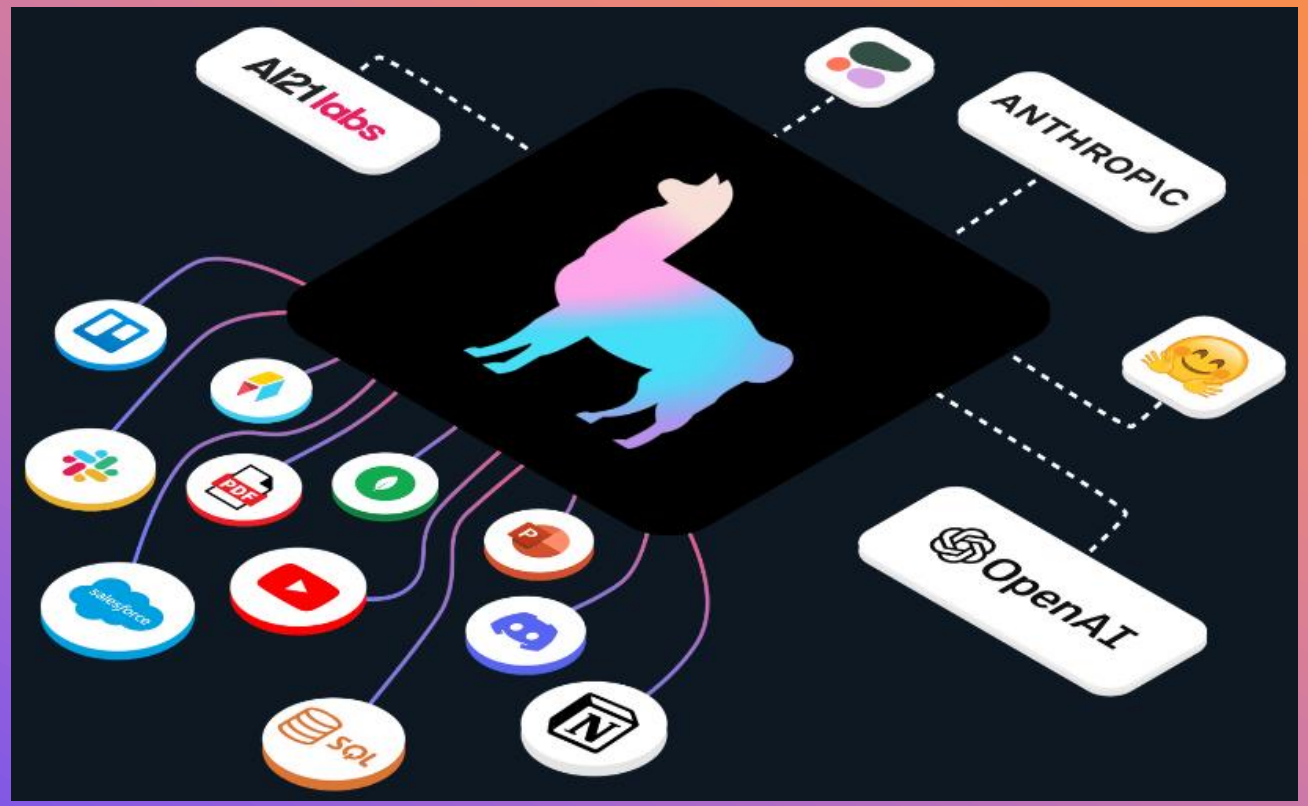
Hands-on LlamaIndex



https://github.com/wenqiglantz/hands-on-llamaindex

LlamaIndex

- **A data framework for your LLM applications**
- **Creator Jerry Liu**
- **Available in both Python and TypeScript**
- **Established on November 13th 2022**
- **Evolved rapidly since then:**
  - 160+ data loaders
  - 40+ LLMs
  - 25+ embeddings
  - 40+ vector stores
  - 35+ agent tools
  - 50+ LlamaPack templates
  - 2.8M monthly downloads, 15K community members, 700+ contributors, 5K+ applications

Hands-on LlamaIndex

AGENDA

Session One

- **RAG High-Level**
- **Query Engines**
  - SubQuestionQueryEngine
  - RouterQueryEngine
- **Data Agents**
  - ReAct Agent
  - OpenAI Agent
- **Evaluation**
  - Evaluation for LLMs
  - Evaluation for retrieval strategies

Session Two

- **Fine-tuning**
  - Fine-tune GPT-3.5
  - Fine-tune open source embedding model
- **LlamaPacks**
  - Neo4j query engine pack
  - Llama Guard moderator pack
- **Sample RAG pipeline deployment**

Retrieval Augmented Generation (RAG)

In RAG, your data is loaded and prepared for queries or “indexed”. User queries act on the index, which filters your data down to the most relevant context. This context and your query then go to the LLM along with a prompt, and the LLM provides a response.

