Starter Tutorial (OpenAI)

This is our famous "5 lines of code" starter example using OpenAI.

> **Tip**
>
> Make sure you've followed the installation steps first.

> **Tip**
>
> Want to use local models? If you want to do our starter tutorial using only local models, check out this tutorial instead.

Download data

This example uses the text of Paul Graham's essay, "What I Worked On". This and many other examples can be found in the `examples` folder of our repo.

The easiest way to get it is to download it via this link and save it in a folder called `data`.

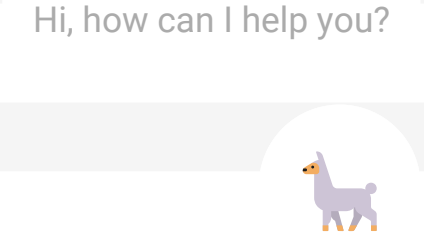
Set your OpenAI API key

LlamaIndex uses OpenAI's `gpt-3.5-turbo` by default. Make sure your API key is available to your code by setting it as an environment variable. In MacOS and Linux, this is the command:

```
export OPENAI_API_KEY=XXXXX
```

and on Windows it is

```
set OPENAI_API_KEY=XXXXX
```



Load data and build an index

In the same folder where you created the `data` folder, create a file called `starter.py` file with the following:

```
from llama_index.core import VectorStoreIndex, SimpleDirectoryReader

documents = SimpleDirectoryReader("data").load_data()
index = VectorStoreIndex.from_documents(documents)
```

This builds an index over the documents in the `data` folder (which in this case just consists of the essay text, but could contain many documents).

Your directory structure should look like this:

```
├── starter.py
└── data
    └── paul_graham_essay.txt
```

Query your data

Add the following lines to `starter.py`

```
query_engine = index.as_query_engine()
response = query_engine.query("What did the author do growing up?")
print(response)
```

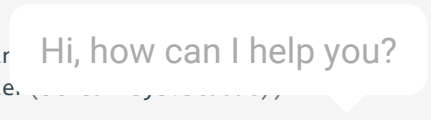
This creates an engine for Q&A over your index and asks a simple question. You should get back a response similar to the following: `The author wrote short stories and tried to program on an IBM 1401.`

Viewing Queries and Events Using Logging

Want to see what's happening under the hood? Let's add some logging. Add these lines to the top of `starter.py`:

```
import logging
import sys

logging.basicConfig(stream=sys.stdout, level=loggir
logging.getLogger().addHandler(logging.StreamHandle
```



You can set the level to `DEBUG` for verbose output, or use `level=logging.INFO` f

Storing your index

By default, the data you just loaded is stored in memory as a series of vector embeddings. You can save time (and requests to OpenAI) by saving the embeddings to disk. That can be done with this line:

```
index.storage_context.persist()
```

By default, this will save the data to the directory `storage`, but you can change that by passing a `persist_dir` parameter.

Of course, you don't get the benefits of persisting unless you load the data. So let's modify `starter.py` to generate and store the index if it doesn't exist, but load it if it does:

```
import os.path
from llama_index.core import (
    VectorStoreIndex,
    SimpleDirectoryReader,
    StorageContext,
    load_index_from_storage,
)

# check if storage already exists
PERSIST_DIR = "./storage"
if not os.path.exists(PERSIST_DIR):
    # load the documents and create the index
    documents = SimpleDirectoryReader("data").load_data()
    index = VectorStoreIndex.from_documents(documents)
    # store it for later
    index.storage_context.persist(persist_dir=PERSIST_DIR)
else:
    # load the existing index
    storage_context = StorageContext.from_defaults(persist_dir=PERSIST_DIR)
    index = load_index_from_storage(storage_context)

# Either way we can now query the index
query_engine = index.as_query_engine()
response = query_engine.query("What did the author do growing up?")
print(response)
```

Now you can efficiently query to your heart's content! But this is just the beginning of what you can do with LlamaIndex.

Tip

Hi, how can I help you?

- learn more about the high-level concepts.
- tell me how to customize things.
- curious about a specific module? check out the component guides.