databricks

Developing Your First LLM App

Mosaic AI Deep Dive

Brian Law – Sr Specialist Solutions Architect

Housekeeping

- This presentation will be recorded and we will share these materials after the session
- We will walk through codes and you can follow along later
- Use the Q&A function to ask questions
- If we do not answer your question during the event, we will follow-up with you afterwards to get you the information you need!
- Please fill out the survey at the end of the session so that we can improve our future sessions

Building gen AI applications on Databricks

Data-centric AI

Gen AI

- Custom models
- Model serving
- RAG

End-to-end AI

- MLOps (MLflow)
- AutoML
- Monitoring
- Governance

Data Science & AI
Mosaic AI

ETL & Real-time Analytics
Delta Live Tables

Orchestration
Workflows

Data Warehousing
Databricks SQL

Data Intelligence Engine
Use generative AI to understand the semantics of your data

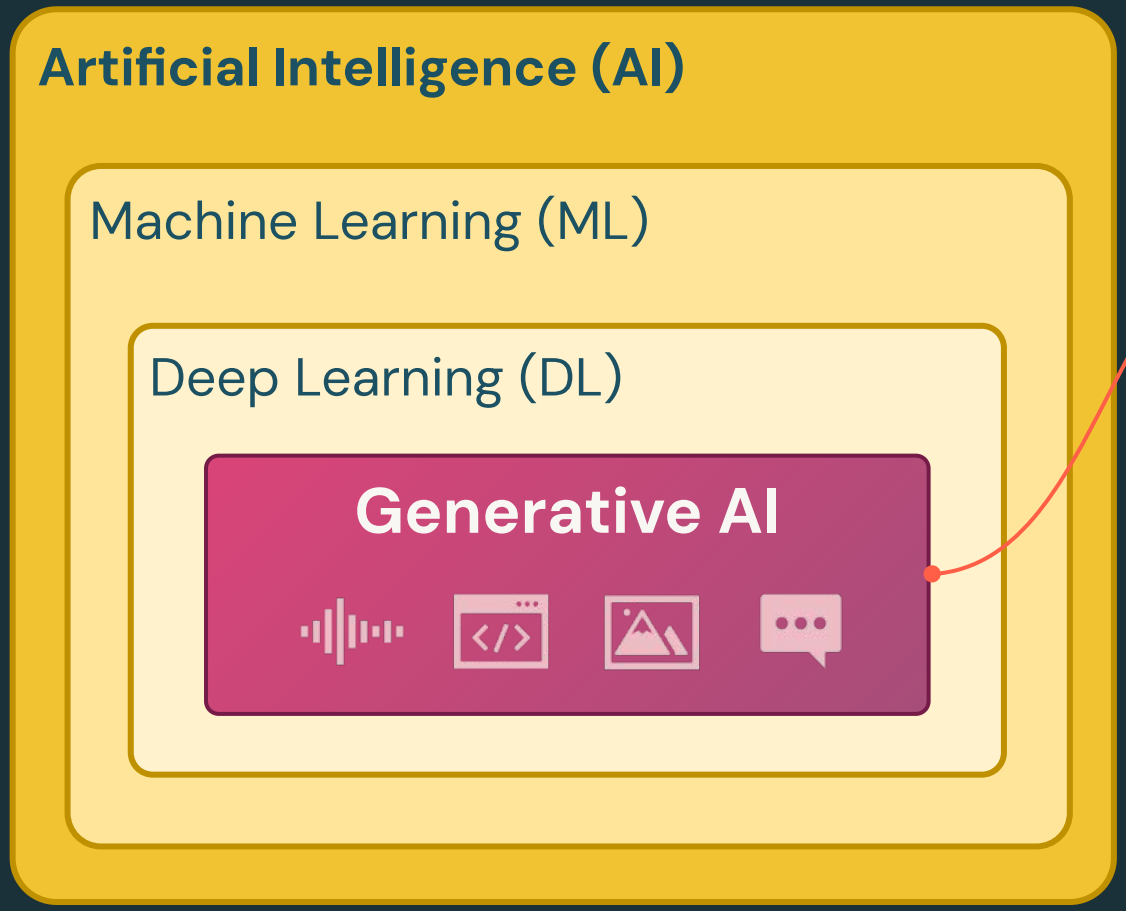
Unity Catalog
Securely get insights in natural language

Delta Lake
Data layout is automatically optimized based on usage patterns

Open Data Lake

All raw data
(Logs, texts, audio, video, images)

What is Generative AI?



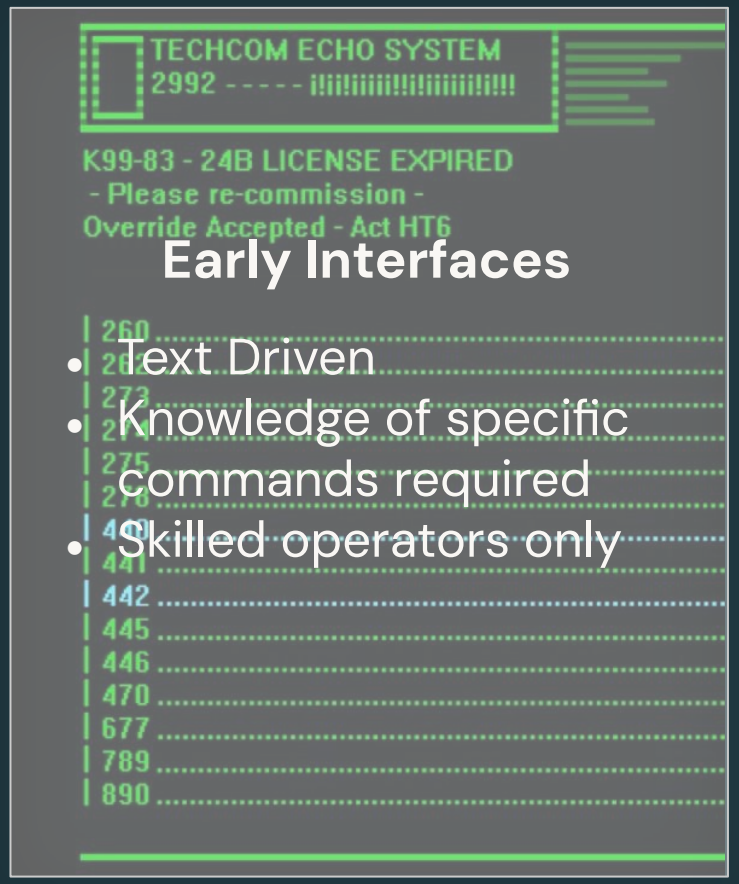
Generative Artificial Intelligence:

Sub-field of AI that focuses on **generating** new content such as:

- Images
- Text
- Audio/music
- Video
- Code
- 3D objects
- Synthetic data

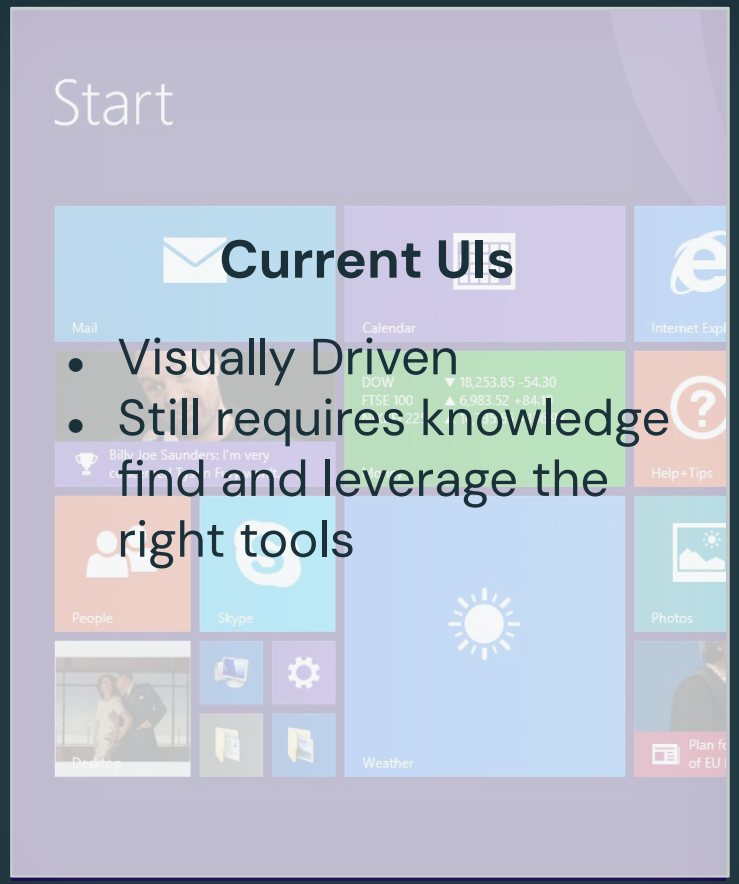
How to think about GenAI and LLMs?

