



 cocoindex-io / cocoindex



[Code](#) [Issues](#) 58 [Pull requests](#) [Discussions](#) [Actions](#) [Projects](#) 1 [Security](#)







cocoindex / examples / multi\_format\_indexing / 

Add file ▾

 georgeh0 chore: bump cocoindex dependency version (#984) ✓

cab0cce · 6 hours ago



| Name   | Name                                   | Last commit date |
|--|--|------------------|
|  ..             |  |                  |
|  source_files   | chore: update source files for the ... | last month       |
|  .env           | chore: set PYTORCH_ENABLE_MPS_FA...    | 3 weeks ago      |
|  README.md      | update examples for cli to cocoin...   | last week        |
|  main.py        | example: add multi_format_index...     | 2 months ago     |
|  pyproject.toml | chore: bump cocoindex depende...       | 6 hours ago      |

README.md



# Build visual document index from PDFs and images with ColPali

 Stars 2.8k

In this example, we build a visual document indexing flow using ColPali for embedding PDFs and images. and query the index with natural language.

We appreciate a star ★ at [CocoIndex Github](#) if this is helpful.

## Steps

### Indexing Flow

1. We ingest a list of PDF files and image files from the `source_files` directory.
2. For each file:
  - **PDF files:** convert each page to a high-resolution image (300 DPI)
  - **Image files:** use the image directly

- Generate visual embeddings for each page/image using ColPali model
3. We will save the embeddings and metadata in Qdrant vector database.

## Query

We will match against user-provided natural language text using ColPali's text-to-visual embedding capability, enabling semantic search across visual document content.

## Prerequisite

[Install Qdrant](#) if you don't have one running locally.

You can start Qdrant with Docker:

```
docker run -p 6333:6333 -p 6334:6334 qdrant/qdrant
```



## Run

Install dependencies:

```
pip install -e .
```



**NOTE:** The `pdf2image` requires `poppler` to be installed manually. Please refer to [this document](#) for the specific installation instructions for your platform.

Setup:

```
cocoindex setup main.py
```



Update index:

```
cocoindex update main.py
```



Run:

```
python main.py
```



## Data Attribution

The example data files used in this demonstration come from the following sources:

### PDF Documents

- **ArXiv Papers:** Research papers sourced from [ArXiv](#), an open-access repository of electronic preprints covering various scientific disciplines.

## Image Documents

- **Healthcare Industry Dataset:** Images from the [vidore/syntheticDocQA\\_healthcare\\_industry\\_test](#) dataset on Hugging Face, which contains synthetic document question-answering data for healthcare industry documents.
- **ESG Reports Dataset:** Images from the [vidore/esg\\_reports\\_eng\\_v2](#) dataset on Hugging Face, containing Environmental, Social, and Governance (ESG) reports.

We thank the creators and maintainers of these datasets for making their data available for research and development purposes.

## About ColPali

This example uses [ColPali](#), a state-of-the-art vision-language model that enables:

- Direct visual understanding of document layouts, tables, and figures
- Natural language queries against visual document content
- No need for OCR or text extraction - works directly with document images

## CocoInsight

I used CocoInsight (Free beta now) to troubleshoot the index generation and understand the data lineage of the pipeline. It just connects to your local CocoIndex server, with Zero pipeline data retention. Run following command to start CocoInsight:

```
cocoindex server -ci main
```



Then open the CocoInsight UI at <https://cocoindex.io/cocoinsight>.