

Welcome to Onyx

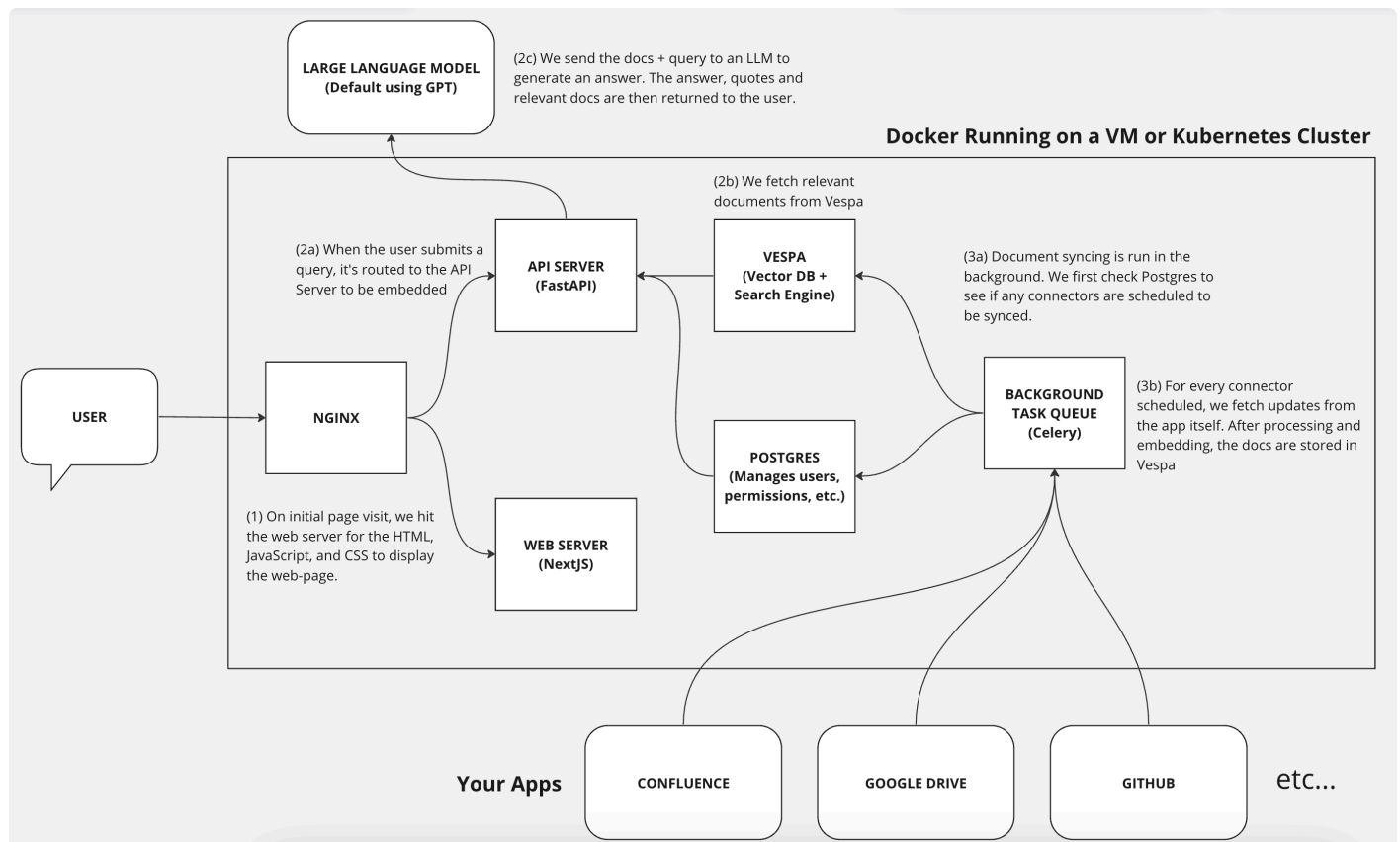
System Overview

Explanation of different system components and flows

This page discusses how Onyx works from a high level. The aim is to give transparency into our designs. That way you can have peace of mind in using Onyx.

Alternatively if you're looking to customize the system or become an open source contributors, this is a great place to start.