A useful analogy for brainstorming usecases



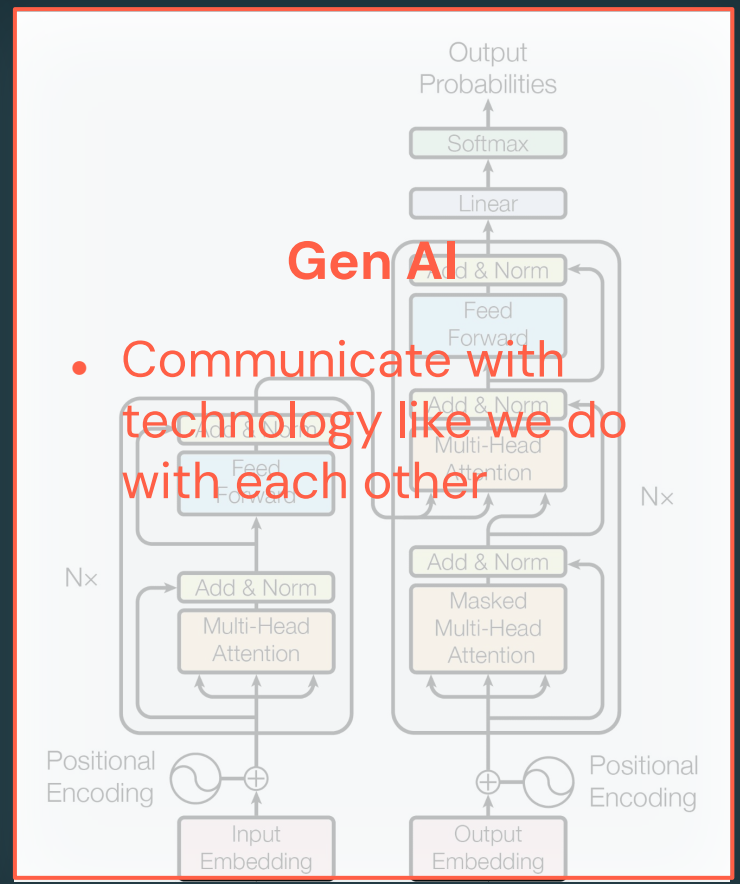
Early Interfaces

- Text Driven
- Knowledge of specific commands required
- Skilled operators only



Current UIs

- Visually Driven
- Still requires knowledge find and leverage the right tools



Gen AI

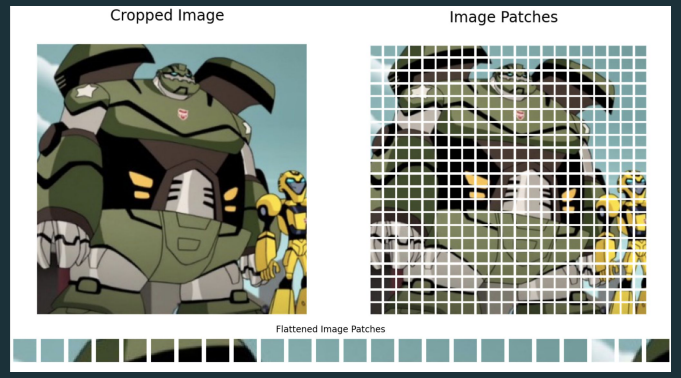
- Communicate with technology like we do with each other

How does GenAI models work

We turn the world into vectors

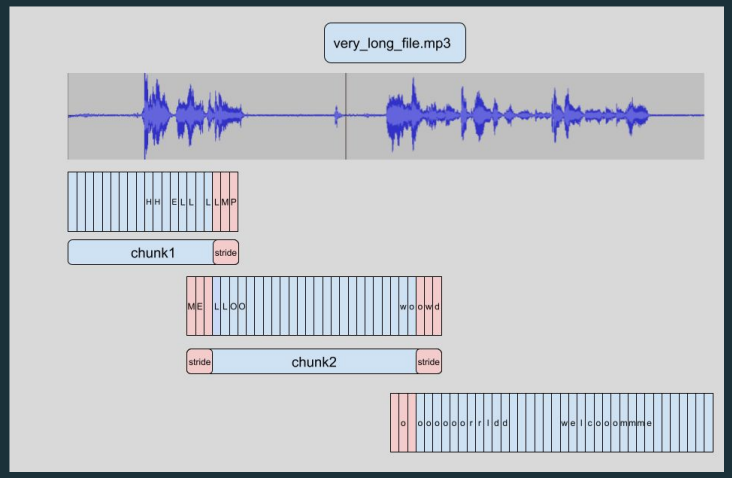
Image

Image Patches



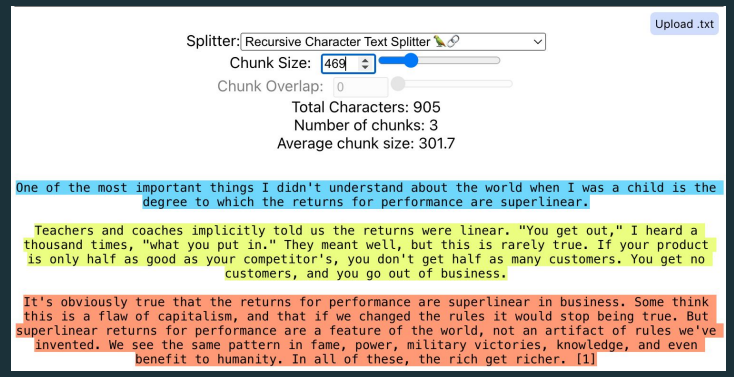
Audio

Audio Segments



Text

Text Chunks



Drill into text parsing

The moon, Earth's only natural satellite, has been a subject of fascination and wonder for thousands of years.

Token

Basic Building block

- The
- Moon
- ,
- Earth’s
- Only
- .....
- years

Sequence

Sequential list of tokens

- The moon,
- Earth’s only natural satellite
- Has been a subject of
- ....
- Thousands of years

Vocab

Complete list of tokens

{

1: "The",

569: "moon",

122: ",",

430: "Earth",

50: "**’s",

...}

Transforming language to model

From Words to Math

English Language	Tokens	Vectors
A language structured with: - an alphanet - Words - sentences - paragraphs	Mathematical encoding of parts of words:	Mathematical encoding of entire sentences and paragraphs:
Ex - Cat - Running	Ex - [40] - [10 12]	Ex: - [0 12 32 127]

Now we understand tokenisation lets look at what makes up an GenAI LLM Application

What makes up a LLM Application?

3 things you need for success

The model

The vector store

The orchestrator

What makes up an LLM Application

The Model

The Model

Key Considerations:

- Proprietary vs Open Source
- Pretraining Knowledge
- Performance vs Latency

What makes up an LLM Application

The vector store

The vector store

Key Considerations:

- Chunking Strategy
- Retrieval Strategy
- Filtering & Finetuning

What makes up an LLM Application

The orchestrator

The orchestrator

Key Considerations:

- Chain Logic
- Evaluations
- Logging and Monitoring

Lab 1: Prompting

Choosing a LLM

Types of LLMs

How to choose a model

Open-Source Models

- Use as **off-the-shelf** or **fine-tune**
- Provides flexibility for customizations
- Can be smaller in size to save cost
- **Commercial / Non-commercial use**

Open-source LLMs:

Non-commercial Use

Meta AI LLaMA

Commercial Use

MISTRAL AI_

mosaicML MPT

Proprietary Models

- Usually offered as **LLMs-as-a-service**
- Some can be **fine-tuned**
- Restrictive licenses for usage and modification

Proprietary LLMs:

ANTHROP\C

ChatGPT

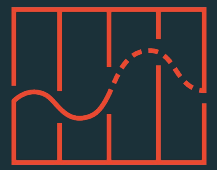
Gemini

Choose the right LLM model flavour

There is no “perfect” model, trade-offs are required.

LLM Model decision criteria

Privacy



Quality

Cost

Latency

Using Proprietary Models (LLMs-as-a-Service)

Pros

- Speed of development
  - Quick to get started and working.
  - As this is another API call, it will fit very easily into existing pipelines.
- Quality
  - Can offer state-of-the-art results

Cons

- Cost
  - Pay for each token sent/received.
- Data Privacy/Security
  - You may not know how your data is being used.
- Vendor lock-in
  - Susceptible to vendor outages, deprecated features, etc.

Using Open Source Models

Pros

- Task-tailoring
  - Select and/or fine-tune a task-specific model for your use case.
- Inference Cost
  - More tailored models often smaller, making them faster at inference time.
- Control
  - All of the data and model information stays entirely within your locus of control.

Cons

- Upfront time investments
  - Needs time to select, evaluate, and possibly tune
- Data Requirements
  - Fine-tuning or larger models require larger datasets.
- Skill Sets
  - Require in-house expertise

Adding knowledge to a model

What makes up a LLM Application?

