Jet AI Project Documentation

Getting Started

1. Clone the repo: https://github.com/ambroquach29/jetai-project

Setting up Backend

2. Navigate to the **/backend** directory
3. Run **npm install** to install server dependencies
4. Create a **.env** file with these values (also refer to .env.example)

```
5. PORT=5000
6. ACCESS_TOKEN=apollo-starter-kit
7. HASURA_URL="http://localhost:8080/v1/graphql"
8. HASURA_ADMIN_SECRET=
9. POSTGRES_URL="postgresql://postgres:postgrespassword@localhost:5432/postgres?schema=public"
```

10. Set up *Postgres database and Hasura GraphQL Engine* running as Docker containers using Docker Compose. The **docker-compose.yml** has the all the configurations needed.
11. Prereq: We need Docker Desktop working on your machine
12. Run **docker compose up -d** to start the containers

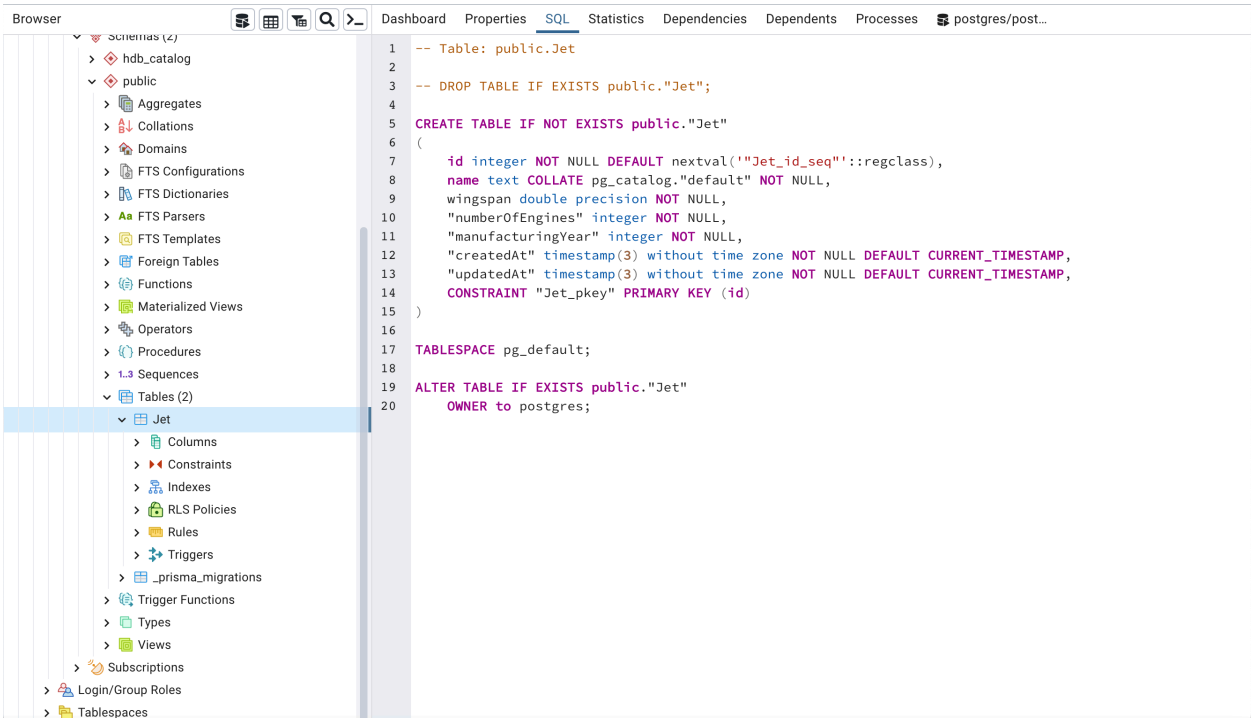
```
❯ docker compose up -d
[+] Running 3/3
 ✔ Container backend-postgres-1              Running                                                    0.0s
 ✔ Container backend-data-connector-agent-1  Healthy                                                    0.0s
 ✔ Container backend-graphql-engine-1        Running                                                    0.0s
❯ docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED        STATUS                  PORTS
                    NAMES
787c0ca764a9   postgres:15                            "docker-entrypoint.s…"   37 hours ago   Up 37 hours             0.0.0.0:54
32->5432/tcp            backend-postgres-1
27b5ac14e091   hasura/graphql-engine:v2.37.0          "/bin/sh -c '\"${HGE_…"   38 hours ago   Up 37 hours (healthy)   0.0.0.0:80
80->8080/tcp            backend-graphql-engine-1
e60609f297c6   hasura/graphql-data-connector:v2.37.0   "/app/run-java.sh"       38 hours ago   Up 37 hours (healthy)   5005/tcp,
0.0.0.0:8081->8081/tcp   backend-data-connector-agent-1
 ~/De/jetai-project/backend  master !9 ?2
```

