



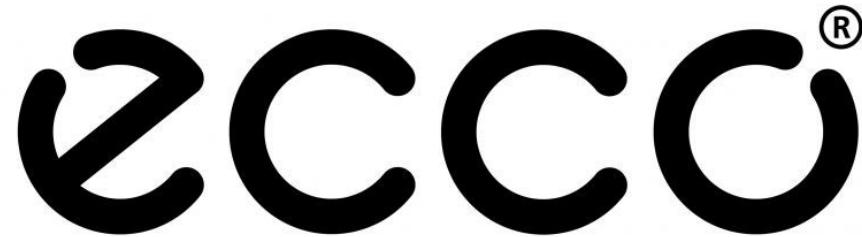
Mastering RAG : Build & Deploy Your AI Agent

Hands-on Workshop

Copenhagen, January 30th, 2024

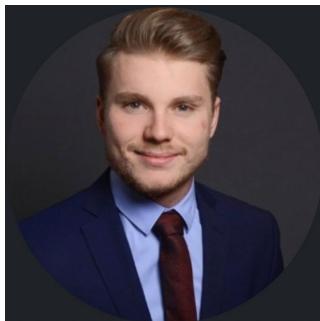
- 4.00 pm - Welcome
- 4.05 pm - Introduction to Generative AI
- 4:30 pm - Hands-on: Build Your Own RAG
- 6.00 pm - Networking, food and drinks
- 7.00 pm - We go home proudly with a working

OUR SPONSOR



➤ Welcome!

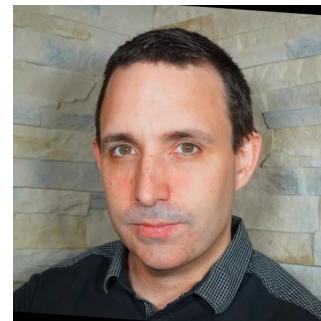
Kai Kühnel
Sales



Sami Kaksonen
Solution Engineer

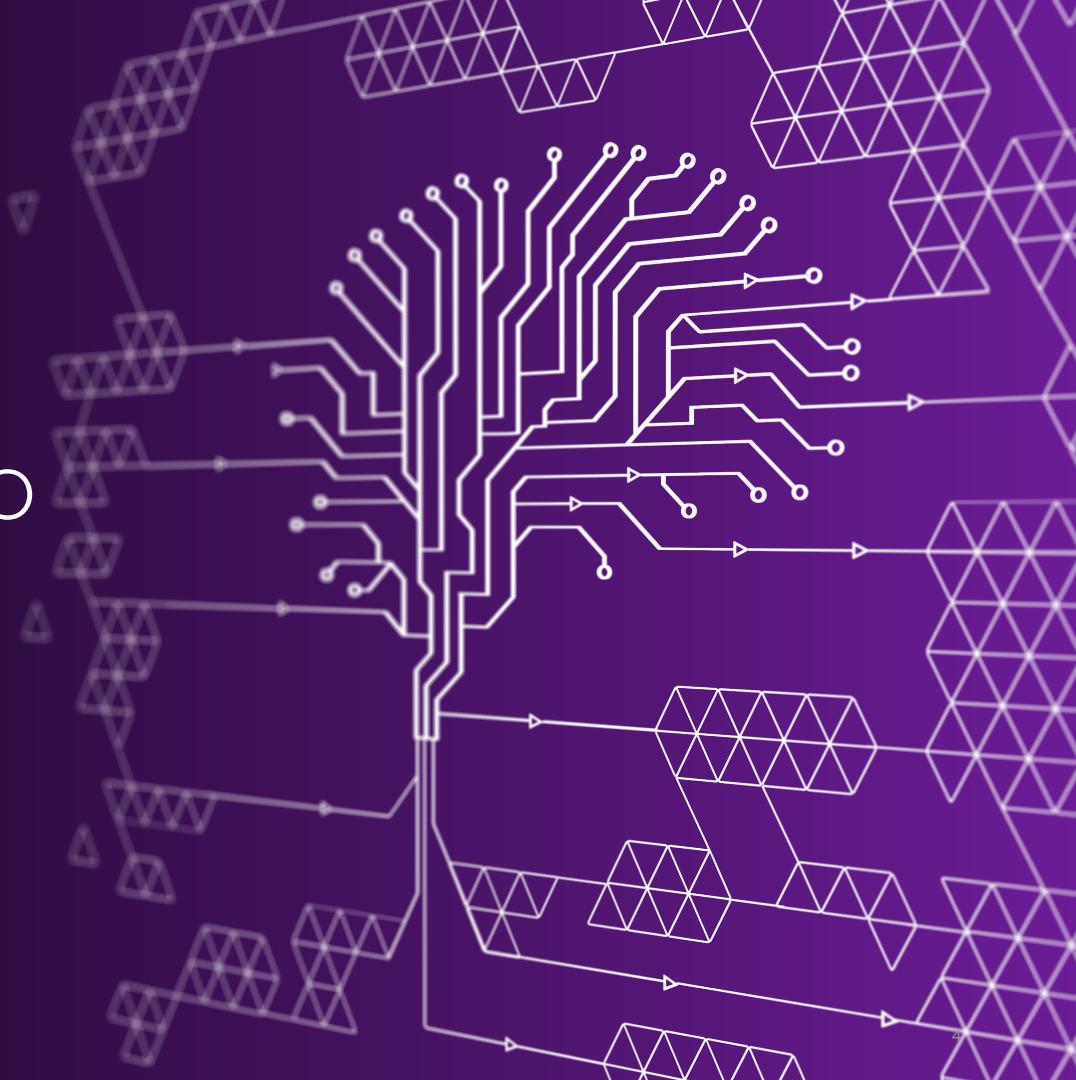


Cédrick Lunven
Software Engineer

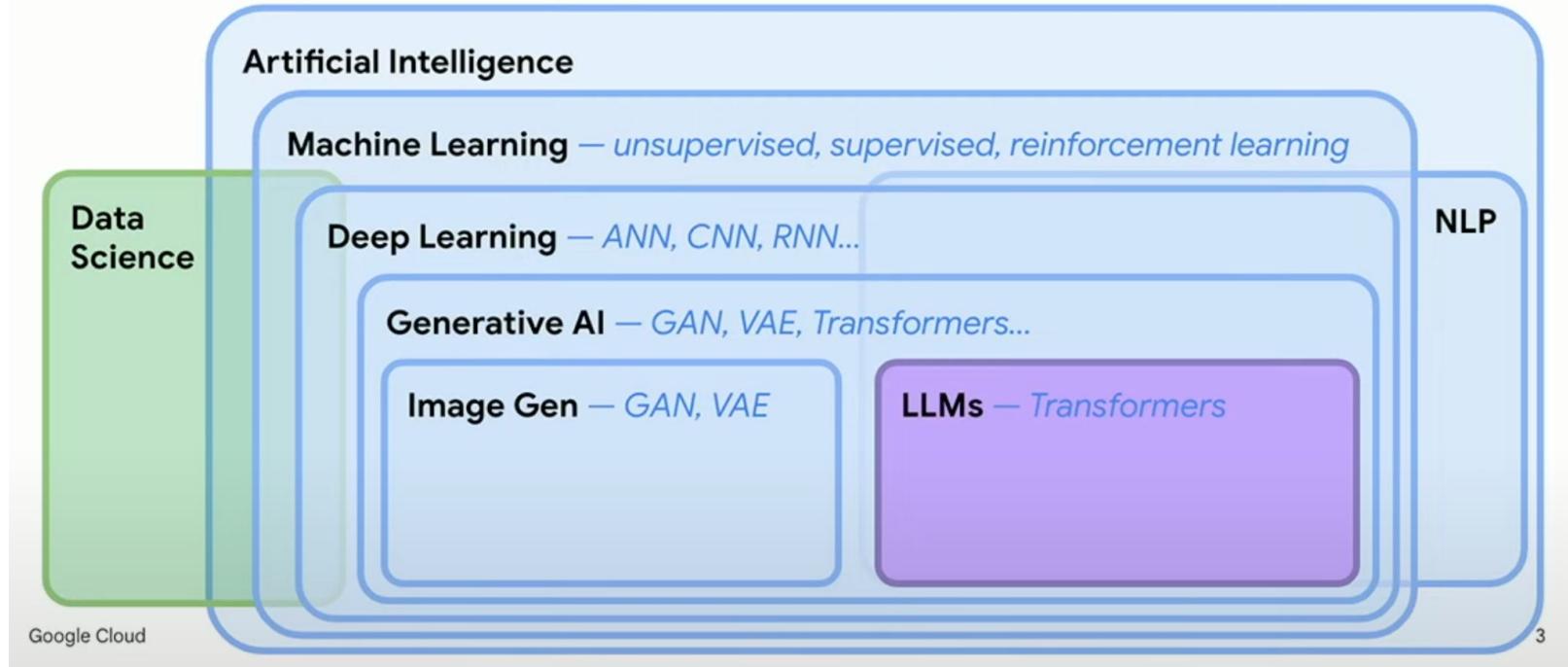




Introduction to Generative AI



➤ The Big picture



➤ Predictive vs Generative AI

Artificial Intelligence

Real-Time AI

Machine Learning

Chatbots &
virtual agents

Code generation

Tailored ads

Content
summaries

Generative AI

- Natural language responses
- Personalized experiences
- Tailored content

Predictive AI

- Quick decisions
- Accurate predictions
- Proactive actions



Interactive, contextualized
experiences

Smart, automatic
operations

Dynamic pricing

Real-time trading

Predictive
maintenance

Inventory
optimization

➤ Large Language Models (LLM)

