Week 15: Augmented Generation & LLMOps

Agenda

In this session, we will discuss:

- Introduction to Retrieval-Augmented Generation
- A workflow for RAG apps
- Implementing RAG Workflows with Vector Databases (including evaluation)
- Using LLMOps to manage Gen AI Apps (case study: RAG App)

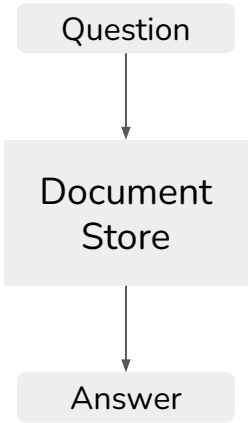
Generative AI Enables Document Q&A at Scale

Our investment process and signal research has evolved closely alongside the latest in data and quantitative techniques. Many of the valuable data sets we leverage today are larger, less structured, and generally more complex in nature relative to what was previously available. This also means they require more robust tools and techniques to analyze. Think in terms of financial news articles, earnings call transcripts, analyst research reports, regulatory filings. As technologies progressed over the years, we were able to benefit from the exponential growth of data and start using more unstructured data.

Dennis Walsh, Goldman Sachs Asset Management

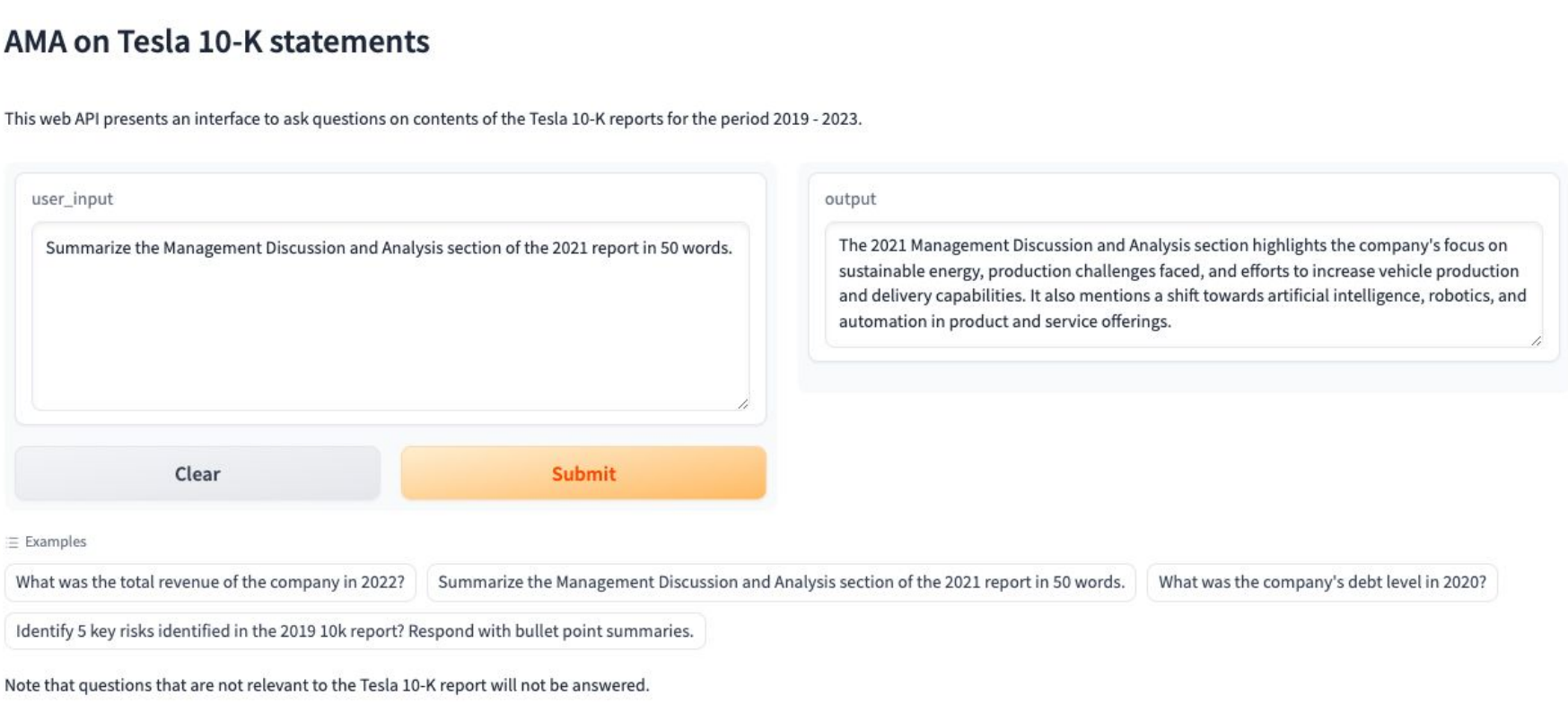
Efficiency benefits include summarizing and synthesizing large volumes of content gathered during the claims lifecycle, including call transcripts, notes, and legal and medical paperwork, which is particularly useful in property and casualty insurance. Companies can compress the claims lifecycle dramatically. Particularly in the life insurance industry, there is significant interest in using generative AI for automation and decision-making in underwriting processes and policy issuance to a broader range of customers without the need for, say, in-person medical exams.

