Joeira Lou S. Dionela

To-Do List App with FastAPI and React

Project Overview

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

- Create, read, update, and delete tasks.
- Mark tasks as completed or pending.
- Filter tasks based on their status.
- Switch between light and dark modes for a more personalized experience.
- All data is saved persistently using a backend API connected to a SQLite database.

Technologies Used

Frontend:

- **React** (with Vite for fast builds and hot reloading)
- **Axios** (for HTTP requests)
- **React Icons** (for UI components)

Backend:

- **FastAPI** (for building the backend API)
- **SQLAlchemy** (for database interaction)
- **SQLite** (lightweight database)

Deployment:

- **Frontend**: Hosted on **GitHub Pages**
- **Backend**: Deployed on **Render**

Features

Backend Features:

- **RESTful API** built with **FastAPI**
- Database connection and task management using **SQLAlchemy**
- **CRUD operations** for tasks (Create, Read, Update, Delete)
- Task filtering by status (completed/pending)
- API documentation automatically generated by **FastAPI**

Frontend Features:

- **Task management interface** for adding, editing, and deleting tasks

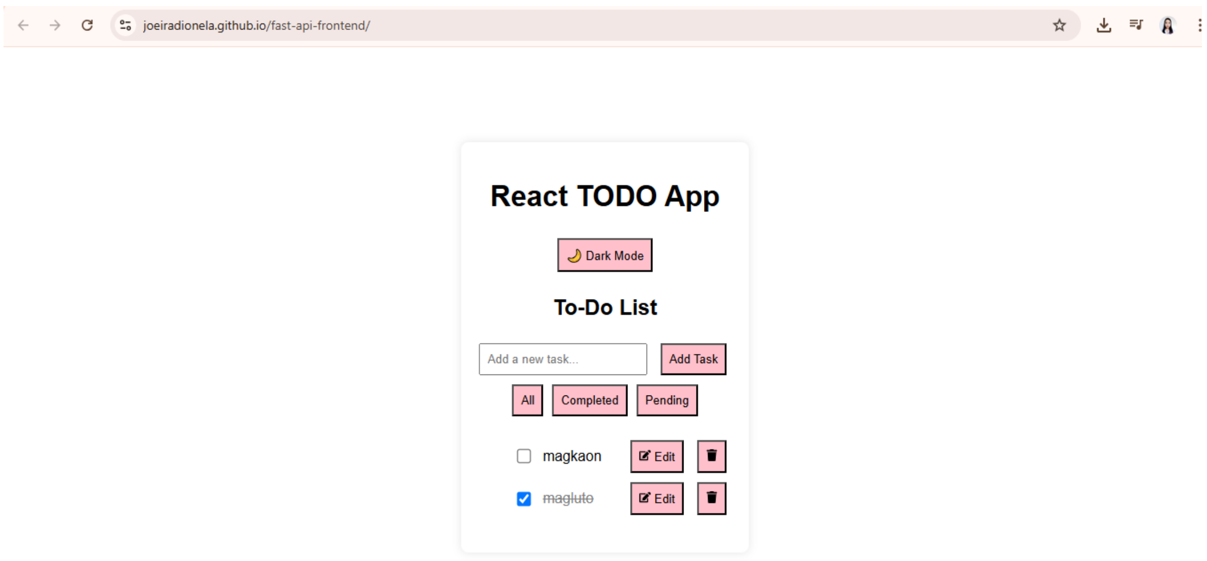
- **Light and Dark mode toggle** for UI personalization
- **Task filter** to view tasks by status (completed, pending, all)
- **Responsive design** for smooth user experience across devices