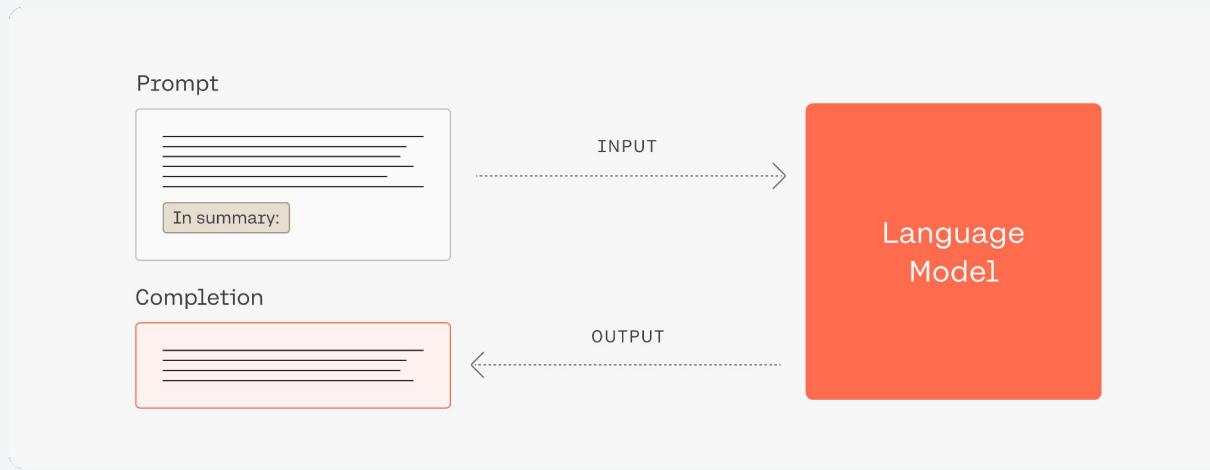


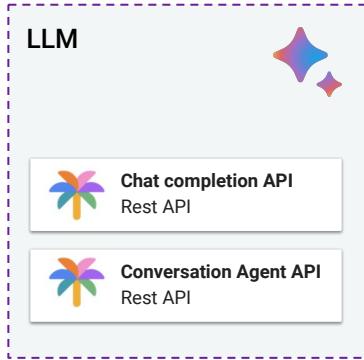
image by [cohere](#)

Large language models are trained on massive amounts of text data.

Designed to process and understand natural language, such as human speech and text.

This allows the model to learn patterns and relationships between words, phrases, and sentences, enabling it to generate coherent and meaningful language output.

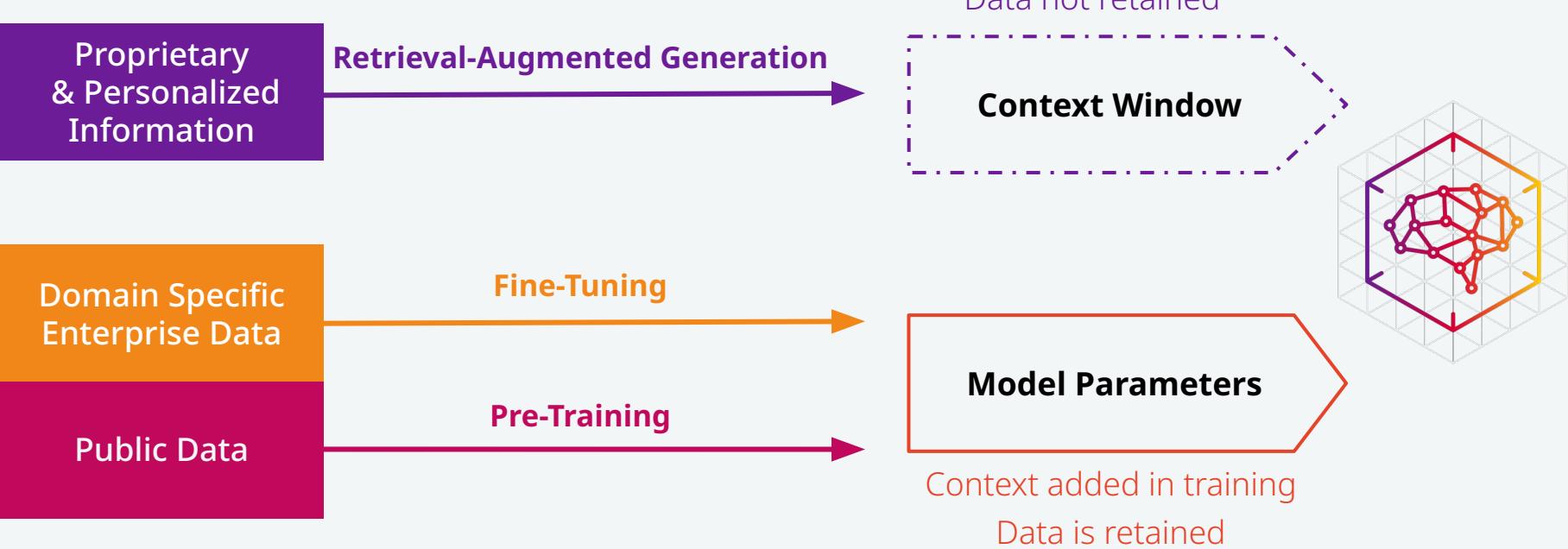
➤ Limitations of “LLM only” mode



- LLMcan be outdated
- LLMDoes not know *your* data
- LLMis not tuned = hard steerability
- LLMHallucinating if not properly prompted
- LLMworks with limited Input windows (tokens)

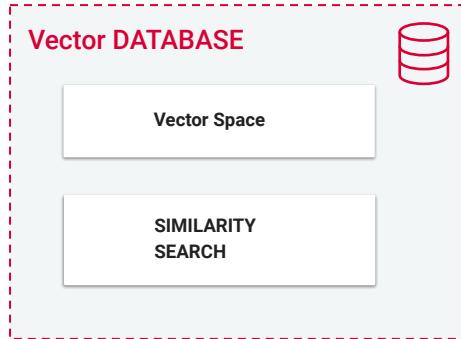
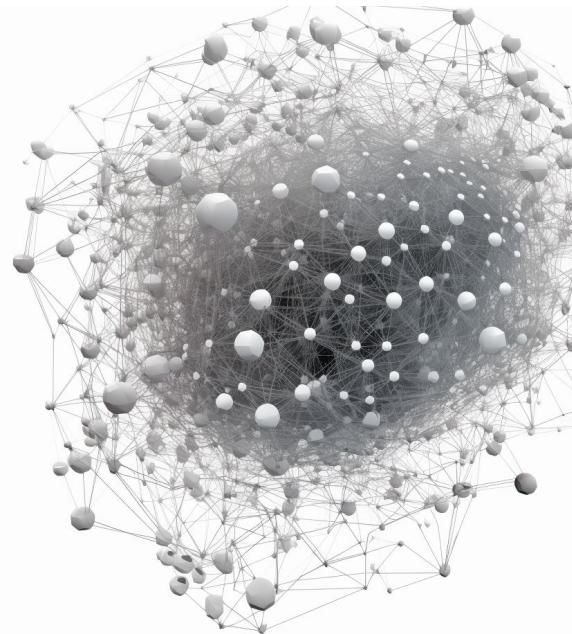
**There is no AI
without Data**