Ernst & Young



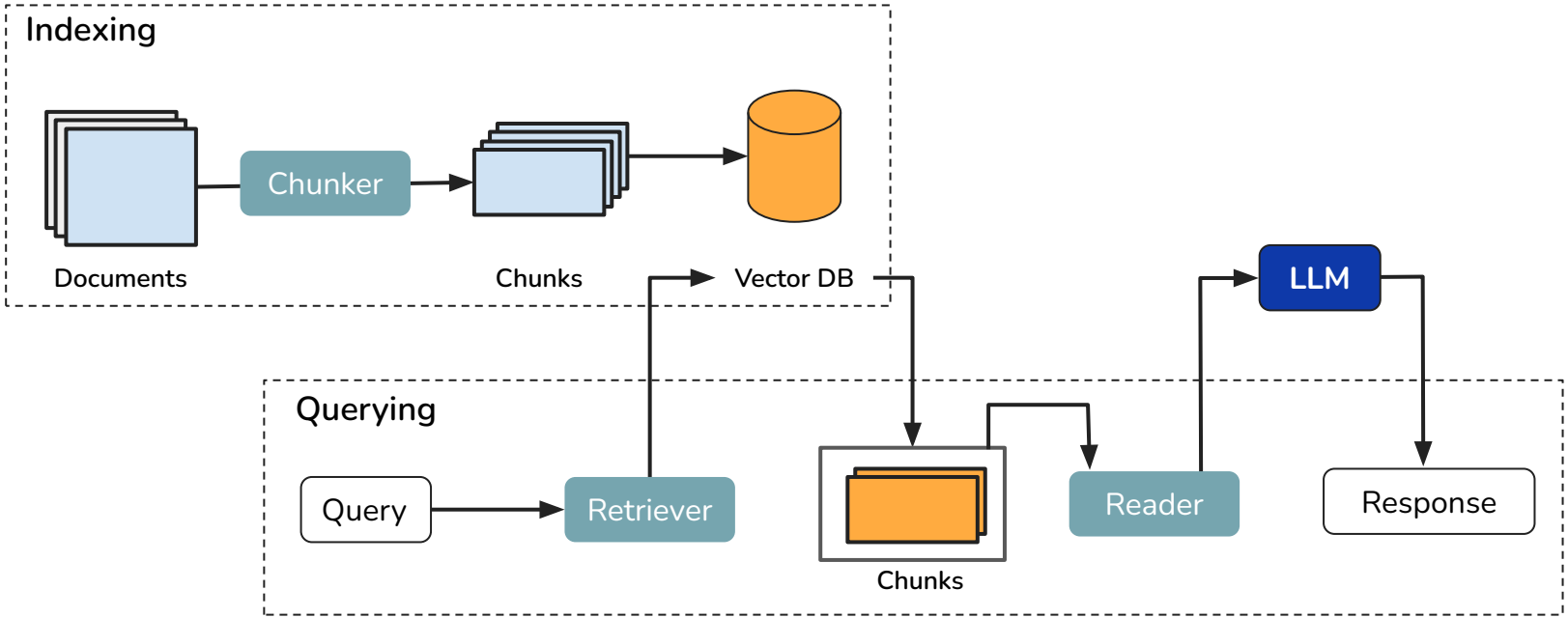
Generative AI is reducing the human effort required in synthesizing information from documents