System Architecture

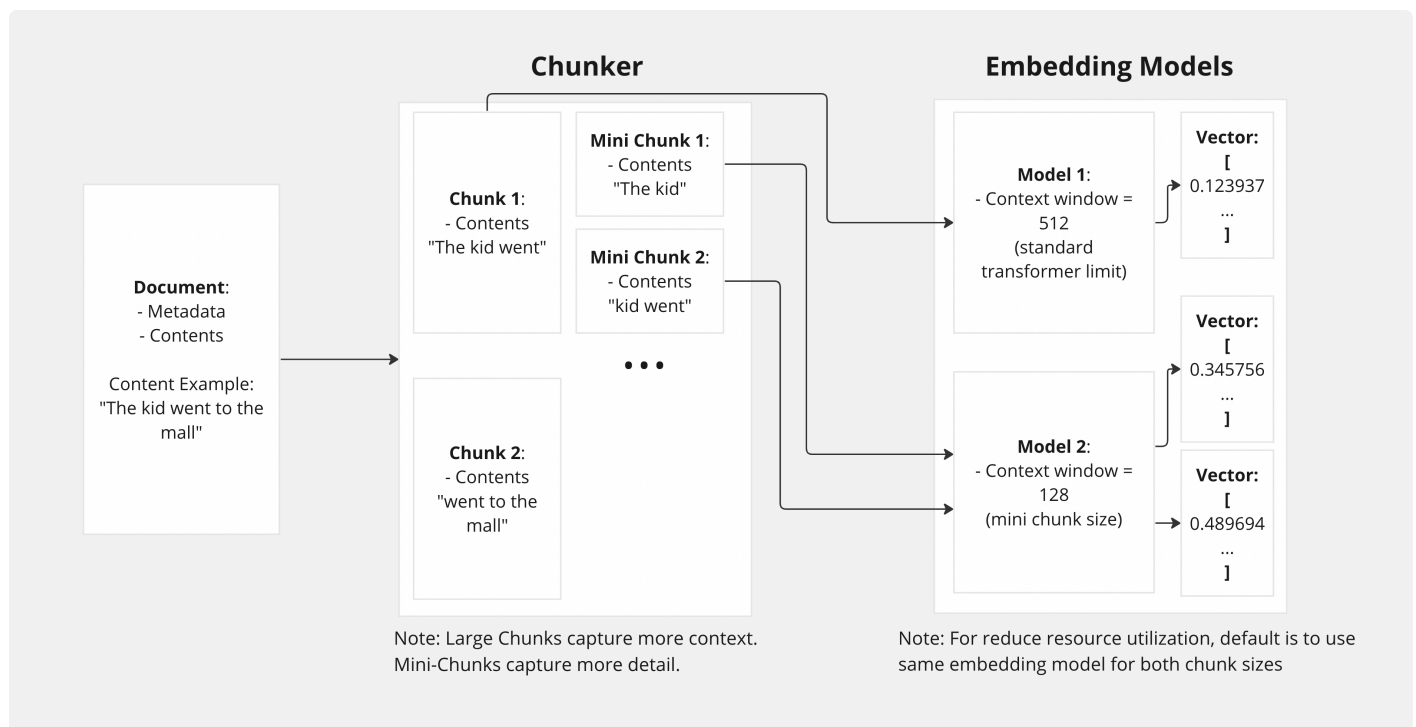


Whether deploying Onyx on a single instance or a container orchestration platform, the flow of data is the same. Documents are pulled and processed via connectors and then persisted in Vespa/Postgres which run in containers within your system.

The only time sensitive data leaves your Onyx setup is when it makes a call to an LLM to generate an answer. The communications to the LLM are encrypted. The data persistence at the LLM API depends on the terms of the LLM hosting service you are using.


We also note that Onyx has some very limited and anonymous telemetry to help us improve the system by identifying bottlenecks and flaky data connectors. You can turn off the telemetry with by setting the `DISABLE_TELEMETRY` env variable to `True`.

Embedding Flow



Each document is split into smaller sections called "Chunks".

