**Retrieval Augmented Generation (RAG)**

Useful Links:

1. <https://www.youtube.com/watch?v=T-D1OfcDW1M>
2. <https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/>
3. https://towardsdatascience.com/intro-to-llm-agents-with-langchain-when-rag-is-not-enough-7d8c08145834

Notes:

Create a folder for your project in your cdrive.

Open VSCode > open folder

Terminal(Top menu bar) > New Terminal

1. **Setup installation**
2. Create a virtual environment: python -m venv ai
3. Activate Environment: ./ai/Scripts/activate
4. (Use deactivate to deactivate the environment)
5. Install python packages: pip install llama-index pypdf python-dotenv pandas

(Pandas for reading csv files, pypdf to read pdf, llama-index to setup the index, python-dotenv to load in some environment variables)

1. **Access Data**
2. Download csv from the Kaggle (<https://www.kaggle.com/datasets/joebeachcapital/world-population-by-country-2023>)
3. Download the pdf from wikipidea (left corner *tools > Download as Pdf*)
4. Create a folder data and places these two files there
5. Additionally, create a notes.txt in the same folder(data). Overall we have 3 files now.
6. **Fetching the OpenAI API key**
7. Create*.env* in the root folder(project\_rag).
8. Browser > <https://platform.openai.com/api-keys> > Create New Secret Key > Give a name and copy the key.
9. Go to .env and write *OpenAI\_API\_KEY = sk-proj-HBcw4MaxY1HO79EhIjlhT3BlbkFJXSTfAPci6TtoEgLOYuZi*
10. **Querying over pandas data**

* Pandas is a famous data science library in python that allows us to read structured data(.csv)

1. Create a min.py in the root folder(project\_rag).
2. Activate the environment variable file (.env). Simply use load\_dotenv.
3. Load the data files(Specify the paths through *os.path.join*) and read the data(*into pandas df*).
4. Create the query engine
5. Create a new file, prompts.py. This file consists of instruction\_str and new\_prompt which tells the query engine what to do with python data.
6. Instantiate the instruction\_str and new\_prompt in main.py
7. **Building agent capabilities**
8. Create the notes engine. Make a new file in the root folder namely notes\_engine.py, write a python function that can be executed by our model.
9. The function simply checks if the file exists if not, it create else it appends the notes
10. Also specify the tool description so that the agent understands which tools to choose from
11. Go to main and create a pipeline for all the tools agent can use.
12. Create a model and agent
13. **Querying Unstructured data**
14. Import all the necessary libraries, here we are gonna use VectorStoreIndex and Embeddings to access and query the unstructured data(pdf)
15. Load the data file

Errors

1. **Activating the environment**

**Issue\_01:**

PS C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag> .\ai\Scripts\activate

>>

.\ai\Scripts\activate : File C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag\ai\Scripts\Activate.ps1 cannot be loaded because running scripts is disabled on

this system. For more information, see about\_Execution\_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.

At line:1 char:1

+ .\ai\Scripts\activate

+ ~~~~~~~~~~~~~~~~~~~~~

+ CategoryInfo : SecurityError: (:) [], PSSecurityException

+ FullyQualifiedErrorId : UnauthorizedAccess

Solution, run this command: **Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned**

**Issue\_02:**

./ai/bin/activate : The term './ai/bin/activate' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the

name, or if a path was included, verify that the path is correct and try again.

At line:1 char:1

+ ./ai/bin/activate

+ ~~~~~~~~~~~~~~~~~

+ CategoryInfo : ObjectNotFound: (./ai/bin/activate:String) [], CommandNotFoundException

+ FullyQualifiedErrorId : CommandNotFoundException

Solution:

>> cd ai

>> cd Scripts

>>.\activate

Notes:

After creating an venv, a new folder is created (ai, in this case). Hence navigate to that folder and then activate your environment.

1. **Querying over Pandas data**

Issue\_03:

(ai) PS C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag> python main.py

Traceback (most recent call last):

File "C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag\main.py", line 4, in <module>

from llama\_index.query\_engine import PandasQueryEngine

ModuleNotFoundError: No module named 'llama\_index.query\_engine'

Solution:

llama\_index.query\_engine import PandasQueryEngine

instead use the below one,

from llama\_index.experimental.query\_engine import PandasQueryEngine

also use from llama\_index.core import PromptTemplate as MyPromptTemplate in prompts.py

Issue\_04:

openai.RateLimitError: Error code: 429 - {'error': {'message': 'You exceeded your current quota, please check your plan and billing details. For more information on this error, read the docs: https://platform.openai.com/docs/guides/error-codes/api-errors.', 'type': 'insufficient\_quota', 'param': None, 'code': 'insufficient\_quota'}}

Solution:

Simply purchase the $10 plan.

1. **Building agent capabilities**

Issue\_04:

File "C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag\note\_engine.py", line 5, in <module>

from llama\_index.tools import FunctionTool

ModuleNotFoundError: No module named 'llama\_index.tools'

Solution:

Use this. from llama\_index.core.tools import FunctionTool

Issue\_05:

(ai) PS C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag> python main.py

Traceback (most recent call last):

File "C:\Users\cvabh\OneDrive\Desktop\Resume\project\_rag\main.py", line 8, in <module>

from llama\_index.agent import ReActAgent

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

ImportError: cannot import name 'ReActAgent' from 'llama\_index.agent' (unknown location)

Solution:

Use this, from llama\_index.core.agent import ReActAgent

from llama\_index.llms.openai import OpenAI

Instead of this,

from llama\_index.agent import ReActAgent

from llama\_index.llms import OpenAI

Summary

Created an advanced Artificial Intelligence agent using RAG (Retrieval Augmented various Generation) framework.

Conducted analysis on both structured and unstructured data for the agent’s use by studying llama index libraries.

Crafted complex prompts to guide the model on what tasks to perform or what information to search for. Examined the operations and functionalities of vectorbases and embeddings in detail, delving into their mechanisms and how they contribute to various natural language processing tasks.

Developed tools to facilitate the agent’s analysis of diverse data formats, enabling retrieval of relevant content based on user prompts. Additionally implemented a note- taking system to allow the agent to write important information to a text file, providing users with access to crucial data at any time.