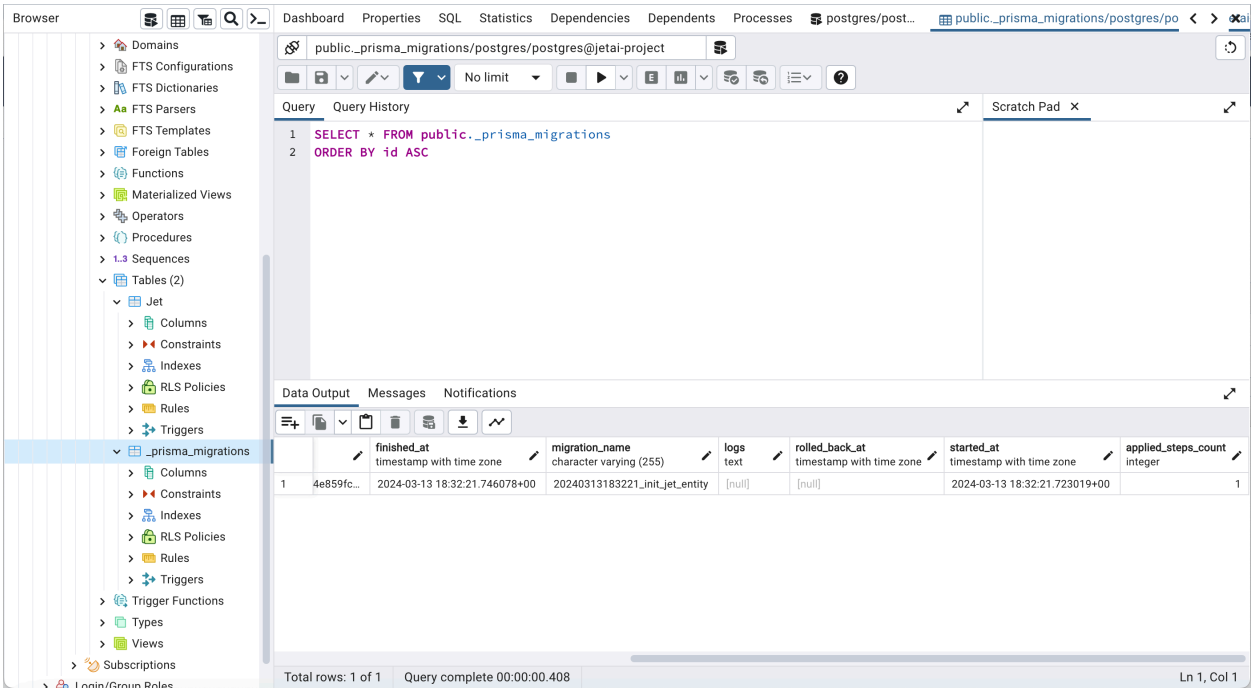
13. Run **docker ps** to check its status
14. Now we've done setting up a *local Postgres database connected Hasura GraphQL Engine* - allow us to communicate with DB via GraphQL queries or REST
15. Initialize *Postgres db* using *Prisma ORM*
16. Navigate to **/backend/src/prisma**

17. Run **npx prisma migrate dev --name init_jet_entity** - This command creates and applies timestamped migration script(s) to our SQL db based on the model(s) defined inside schema.prisma

```
❯ npx prisma migrate dev --name init_jet_entity
Environment variables loaded from ../../.env
Prisma schema loaded from prisma/schema.prisma
Datasource "db": PostgreSQL database "postgres", schema "public" at "localhost:5432"

Applying migration `20240313183221_init_jet_entity`

The following migration(s) have been created and applied from new schema changes:

migrations/
  └─ 20240313183221_init_jet_entity/
    └─ migration.sql

Your database is now in sync with your schema.

✔ Generated Prisma Client (v5.11.0) to ./../../node_modules/@prisma/client in 104ms

 ~/De/jetai-project/backend/s/schema  master !3 ?2                                  ✔ 4s
```

18. We can use *PgAdmin4* to verify the state of our DB, *Prisma* also provides a *migrations table* that keeps record of our migration history