Demo



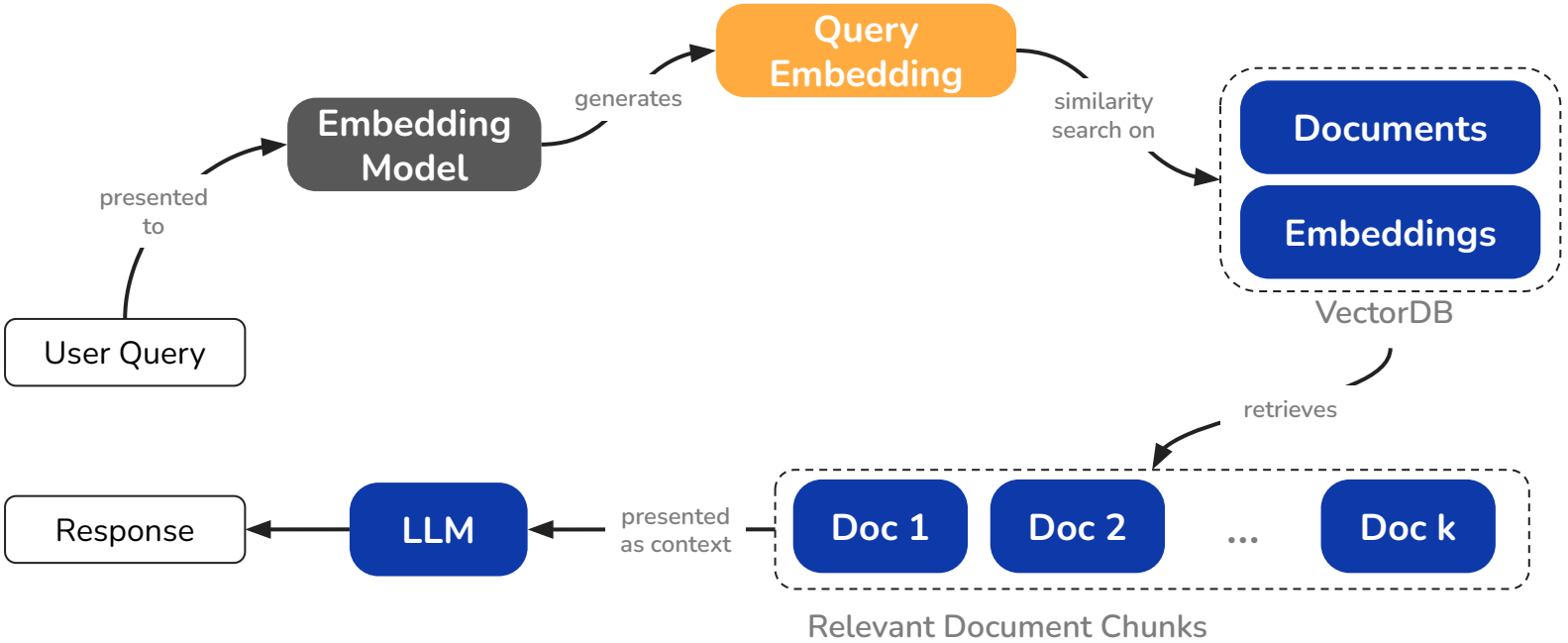
A Workflow for Retrieval-Augmented Generation

Retrieval-augmented generation (RAG) is a technique that enhances generative AI models by incorporating external data sources to improve accuracy and relevance in text generation