3 things you need for success

The model

The vector store

The orchestrator

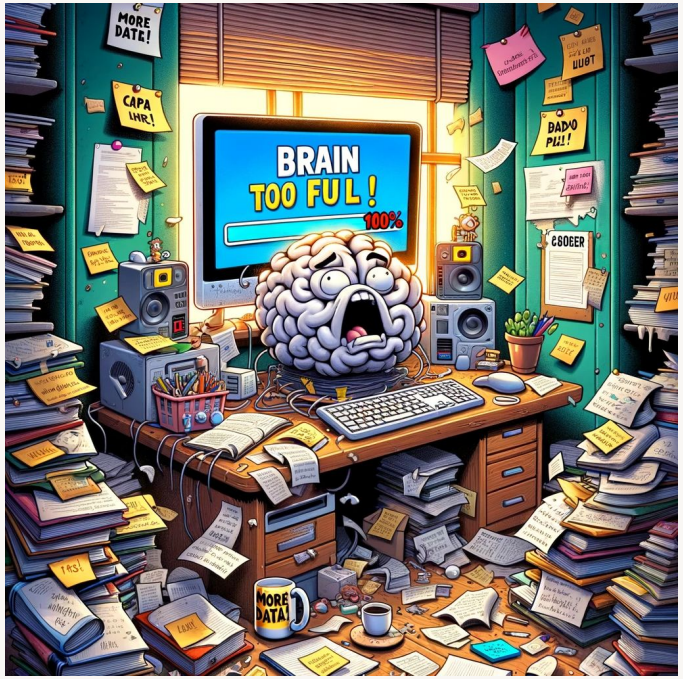
Why do we need a Vectorstore?

Pretrained models:

DO NOT Understand your business

Are too expensive for most to build

have prompt length limits



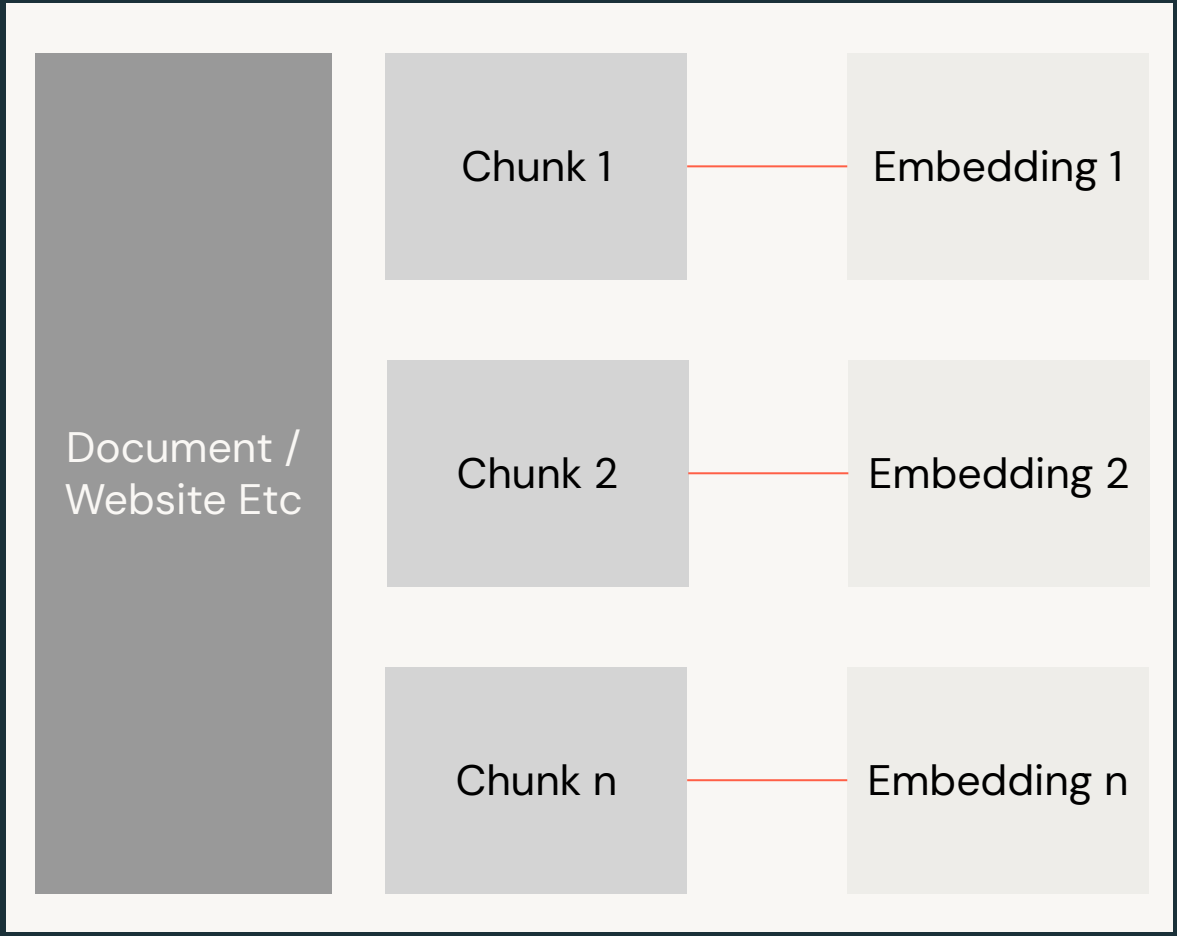
How do vectorstores work?

Key ingredients

- The source documents
- An embedding model
- A search index

Ingesting Documents

And making them searchable



We will:

- Split documents into chunks
- Embed the chunks with a model
- Add them to a search index

Lab 2: Working with VectorDBs

Putting it all together

What makes up a LLM Application?