➤ Adding Enterprise Context to GenAI

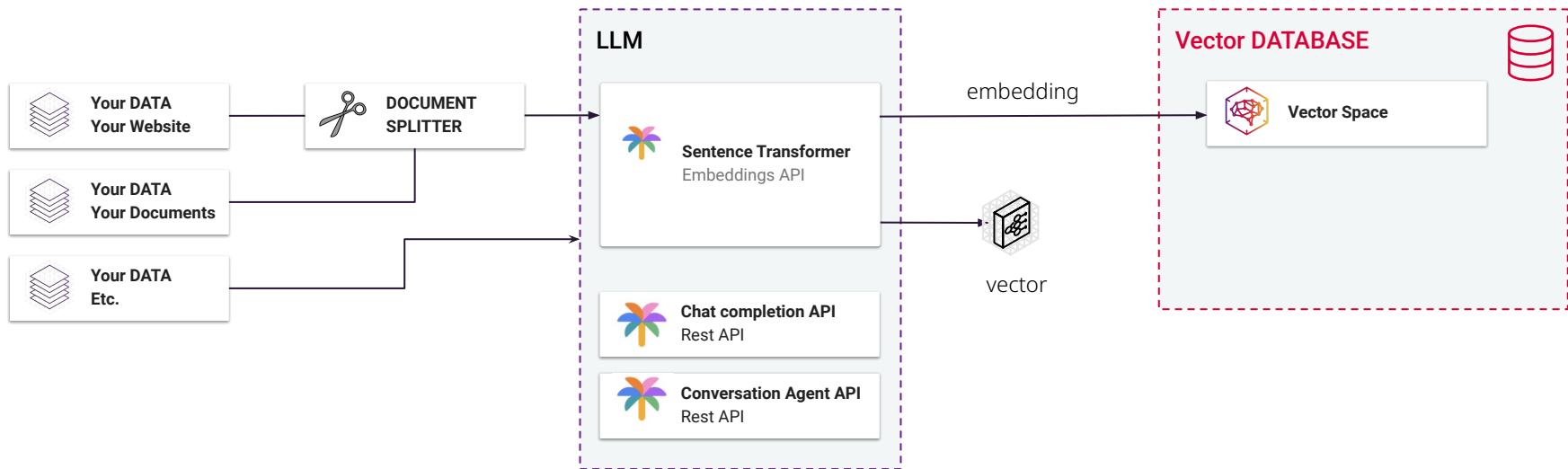


➤ Good Vector Database ?

- Handling **A LOT** of vector
- Effective Search Algorithms
- Performant, Resilient
- Dynamic (vector sizes)
- Meta Data Filtering
- Keyword search
- Semantic Caching
- Chat History
- Key Value cache

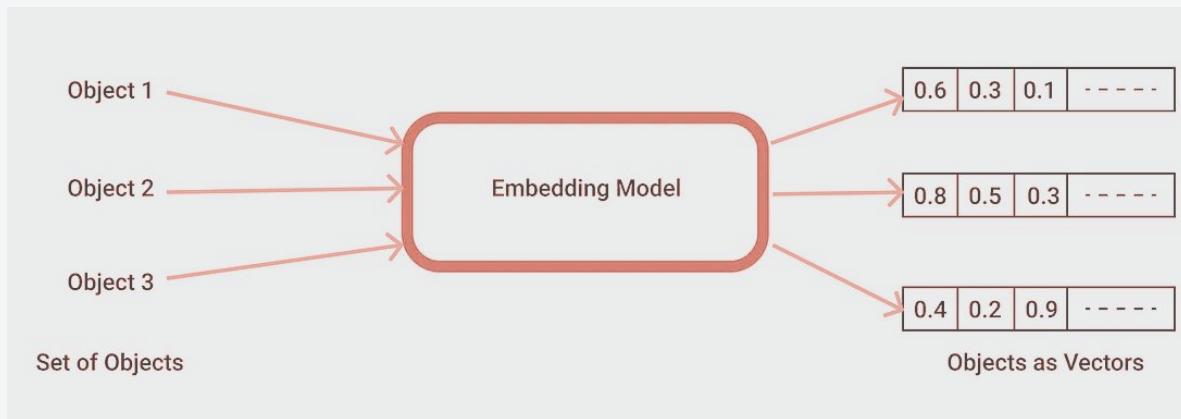


➤ Adding Context: **Vectorization**



What is Embedding?

Take source objects (text, images, sound, movies) and create Vector format representation of the context.
This allows for Similarity Search on the Database finding Semantically comparable objects.



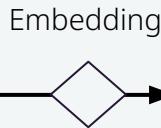
**Goal is to capture semantics and context.
Processed by a ML model**

What is Vector?

Vector is a multi-dimensional numerical representation of text/image/video

"To create a security token that can be used to log into a database, select Token Management from the User Management menu. Then, choose an appropriate role for the user, and click the Generate Token button. Copy the token details to a safe place, as the secret that is shown can never be reproduced in the Astra console for security reasons."

Raw Text

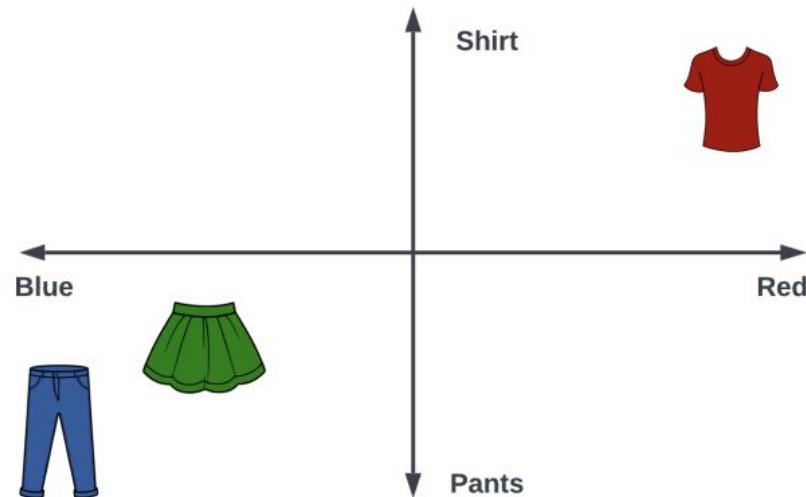


```
[ -0.29334254562854767, 0.06338247656822205, 0.03711941838264465, 0.06770425289869308, 0.030722564086318016, -0.03855780512094498, 0.05715630576014519, 0.0125797235965729, 0.0320907607645987, 0.018565965816378593, 0.005725057329982519, 0.003278332995250821, 0.019661232829093933, 0.008483093231916428, 0.01011530589312315, -0.0686598121199036, -0.02742725796997547, 0.004272086545825005, 0.006464742589741945, 0.033381473273038864, -0.06456394493579865, -0.001686627672735039, -0.02538292482495308, -0.01352921282730103, 0.006745325401425362, -0.09090574085712433, -0.004533074174914563, -0.0157530866495653, -0.0293307810218048, -0.0135655625995397568, 0.0373402051626763596, -0.013345368206501007, -0.04631896317005157, 0.01822153415060043, -0.030514687299728394, 0.060879088993823624, -0.015947293490171432, -0.0043849475760161877, 0.01510275304317474, 0.03181201219558716, 0.004961538943462074, -0.00960769411764431, -0.0026698557194322348, -0.02120211534202099, -0.02445143461227417, 0.018808122724294662, -0.04526928439736366, -0.0350750833749771, 0.01936023831367494, -0.04246317967772484, -0.04538712650537491, 0.003057188587262265, -0.02645616978406906, 0.00433978391812516, -0.00498298677004576, 0.0019529943620089102, -0.015389597043395042, -0.008066490292549133, -0.04361098087349, 0.018591511994600296, -0.0082497593910911259, 0.031464317948191284, 0.005365008022636175, -0.034285105764865875, -0.04675269049406052, 0.06229010224342346, -0.0160919959575867, 0.03467985361814, 0.005365008022636175, -0.034285105764865875, -0.04675269049406052, 0.06229010224342346, -0.0160919959575867, -0.038237784057855606, -0.01697474904358387, 0.0023320959880948067, -0.028734117463515, -0.07216104120016098, 0.04663623124361038, 0.028389146806120872, -0.02821142040193081, -0.06294400244951248, 0.00343078840219814, 0.07920042425394058, 0.007338271476328373, 0.0650653690909162, -0.06103818118572235, 0.06294400244951248, 0.00343078840219814, 0.07920042425394058, 0.007338271476328373, 0.0650653690909162, -0.0252226146276474, 0.02745004184540865, -0.01720043271780014, 0.04627233743690625, -0.05018896237015724, 0.01577943935909336, -0.026586400344967842, -0.0197460112306118, -0.00036689057014882565, -0.016816521063447, -0.0254684840233205958, 0.00071008078211757, 0.04524853080511093, 0.001050888327816938, -0.00547241867231131, 0.01160429238188224907, -0.042706481282711, -0.02004644088447094, -0.06824997067451477, -0.08084388822317123, -0.081672713161051483, 0.038480401039123535, -0.04149484634399414, 0.0621405728161335, 0.01636849343776703, -0.02775057591497898, 0.02410232089459896, 0.021344885230064392, 0.056428126990795135, 0.02979239635169506, -0.05207456275820732, 0.00429974822374946, 0.03471621247467, 0.034210272893476486, 0.0010842653136933222, 0.01124250236896607, 0.0379135665607452, -0.004098605364561081, 0.01202376637664845, 0.0216593053190205, 0.03850701451301575, -0.03979567810893059, 0.024909289553761482, 0.003612052416289236, 0.03026977797636988, 0.03532775491476059, 0.04048445075750351, -0.02123659290733253, 0.05895552039146423, 0.04913758486509323, -0.047305576503276825, 0.05272323951426506, 0.01215427809438705, -0.02513653226196766, -0.0105582932010293, -0.049685653299093246, 0.032950107008218765, -0.007436738815158606, -0.07494320720434189, -0.04471016082201004, 0.0316404938697815, -0.029877835884690285, -0.020543526858901354, 0.0253277961647129, 0.011234065517783165, 0.07374250143766403, 0.04288359731435776, 0.03435317426919937, -0.02951200306415558, -0.09385887533426285, -0.00531736761315582, 0.0170515943467617, -0.00934696663171053, 0.01293235830962658, 0.02108096517622471, 0.030062183737754822, 0.004270109347999096, -0.005795920733362436, 0.006119553931057453, -0.009726069867610931, -0.01605408842633233, -0.1282315403220337, 0.005963715258985758, -0.01607099547982216...]
```

