DRF vs FastAPI Comparison

FEATURE	Django REST Framework (DRF)	FastAPI
Framework Type	Based on Django, full-featured	Lightweight, asynchronous Python framework
Speed	Slower, due to synchronous nature	Much faster thanks to async support
Flexibility	Opinionated, tightly coupled with Django ORM	Very flexible with pydantic and SQLAlchemy
Learning Curve	Easier for Django users	Steeper for beginners unfamiliar with async
Auto Docs	DRF browsable API	Swagger UI + ReDoc by default
Community Support	Large, mature ecosystem	Growing, especially among microservices devs

Table 1. show the difference between Django REST Framework (DRF) and FastAPI

Advantages of DRF:
• Full integration with Django admin and ecosystem.
• Great for monolithic apps with complex auth and admin needs.

Advantages of FastAPI:
• High performance with async I/O.
• Modern Python type hinting and validation (pydantic).
• Better for microservices and async-heavy APIs.

Challenges and Solutions

Challenge	Description	Solution
Database Connection on Render	FastAPI couldn't detect DATABASE_URL	Used os.environ.get() and fallback for local dev
Missing dependencies in frontend	React wouldn’t build initially	Ensured react-scripts and static files were correctly configured
CORS issues between frontend and backend	React failed to fetch from backend	Installed fastapi-cors, allowed frontend origin
Broken orm_mode warning	Pydantic updated to from_attributes	Updated model configs to reflect this change
Git push issues	Remote was ahead	Pulled latest first, then force pushed

Frontend render deploy issues	Environment not detected	Used static site deploy, ensured npm run build was correct
Neon DB not connecting	Confusing UI for credentials	Used test connection, copied full PostgreSQL URI

Table 2. shows the challenges encountered in development and deployment and how I solved them for both DRF and FastAPI.

Why I used Neon DB instead of render's own postgreSQL DB?

Free version of render doesn't allow to build more than one database. And for previous activities to be safe I used Neon DB instead, and it's easy to use.