3 things you need for success

The model

The vector store

The orchestrator

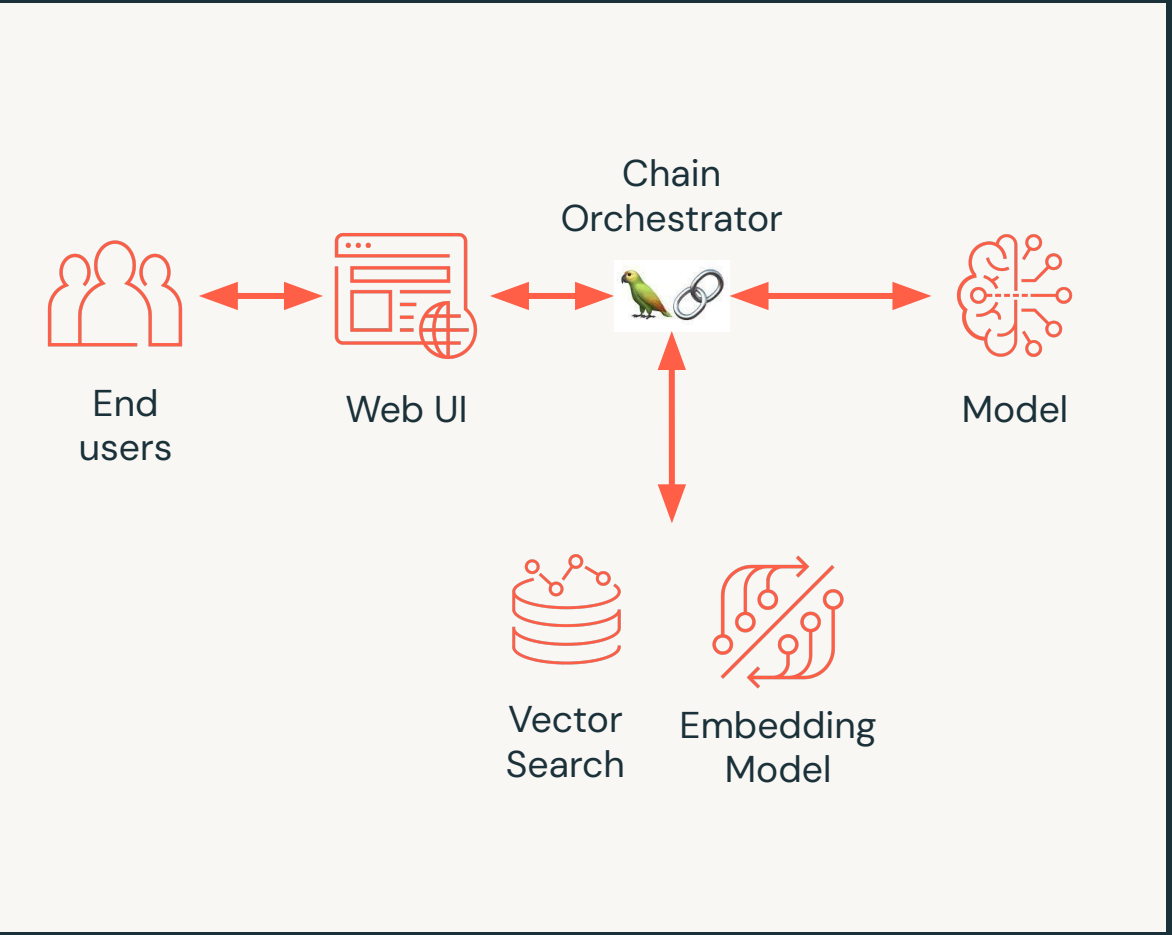
Orchestration and Logic

For a successful RAG

User Question ->

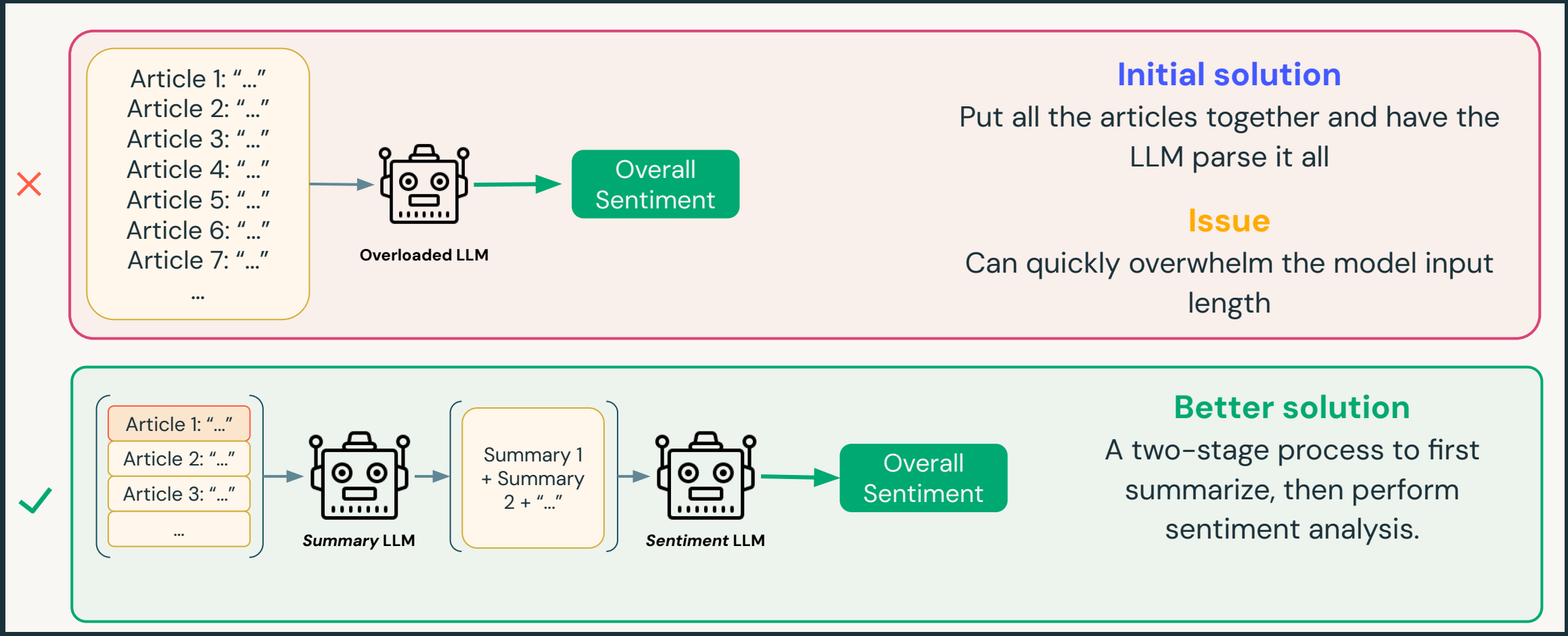
- Orchestrator
- Embedding Model
- Vector Search
- Orchestrator
- Model
- Answer

-> User



Orchestrators can get more complex

Think about the business logic



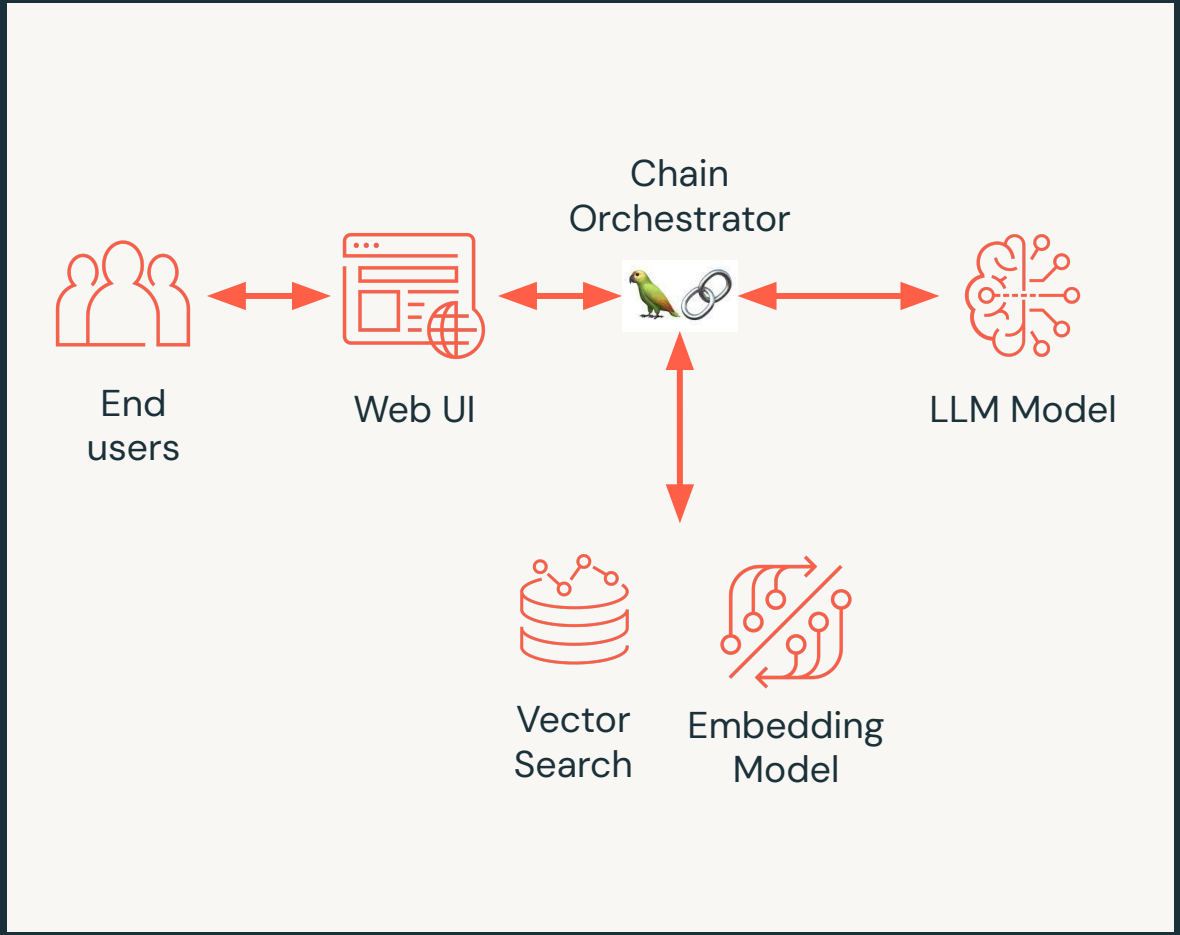
Example Advanced Logic

For better RAG

User Question ->

- Orchestrator
- Query Expansion with LLM Model
- Orchestrator
- Relevancy Check with LLM Model
- Orchestrator
- Embedding Model
- Vector Search
- Reranking with LLM Model
- Orchestrator
- Answer with LLM Model

