RAG - Documentation

**Prerequisites :**

* Python
* Docker and Docker Compose
* Hugging Face API token (I have already added my token, if you want you can change it to yours at line 20)

**Installation :**

* Clone the repository:

https://github.com/AkashHEAT/Retrieval-Augmented-Generation-task-.git

* Install Python Dependencies:

pip install fastapi uvicorn pymilvus sentence-transformers requests beautifulsoup4 huggingface\_hub

* Start the milvus services:

Create a new folder named milvus\_docker and put the docker-compose.yml file in it and run the following command.

docker-compose -f milvus\_docker/docker-compose.yml up

**Running the application :**

**Accessing using Terminal:**

1. Start the FastAPI Server:

python main.py

1. Test the Endpoints:

* Load Data:

curl -X POST 'http://127.0.0.1:8000/load' \

-H 'Content-Type: application/json' \

-d '{

"url": "<https://en.wikipedia.org/wiki/Generative_artificial_intelligence>"

}'

* Query Data:

curl -X POST 'http://127.0.0.1:8000/query' \

-H 'Content-Type: application/json' \

-d '{"query": "academic discipline of artificial intelligence"}'

You could able to see the outputs in the terminal.

**Accessing using Swagger UI (**prefer**)**

1. Once your FastAPI server is running, you can access the Swagger UI at http://127.0.0.1:8000/docs
2. **Interact with the API**
3. Load Data :

* Click on the /load endpoint to expand it.
* Click on the "Try it out" button.
* Enter the JSON payload in the request body, for example:

{

"url": "<https://en.wikipedia.org/wiki/Generative_artificial_intelligence>"

}

- Click the "Execute" button to send the request.

* You will see the response from the server below the request.

1. Query Data :

* Click on the /query endpoint to expand it.
* Click on the "Try it out" button.
* Enter the JSON payload in the request body, for example:

{

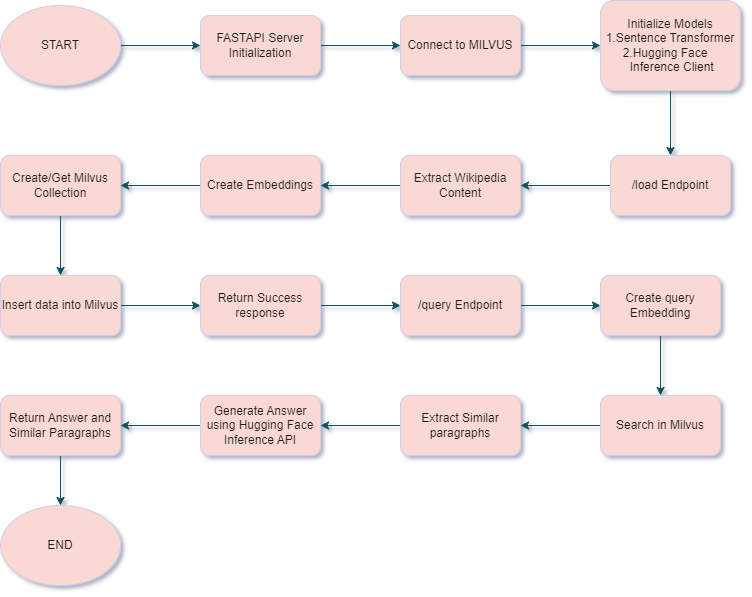
"query": "academic discipline of artificial intelligence"

}

- Click the "Execute" button to send the request.

* You will see the response from the server below the request.

**Code flow:**



**Explanation :**

1. **Imports**: Import necessary libraries and modules.

2. **FastAPI App**: Initialize the FastAPI application.

3. **Milvus Connection**: Connect to the Milvus vector database.

4. **Model Initialization**: Initialize the SentenceTransformer model and the Hugging Face Inference Client.

5. **Data Models**: Define Pydantic models for the Wikipedia URL and query input.

6. **Extract Wikipedia Content**: Function to extract text content from a Wikipedia page.

7. **Create Milvus Collection**: Function to create a Milvus collection for storing embeddings.

8. **/load Endpoint**: Endpoint to load data from a Wikipedia URL, create embeddings, and store them in Milvus.

9. **/query Endpoint**: Endpoint to query the stored data, search for similar content in Milvus, and generate an answer using the Hugging Face Inference API.