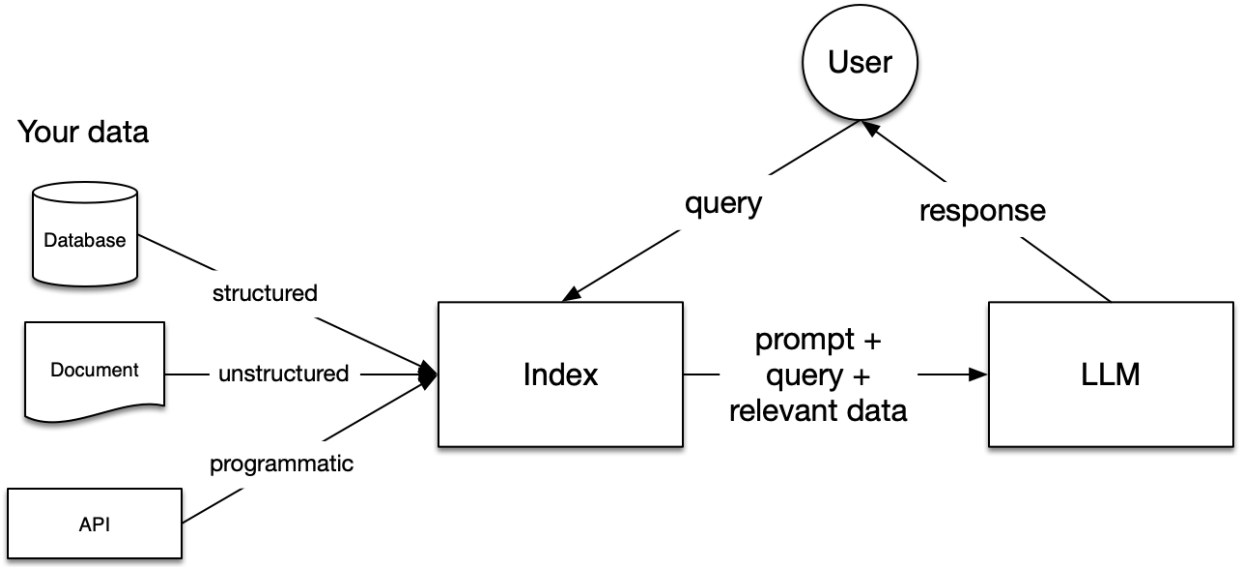
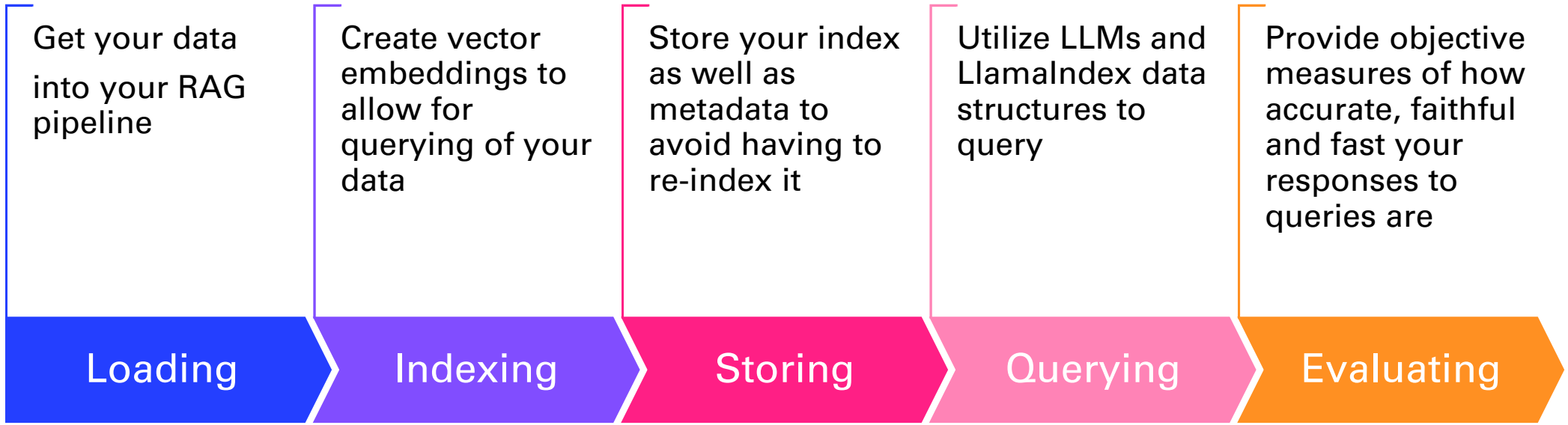