-> User



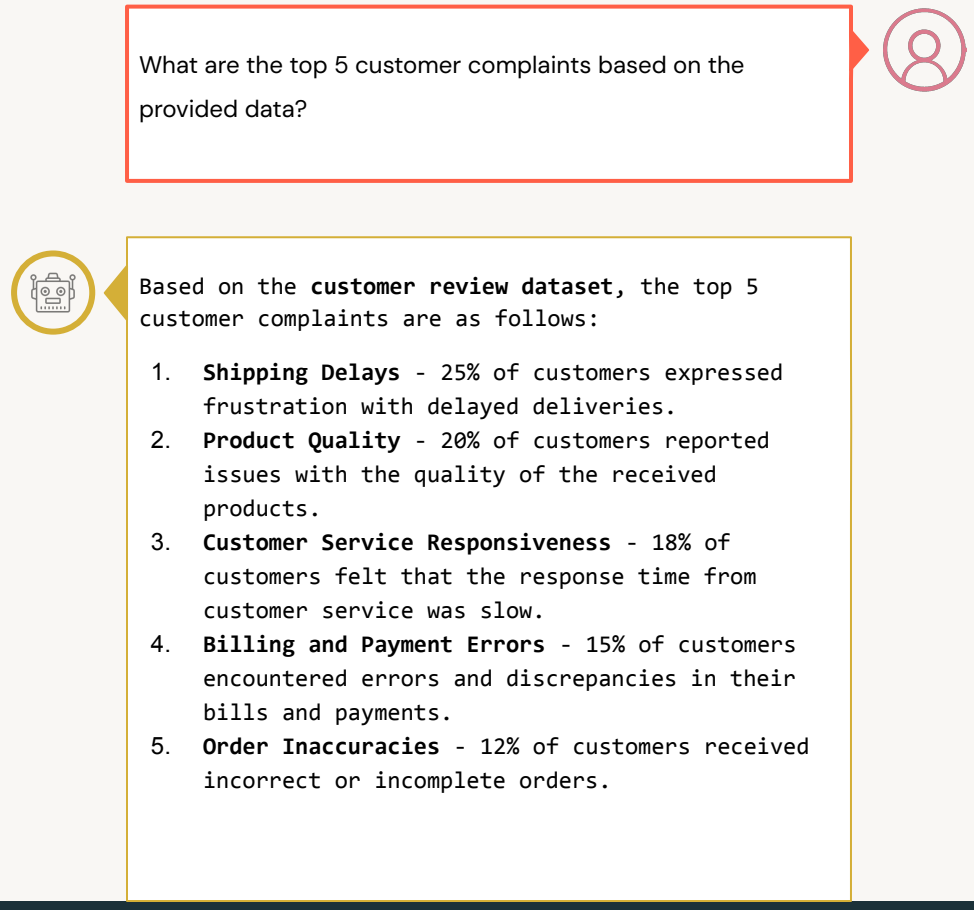
Lab 3: Assembling it all together

From Concept to Reality: Architecting your application

LLM Business Use Cases

Customer Engagement

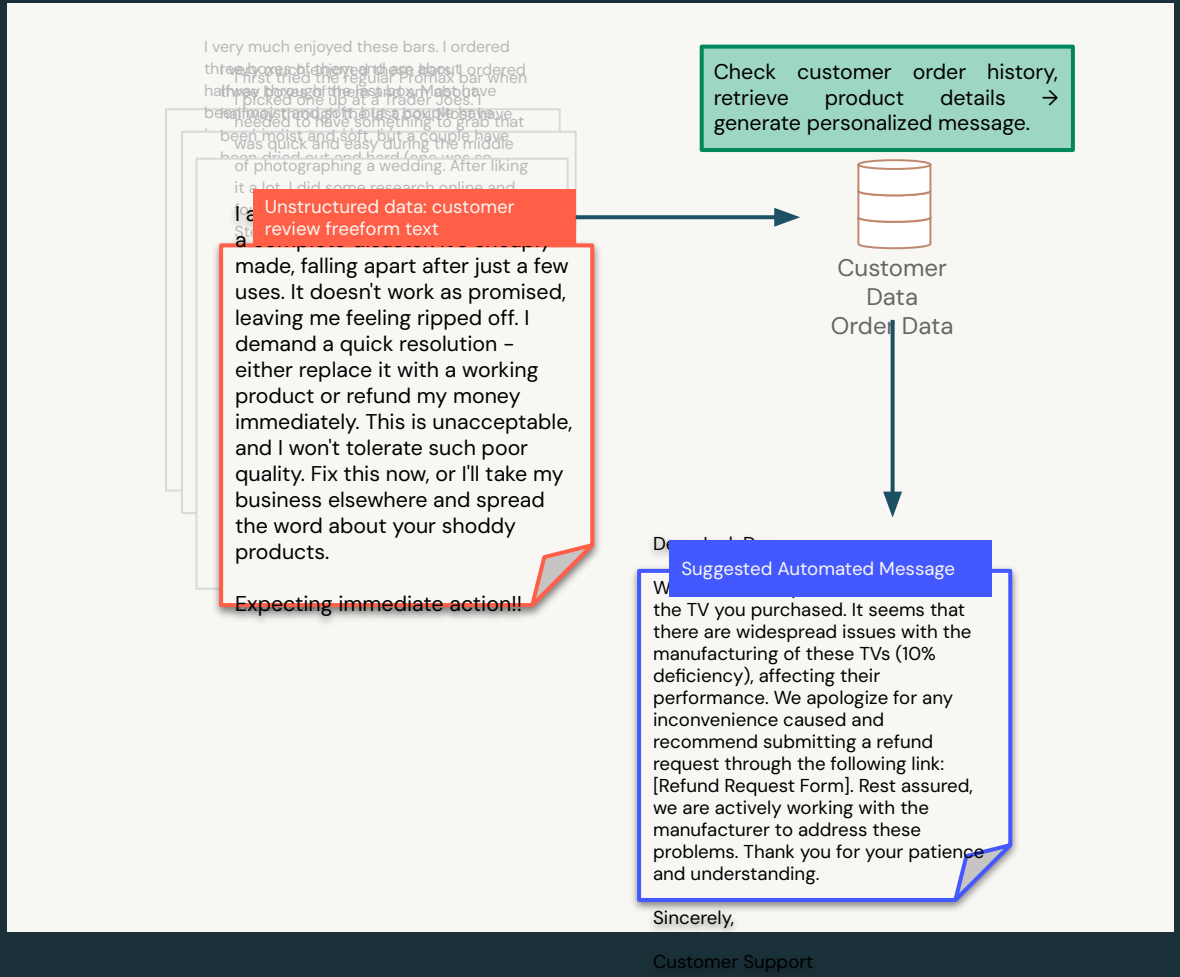
- Personalization and customer segmentation:
  - Provide personalized product/content recommendation based on customer behaviour and preferences
- Feedback Analysis
- Virtual assistants



LLM Business Use Cases

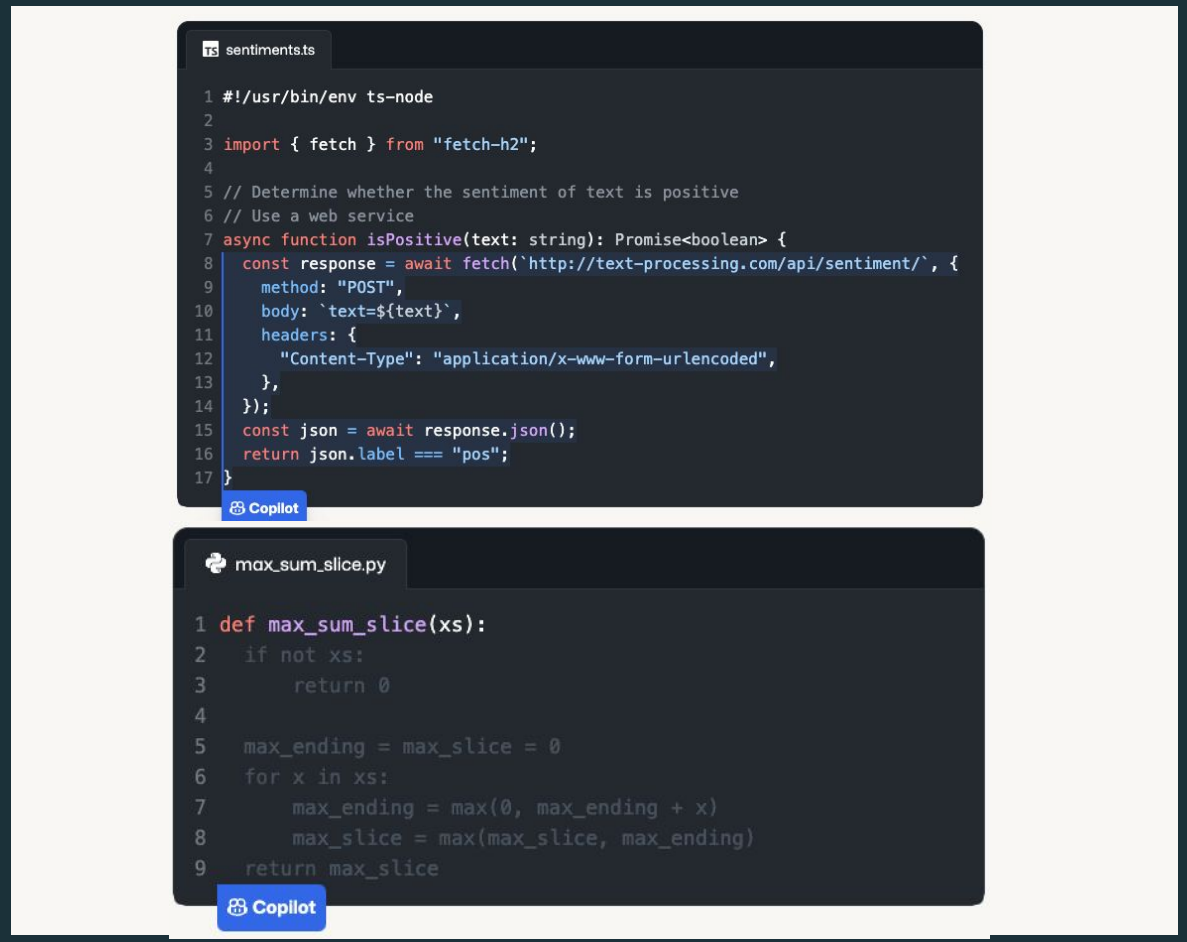
Process automation and efficiency

- Customer support augmentation and automated question answering
- Automated customer response
  - Email
  - Social media, product reviews
- Sentiment analysis, prioritization