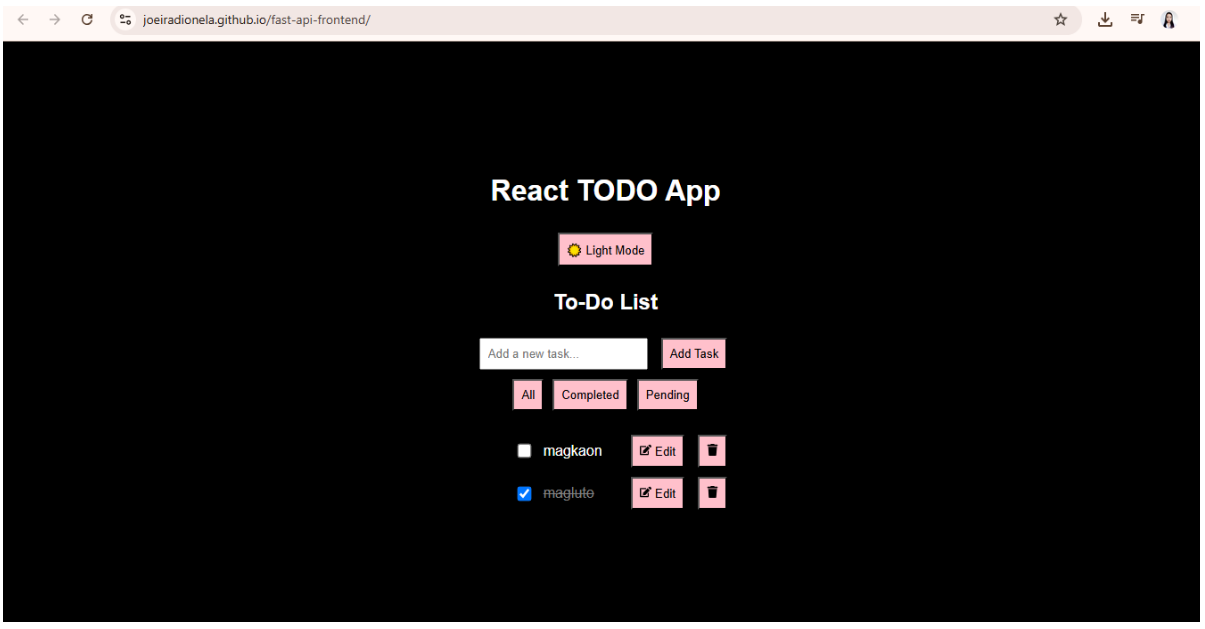
Deployment Links

- **Frontend:** https://joeiradionela.github.io/fast-api-frontend/
- **Backend:** https://fast-api-backend-dmz8.onrender.com/docs
- **Backend_url:** https://fast-api-backend-dmz8.onrender.com/todos/