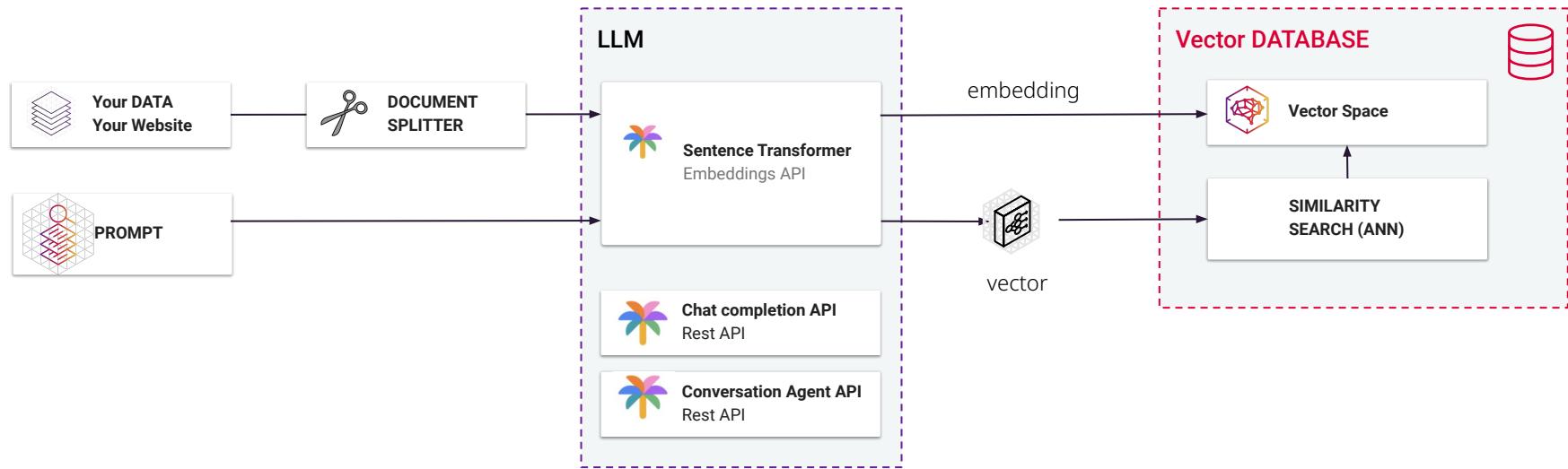
Vector

➤ What is Vector Search?

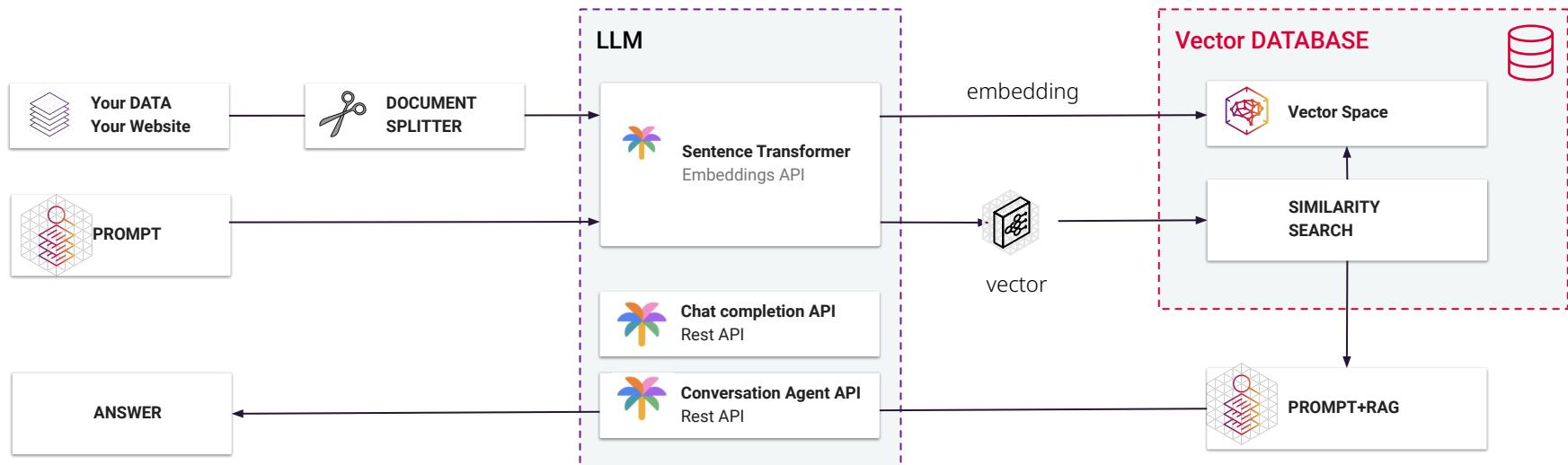
- Vector search finds objects that have similar meaning
- Vector search understands MEANING
- Vectors created from EXISTING data through EMBEDDINGS



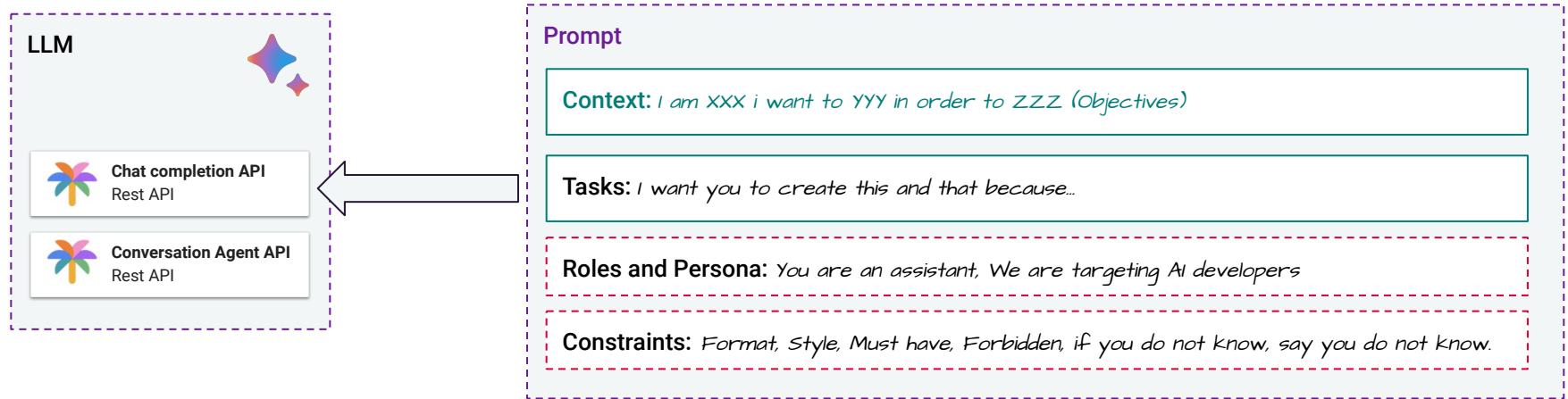
➤ Retrieving Context: **Semantic Search**



