Bongo, Reige J
BSIT -3R5
Applications Development and Emerging Technologies

Building a Full-Stack

To-Do List Application with FastAPI Documentation

Task:

Demonstrate your ability to integrate a front-end React application with a FastAPI backend by developing a fully functional To-Do List application. This includes implementing CRUD operations, task filtering, and a dark mode feature.

Objectives:

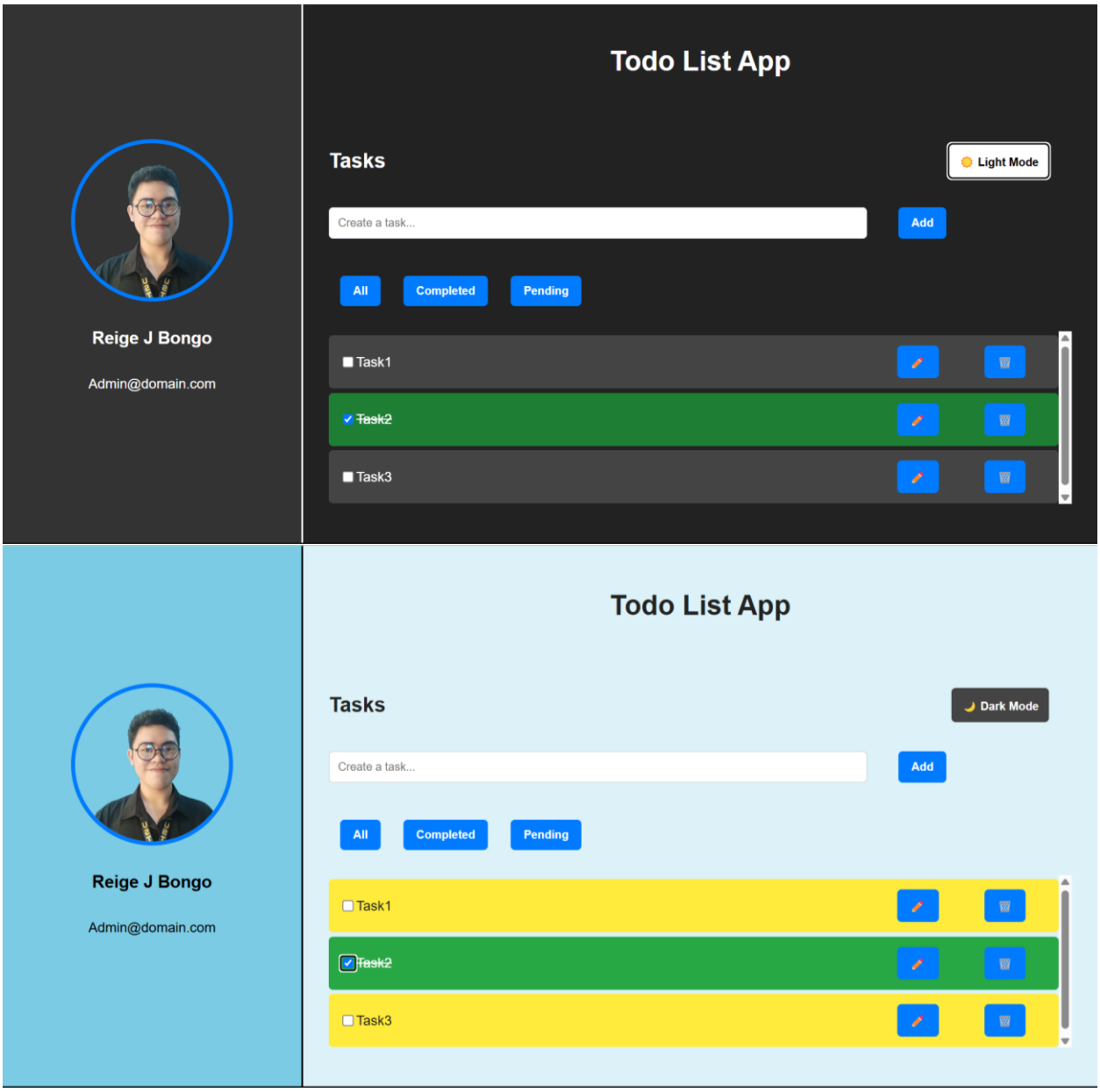
1. Learn full-stack development
2. Use REST API for frontend-backend interaction
3. Practice deploying frontend (GitHub Pages) & backend (Render)
4. Build something simple and usable