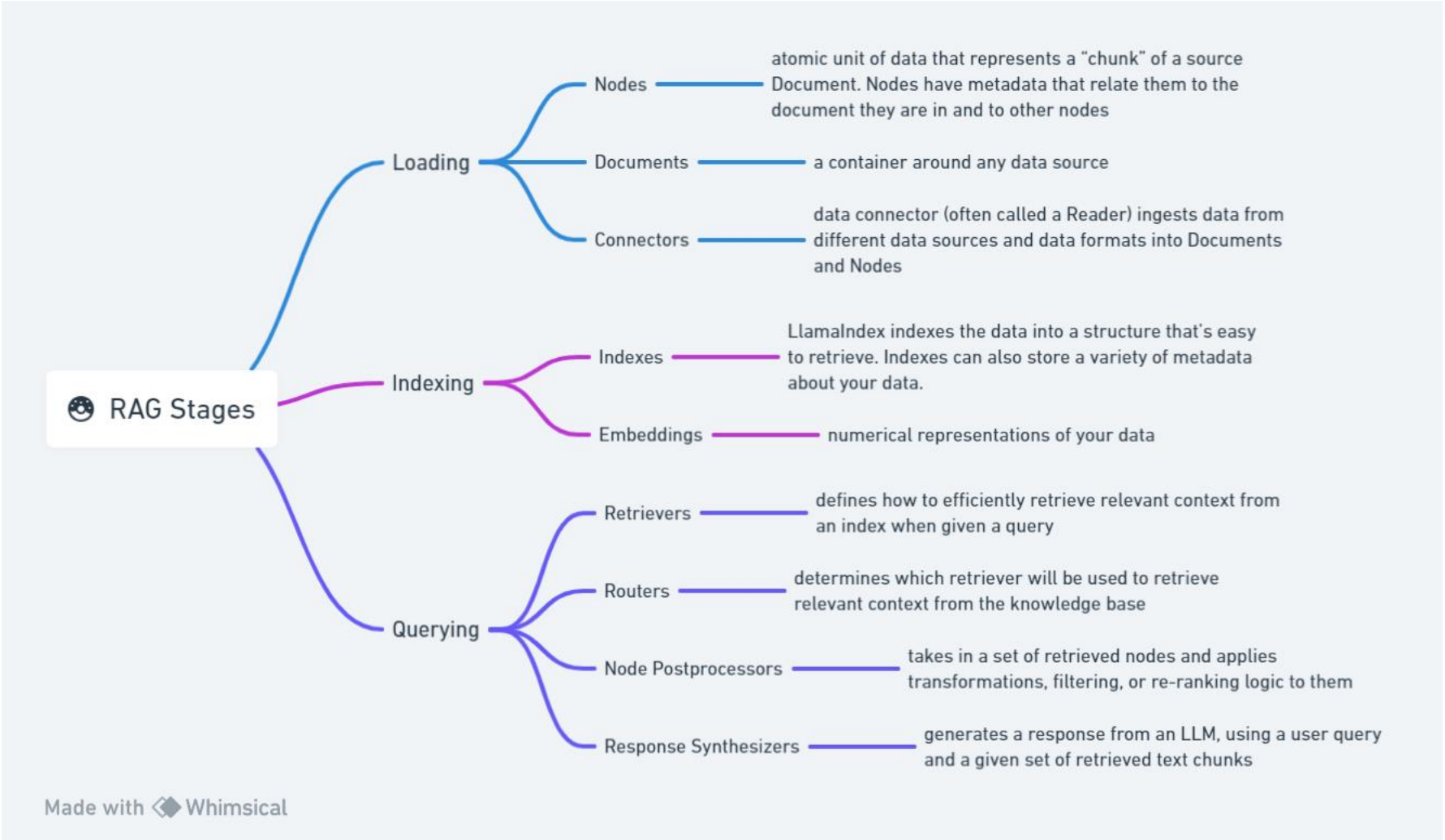
Image source: https://docs.llamaindex.ai/en/stable/getting_started/concepts.html

Stages of RAG



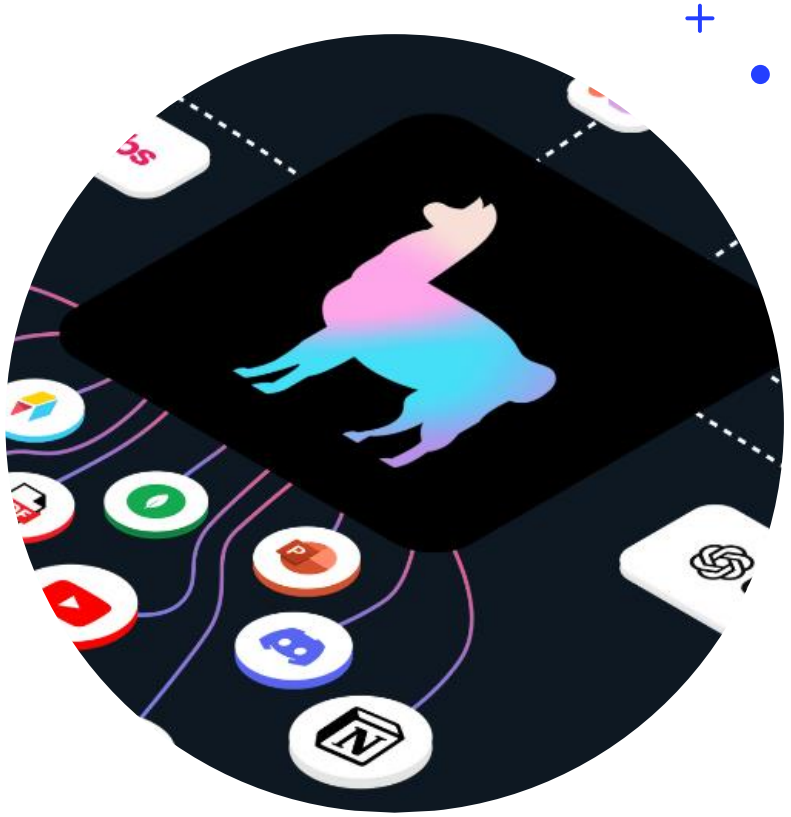
Reference: https://docs.llamaindex.ai/en/stable/getting_started/concepts.html