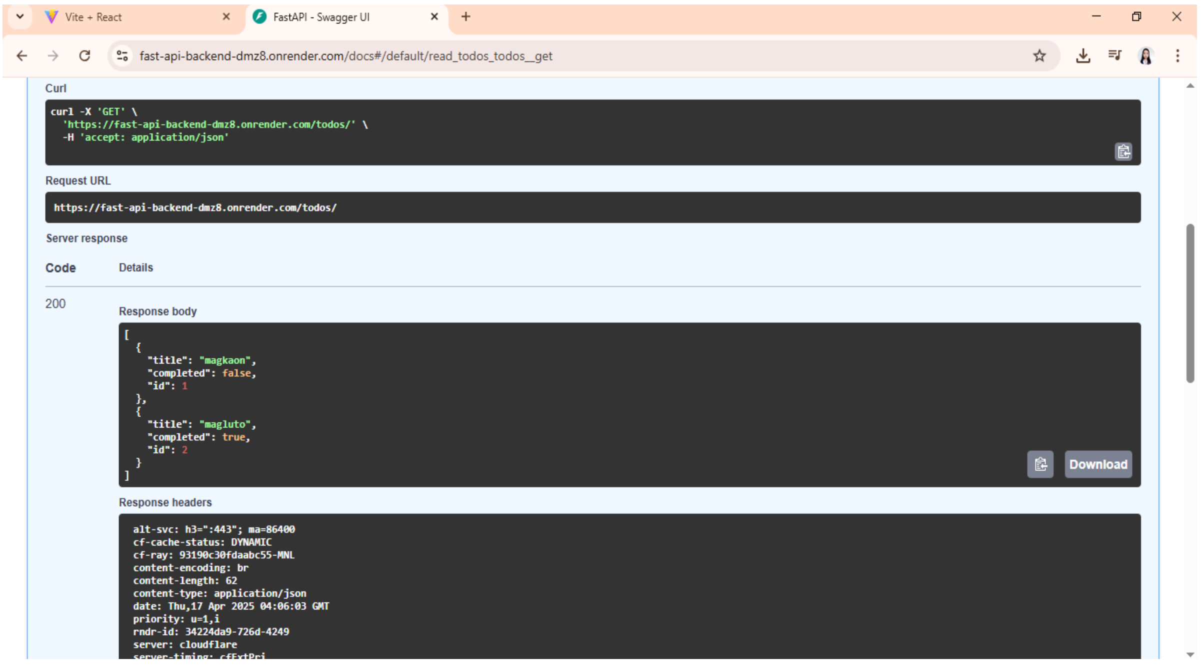
Screenshots

- **Frontend:**





- **Backend:**



fast-api-backend-dmz8.onrender.com/docs#/default/read_todos_todos__get

Curl

```
curl -X 'GET' \
  'https://fast-api-backend-dmz8.onrender.com/todos/' \
  -H 'accept: application/json'
```

Request URL

```
https://fast-api-backend-dmz8.onrender.com/todos/
```

Server response

Code Details

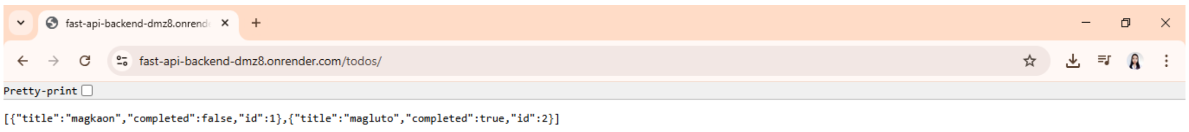
200

Response body

```
[
  {
    "title": "magkaon",
    "completed": false,
    "id": 1
  },
  {
    "title": "magluto",
    "completed": true,
    "id": 2
  }
]
```

Response headers

```
 alt-svc: h3=":443"; ma=86400
 cf-cache-status: DYNAMIC
 cf-ray: 93190c30fdaabc55-MNL
 content-encoding: br
 content-length: 62
 content-type: application/json
 date: Thu,17 Apr 2025 04:06:03 GMT
 priority: u=1,i
 rndr-id: 34224da9-726d-4249
 server: cloudflare
```



fast-api-backend-dmz8.onrender.com/todos/

Pretty-print

```
[{"title":"magkaon","completed":false,"id":1},{"title":"magluto","completed":true,"id":2}]
```

Challenges and Learnings

Backend (FastAPI) Challenges:

- **CORS Configuration: Initially struggled with setting up CORS for frontend-backend communication across different domains.**

- **Error Handling & Validation: Took time to set up proper error messages and response formats for client-side handling.**

Frontend (React) Challenges:

- **State Management: Faced difficulties ensuring that the frontend UI updated correctly when tasks were added, updated, or deleted.**
- **API Integration: Worked through issues with asynchronous requests and ensuring smooth UI updates during slow API responses.**

Database and Deployment Challenges:

- **SQLAlchemy: Worked on setting up the database models and relationships properly.**
- **Deployment: Had issues with setting environment variables and configuring base URLs for GitHub Pages and Render.**

Learnings:

- **Overcoming these challenges improved my understanding of FastAPI, React, and deployment processes.**
- **I learned the importance of clear error handling, efficient state management, and smooth deployment setups.**