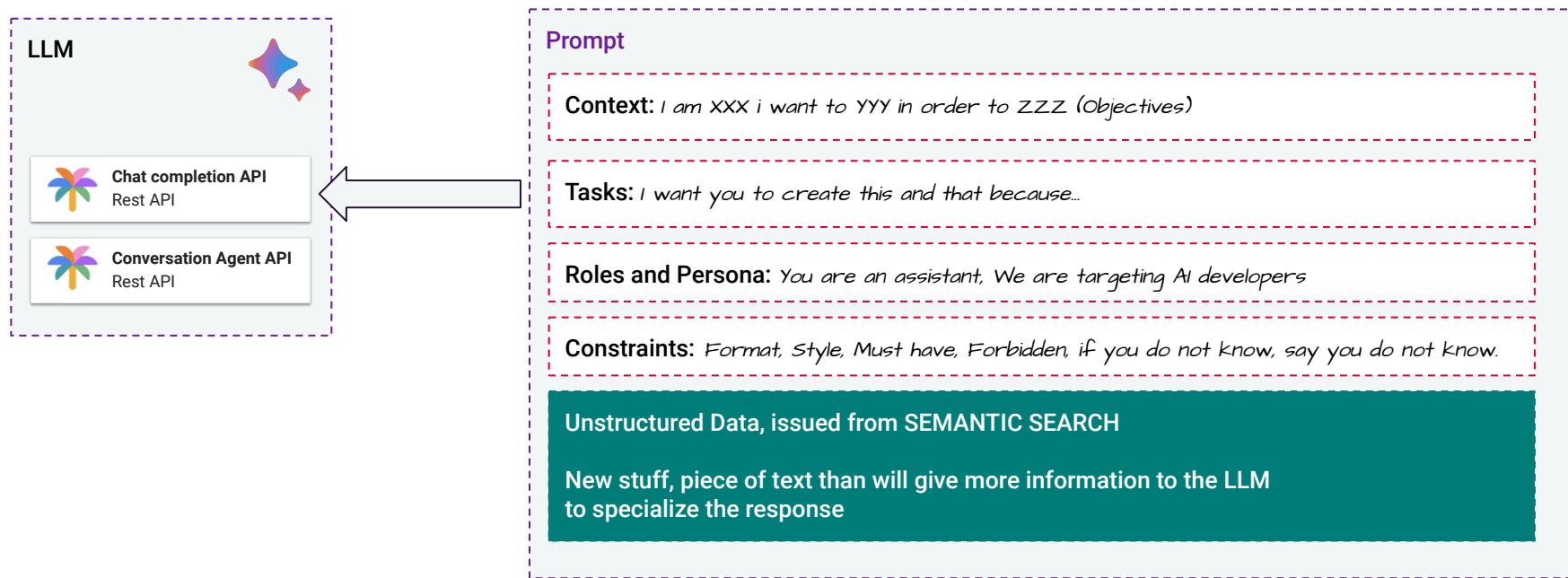
➤ RAG: Prompt Engineering



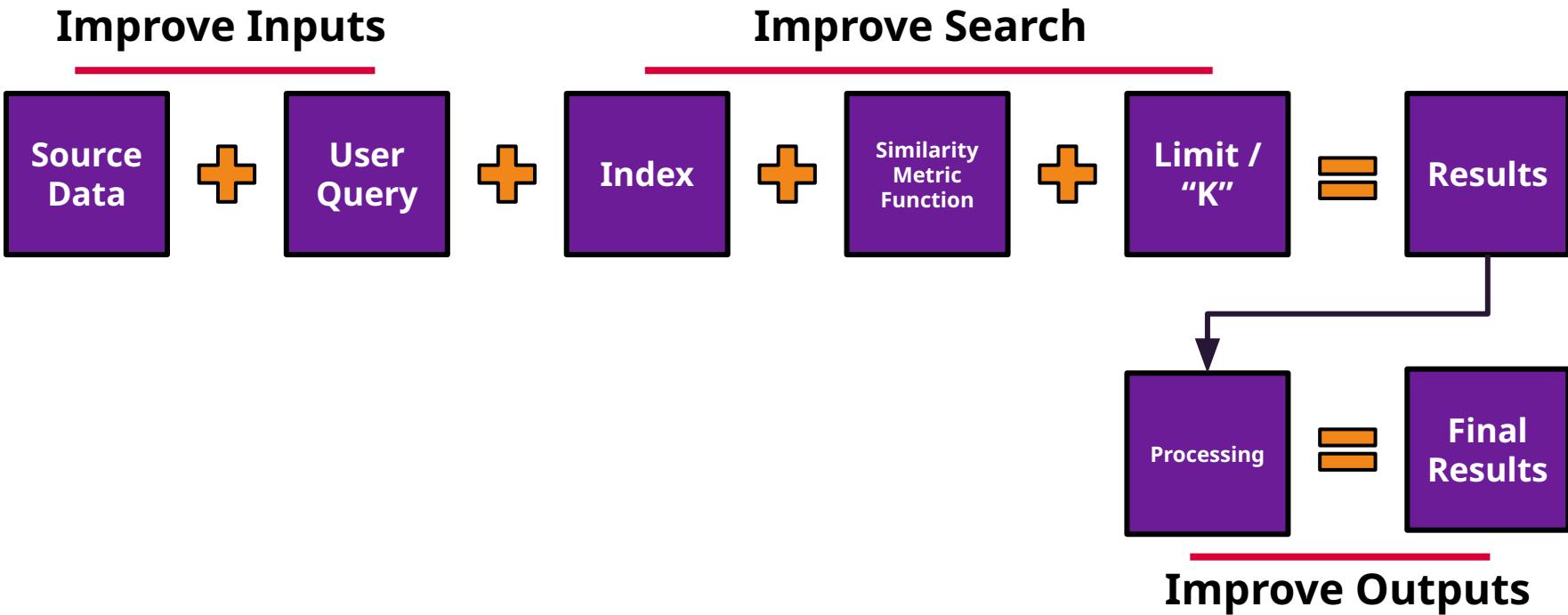
➤ Prompt Engineering



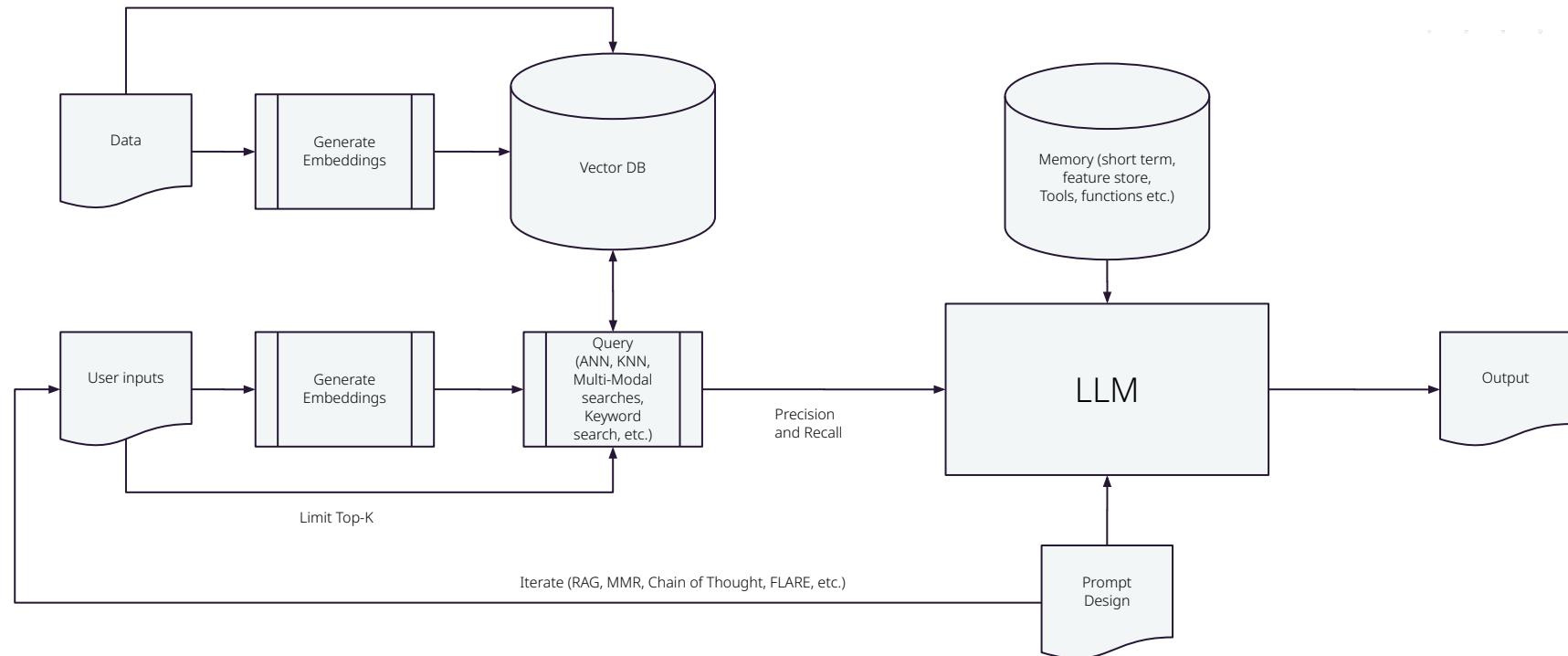
➤ Prompt Engineering



➤ How to improve relevance



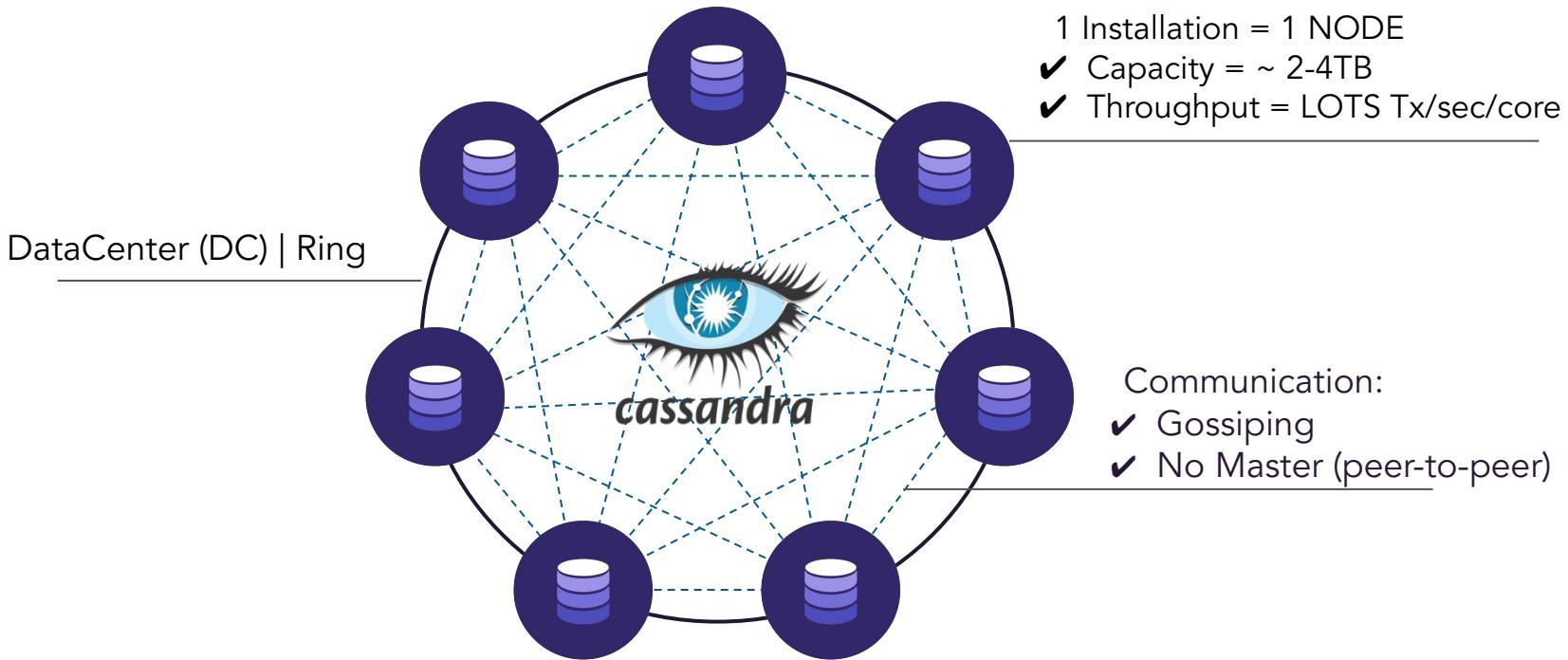
➤ Putting it all together





➤ Apache Cassandra™
as a vector database

➤ Nosql Distributed database



Apache Cassandra®