Important Concepts within RAG Stages



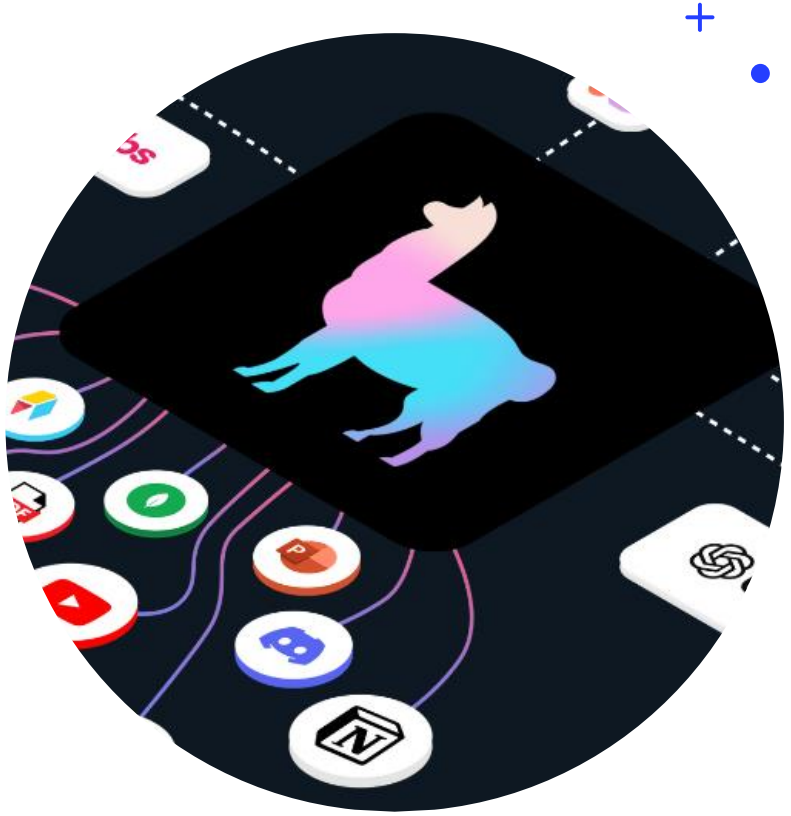
Query Engine

- Query engine is a generic interface that allows you to ask question over your data.
- A query engine takes in a natural language query, and returns a rich response. It is most often (but not always) built on one or many indexes via retrievers. You can compose multiple query engines to achieve more advanced capability.



Query Engines

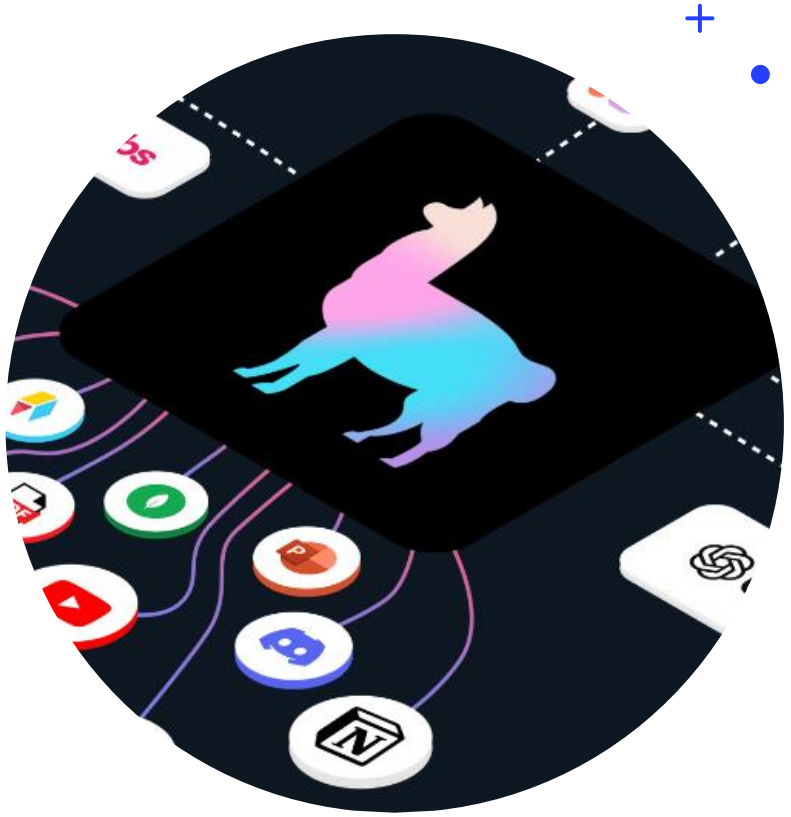
- SubQuestionQueryEngine
  - designed to tackle complex queries with more than one data source
  - queries to compare and contrast certain categories
  - generates sub-questions given descriptions of data sources
  - executes sub-questions on the selected data source
  - gathers sub-responses
  - synthesizes a final answer
- RouterQueryEngine
  - many different techniques for LLM-based queries over your private data: summarization, top-k semantic search, complex queries such as compare and contrast
  - one single interface which routes your queries to different query engines



Query Engine Notebook Walkthrough

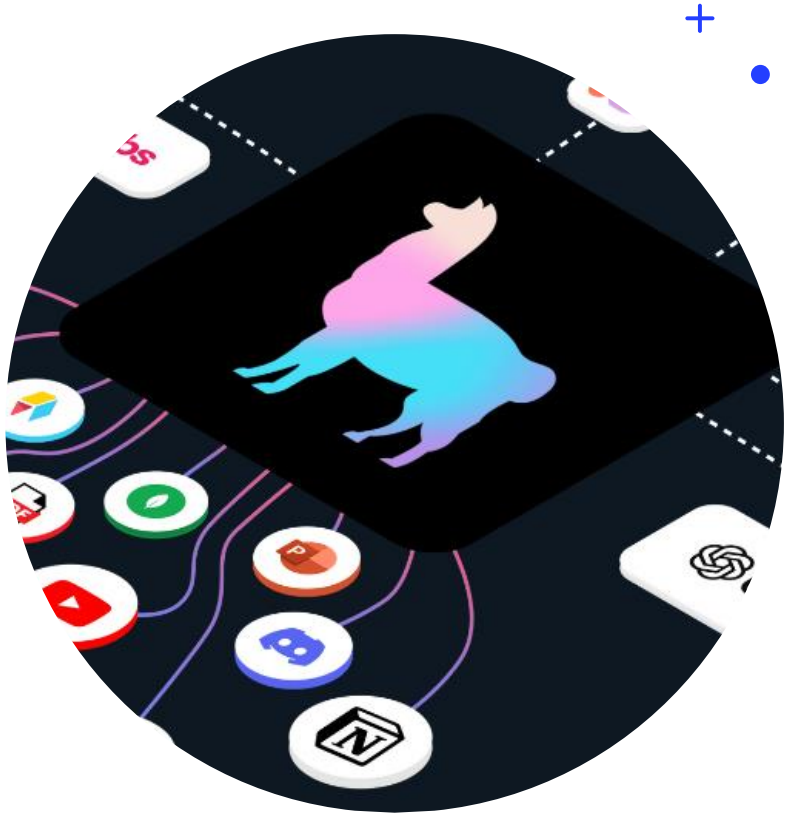
Data Agents

- LLM-powered knowledge workers that can intelligently perform various tasks over your data, in both a “read” and “write” function.
  - Perform automated search and retrieval over different types of data.
  - Call any external service API in a structured fashion, and process the response + store it for later.
- Agents are a step beyond query engines in that they can not only “read” from a static source of data, but can dynamically ingest and modify data from a variety of different tools.



