Question & Answer RAG pipeline

- Project Presentation -

Balázs Menkó

KPMG
Data Scientist / ML Engineer
Job Application

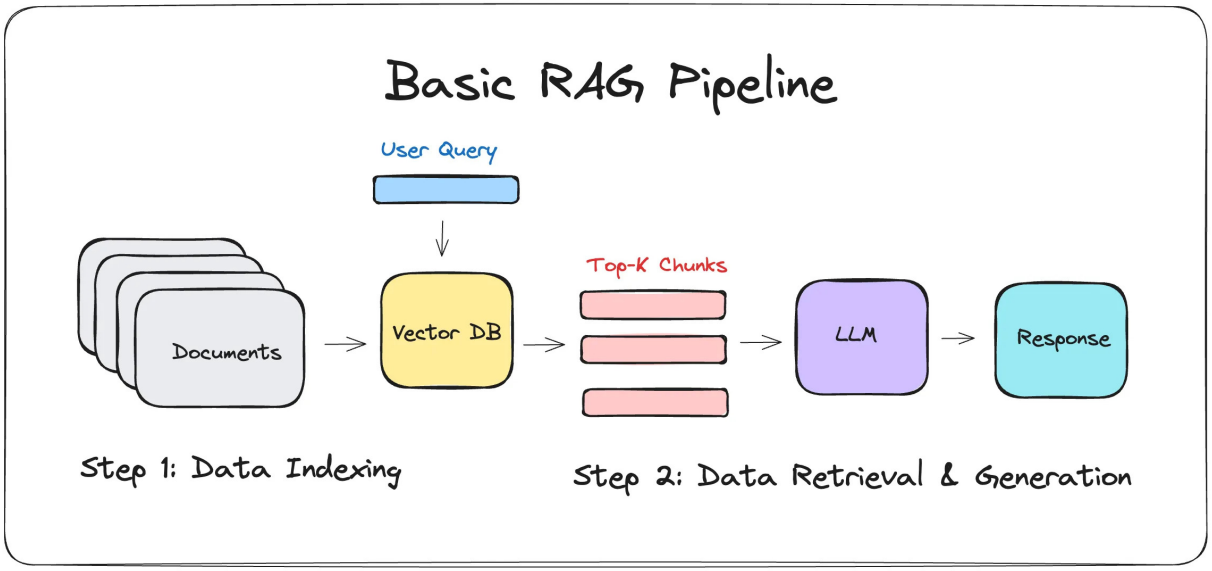
November 14, 2025

Project Description

- Build a Question–Answering assistant
- Use a Retrieval-Augmented Generation (RAG) approach
  - Vector DB
  - Retriever-ranker layer
  - Pretrained LLM layer to generate final answer
- Source documents: Wikipedia / Board Game Manual PDFs

RAG Pipeline Overview

1. Ingest data → chunk → embed (ChromaDB)
2. Retrieve with SentenceTransformer
3. Re-rank with CrossEncoder
4. Generate answers with Llama3.2 (Ollama)



Source: medium.com/rag

Vector Store: ChromaDB

- Persistent embedding database
- Stores text lines/chunks + metadata (source file + line/chunk id)
- Embedding saved to `chroma.sqlite3`

Retrieval + Ranking

- **Retriever:**
  - `SentenceTransformer("multi-qa-MiniLM-L6-cos-v1")`
  - Finds semantic matches
- **Ranker:**
  - `CrossEncoder("cross-encoder/ms-marco-MiniLM-L-6-v2")`
  - Scores semantic matches
- Produces clean, reliable context for answer generation
- Source: Hugging Face

Pretrained LLM

- OpenAI API - not free
- Gemini API - no response

- Local model served via Ollama
- Used model: **Llama3.2** & **Mistral**
- Ensures:
  - free usage
  - reproducibility
  - data privacy (no cloud calls)

Wikipedia Search RAG pipeline

- Extract (max 4) keyword with **Llama3.2**
- Fetch Wikipedia articles to a `.txt` document
- Use the pipeline (Chroma + Retrieval/Ranking + LLM)
- Problem: hard to evaluate later

Board Game Manual RAG pipeline

- Source of idea: a YouTube video
- 11 board game rules in PDF file
- Same pipeline

Result Testing with `PyTest` + **Mistral** model:

- 5 question with answers
- Mistral model decides whether an answer is correct or not
- 4 test passed + 1 failed (Llama could not answer)
- A full test took around 15 minutes

Streamlit Application

- Built for Board Game Rule Q&A project
- Minimal interface to get a question
- Show the answer after the generation process finished

Docker Containerization

- Hardest part, since I am not experienced in Docker
- Watched different videos on the topic
- Used LLMs to generate dockerization (OpenAI + Gemini)

Ensuring Accessibility for End Users

- Web-based App using Streamlit
- Cloud Hosting (AWS, MS Azure)

Further Development Opportunities

Wikipedia Q&A

- Multi-turn chat
- Source designation: article links
- Remove downloaded data after finished conversation

Board Game Manual Q&A

- Multi-turn chat
- Advanced Streamlit UI
- More precise source designation
- OCR support for scanned PDFs