Implementing a RAG Workflow

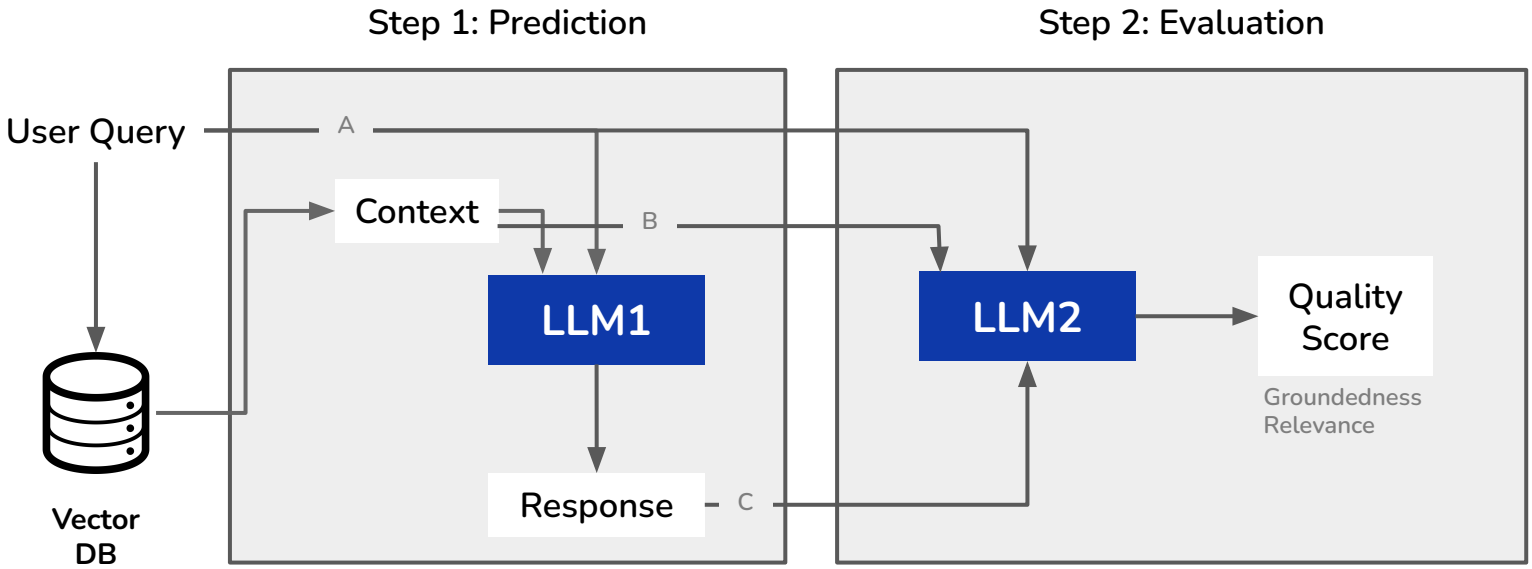
Given input queries, vector databases retrieve relevant documents using similarity search as context. The LLM uses this context to answer user queries.