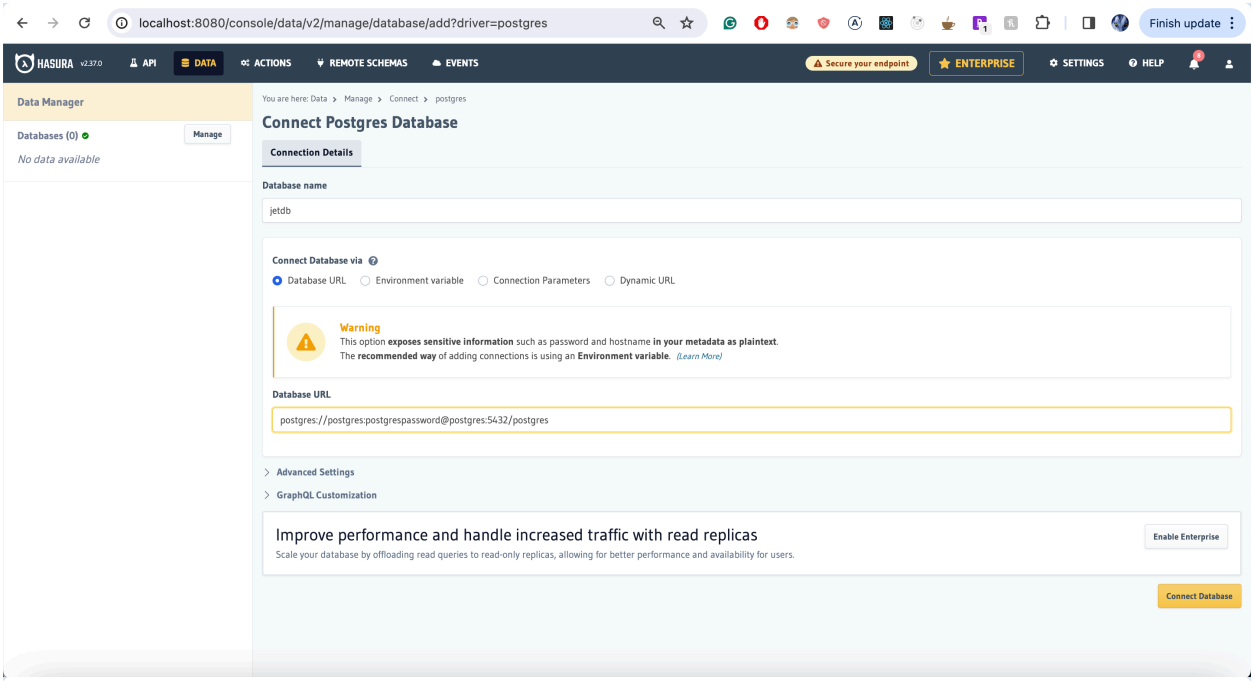


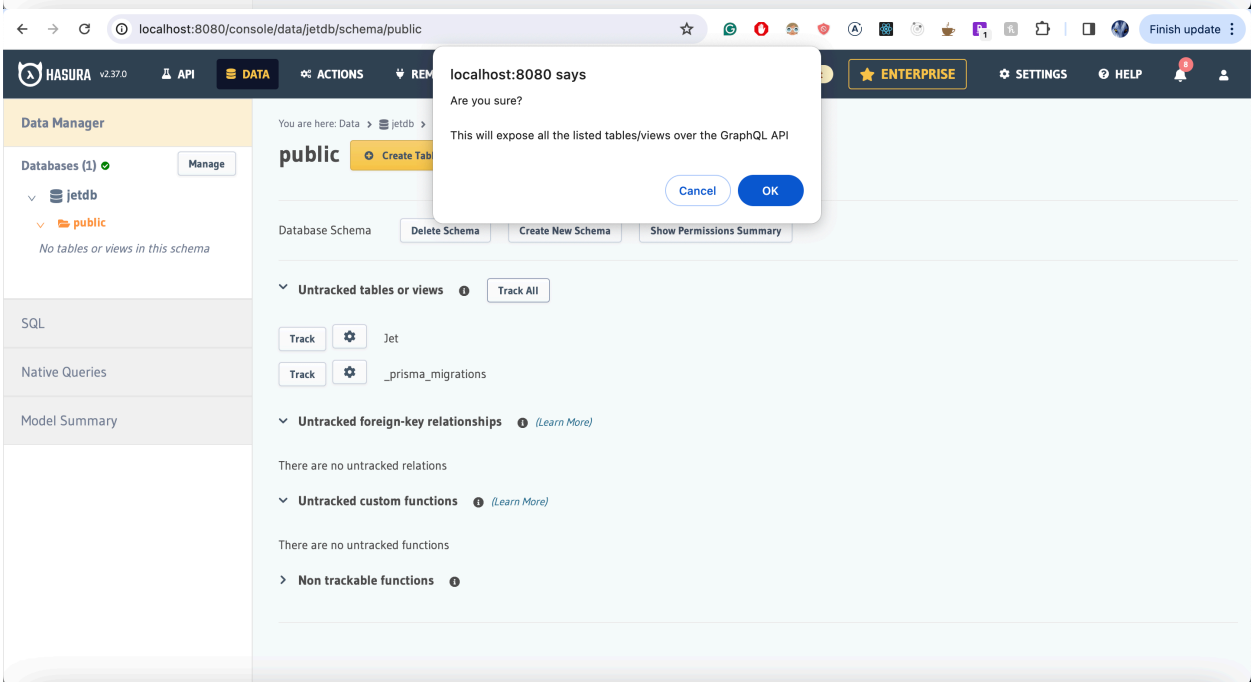


19. Navigate back to **/backend** directory and run **npm run dev** to start the Apollo server

```
❯ npm run dev

> jetai-backend@1.0.0 dev
> ts-node-dev src/server.ts

[INFO] 01:35:22 ts-node-dev ver. 2.0.0 (using ts-node ver. 10.9.2, typescript ver. 5.4.2)
(node:67373) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)


        🚀 Server is running!
        🎮 Listening on port 5000
        🔰 Query at http://localhost:5000/graphql
```