LLM Business Use Cases

Code generation and developer productivity



- Code completion, boilerplate code generation
- Error detection and debugging
- Convert code between languages
- Write code documentation
- Automated testing
- Natural language to code generation
- Virtual code assistant for learning to code

LLMs are designed to address certain tasks

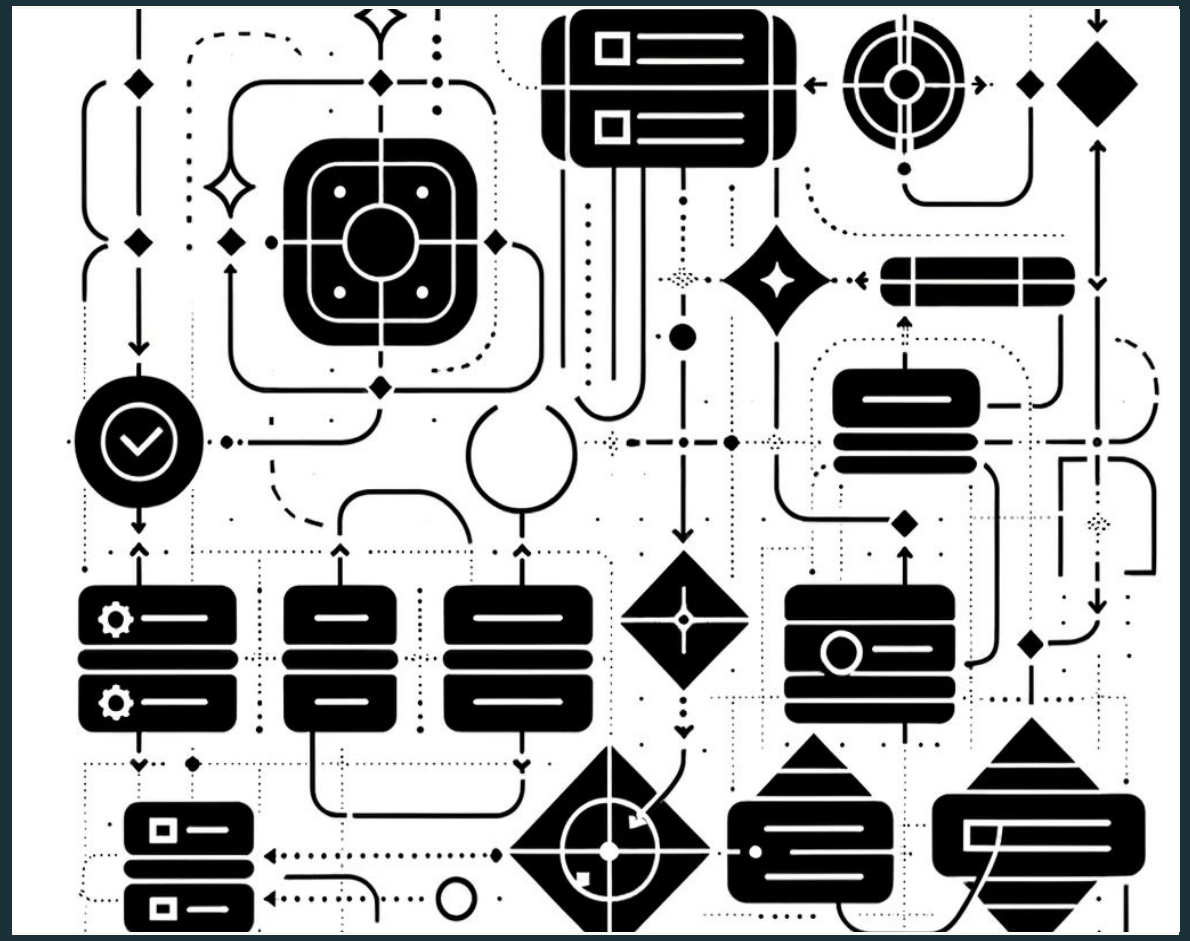
Common LLM Tasks

Content Creation and Augmentation	Generating coherent and contextually relevant text. LLMs excel at tasks like text completion, creative writing, story generation, and dialogue generation.
Summarization	Summarizing long documents or articles into concise summaries. LLMs provide an efficient way to extract key information from large volumes of text.
Question Answering	Comprehend questions and provide relevant answers by extracting information from their pre-trained knowledge.
Machine Translation	Automatically converting a text from one language to another. LLMs are also capable to explain language structure such as grammatical rules.
Classification	Categorizing text into predefined classes or topics. LLMs are useful for tasks like topic classification, spam detection, or sentiment analysis.
Named Entity Recognition (NER)	Identifying and extracting named entities like names of persons, organizations, locations, dates, and more from text.
Tone / Level of content	Adjusting the text's tone (professional, humorous, etc.) or complexity level (e.g., fourth-grade level).
Code generation	Generating code in a specified programming language or converting code from one language to another.

From business logic to prompts

Map out how to complete the task:

- Information Required
- Key Steps
  - Align to LLM Tasks
- Decision Points

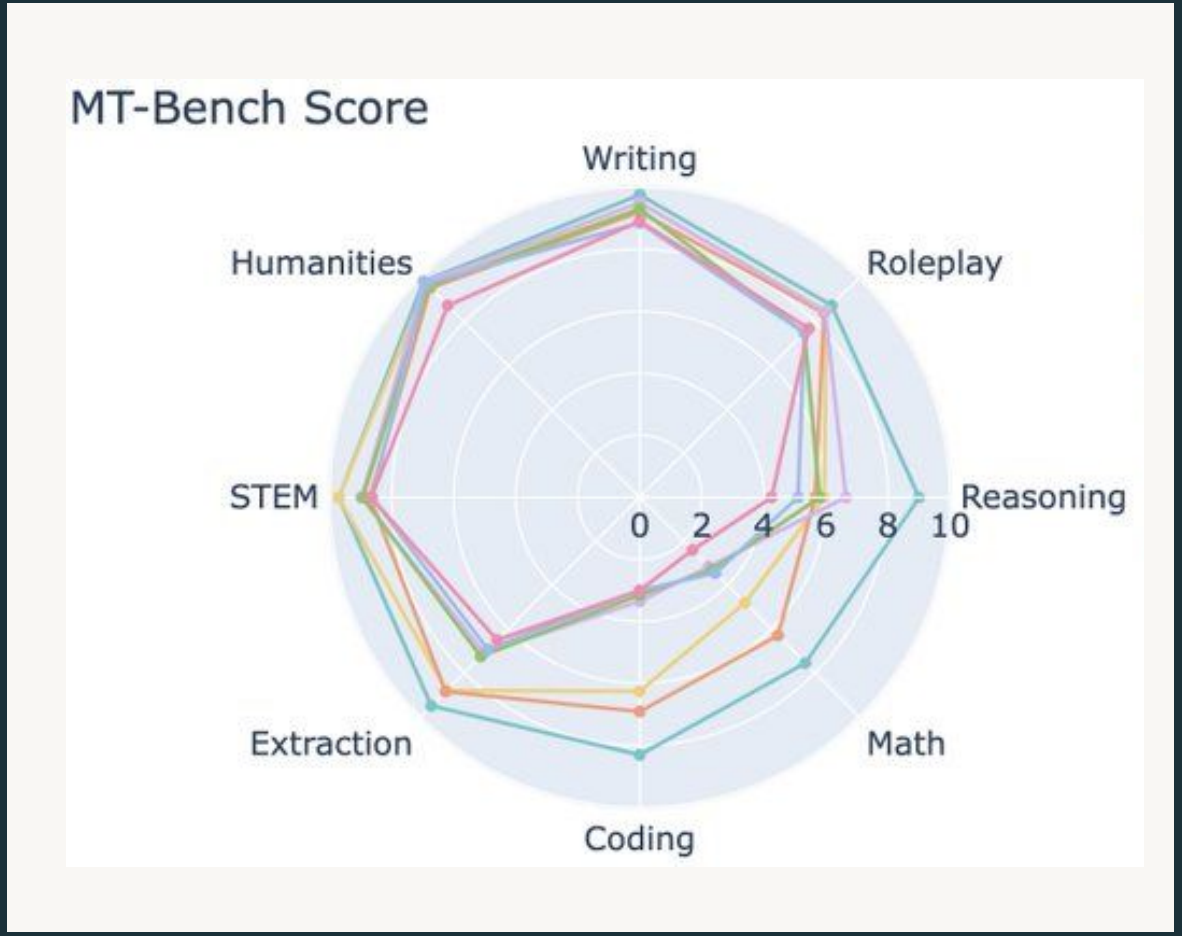


How to evaluate your app?
The Model

Good performance is subjective

Test with representative questions

- Public Benchmarks are like:
  - ENTER scores - indicative but not the most relevant
- Metrics exist like relevancy etc
  - But are experimental



Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions	17.8	18.1	22.7	28.0	23.0	29.5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA						78.6	84.5	85.0
Natural Questions	17.8	18.1	22.7	28.0	23.0	29.5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Massive Multitask Language Understanding:
https://paperswithcode.com/dataset/mmlu

University general knowledge type questions

Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions	17					5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Trivia Question and Answers:
https://nlp.cs.washington.edu/triviaqa/

Pub Trivia

Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions							31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Natural Questions
https://paperswithcode.com/dataset/natural-questions

Wikipedia questions and answers

Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions	17					5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Grade School Math Problems
https://paperswithcode.com/dataset/gsm8k

High school math questions

Common LLM metric tables

Source: https://ai.meta.com/llama/

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions	17.8	18.1	22.7	28.0	23.0	29.5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8

Does your business problem resemble one of these?

Common Scenario!

Happens at all our customers

Wow, this RAG POC was awesome, it *seems like* it can answer everything correctly!

How can we be so sure?

I tested 10 questions and it looked good to me 🤷🏽

Uhmmm… I don't think that approach will scale

MLflow Evaluate for LLMs

Checking how good your app is!

Overview:

- **Batch Compare**: Compare Models with on many questions
- **Rapid and Scalable Experimentation**: Evaluate unstructured outputs automatically
- **Cost-Effective**: Automate evaluations with LLMs

```
from mlflow.metrics.genai.metric_definitions import answer_relevance

answer_relevance_metric = answer_relevance(model="endpoints:/gpt-4")

results = mlflow.evaluate(
 model,
 eval_df,
 model_type="question-answering",
 evaluators="default",
 predictions="result",
 extra_metrics=[answer_relevance_metric, mlflow.metrics.latency()],
 evaluator_config={
  "col_mapping": {
   "inputs": "questions",
   "context": "source_documents",
  }
 }
)
print(results.metrics)


results.tables["eval_results_table"]
```

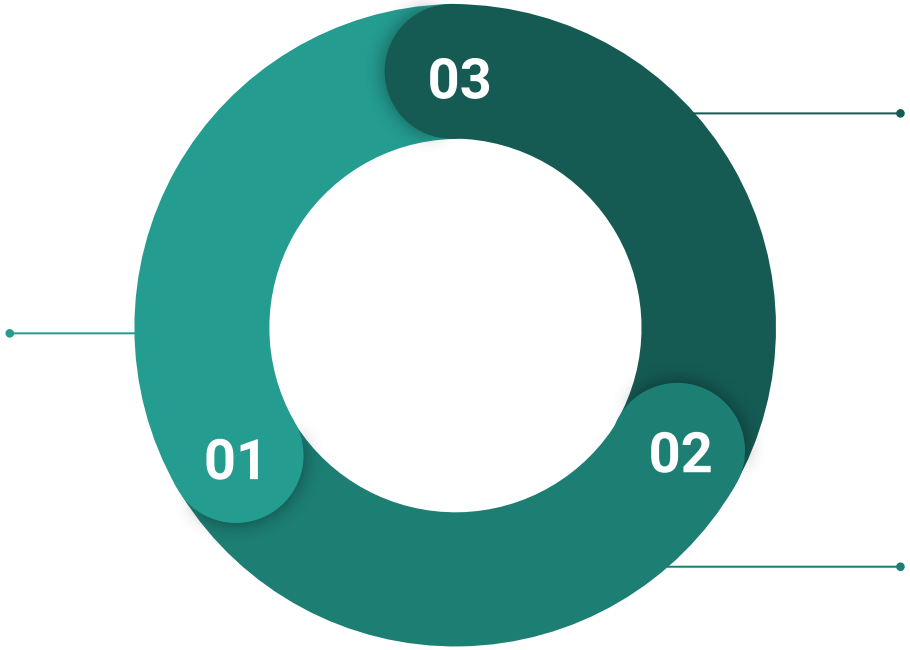
How to evaluate your app?
The Full RAG

How can we assess a RAG?

First let us look at the process

User Asks a Question

- Could be Specific
- Could be Random
- Could be Broad



Generation

- Given context we need to generate a fluid reply

Retrieval

- Given a question we need to find the right context

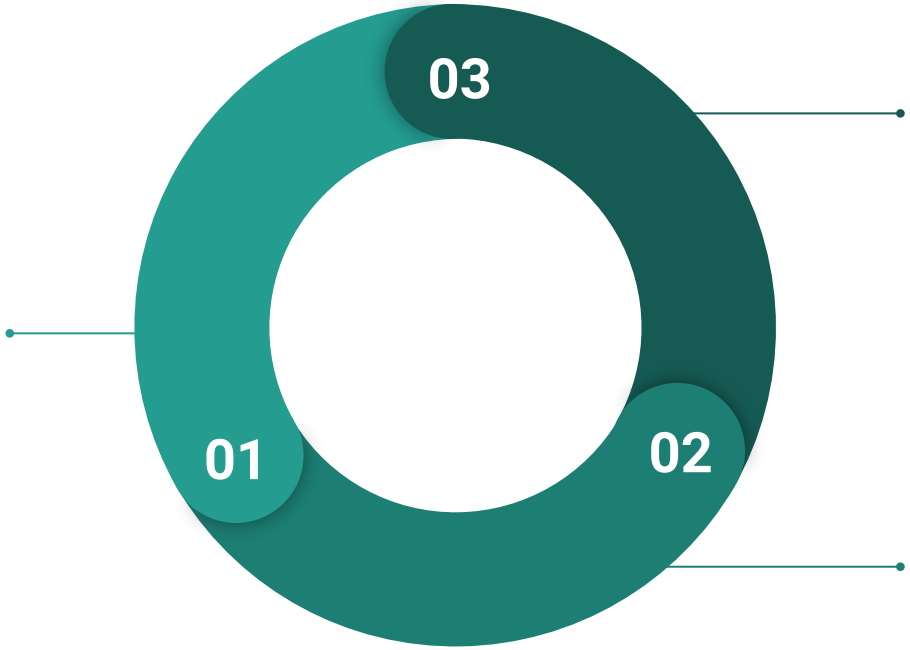
How can we assess a RAG?

First let us look at the process

User Asks a Question

- Could be Specific
- Could be Random
- Could be Broad

Questions will vary on use case!



Generation

- Given context we need to generate a fluid reply

Retrieval

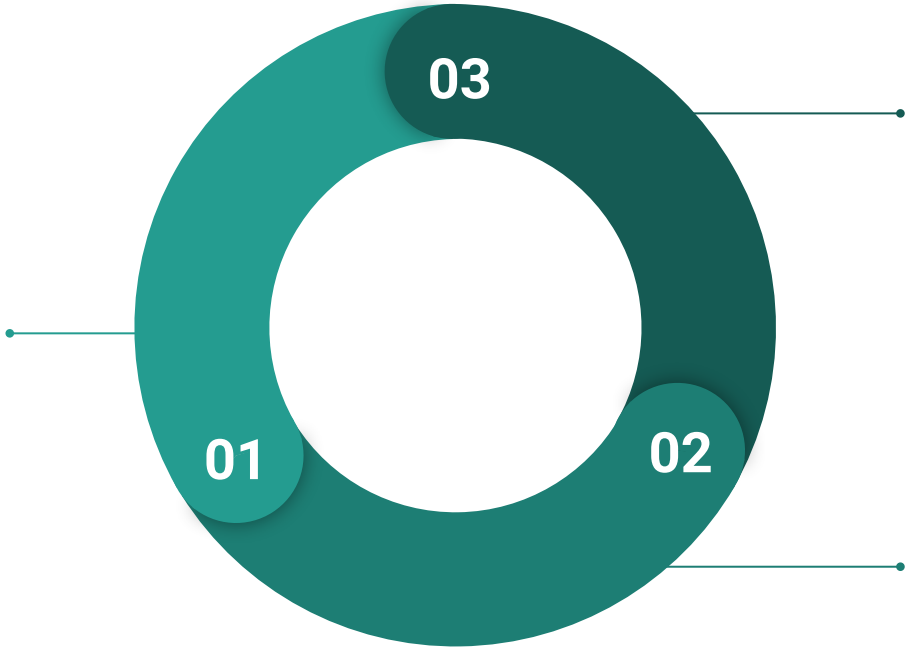
- Given a question we need to find the right context

How can we assess a RAG?