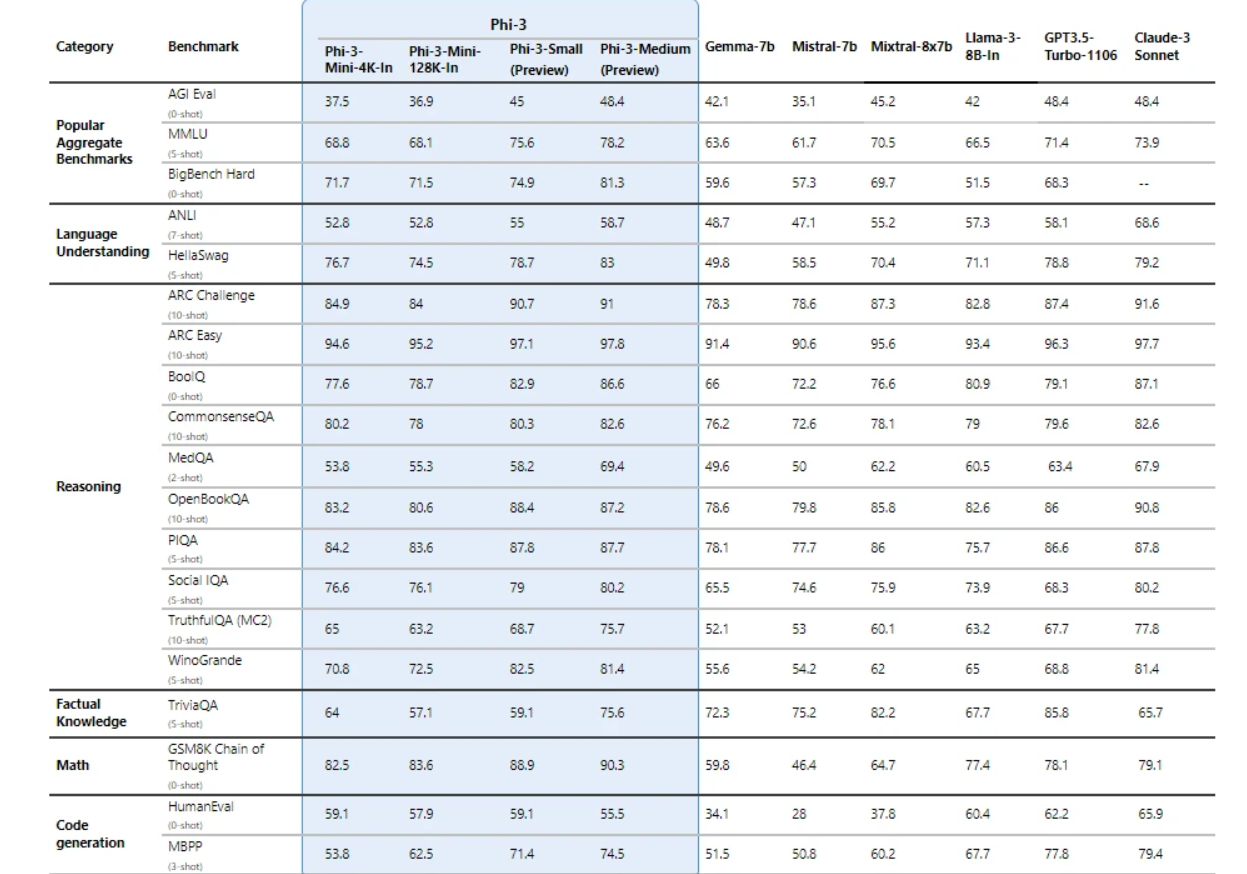
10. **Root Endpoint**: A simple root endpoint to check if the API is running.

11. **Run the Server**: Start the FastAPI server.

**Analysis:**

Why Microsoft/Phi-3-mini-4k-instruct ?

* **High Performance in a Small Package:** Despite its relatively small size, the model achieves state-of-the-art results across a variety of benchmarks, outperforming larger models in several domains.
* **Cost-Effective:** The compact size of Phi-3-Mini-4K-Instruct makes it more cost-effective to deploy compared to larger models, reducing computational and financial overheads.
* **Versatile Applications:** The model's ability to handle a wide range of tasks, from conversational AI to code generation, makes it a versatile tool for developers and organizations.



why uvicorn ?

🡺It's optimized for speed and minimal resource usage, making it a good choice for high-performance applications.

🡺Uvicorn supports features like hot reloading, which automatically reloads the server when code changes are detected, making development faster and more efficient.

Sentence transformer:

🡺 all-MiniLM-L6-v2 (<https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>)

Milvus-Vector DB:

🡺Installation reference(<https://milvus.io/docs>)

Start the Milvus services using Docker Compose: docker-compose up -d

Verify that the containers are running: docker-compose ps

To stop Milvus, run: docker-compose down