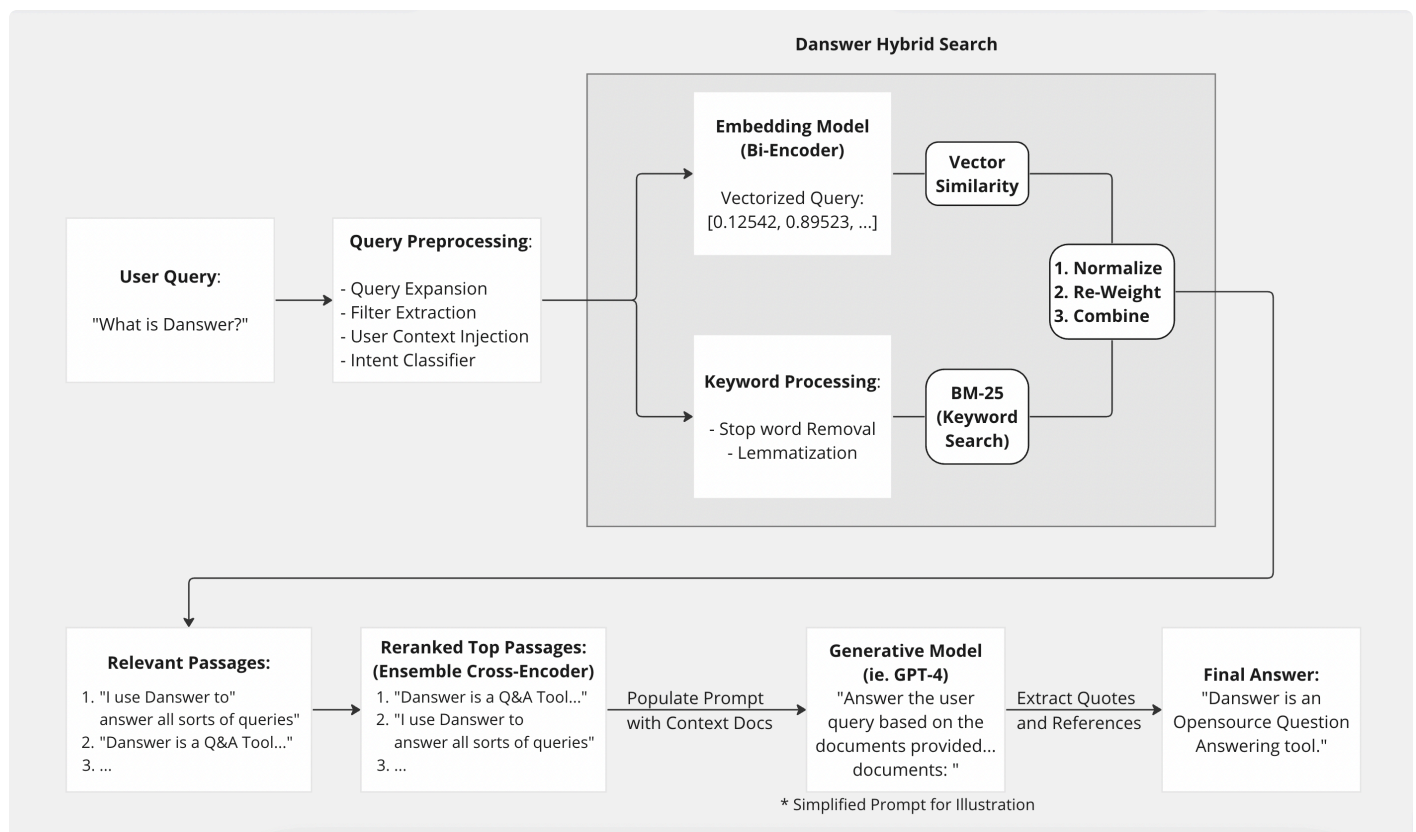
By passing chunks to the LLM instead of full documents, we are able to reduce noise to the model by only passing in the relevant sections of the document. Additionally this is significantly more cost efficient as LLM services generally charge per token. Finally by

embedding chunks rather than full documents, we are able to retain more detail as each  vector embedding is only capable of encoding a limited amount of information.

The addition of mini-chunks takes this concept even further. By embedding at different sizes, Onyx can retrieve both high level context and details. Mini-chunks can also be turned on/off via environment variables as generating multiple vectors per chunk can slow down document indexing if deployed on less capable hardware.

In choosing our embedding model, we use the latest state-of-the-art biencoder that is small enough to run on CPU while maintaining subsecond document retrieval times.

Query Flow



This flow is often updated as we're constantly striving to push the capabilities of the retrieval pipeline with the most recent advancements coming out of research and the opensource community. Also note that many of the parameters of this flow such as how much documents to retrieve, how many to rerank, what models to use, which chunks are passed to the LLM, etc. are all configurable.

If there are any questions, don't hesitate to reach out to the maintainers!



[◀ Multilingual Setup ▶](#)

[Contact Us ▶](#)

Powered by Mintlify