

January 12, 2024

SHOWCASE • LLM

# Pathway is now available in LlamaIndex, here is how to get started!

You can now use Pathway in your RAG applications which enables always up-to-date knowledge from your documents to LLMs with LlamaIndex integration.

---

Pathway is now available on [LlamaIndex](#), a data framework for LLM-based applications to ingest, structure, and access private or domain-specific data. You can now query Pathway and access up-to-date documents for your RAG applications from LlamaIndex using Pathway [Reader](#) and [Retriever](#).

With this new integration, you will be able to use Pathway vector store natively in LlamaIndex, which opens up endless new possibilities! In this article, you will have a quick dive into Pathway + LlamaIndex to explore how to create a simple, yet powerful RAG solution using PathwayRetriever.

## Why Pathway?

Pathway offers an indexing solution that is always up to date without the need for traditional ETL pipelines, which are needed in regular VectorDBs. It can monitor several data sources (files, S3 folders, cloud storage) and provide the latest information to your LLM application.

## Learning outcomes

You will learn how to create a simple RAG solution using Pathway and LlamaIndex.

This article consists of:

- Create data sources. Define data sources Pathway will read and keep the vector store updated.
- Creating a transformation pipeline (parsing, splitting, embedding) for loading documents into Vector store
- Querying your data and getting answers from LlamaIndex.

## Prerequisites

### Installing Pathway and LlamaIndex.

```
$ pip install pathway  
$ pip install llama-index
```

### Setting up a folder

To start, you need to create a folder Pathway will listen to. Feel free to skip this if you already have a folder on which you want to build your RAG application. You can also use Google Drive, Sharepoint, or any other source from [pathway-io](https://pathway.io).

```
$ mkdir -p 'data/'
```

### Set up OpenAI API Key

```
import getpass  
import os  
  
# omit if embedder of choice is not OpenAI  
if "OPENAI_API_KEY" not in os.environ:  
    os.environ["OPENAI_API_KEY"] = getpass.getpass("OpenAI API Key:")
```

### Define data sources

Pathway can listen to many sources simultaneously, such as local files, S3 folders, cloud storage, and any data stream.

See [pathway-io](#) for more information.

You can easily connect to the data inside the folder with the Pathway file system connector. The data will automatically be updated by Pathway whenever the content of the folder changes.

```
import pathway as pw

data_sources = []
data_sources.append(
    pw.io.fs.read(
        "./data",
        format="binary",
        mode="streaming",
        with_metadata=True,
    ) # This creates a `pathway` connector that tracks
    # all the files in the ./data directory
)
```

## Create the document indexing pipeline

Now that the data is ready, you must create the document indexing pipeline. The transformations should be a list of `TransformComponent` s ending with an `Embedding` transformation.

First, split the text using `TokenTextSplitter` , then embed it with `OpenAIEmbedding` .

Finally, you can run the server with `run_server` .

```
from llama_index.retrievers import PathwayVectorServer
from llama_index.embeddings import OpenAIEmbedding
from llama_index.node_parser import TokenTextSplitter

embed_model = OpenAIEmbedding(embed_batch_size=10)

transformations_example = [
    TokenTextSplitter(
        chunk_size=150,
        chunk_overlap=10,
        separator=" ",
    ),
    embed_model,
]

processing_pipeline = PathwayVectorServer(
    *data_sources,
    transformations=transformations_example,
)

# Define the Host and port that Pathway will be on
PATHWAY_HOST = "127.0.0.1"
PATHWAY_PORT = 8754

# `threaded` runs pathway in detached mode, you have to set it to False when run
# for more information on `with_cache` check out https://pathway.com/developers
processing_pipeline.run_server(
    host=PATHWAY_HOST, port=PATHWAY_PORT, with_cache=False, threaded=True
)
```

Awesome! The vector store is now active, you're set to start sending queries.

## Create LlamIndex Retriever and create Query Engine

```
from llama_index.retrievers import PathwayRetriever
```

```
retriever = PathwayRetriever(host=PATHWAY_HOST, port=PATHWAY_PORT)
retriever.retrieve(str_or_query_bundle="what is pathway")
```

```
from llama_index.query_engine import RetrieverQueryEngine
```

```
query_engine = RetrieverQueryEngine.from_args(
    retriever,
)
```

```
response = query_engine.query("What is Pathway?")
print(str(response))
```

```
Out[]: Empty Response
```

As you can see, the LLM cannot respond clearly as it lacks current knowledge, but this is where Pathway shines. Add new data to the folder Pathway is listening to, then ask our agent again to see how it responds.

To do that, you can download the repo readme of Pathway into our `data` folder:

```
$ wget 'https://raw.githubusercontent.com/pathwaycom/pathway/main/README.md' -C
```



Try again to query with the new data:

```
response = query_engine.query("What is Pathway?")
print(str(response))
```

```
Out[]: Pathway is a Python framework that allows for high-throughput and low-latency
```



As you can see, after downloading the document to the folder Pathway is listening to, changes are reflected to the query engine immediately. LLM responses are up to date

with the latest changes in the documents which would require extra ETL steps in regular Vector DBs.

## Conclusion

With the integration of Pathway within LlamaIndex, you can now access up-to-date documents for your RAG applications from LlamaIndex. You should now be able to use Pathway Reader and Retriever to connect to your data sources and monitor for changes, providing always up-to-date documents for your LlamaIndex application.

If you are interested in building RAG solutions with Pathway, don't hesitate to read [how the vector store pipeline is built with Pathway](#). To learn more about the possibilities of combining the live indexing pipeline of Pathway and LLMs, check out [real-time RAG alerting with Pathway](#) and [ingesting unstructured data to structured](#).

## Related articles



### Build an LLM App

Mohamed Malhou 2023-07-20



### Use LLMs for notifications

Pathway Team 2023-11-17

U  
T  
B



Pathway Team

#LLM #RAG #GPT #OpenAI #LlamaIndex

### Share this article



C  
-  
F

Showcases

← Always up-to-date Vector Data Indexing pipeline

Showcases

Build an LLM App →

-

Success Stories

Our Story

Careers

Events

## Developers

---

Documentation

Tutorials

Showcases

## About

---

Legal & GDPR

Equal opportunity employer

Privacy policy

Licensing

Media kit

Glossary

## Contact

---

Let's talk

### Chat with us on Discord

Pathway

96bis Boulevard Raspail

Agoranov

75006 Paris, France

[contact@pathway.com](mailto:contact@pathway.com)

© 2021-2023 Pathway