First let us look at the process

User Asks a Question

- Could be Specific
- Could be Random
- Could be Broad



Generation

- Given context we need to generate a fluid reply

Retrieval

- Given a question we
to find the right cont

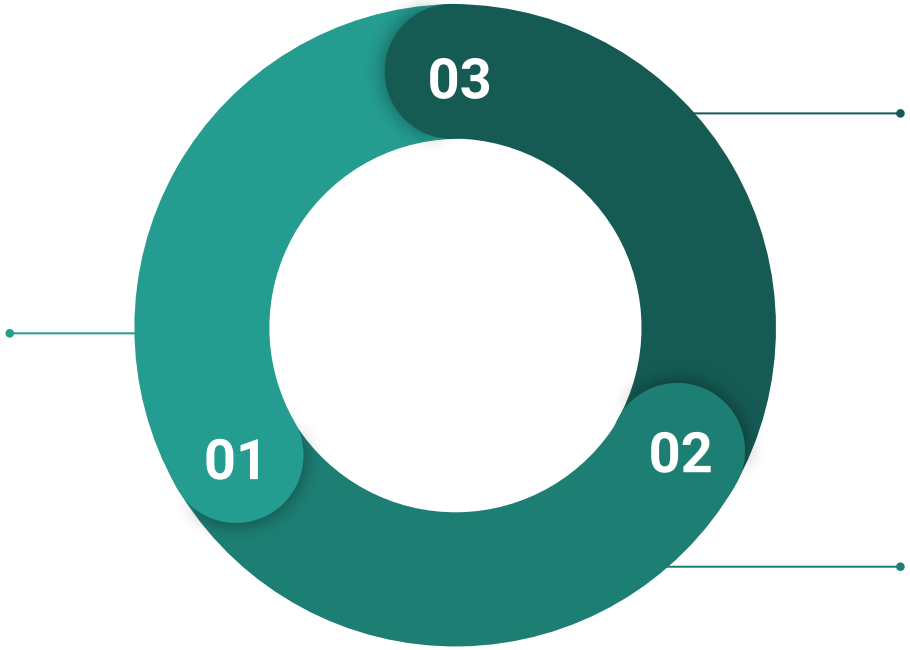
Retrieval depends on ingest

How can we assess a RAG?

First let us look at the process

User Asks a Question

- Could be Specific
- Could be Random
- Could be Broad



Generation

- Given context
generate a

Generation depends on context

Retrieval

- Given a question we need to find the right context

Best Practices

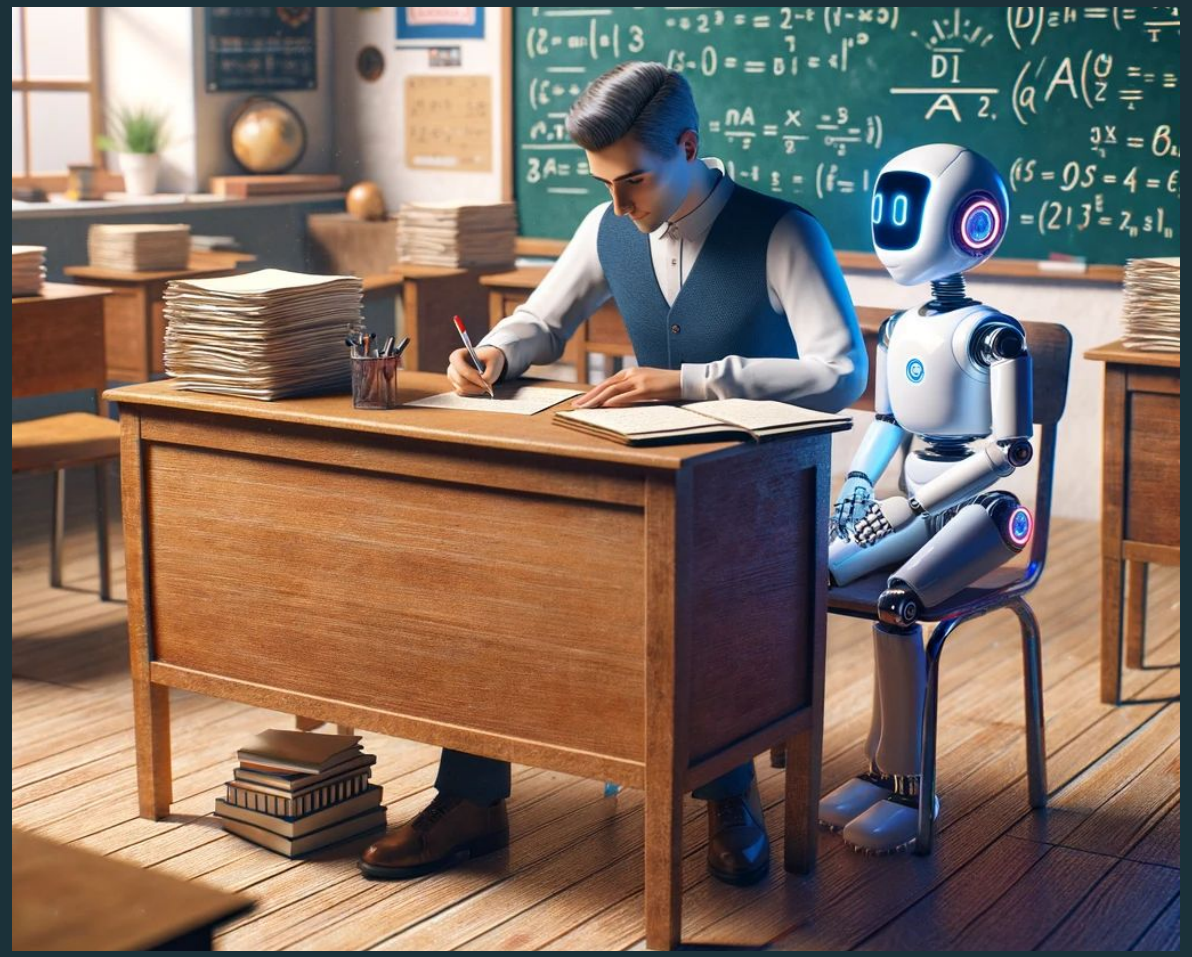
Starting your evals

Build out a set of representative questions:

- Either manually or synthetically

Include hack prompts as well

Test early and often with power users



What makes up a LLM Application?

3 things you need for success

The model

The vector store

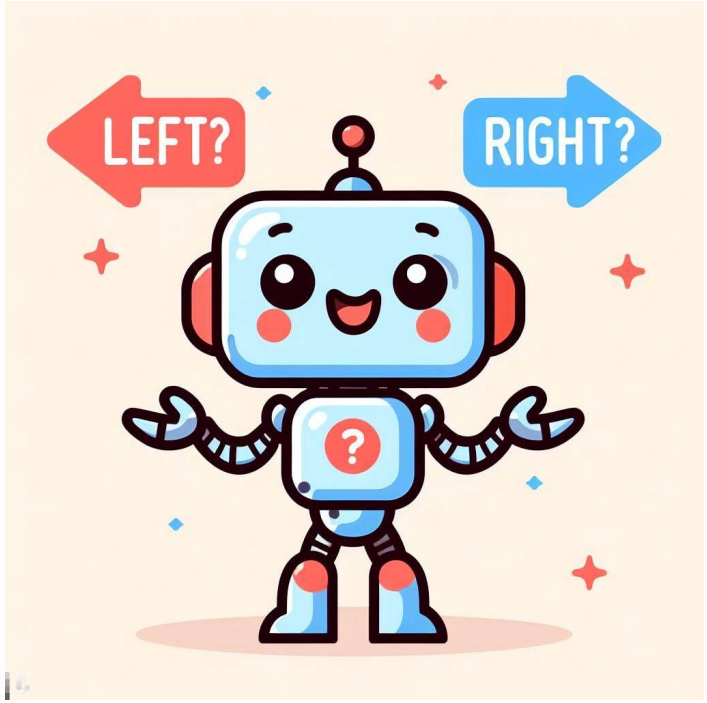
The orchestrator

But how about finetuning?

RAG vs Model

RAG Architecture

- Fast to Develop
- Increased Ops
- Hard to evaluate



Finetune Model

- Expensive & Slow to build
- Not up to date
- Hard to be precise

Common Questions

RAG vs Model

RAG Architecture

- Fast to Develop
- Increased Ops
- Hard to evaluate

Finetune Model

- Expensive & Slow to build
- Not up to date
- Hard to be precise

Thank You

Join us on 5th April for Part 2:

LLM Fundamentals
Accelerating LLM Apps to Production

9.30am IST | 12pm SGT | 3pm AEDT

https://pages.databricks.com/apj-databricks-for-practitioners-series.html

Link is also included under the “Resources” section on the right of your console