Data Agents

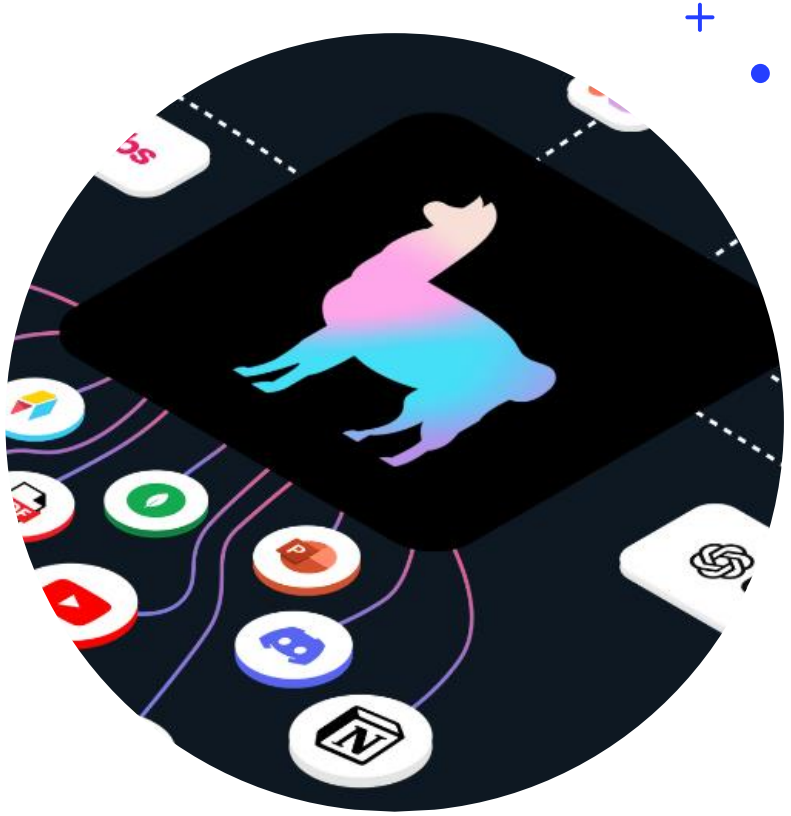
- ReAct Agent
  - Short for Reasoning and Acting
  - For each chat interaction, the agent enters a reasoning and acting loop:
  - First, decide whether to use the query engine tool and which query engine tool to use to come up with appropriate input
  - Query with the query engine tool and observe its output
  - Based on the output, decide whether to repeat the process or give a final response
- OpenAI Agent
  - An OpenAI (function calling) Agent
  - Uses the OpenAI function API to reason about whether to use a tool, and returning the response to the user



Data Agents Notebook Walkthrough

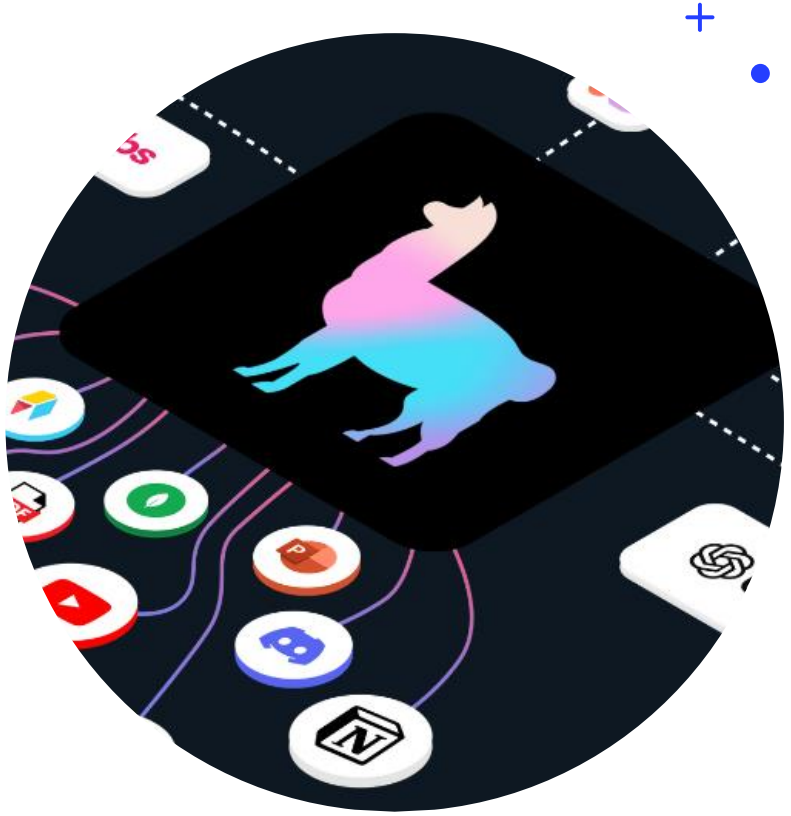
Evaluation

- Evaluation and benchmarking are crucial concepts in LLM development. To improve the performance of an LLM app (RAG, agents), you must have a way to measure it.
- LlamaIndex offers key modules to measure the quality of generated results as well as the retrieval quality.
  - **Response Evaluation**: Does the response match the retrieved context? Does it match the query? Does it match the reference answer or guidelines?
  - **Retrieval Evaluation**: Are the retrieved sources relevant to the query?



