N26 AI Engineer Interview

RAG presentation
02/11/2025
Gowtham B C

Architecture Overview

- Gemini api key: Dev time hit threshold with gemini and got shadowbanned.
- v1.0: Local ollama Gemma 3n models for LLM, nomic-embed-text for embedding (link)
- V2.0: Working solution with Gemini Embedding and Gemini 2.5 Flash (link)

Features implemented:

- UI to ingest vectors (http://localhost:8000/)
- Qdrant for Vector DB and search engine (http://localhost:6333/dashboard)
- Reranker algorithm (cross-encoder/ms-marco-MiniLM-L-6-v2)
- Api test cases using Pytest and Good exception handling
- Clear all Vector Collections
- Swagger for API docs (http://localhost:8000/docs)
- Model Drift score for DB health check (Index Coherence score)
- Git Repository (https://github.com/Gowtham171996/N26-Gowtham-AI-Engineer)
- Dockerised & docker-compose run in any hardware (Qdrant, Ollama, API-Backend)

Assumptions & Trade-offs

File type

- Explicitly mentioned as .txt, but can take word, pdf with texts. Mounted to local “data” directory to ingest real time (Poor for tables pypdf , imaged pdf OCR)

Track of Model Vector drift

- Vector index drift Score identification along with updated time

When deployed will be on GPU Cloud

- While it is fast, it is expensive for POC. While our own models are free to run with GPU enabled hardware.

From PoC to production (15k parallel requests)

Features to add

- Add redis (A bit higher TTL)
- Deploy on Kubernetes
- Reverse proxy of Traefik, Nginx with load balancer
- Run multiple instances of api-backend container
- CI/CD pipelines (Built, Has issues)
- Vanilla HTML to framework such as Typescript React for UI
- Load balancer

(Single instance with uvicorn can handle approx 7k parallel requests, for time consuming tasks, thread pool should be used. If needed Gunicorn can be used)

Monitoring & Metrics

- Document permission Management (Private: User only, Public: All user) (Qdrant supports its efficiently)
- Simple sqlite User Auth Management
- Config in yaml to Config from SQLite table only for Admin
- Real time logs view Dashboard (Like docker logs, only to Admin)
- Rate limiter per minute & Security headers (Geo fencing if needed)
- Different pipelines for images and tables
- Kafka Message queue
- Grafana, Telemetry and prometheus

Personal takenote

- While in free tier i should have been wary about gemini embedding. I ran POC and it blocked me. So I have implemented both local RAG as well as Cloud RAG.
- While it was fun weekend coding, there was certainly time pressure of deadline
- Could have cleaned code better
- Reduced docker size
- More time integration testing
- But was very happy when docker compose ran in peer testing.

Looking forward for next round of interview!! Cheers!!

RAG ASSIGNMENT DOCUMENTATION

N26 AI Engineer role

02/11/2025

Gowtham B C
bcgowtham17@gmail.com

RAG ASSIGNMENT DOCUMENTATION

Declaration:

Since Gemini key wasn't given, I ran through **free tier quota** for embedding and LLM and couldn't develop or build using cloud. Hence local ollama docker was developed, once out of Shadow ban I refactored with Gemini cloud to complete the assignment.

While 90% of the code is same and 100% of concept is same, it is totally free as well.

I may try to make the entire solution live as well: https://rag.solventumrnd.com/ (Not in scope of assignment)

Prerequisite

Run all these commands to ensure, Cuda enabled Nvidia graphics card and docker is up and running on hardware. Stable internet and 20GB of hardware memory

1. docker

2. open docker desktop application

If 1 and 2 is command not found then Nvidia GPU is disabled.

If Nvidia graphics card is configured, please visit docker compose file and uncomment the line from 72 to 78 i.e. Deploy section By default, GPU is disabled it since it is too slow on CPU, please enable it.

Steps to run:

Please follow every step carefully:

1. Have signed in GitHub account:
   Repo link: https://github.com/Gowtham171996/N26-Gowtham-AI-Engineer
   Download the file: https://github.com/Gowtham171996/N26-Gowtham-AI-Engineer/releases/tag/v2.1

2. Extract the zipped file

3. Open command prompt in folder path and run **DIR** to verify right path

4. Make sure n26rag-compose.yml file is visible. If not you are in wrong path.

5. Run "**docker compose -f n26rag-compose.yml up -d**"

RAG ASSIGNMENT DOCUMENTATION

(Make sure with Cuda GPU enabled or only CPU) Read prerequisites.

6. **Wait for 10mins** (pip requirements are very bulky in size)
   The backend will say "**Application complete**". Till then it is kept on running. So **please wait.**

7. Open the URLs:
   - http://localhost:8000/ (for main application)
   - http://localhost:6333/dashboard (for qdrant)
   - http://localhost:8000/docs (for docs)