Undisputed Leader of Scale and Reliability

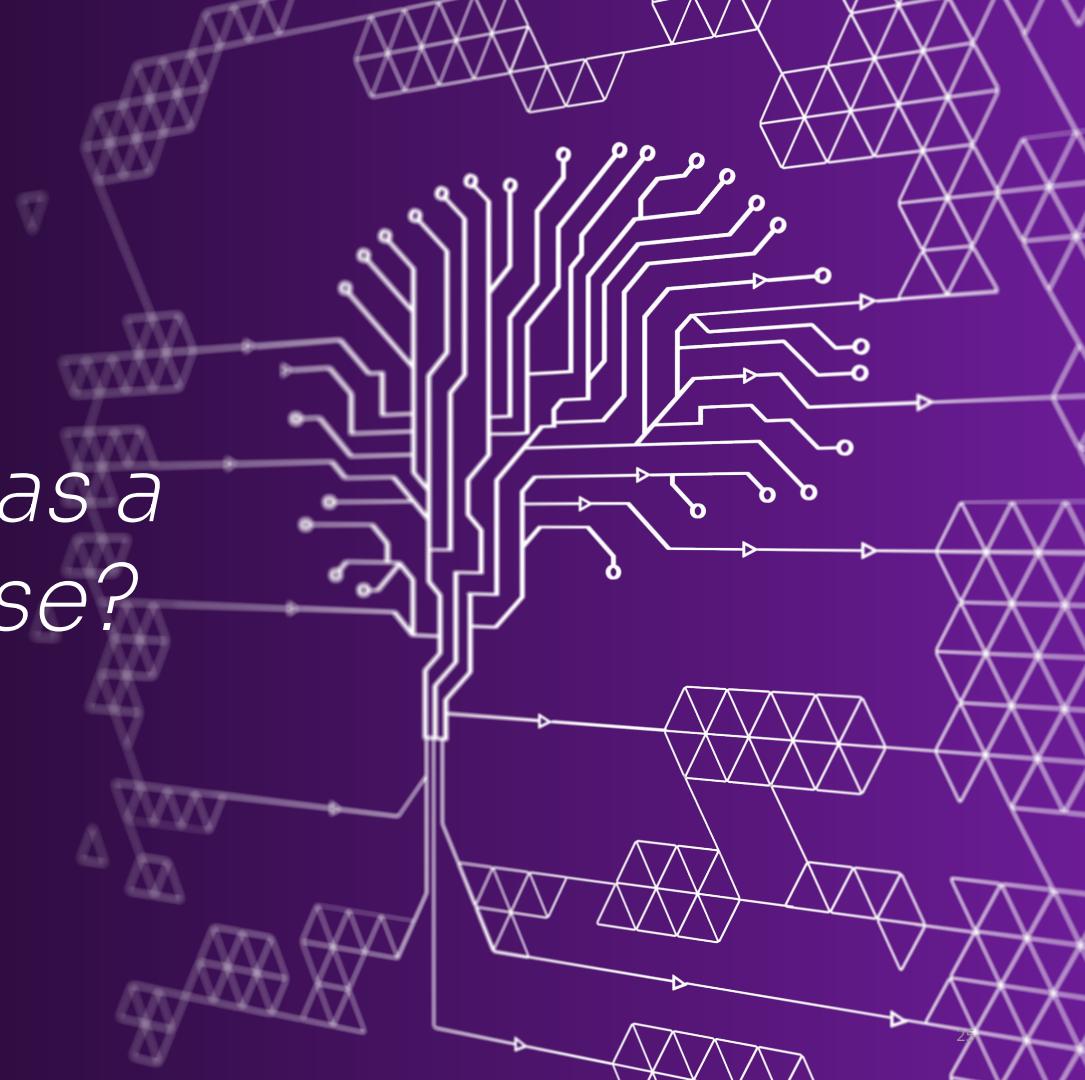
Apache Cassandra at Apple Scale and Scope

- Over three hundred thousand instances
- Hundreds of petabytes of data
- Over two petabytes per cluster
- Millions of queries per second
- Thousands of clusters
- Thousands of applications

Instances	Storage	Density
QPS	Clusters	Applications

APACHECON

» Why Astra DB as a vector database?