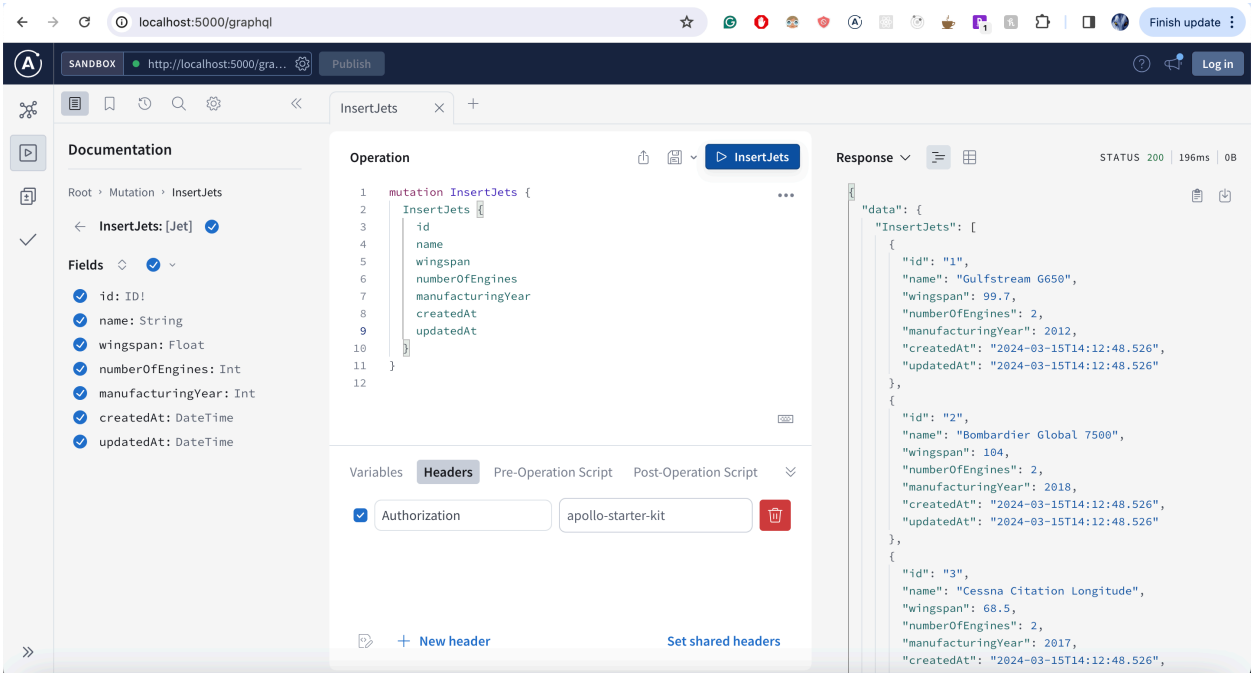
20. Go to **http://localhost:8080** on your browser, under the **Data** tab connect to Database via URL postgres://postgres:postgrespassword@postgres:5432/postgres





21. After you connected to Postgres on Hasura, you will see something like the screenshot above. Click on **Track All**.

22. Go to http://localhost:5000/graphql and run the InsertJets query as shown below to populate our DB



23. Here are the links to other GraphQL Queries and Mutations that you can run on the GraphQL API (Headers and Variables included): [QueryJets](), [InsertJet](), [UpdateJet](), [DeleteJet]()

Setting up LLM

24. Navigate to the **/llm** directory
25. Run **source llm-env/bin/activate** to activate the virtual environment
26. Replace **openai_api_key** in **main.py** with my OpenAI key below

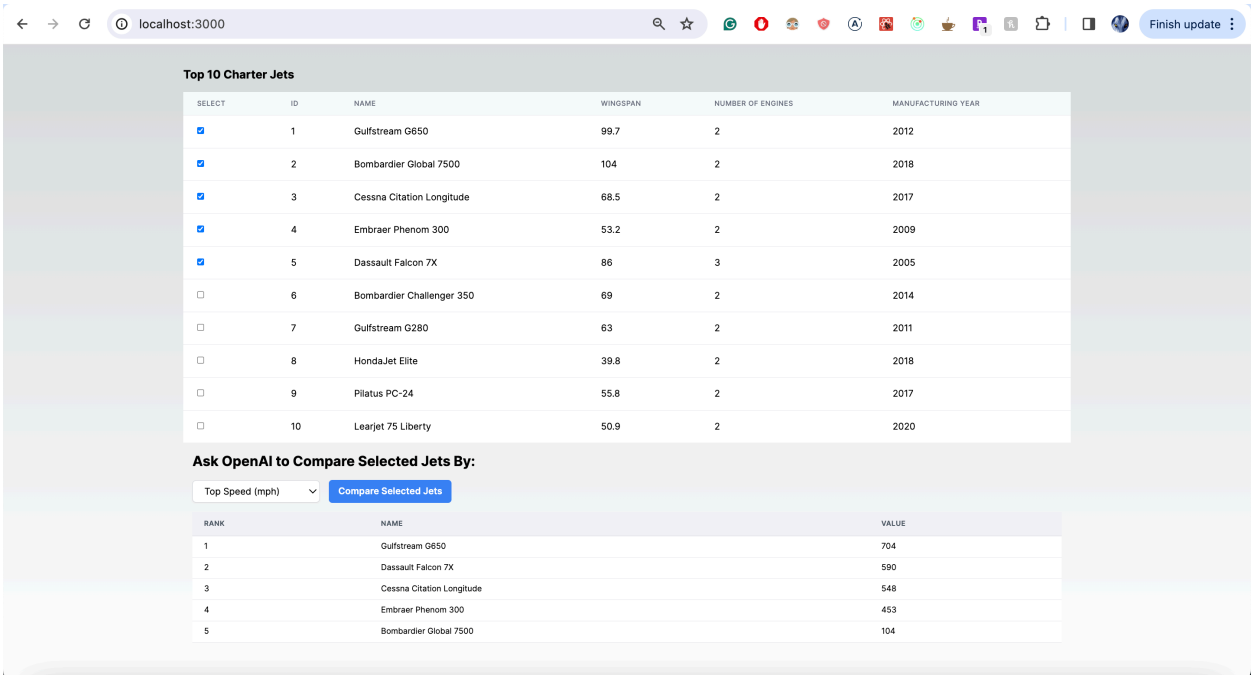
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

