Document Loading

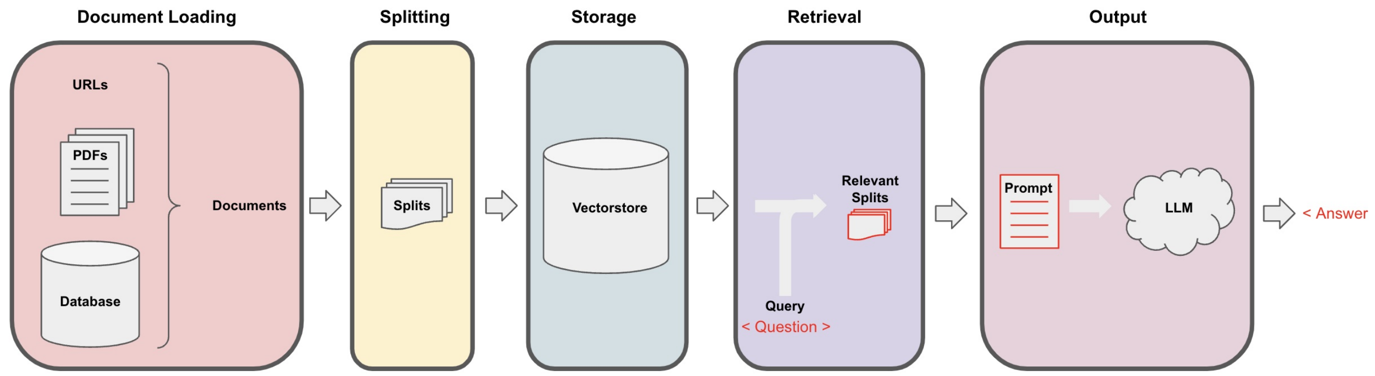
Note to students.

During periods of high load you may find the notebook unresponsive. It may appear to execute a cell, update th
completion number in brackets [#] at the left of the cell but you may find the cell has not executed. This is particularly obvious on print statements when there is no output. If this happens, restart the kernel using the command under the Kernel tab.

Retrieval augmented generation

In retrieval augmented generation (RAG), an LLM retrieves contextual documents from an external dataset as part of its execution.

This is useful if we want to ask question about specific documents (e.g., our PDFs, a set of videos, etc).



In [ ]:
```
#! pip install langchain
```

In [ ]:
```
import os
import openai
import sys
sys.path.append('../..')

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

openai.api_key  = os.environ['OPENAI_API_KEY']
```

PDFs

Let's load a PDF [transcript (https://see.stanford.edu/materials/aimlcs229/transcripts/MachineLearning-Lecture01.pdf)](https://see.stanford.edu/materials/aimlcs229/transcripts/MachineLearning-Lecture01.pdf) from Andrew Ng's famous CS229 course! These documents are the result of automated transcription so words and sentences are sometimes split unexpectedly.

In [ ]:
```
# The course will show the pip installs you would need to install packages c
# These packages are already installed on this platform and should not be ru
#! pip install pypdf
```

In [ ]:
```
from langchain.document_loaders import PyPDFLoader
loader = PyPDFLoader("docs/cs229_lectures/MachineLearning-Lecture01.pdf")
pages = loader.load()
```

Each page is a `Document`.

A `Document` contains text (`page_content`) and `metadata`.

In [ ]:
```
len(pages)
```

In [ ]:
```
page = pages[0]
```

In [ ]:
```
print(page.page_content[0:500])
```

In [ ]:
```
page.metadata
```

YouTube

In [ ]:
```
from langchain.document_loaders.generic import GenericLoader
from langchain.document_loaders.parsers import OpenAIWhisperParser
from langchain.document_loaders.blob_loaders.youtube_audio import YoutubeAuc
```

In [ ]:
```
# ! pip install yt_dlp
# ! pip install pydub
```

Note: This can take several minutes to complete.

In [ ]:
```
url="https://www.youtube.com/watch?v=jGwO_UgTS7I"
save_dir="docs/youtube/"
loader = GenericLoader(
    YoutubeAudioLoader([url],save_dir),
    OpenAIWhisperParser()
)
docs = loader.load()
```

```
In [ ]: docs[0].page_content[0:500]
```

URLs

```
In [ ]: from langchain.document_loaders import WebBaseLoader

        loader = WebBaseLoader("https://github.com/basecamp/handbook/blob/master/37s
```