➤ DataStax Astra DB

- Apache Cassandra® in the cloud!
- <https://astra.datastax.com/>
- Free tier available!



› Fast and Easy

Things you don't have to worry
about

Scale

Cost Efficiency

Security

Reliability

Low latency

Performance

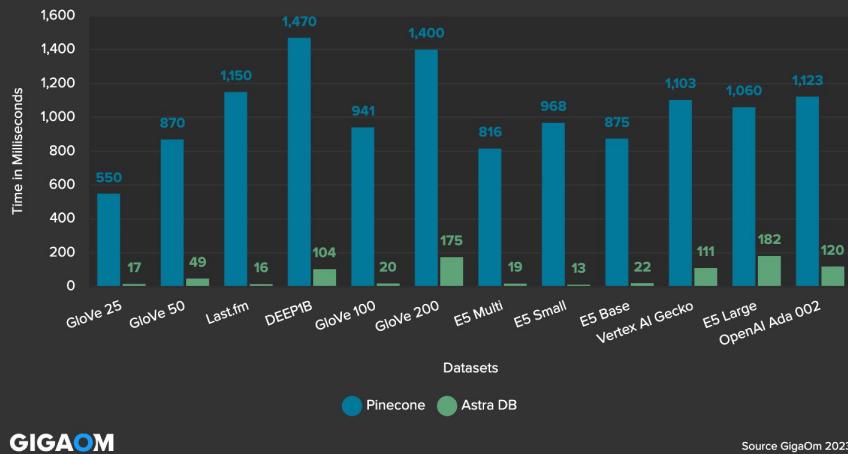


10x Faster than any Vector DB

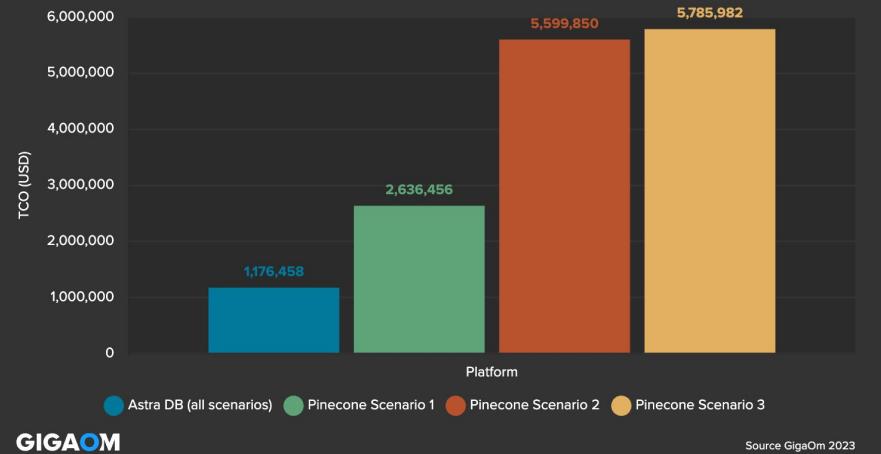
Scales to Billions of Rows

Industry Leading Performance & TCO

SEARCH QUERY RESPONSE DURING INDEXING/TRAINING AT 99TH PERCENTILE (LOWER IS BETTER)

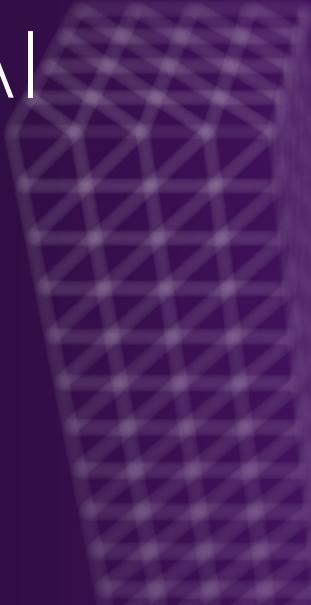


3-YEAR TOTAL COST OF OWNERSHIP (LOWER IS BETTER)



...For AI Apps & Services

Now let's build your own GenAI application



The screenshot shows a web-based application interface. At the top right, there are standard browser controls: Share, Star, Print, and a three-dot menu. Below the header, a large wireframe cube is displayed, representing a complex data structure or search space.

The main content area has a white background. On the left, there's a sidebar with a "Logout" button. The main panel starts with a heading "What is this app?". Below it, a descriptive text explains that the app is a Chat Agent designed to handle enterprise context for meaningful responses. It notes that foundational language models are not trained on enterprise data, so they lack context about the user's organization.

Next is a section titled "What does it know?", which lists pre-loaded information such as the DataStax homepage, vector search for generative AI apps, Astra DB Datasheet, Digital Champion Priceline, and Luna for Apache Cassandra. It also includes a note that users can start interacting with their personal assistant based on this information.

Below this is a section for "Add additional content", which allows users to drop PDF or Text files into an upload box. A note cautions users to be careful with the "Delete context" button, as it removes preloaded content.

The bottom part of the interface is a chat window. It shows a message from a bot asking if the user is ready to help, followed by a question from the user "What's up?". There's also a "Manage app" button at the bottom right.

› Prerequisites status check

- Go to <https://github.com/clun/build-your-own-rag-chatbot>

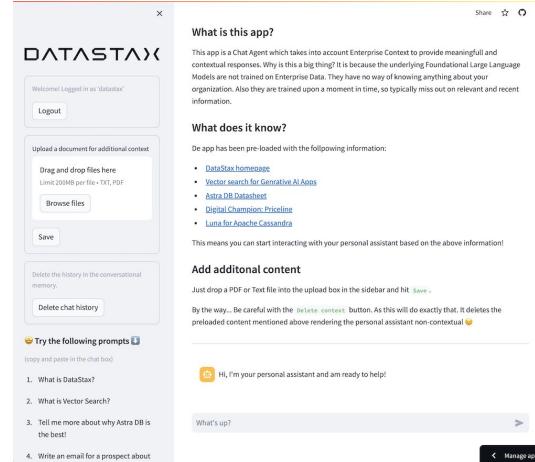


- A Github account
- Google Colab
- Accounts created during the session:
 - DataStax Astra DB
 - OpenAI account
 - Streamlit