27. Run **uvicorn main:app –reload** to start the running our LLM service using Fast API - Check http://localhost:8000/docs for Swagger endpoint specifications

```
❯ uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['/Users/ambroquach/Desktop/jetai-project/llm']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [85898] using StatReload
/Users/ambroquach/Desktop/jetai-project/llm/llm-env/lib/python3.12/site-packages/langchain/llms/__init__.py:548: LangChainDeprecat
ionWarning: Importing LLMs from langchain is deprecated. Importing from langchain will no longer be supported as of langchain==0.2
.0. Please import from langchain-community instead:

`from langchain_community.llms import OpenAI`.

To install langchain-community run `pip install -U langchain-community`.
  warnings.warn(
/Users/ambroquach/Desktop/jetai-project/llm/llm-env/lib/python3.12/site-packages/langchain_core/_api/deprecation.py:117: LangChain
DeprecationWarning: The class `langchain_community.llms.openai.OpenAI` was deprecated in langchain-community 0.0.10 and will be re
moved in 0.2.0. An updated version of the class exists in the langchain-openai package and should be used instead. To use it run `
pip install -U langchain-openai` and import as `from langchain_openai import OpenAI`.
  warn_deprecated(
INFO:     Started server process [85900]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:52613 - "GET /docs HTTP/1.1" 200 OK
INFO:     127.0.0.1:52613 - "GET /openapi.json HTTP/1.1" 200 OK
```

