Rehana Nicole Ruilan

BSIT-3R1

Documentation: Full-Stack To-Do List App (FastAPI + React)

FastAPI vs Django REST Framework (DRF)

FastAPI

- **Type Hinting**: Native support with Pydantic makes validation and data modeling easy.
- **Speed**: Built on ASGI (Starlette), making it faster than Django REST in many use cases.
- **Auto Documentation**: Generates Swagger and ReDoc automatically.
- **Ease of Use**: Lightweight and minimalistic syntax.

Django REST Framework

- **Integrated Admin Panel**: Comes with Django's powerful admin tools.
- **Robust ORM**: Built-in with Django's mature ORM.
- **Heavier Setup**: More boilerplate code and configurations.
- **Slower**: Built on WSGI, not ASGI.

Conclusion: FastAPI is preferred for faster development and simpler APIs, especially when you want to keep things light and async-capable.

Challenges & Solutions

Challenge 1: CORS Issues

Problem: Netlify frontend couldn't fetch from Render backend due to blocked CORS policy. **Solution**: Added CORS middleware to main.py:

```
app.add_middleware(
    CORSMiddleware,
    allow_origins=["https://fastapi-pit4.netlify.app"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)
```

Challenge 2: Incorrect API Endpoint Structure

Problem: API URL in React was missing /todos/ path, causing 404 or CORS errors.
Solution: Corrected API constant to:

```
const API = "https://fastapi-pit4.onrender.com/todos/";
```

Challenge 3: Render PostgreSQL Integration (Optional)

Problem: Render doesn't persist SQLite between deployments. **Solution**: Switched to PostgreSQL and stored the connection string in Render's Environment Variables using the DATABASE_URL key.

Deployment Summary

Backend

- **Platform**: Render
- **Live URL**: https://fastapi-pit4.onrender.com
- **Start Command**: ./start.sh
- **CORS Configured**: Yes

Frontend

- **Platform**: Netlify
- **Live URL**: https://fastapi-pit4.netlify.app
- **Build Command**: npm run build
- **Publish Directory**: dist/

Conclusion

This project was a hands-on implementation of a modern web stack. It improved my understanding of:

- Frontend-backend integration
- API routing and data models
- Real-world deployment of full-stack applications

With a clean UI, complete CRUD functionality, task filtering, and dark mode toggle, this To-Do List App is a minimal but powerful example of using FastAPI + React in production.