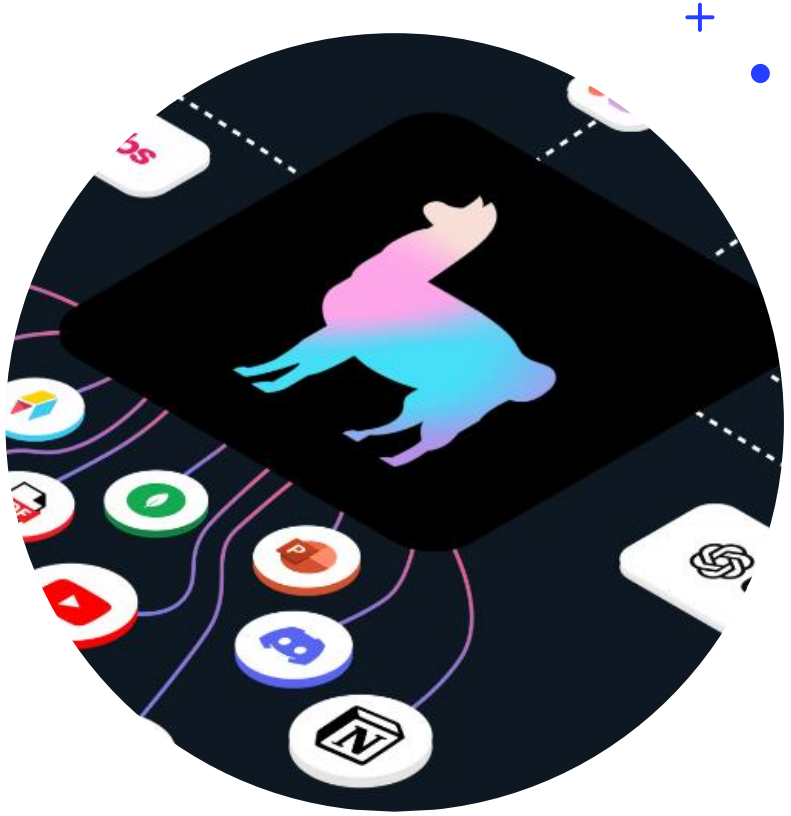
Evaluation Modules

- **Correctness**: Whether the generated answer matches that of the reference answer given the query (requires labels).
- **Semantic Similarity** Whether the predicted answer is semantically similar to the reference answer (requires labels).
- **Faithfulness**: Evaluates if the answer is faithful to the retrieved contexts (in other words, whether there's hallucination).
- **Context Relevancy**: Whether retrieved context is relevant to the query.
- **Answer Relevancy**: Whether the generated answer is relevant to the query.
- **Guideline Adherence**: Whether the predicted answer adheres to specific guidelines.



Evaluations

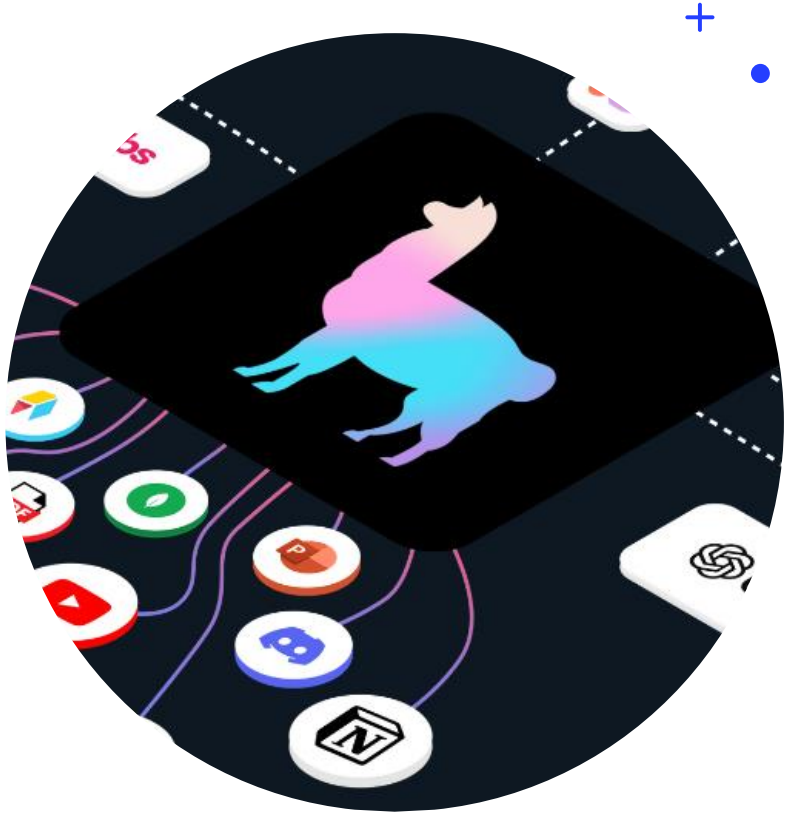
- Evaluation-Driven Development
  - Evaluation for LLMs
    - gpt-3.5-turbo
    - zephyr-7b-beta
  - Evaluation for retrieval strategies
    - Recursive retriever + document agent
    - Metadata replacement + node sentence window