[Notebook]
implementing_rag.ipynb

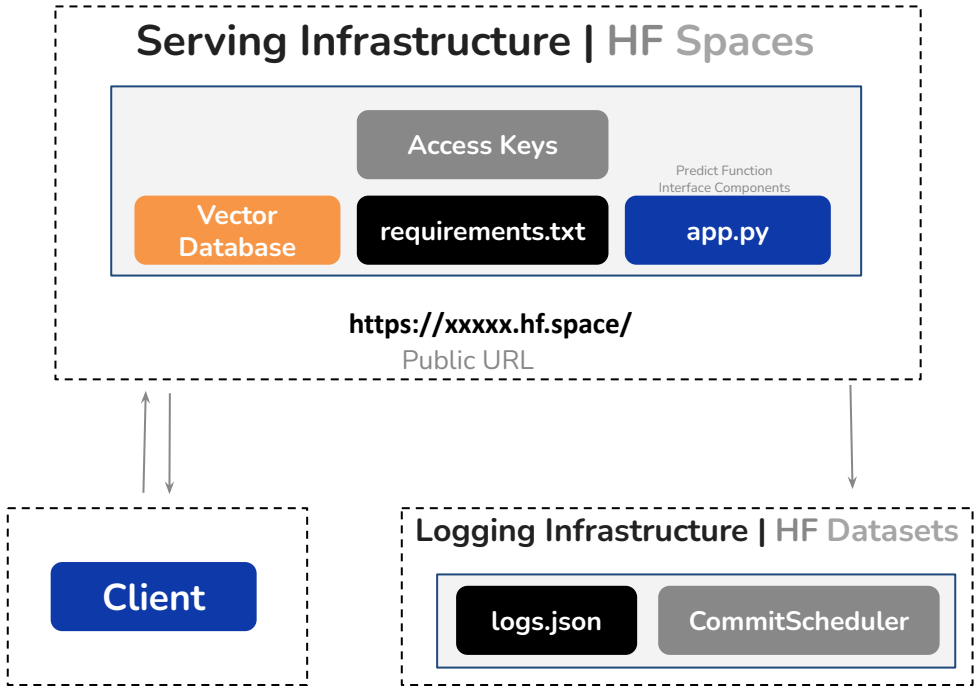
Evaluating RAG with LLM-as-a-Judge

Given input query, context and LLM response, rater LLMs judge whether: (a) The response is grounded in the context, (b) The response is relevant to the query



[Notebook]
implementing_rag.ipynb

Deploying LLMs for RAG with HuggingFace Spaces



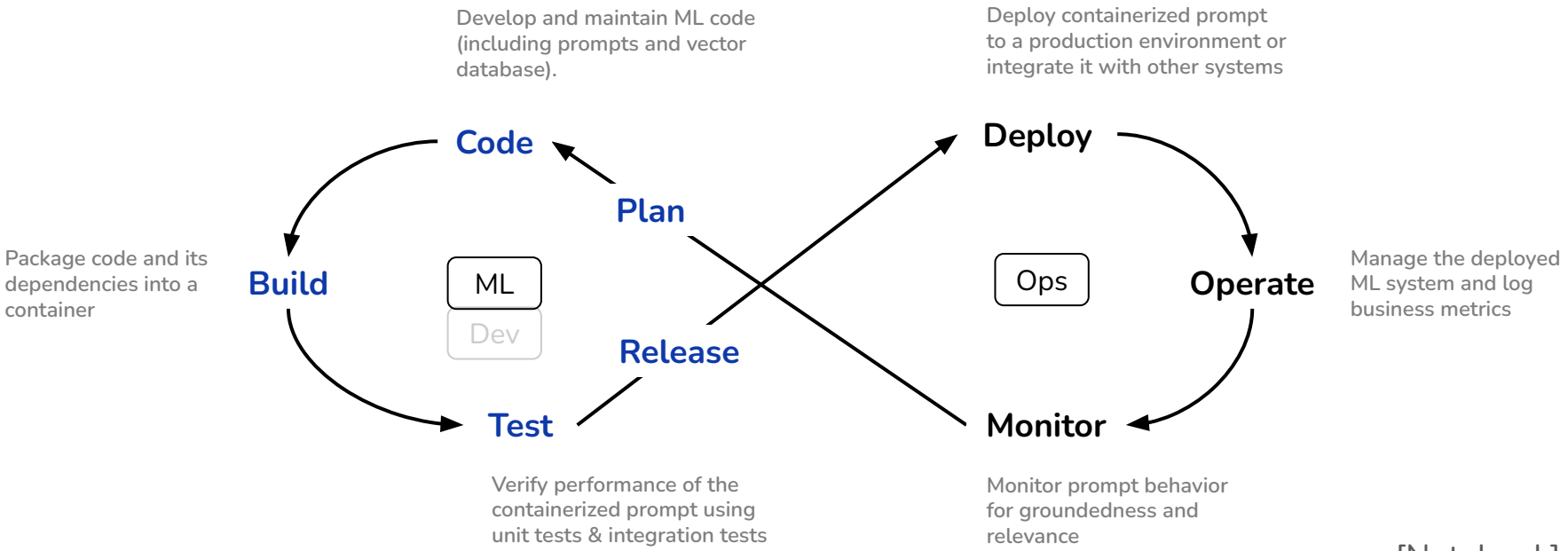
Customers and other stakeholders access the public URL or API for predictions

