Local RAG Notes

why use rag

- **Why RAG locally?**
  - ◇ query with context
    - ■ input → process → output
      -improve output by feeding more relevant input
  - ◇ no gate keepers
  - ◇ privacy

how it works

- **How RAG Works**
  - ◇ stands for: Retrieval-Augmented Generation
  - ◇ Two LLM's
    - ■ chat LLM
    - ■ embedding LLM

  - ◇ Vocabulary:
    - ■ embedding
      - creating some numerical representation of data
      - usually done by converting data into vectors

    - ■ indexing:
      - storing the vectors in a way that makes searching easier
      - perhaps storing similar vectors next to each other

    - ■ semantic similarity
      - how close two vectors match
      - comparison is done by comparing direction of vectors
      - specifically it looks at the cosine similarity
        → 1 means vectors are same
        → -1 means vectors are opposite
        → 0 means vectors are perpendicular, no relationship

  - ◇ steps
    - ■ **INGEST DATA**
      - document is vectorized
        → document is turned into chunks
        → each chunk is turned into a vector
        → each chunk gets a vector id
        → done by embedding LLM

      - data is stored in a db for each chunk
        → key: is id of vector chunk
        → value: is chunk

■ **QUERY DATA**
- user submits a query
- query is vectorized
- db **retrieves** chunks with semantic similarity closest to 1
  → these chunks are the most related to query
- query is **augmented** with the most related chunks
- reponse is **generated** from augmented-query
  → done by chat LLM

download models

• **Download LLM model**
◇ download chat LLM
■ Qwen: https://huggingface.co/Qwen/Qwen2-7B
■ files an version → download 3 GB model
◇ download embedding LLM
■ Nomic: https://huggingface.co/nomic-ai/nomic-embed-text-v1.5-GGUF
■ files and versions → downloaded 274 MB model

• place models in a folder called models

download pdf file

• **Download Test Pdf Input File**

```
mkdir files
cd files
wget https://oral.history.ufl.edu/wp-content/uploads/sites/15/A-Peoples-History-of-Gainesville-Timeline-.pdf
```

create model file

• create a model file for the chat and embedding models
◇ Ollama is a wrapper for Llama.cpp
◇ therefore it only works with GGUF file formats
◇ GGUF must split it up into pieces (helps performance)
■ computer won't have to load entire LLM in RAM

```
# inside models/NomicModelfile
FROM nomic-embed-text-v1.f16.gguf
```

```
# inside models/QwenModelfile
FROM qwen2-7b-instruct-q2_k.gguf
```

start docker

- **Create Docker-Compose file**

```
# inside docker-compose.yml

services:
 ollama:
   image: ollama/ollama
   ports:
    - "11434:11434"
   volumes:
    - ollama_data:/root/.ollama
    - ./models:/models

   environment:
    - OLLAMA_KEEP_ALIVE=5m  # Optional: controls how long models stay loaded in memory

 chroma:
   image: chromadb/chroma
   ports:
    - "8000:8000"
   environment:
    - IS_PERSISTENT=TRUE
    - PERSIST_DIRECTORY=/chroma/chroma
   volumes:
    - chroma_data:/chroma

volumes:
 ollama_data:
 chroma_data:
```

- **Create Container**

```
docker-compose up -d
```

register models

- **Register Models**
  ◇ register models by running model files

```
# registers Qwen Model
docker exec -it rag-ollama-1 ollama create qwenhd -f /models/QwenModelfile

# registers Nomic Model
docker exec -it rag-ollama-1 ollama create nomichd -f /models/NomicModelfile
```

create rag code

- **Create RAG code**
  ◇ Create code to ingest documents with Embedding LLM

```
# inside ingest.py

from langchain_community.document_loaders import WebBaseLoader, PyPDFLoader
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain_chroma import Chroma
from langchain_ollama import OllamaEmbeddings

# Initialize Ollama embeddings
embeddings = OllamaEmbeddings(model="nomic-embed-text")

# Load Web documents
loader = WebBaseLoader([
"https://en.wikipedia.org/wiki/Gainesville,_Florida",
"https://en.wikipedia.org/wiki/University_of_Florida",
])
web_docs = loader.load()

# Load Pdf documents
# each page is a document
pdf_path = "files/A-Peoples-History-of-Gainesville-Timeline-.pdf"
pdf_loader = PyPDFLoader(pdf_path)
pdf_docs = pdf_loader.load()


# Combine all documents
all_docs = web_docs + pdf_docs
print(f"Total documents: {len(all_docs)}")



# Split documents
text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=50)
splits = text_splitter.split_documents(all_docs)

# Create vector store
vectorstore = Chroma.from_documents(
   documents=splits,
   embedding=embeddings,
   persist_directory="chroma"
)

print("Data ingested successfully!")
```

◇ Create code to query Chat LLM

```
# inside query.py

from langchain_chroma import Chroma
from langchain_ollama import OllamaEmbeddings, OllamaLLM
from langchain.chains import RetrievalQA

# Initialize components
embeddings = OllamaEmbeddings(model="nomic-embed-text")
llm = OllamaLLM(model="qwenhd")

vectorstore = Chroma(
   embedding_function=embeddings,
   persist_directory="chroma"
)

# Create retriever
# returns top three document chunks
retriever = vectorstore.as_retriever(search_kwargs={"k": 3})
```

```
# Create RAG chain
qa_chain = RetrievalQA.from_chain_type(
    llm=llm,
    chain_type="stuff",
    retriever=retriever
)

# Query
while True:
    query = input("\nEnter your question (or 'quit' to exit): ")
    if query.lower() == 'quit':
        break

    result = qa_chain.invoke({"query": query})
    print()
    print(result["result"])
```

create python env

- **Create requirements.txt**

```
chromadb>=0.4.22
ollama>=0.1.14
langchain>=0.1.0

langchain-chroma==0.2.4
langchain-ollama==0.3.3

sentence-transformers>=2.2.2
beautifulsoup4>=4.12.0  # Required for WebBaseLoader
pypdf
```

- **Create Conda environment**

```
conda create --name ml-rag python=3.12

# activate
conda activate ml-rag

# install requirements.txt
pip install -r requirements.txt
```

run rag code

- **Ingest Documents**

```
export USER_AGENT=agenthd
python ingest.py
```

- **Query Documents**

```
python query.py
```

- Verify it works by asking a question related to documents
  - ◇ ask these questions:
    - Where is Gainesville located?
    - What sports are played in Gainesville?
    - When was Gainesville founded?
    - What is the Matrix?