Tech Stack

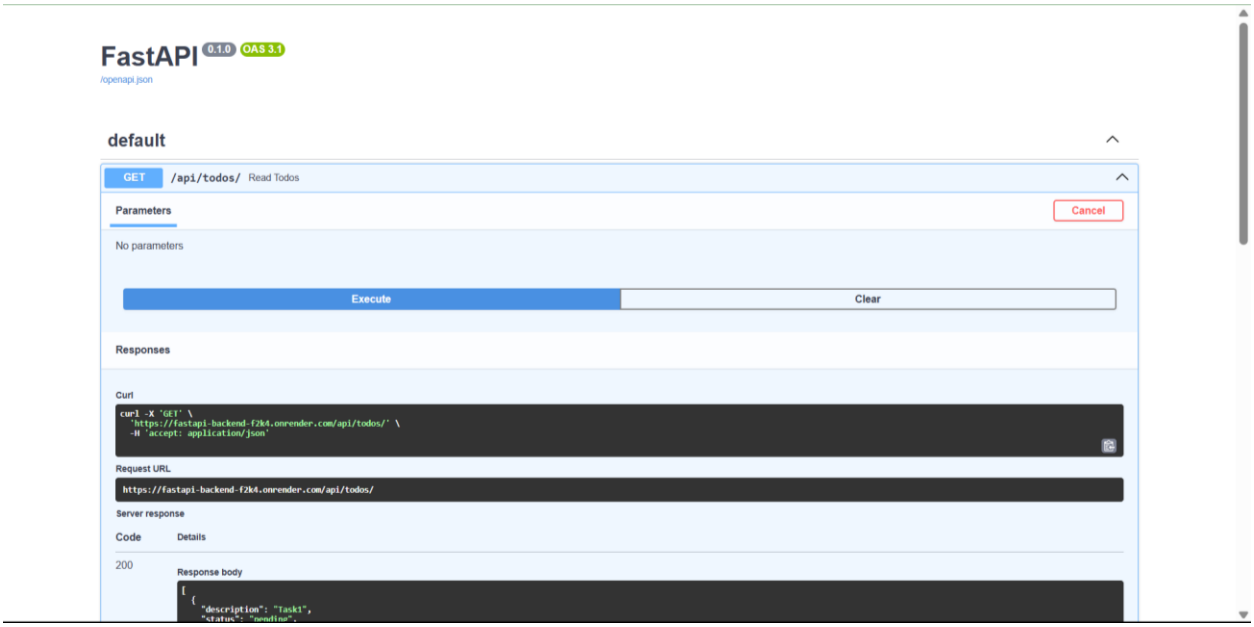
Layer	Technology
Frontend	React + Vite
Backend	FastAPI
Database	SQLite
Deployment	GitHub Pages (Frontend), Render (Backend)

In this case, SQLite was used instead of PostgreSQL for this project because Render only provides one free PostgreSQL database per account. Since the backend service already used in previous activity, SQLite was chosen as a simple and effective alternative for a small-scale todo list app.

Front end:


Todo List App
Tasks
Light Mode
Create a task...
Add
All
Completed
Pending
Task1
Task2
Task3
Reige J Bongo
Admin@domain.com
Todo List App
Tasks
Dark Mode
Create a task...
Add
All
Completed
Pending
Task1
Task2
Task3
Reige J Bongo
Admin@domain.com

Backend (Render)



Difference between Django REST Framework (DRF) and FastAPI (highlight key features, advantages, disadvantages)

- Django REST Framework is an open-source and high-level python based framework. Django comes with many built-in features like Object-Relational Mapping, Admin interface, User authentication system, and Form handling. Django aims to simplify the development process by promoting clean and pragmatic coding style which makes applications more robust and scalable. The advantages of using Django are secure, scalable, and browsable API. The disadvantages on the other hand are its performance and learning curve. FastAPI on the other hand is a lightweight, modern, and high-performance web framework for building APIs in python. Fast API offers speed, safety, automatic documentation, and asynchronous support. Fast API is super lightweight and fast making it ideal for microservices, APIs, and real-time applications. Also developer friendly and great async support. However, FastAPI has a younger ecosystem with only few plugins and extensions, and lacks of built-in ORM, admin panel, and templating unlike Django. The key difference between Django REST Framework and FastAPI is their speed, purpose,

and design. Django is suitable for all-in-one solution and full web applications while FastAPI is suitable for fast, scalable APIs and microservices.

Challenges you encountered in development and deployment and how you solved them for both DRF and FastAPI.

- The challenges I've encountered so far for Django REST Framework is that it is very overwhelming at first since there are so many layers like views, serializers, models, and etc. The most common error that I've encountered during development is the URL structure, CORS headers, unable to connect the database and API related issues. Luckily, we've solve these kinds of challenges since we are focusing on Django in our IT322 subject which is Integrative Programming and Technologies.

- The challenges I've encounter for FastAPI is that unlike Django where there are so many python files at the start of the development, FastAPI don't have a single file and you have to manually add it. And when it comes to built-in tools, FastAPI don't have tools like admin panel or ORM by default, which requires extra setup for things like authentication. Right now I'm still practicing this new framework so that we can have an option what backend to use for our future project especially our CAPSTONE.