Logs of requests and predictions are periodically logged to a HF Dataset for monitoring

[Demo]

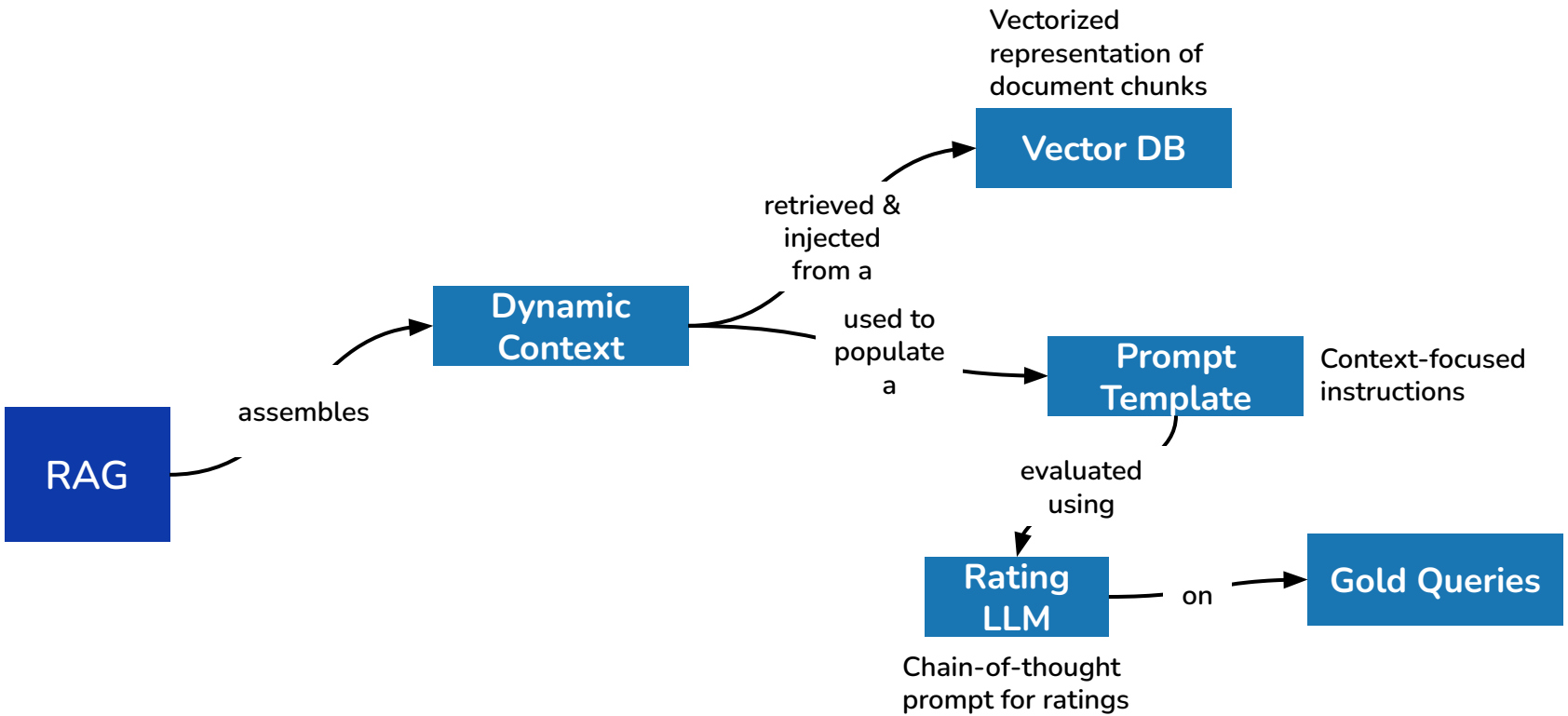
Implementing LLMOps for RAG

LLMOps is a subset of MLOps, and all steps stay the same. The implementation is simpler since we do not need to package and deploy a model, rather we package and deploy the prompt and the vector database.



[Notebook]
monitoring_rag.ipynb

Summary



Summary