8. The UI will load with filed ingested. if not click "Trigger full ingestion" button

9. Ask a query to the model and observe the links it pulled information from.

10. Want to modify ingestion? In "data" folder add a new file (txt, word doc, pdf with text)

11. Delete the vector collection by clicking on UI

12. Perform step 7 to ingest new file.

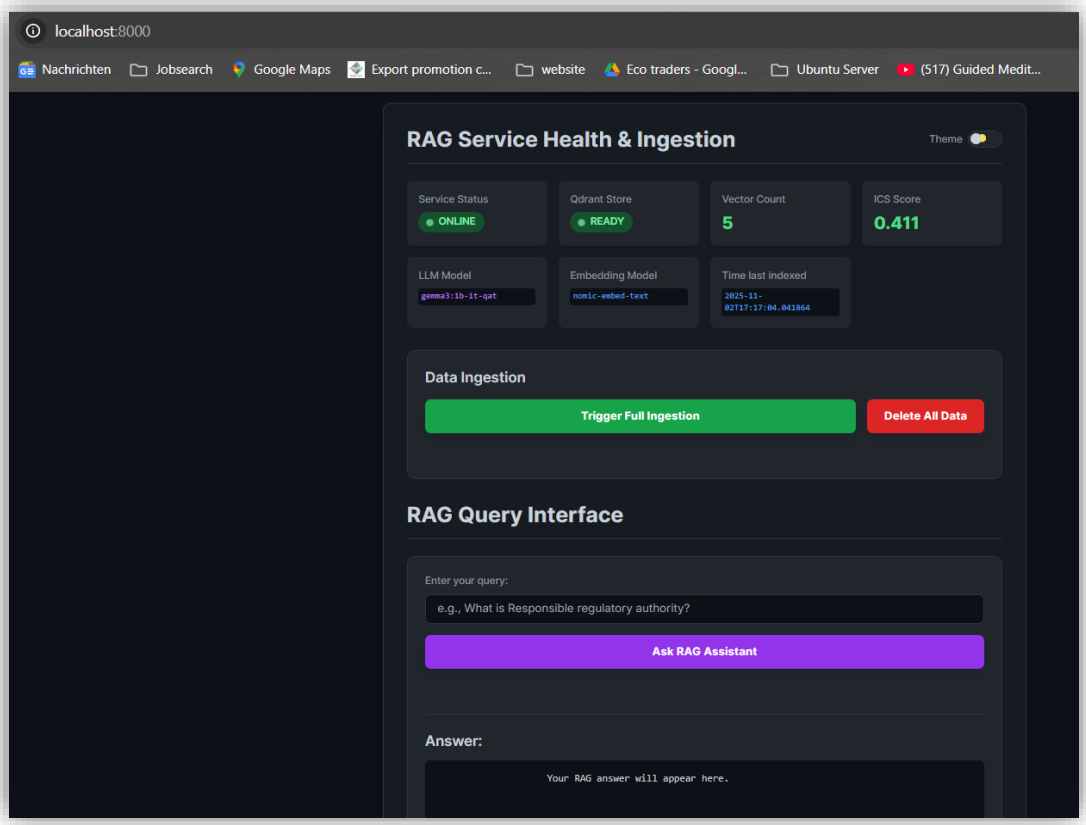
13. Perform step 8 with newly added file.

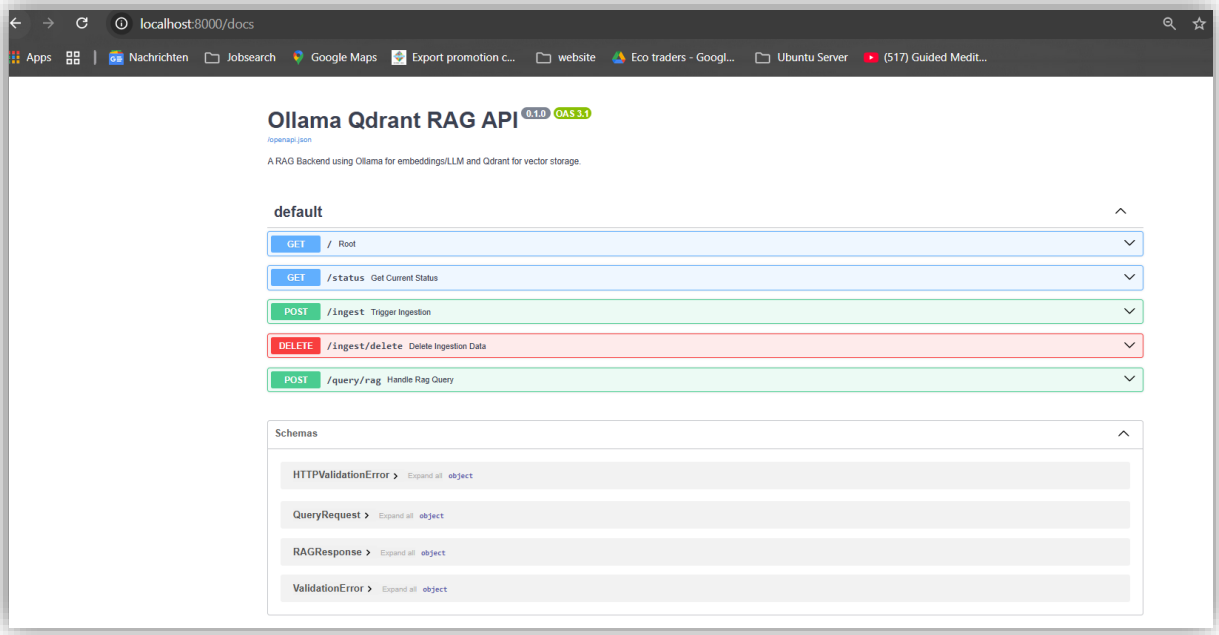
Known Issues:

1. Docker build images are heavy. (For Cuda enabled environment, along with reranker algorithm. PIP requirements are too much.)

RAG ASSIGNMENT DOCUMENTATION

Gallery





RAG ASSIGNMENT DOCUMENTATION