EDD Notebook Walkthrough

Finetuning

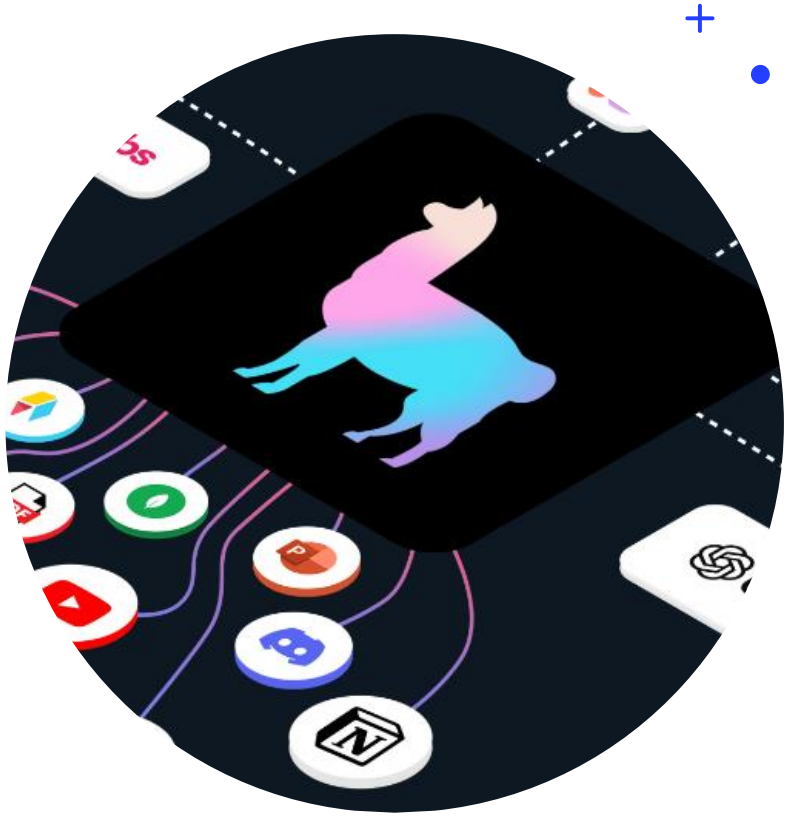
- Finetuning a model means updating the model itself over a set of data to improve the model in a variety of ways. This can include improving the quality of outputs, reducing hallucinations, memorizing more data holistically, and reducing latency/cost.
- The core of LlamaIndex's finetuning toolkit revolves around in-context learning / retrieval augmentation, which involves using the models in inference mode and not training the models themselves.
- While finetuning can be also used to "augment" a model with external data, finetuning can complement retrieval augmentation.



Finetuning

- Finetune embedding models
  - allows for more meaningful embedding representations over a training distribution of data, leads to better retrieval performance.

- Finetune LLMs
  - allows it to learn a style over a given dataset.
  - allows it to learn a DSL that might be less represented in the training data (e.g. SQL).
  - allows it to correct hallucinations/errors that might be hard to fix through prompt engineering.
  - allows it to distill a better model (e.g. GPT-4) into a simpler/cheaper model (e.g. gpt-3.5, Llama 2).



Finetuning Notebook Walkthrough

LlamaPacks

- Pre-packaged low-code templates to help jump start your LLM application development
- 50+ packs curated since Nov. 2023