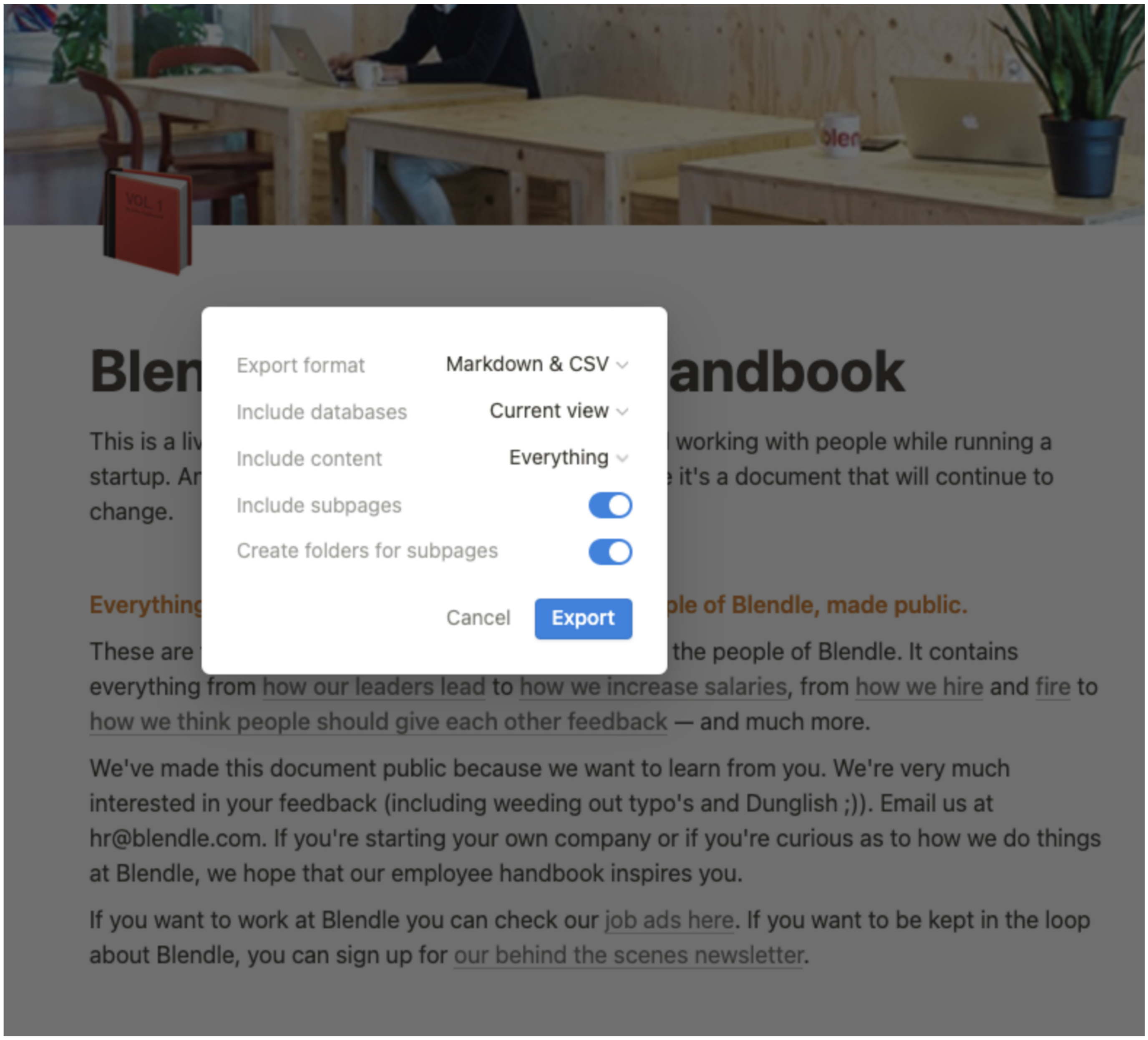
```
In [ ]: docs = loader.load()
```

```
In [ ]: print(docs[0].page_content[:500])
```

Notion

Follow steps here (https://python.langchain.com/docs/modules/data_connection/document_loaders/integrations/notion) for an example Notion site such as this one (https://yolospace.notion.site/Blendle-s-Employee-Handbook-e31bff7da17346ee99f531087d8b133f):

- Duplicate the page into your own Notion space and export as `Markdown / CSV`.
- Unzip it and save it as a folder that contains the markdown file for the Notion page.


Blen andbook

This is a liv working with people while running a startup. Ar it's a document that will continue to change.

Everything ple of Blendle, made public.

These are the people of Blendle. It contains everything from how our leaders lead to how we increase salaries, from how we hire and fire to how we think people should give each other feedback — and much more.

We've made this document public because we want to learn from you. We're very much interested in your feedback (including weeding out typo's and Dunglish ;)). Email us at hr@blendle.com. If you're starting your own company or if you're curious as to how we do things at Blendle, we hope that our employee handbook inspires you.

If you want to work at Blendle you can check our job ads here. If you want to be kept in the loop about Blendle, you can sign up for our behind the scenes newsletter.

Export format: Markdown & CSV
Include databases: Current view
Include content: Everything
Include subpages: on
Create folders for subpages: on

```
In [ ]: from langchain.document_loaders import NotionDirectoryLoader
        loader = NotionDirectoryLoader("docs/Notion_DB")
        docs = loader.load()
```

```
In [ ]: print(docs[0].page_content[0:200])
```

```
In [ ]: docs[0].metadata
```

```
In [ ]:
```