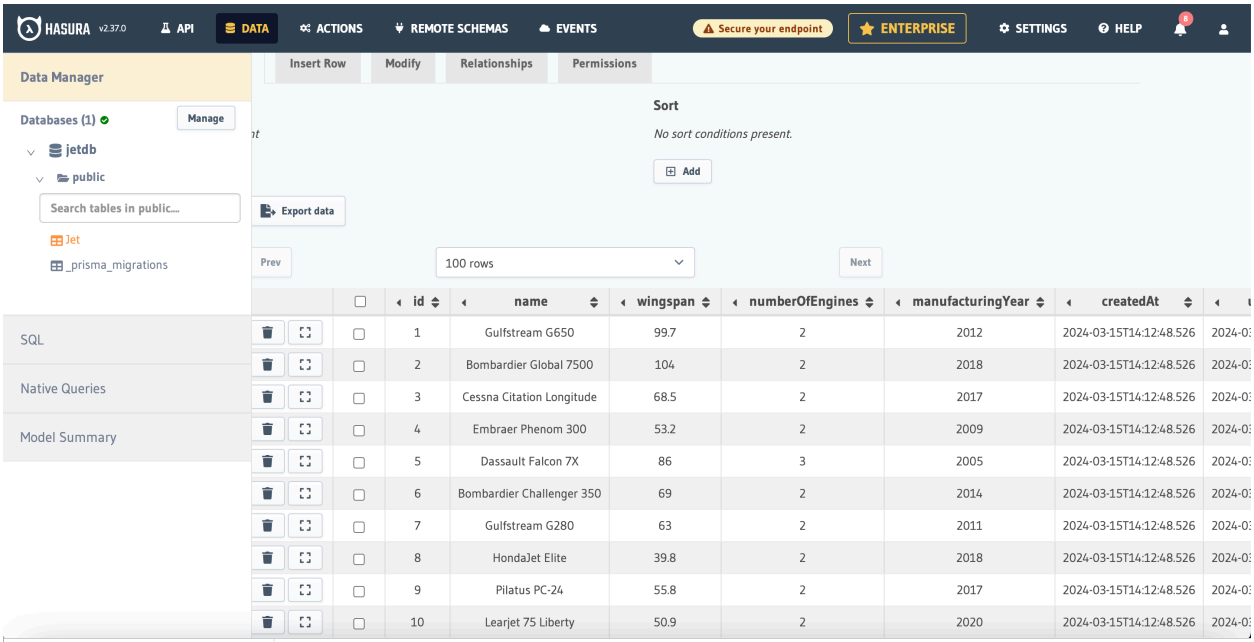
Setting up Frontend

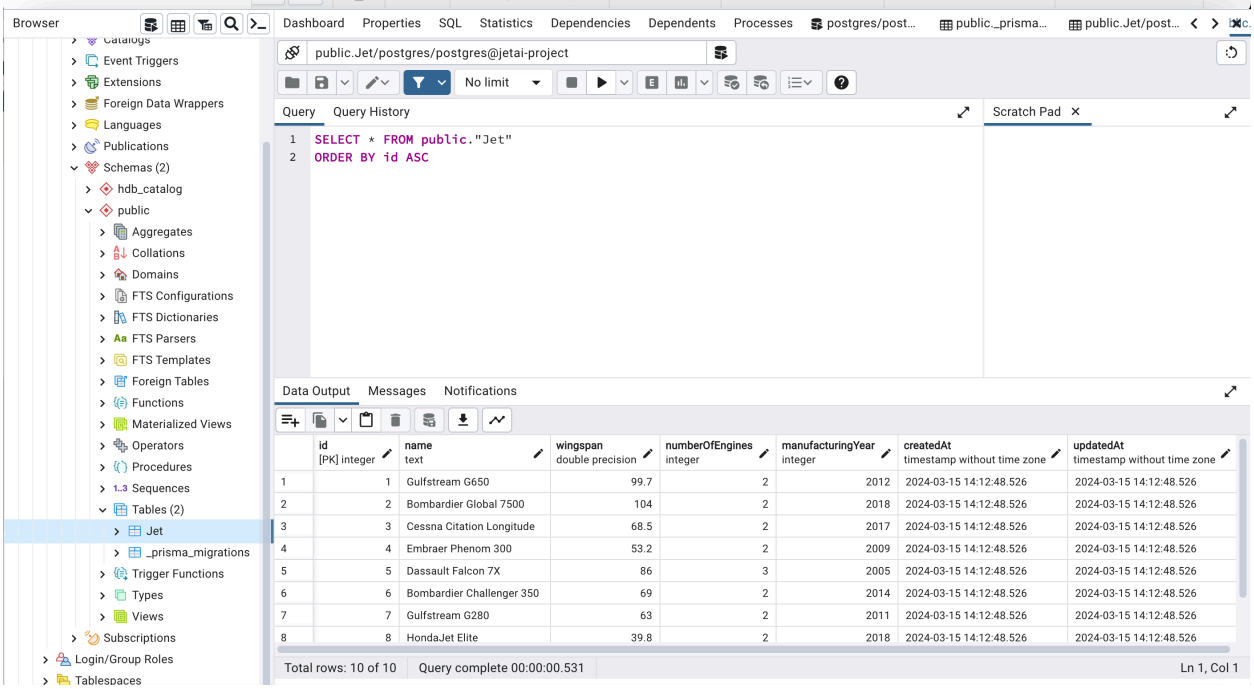
28. Navigate to the **/frontend** directory and run **npm install** then **npm run dev** to start the NextJS app.
29. Go to http://localhost:3000 to start using the Jet AI Comparison Tool