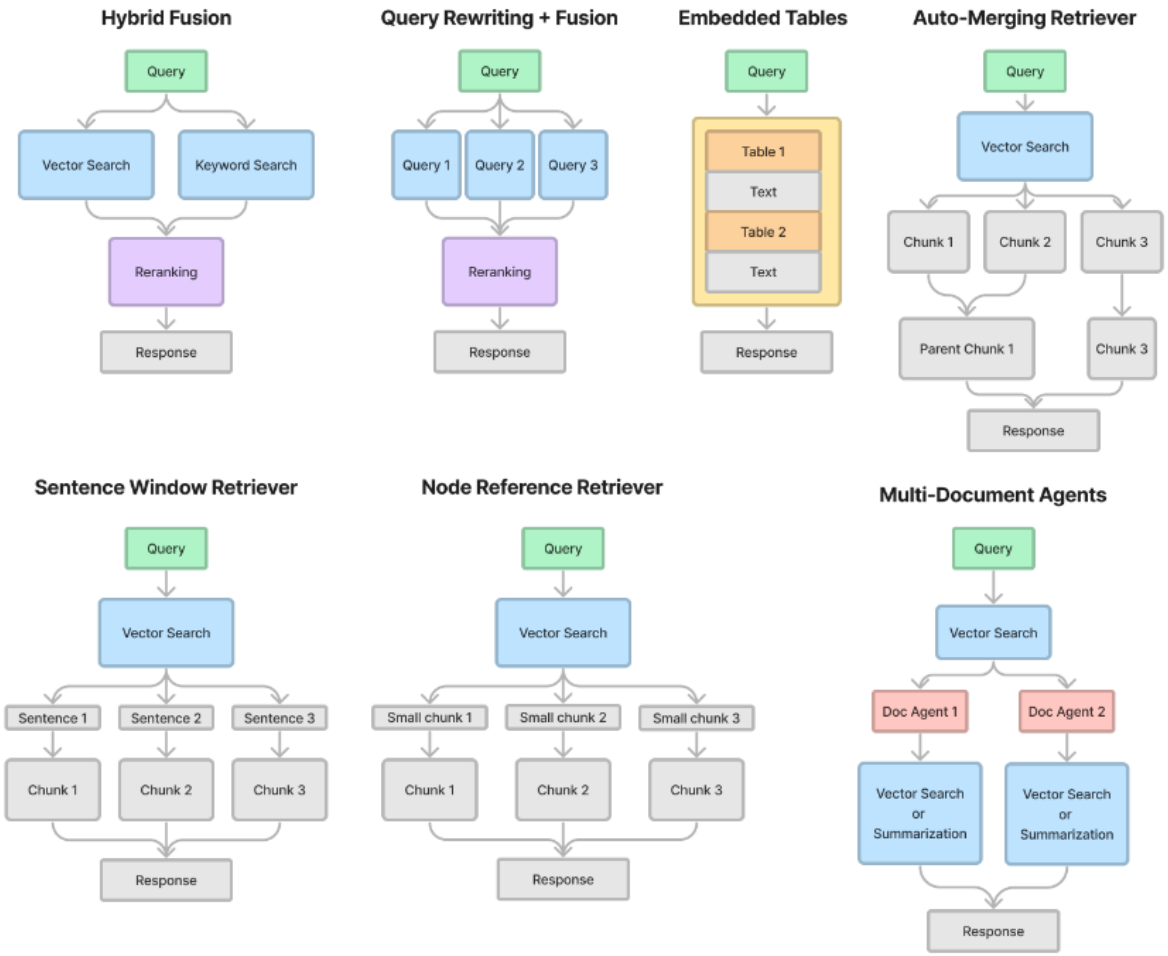
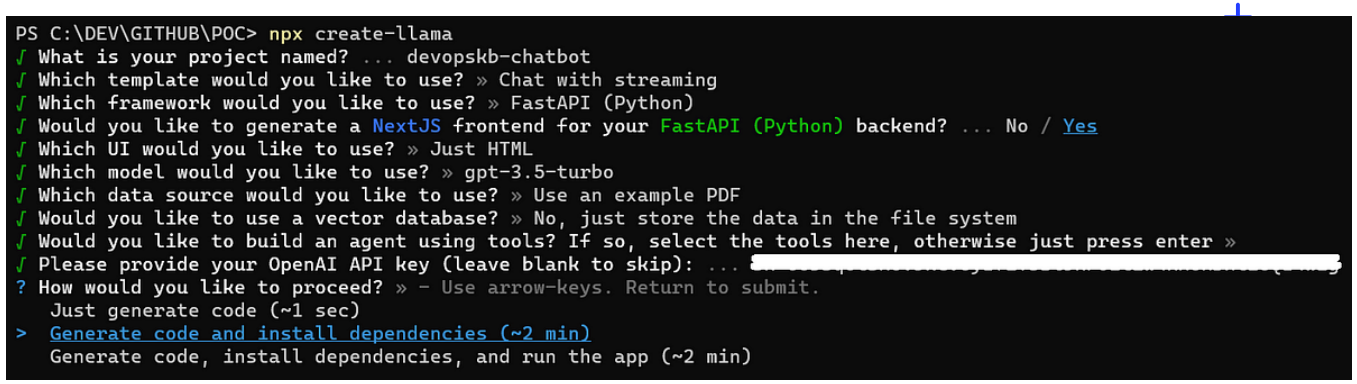


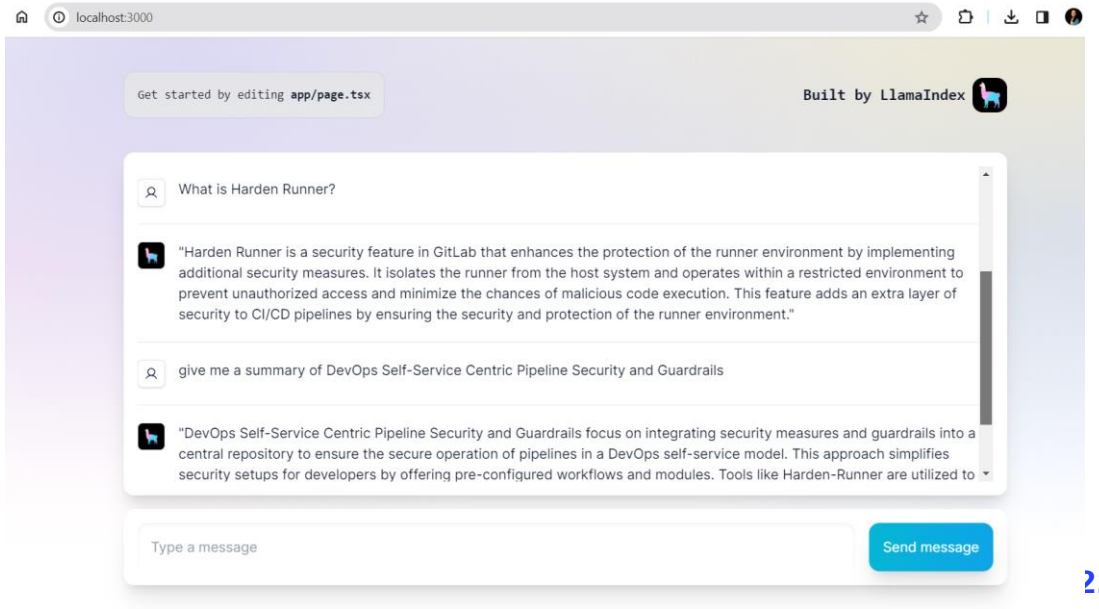
Image source: [LlamaIndex X post on seven advanced retrieval LlamaPacks]

LlamaPacks Notebook Walkthrough

From Notebook to Microservice

- create-llama, a command line tool to generate LlamaIndex apps.
- “npx create-llama”
- Convert Colab notebook to microservice
- Chatbot deployment





THANK YOU

https://github.com/wenqiglantz/hands-on-llamaindex