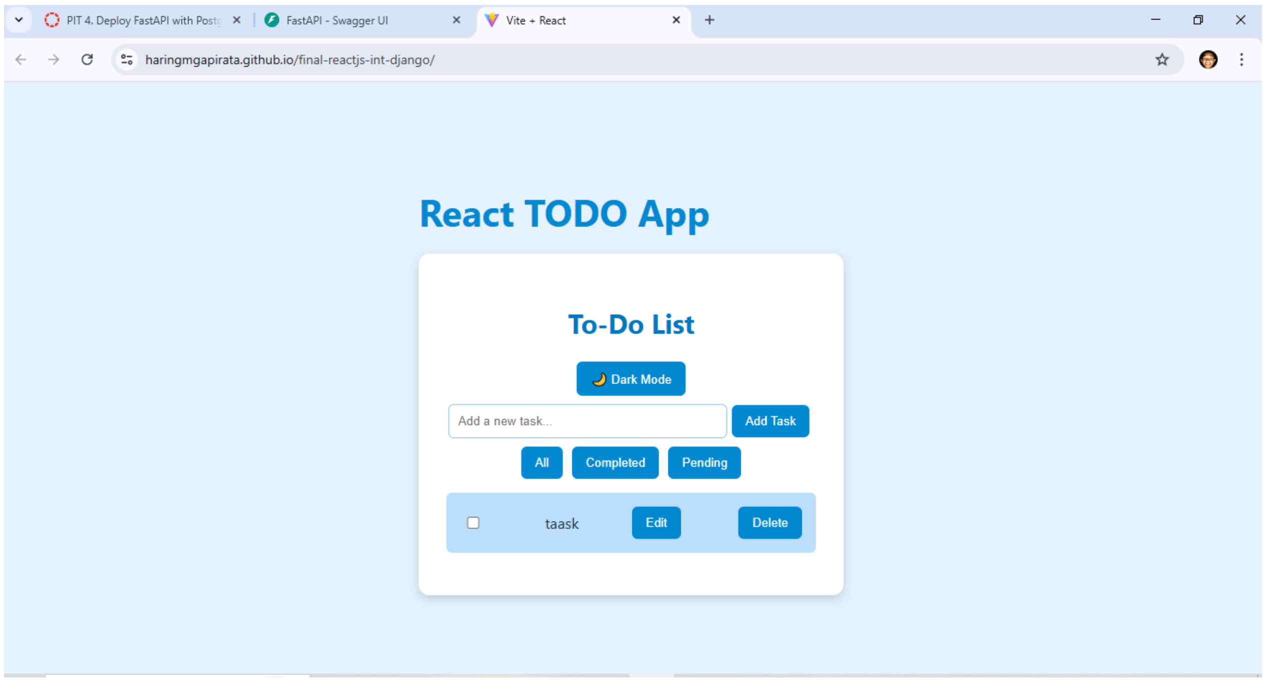
Feature	FastAPI	Django REST Framework (DRF)
Framework Type	Web framework focused on APIs	A toolkit for building APIs on top of Django
Performance	Very fast, asynchronous, and high-performance	Generally slower than FastAPI, synchronous
Asynchronous Support	Fully supports asynchronous programming	Limited async support, primarily synchronous
Ease of Use	Easy to use with automatic validation and docs	Easy to use but requires Django knowledge
Learning Curve	Shorter learning curve due to simplicity	Steeper, as it builds on Django's complexity
Validation	Automatic and built-in data validation	Needs serializers and custom validation

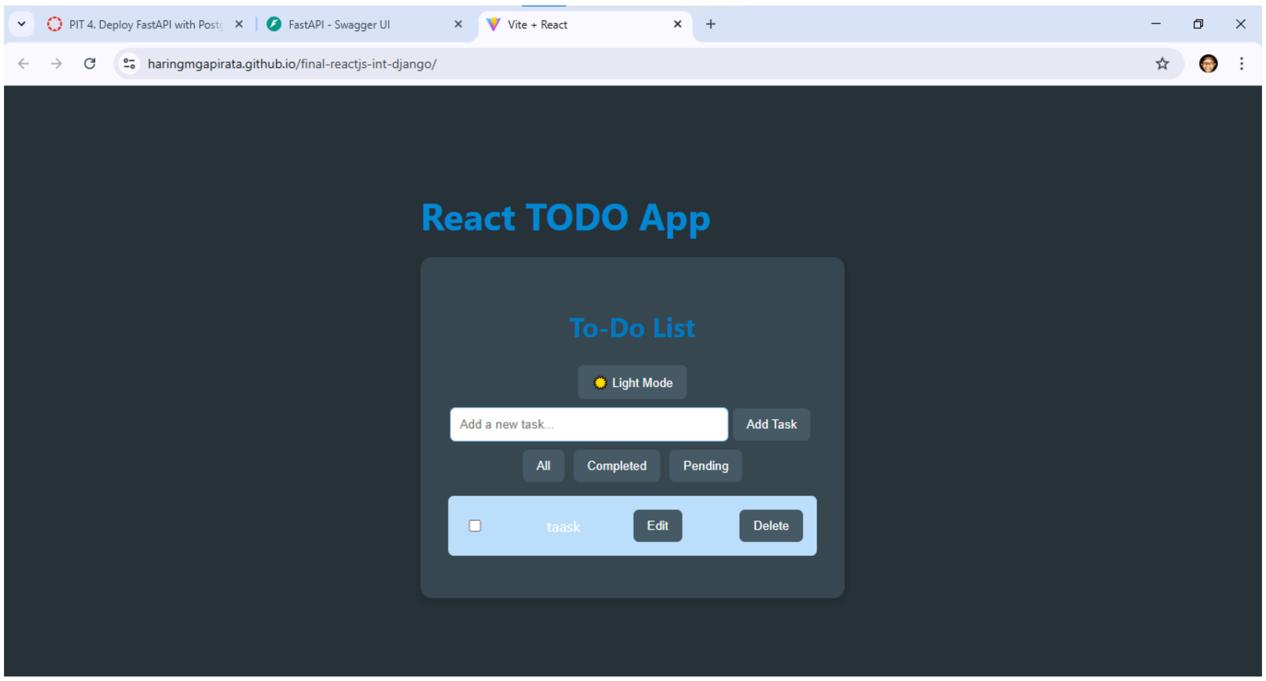
Technologies Used:

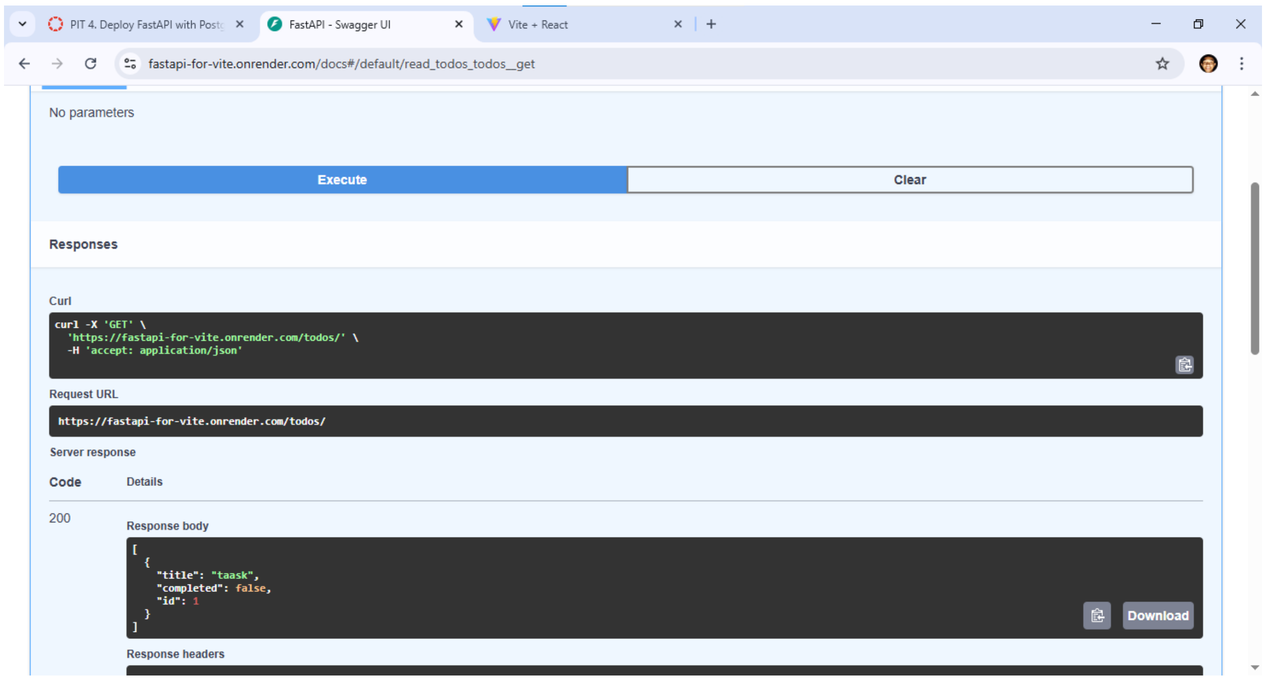
- Frontend: React with Vite
- Backend: FastAPI with Postgresql
- Database: Postgresql
- Deployment:
  - Backend deployed on Render ○

    Frontend hosted on GitHub Page

Screenshots:






PIT 4. Deploy FastAPI with Postg
FastAPI - Swagger UI
Vite + React
fastapi-for-vite.onrender.com/docs#/default/read_todos_todos__get

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'https://fastapi-for-vite.onrender.com/todos/' \
  -H 'accept: application/json'
```

Request URL

```
https://fastapi-for-vite.onrender.com/todos/
```

Server response

Code	Details
200	Response body

```
[
  {
    "title": "taask",
    "completed": false,
    "id": 1
  }
]
```

Download

Response headers

Live Links:

Backend: https://fastapi-for-vite.onrender.com/docs

Frontend: https://haringmgapirata.github.io/final-reactjs-int-django/

Challenges Faced:

Django REST Framework (DRF):

During my work with Django REST Framework, I encountered a major challenge when deploying the project to Render. The initial difficulty stemmed from pushing the code to GitHub, which ended up delaying the deployment more than I had anticipated. I had to spend considerable time identifying and fixing the issues before the project was successfully uploaded to the repository. Although the deployment proceeded without issues afterward.

FastAPI:

When I moved over to FastAPI, the process felt much more seamless. My existing experience with Django REST Framework made it easier to adjust and implement the needed features smoothly. FastAPI's clean design and clear documentation made backend setup and deployment hassle-free. Since the framework is intuitive and I was already familiar with similar technologies, I was able to complete everything quickly and without any major roadblocks.