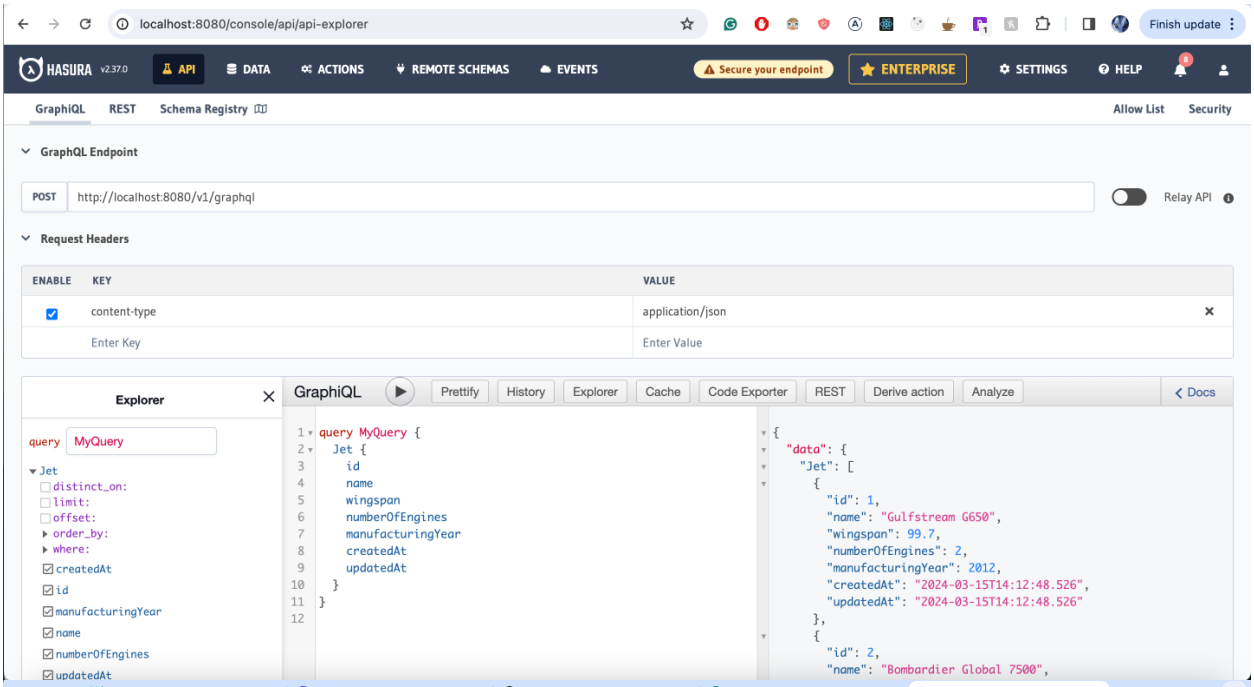


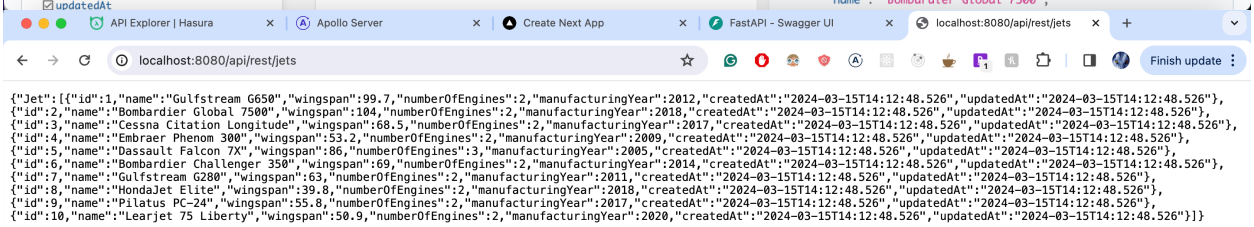
~~Thank you for your time and reviewing this project. Feel free to reach out to me if you have any question or there's any hiccup in the process of getting it to run.~~

Other related screenshots/examples:

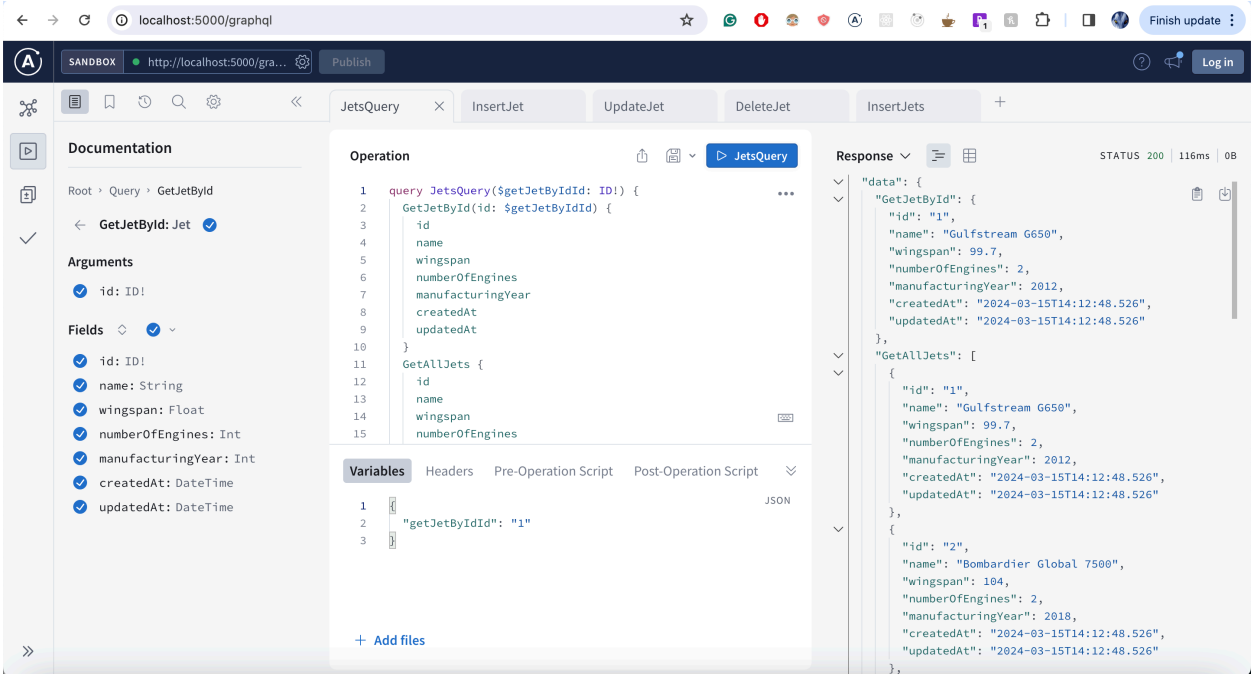


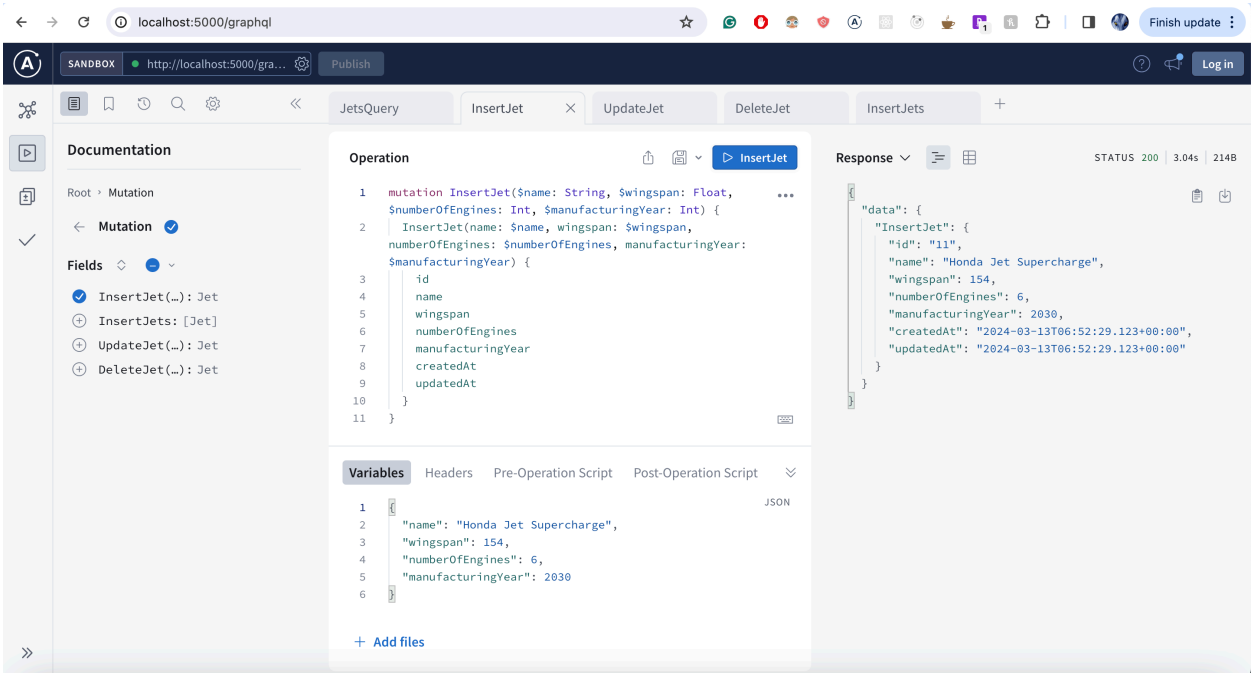







```
{"Jet":[{"id":1,"name":"Gulfstream G650","wingspan":99.7,"numberOfEngines":2,"manufacturingYear":2012,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":2,"name":"Bombardier Global 7500","wingspan":104,"numberOfEngines":2,"manufacturingYear":2018,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":3,"name":"Cessna Citation Longitude","wingspan":68.5,"numberOfEngines":2,"manufacturingYear":2017,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":4,"name":"Embraer Phenom 300","wingspan":53.2,"numberOfEngines":2,"manufacturingYear":2009,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":5,"name":"Dassault Falcon 7X","wingspan":86,"numberOfEngines":3,"manufacturingYear":2005,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":6,"name":"Bombardier Challenger 350","wingspan":69,"numberOfEngines":2,"manufacturingYear":2014,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":7,"name":"Gulfstream G280","wingspan":63,"numberOfEngines":2,"manufacturingYear":2011,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":8,"name":"HondaJet Elite","wingspan":39.8,"numberOfEngines":2,"manufacturingYear":2018,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":9,"name":"Pilatus PC-24","wingspan":55.8,"numberOfEngines":2,"manufacturingYear":2017,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"},
{"id":10,"name":"Learjet 75 Liberty","wingspan":50.9,"numberOfEngines":2,"manufacturingYear":2020,"createdAt":"2024-03-15T14:12:48.526","updatedAt":"2024-03-15T14:12:48.526"}]}
```


localhost:5000/graphql
Finish update
SANDBOX
http://localhost:5000/gra...
Publish
Log in
JetsQuery
InsertJet
UpdateJet
DeleteJet
InsertJets
Documentation
Root › Mutation
Mutation
Fields
InsertJet(…): Jet
InsertJets: [Jet]
UpdateJet(…): Jet
DeleteJet(…): Jet
Operation
InsertJet
mutation InsertJet($name: String, $wingspan: Float, $numberOfEngines: Int, $manufacturingYear: Int) {
InsertJet(name: $name, wingspan: $wingspan, numberOfEngines: $numberOfEngines, manufacturingYear: $manufacturingYear) {
id
name
wingspan
numberOfEngines
manufacturingYear
createdAt
updatedAt
}
}
Response
STATUS 200 | 3.04s | 214B
{
"data": {
"InsertJet": {
"id": "11",
"name": "Honda Jet Supercharge",
"wingspan": 154,
"numberOfEngines": 6,
"manufacturingYear": 2030,
"createdAt": "2024-03-13T06:52:29.123+00:00",
"updatedAt": "2024-03-13T06:52:29.123+00:00"
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"name": "Honda Jet Supercharge",
"wingspan": 154,
"numberOfEngines": 6,
"manufacturingYear": 2030
}
Add files