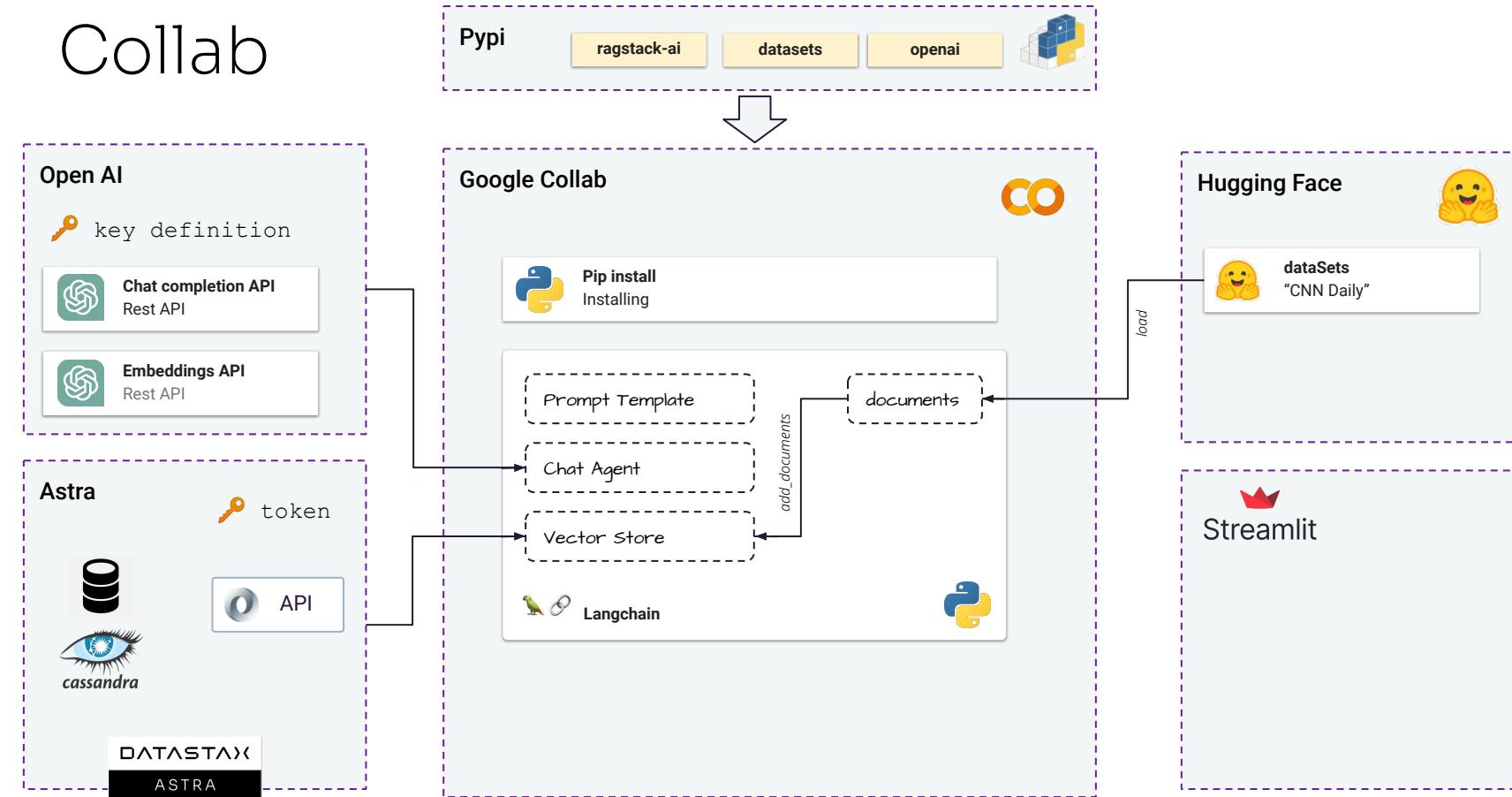


What you'll learn

- How to leverage [DataStax RAGStack](#) for production-ready use of the following components:
 - The [Astra DB Vector Store](#) for Semantic Similarity search
 - [LangChain](#) for linking OpenAI and Astra DB
- How to use [OpenAI's Large Language Models](#) for Q&A style chatbots
- How to use [Streamlit](#) to easily deploy your awesome app to the internet for everyone to see!



Collab



LLM Framework

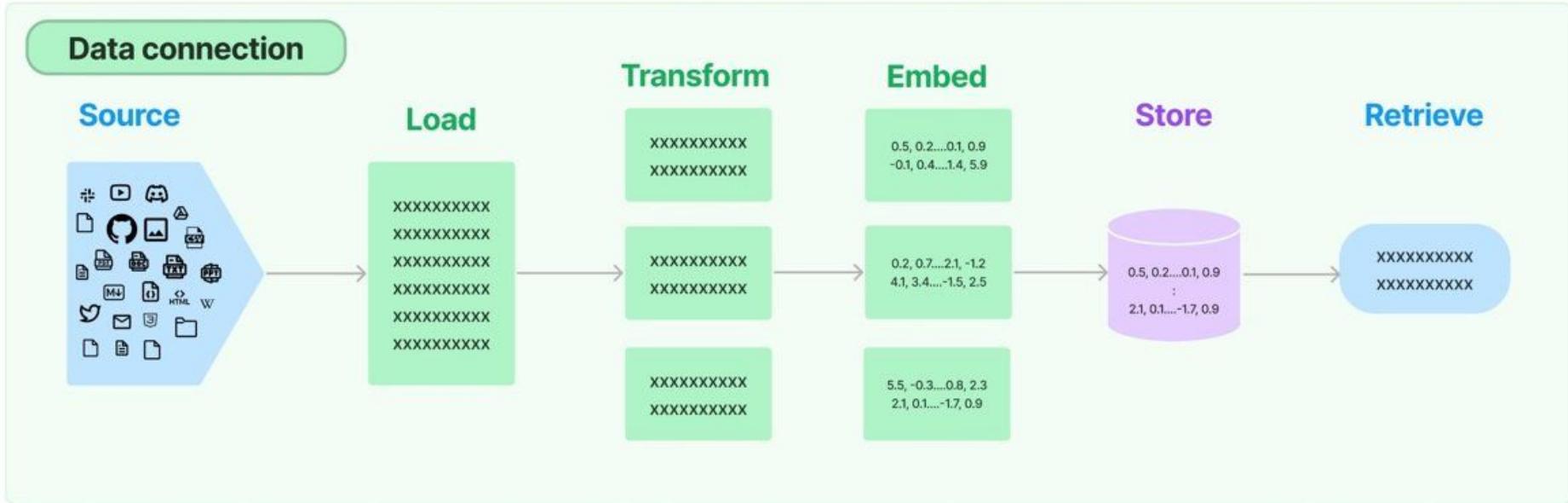
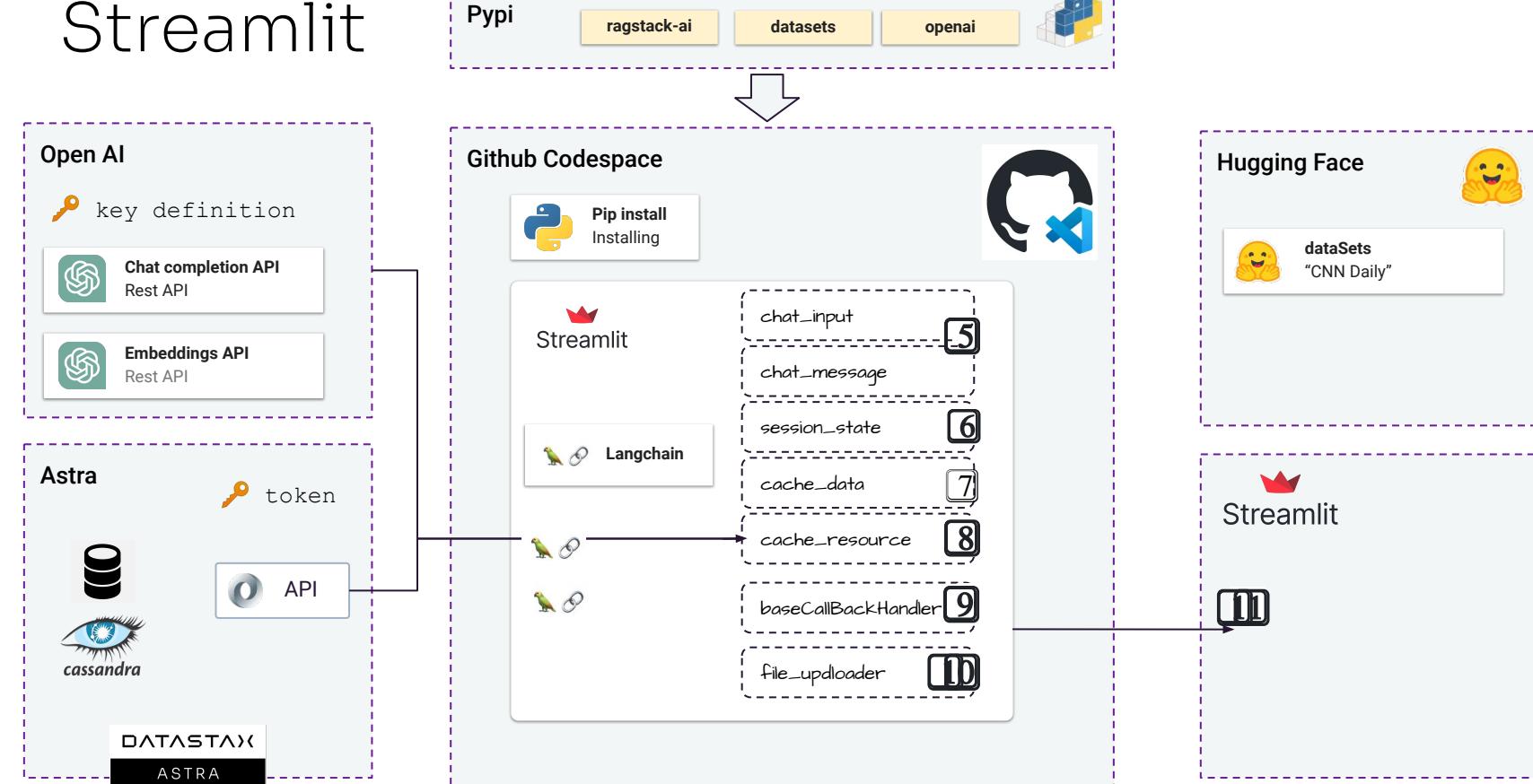


Image Credit: <https://python.langchain.com/>

Streamlit



DATASTAX

Thank
You



DATASTAX