```
pip install gradio langchain_openai pinecone
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.11/dist-packages (5.23.1)
Requirement already satisfied: langchain_openai in /usr/local/lib/python3.11/dist-packages (0.3.10)
Requirement already satisfied: pinecone in /usr/local/lib/python3.11/dist-packages (5.4.2)
Requirement already satisfied: aiofiles<24.0,≥22.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (23.2.1)
Requirement already satisfied: anyio<5.0,≥3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.9.0)
Requirement already satisfied: fastapi<1.0,≥0.115.2 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.115.12
Requirement already satisfied: ffmpy in /usr/local/lib/python3.11/dist-packages (from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.8.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (1.8.0)
Requirement already satisfied: groovy~=0.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.2)
Requirement already satisfied: httpx≥0.24.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub≥0.28.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.29.3
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,≥2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,≥1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (1.26.4)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.10.15)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from gradio) (24.2)
Requirement already satisfied: pandas<3.0,≥1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,≥8.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (11.1.0)
Requirement already satisfied: pydantic≥2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.10.6)
Requirement already satisfied: pydub in /usr/local/lib/python3.11/dist-packages (from gradio) (0.25.1)
Requirement already satisfied: python-multipart≥0.0.18 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.0.2
Requirement already satisfied: pyyaml<7.0,≥5.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (6.0.2)
Requirement already satisfied: ruff≥0.9.3 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.11.2)
Requirement already satisfied: safehttpx<0.2.0,≥0.1.6 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.10.0)
Requirement already satisfied: starlette<1.0,≥0.40.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.46.1)
Requirement already satisfied: tomlkit<0.14.0,≥0.12.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.13.2
Requirement already satisfied: typer<1.0,≥0.12 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.15.2)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.12.2)
Requirement already satisfied: uvicorn≥0.14.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.34.0)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.8.0→gradio) (2
Requirement already satisfied: websockets<16.0,≥10.0 in /usr/local/lib/python3.11/dist-packages (from gradio-client==1
Requirement already satisfied: langchain-core<1.0.0,≥0.3.48 in /usr/local/lib/python3.11/dist-packages (from langchain_
Requirement already satisfied: openai<2.0.0,≥1.68.2 in /usr/local/lib/python3.11/dist-packages (from langchain_openai)
Requirement already satisfied: tiktoken<1,≥0.7 in /usr/local/lib/python3.11/dist-packages (from langchain_openai) (0.9
Requirement already satisfied: certifi≥2019.11.17 in /usr/local/lib/python3.11/dist-packages (from pinecone) (2025.1.3
Requirement already satisfied: pinecone-plugin-inference<4.0.0,≥2.0.0 in /usr/local/lib/python3.11/dist-packages (from
Requirement already satisfied: pinecone-plugin-interface<0.0.8,≥0.0.7 in /usr/local/lib/python3.11/dist-packages (from
Requirement already satisfied: python-dateutil≥2.5.3 in /usr/local/lib/python3.11/dist-packages (from pinecone) (2.8.2
Requirement already satisfied: tqdm≥4.64.1 in /usr/local/lib/python3.11/dist-packages (from pinecone) (4.67.1)
Requirement already satisfied: urllib3≥1.26.0 in /usr/local/lib/python3.11/dist-packages (from pinecone) (2.3.0)
Requirement already satisfied: idna≥2.8 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,≥3.0→gradio) (3.1
Requirement already satisfied: sniffio≥1.1 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,≥3.0→gradio) (
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-packages (from httpx≥0.24.1→gradio) (1
Requirement already satisfied: h11<0.15,≥0.13 in /usr/local/lib/python3.11/dist-packages (from httpcore==1.*→httpx≥0
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub≥0.28.1→gradio
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from huggingface-hub≥0.28.1→gradio
Requirement already satisfied: langsmith<0.4,≥0.1.125 in /usr/local/lib/python3.11/dist-packages (from langchain-core<
Requirement already satisfied: tenacity≠8.4.0,<10.0.0,≥8.1.0 in /usr/local/lib/python3.11/dist-packages (from langcha
Requirement already satisfied: jsonpatch<2.0,≥1.33 in /usr/local/lib/python3.11/dist-packages (from langchain-core<1.0
Requirement already satisfied: distro<2,≥1.7.0 in /usr/local/lib/python3.11/dist-packages (from openai<2.0.0,≥1.68.2-
Requirement already satisfied: jiter<1,≥0.4.0 in /usr/local/lib/python3.11/dist-packages (from openai<2.0.0,≥1.68.2→
Requirement already satisfied: pytz≥2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,≥1.0→gradio) (
Requirement already satisfied: tzdata≥2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,≥1.0→gradio)
Requirement already satisfied: annotated-types≥0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic≥2.0→g
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-packages (from pydantic≥2.0→gra
Requirement already satisfied: six≥1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil≥2.5.3→pineco
Requirement already satisfied: regex≥2022.1.18 in /usr/local/lib/python3.11/dist-packages (from tiktoken<1,≥0.7→lang
```

```
!pip install torch transformers sentence-transformers datasets
!pip install keybert pymupdf pinecone-client pdfplumber
!pip install langchain langgraph langchain_community langchain_pinecone langchain_huggingface

!pip uninstall numpy
!pip install numpy==1.25.0
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.11/dist-packages (2.6.0+cu124)
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.50.0)
Requirement already satisfied: sentence-transformers in /usr/local/lib/python3.11/dist-packages (3.4.1)
Requirement already satisfied: datasets in /usr/local/lib/python3.11/dist-packages (3.4.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from torch) (3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.11/dist-packages (from torch) (4.12.
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from torch) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from torch) (2024.12.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torc
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch)
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in /usr/local/lib/python3.11/dist-packages (from torch) (9.1
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in /usr/local/lib/python3.11/dist-packages (from torch) (12
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in /usr/local/lib/python3.11/dist-packages (from torch) (11.
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in /usr/local/lib/python3.11/dist-packages (from torch) (
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in /usr/local/lib/python3.11/dist-packages (from torch) (
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in /usr/local/lib/python3.11/dist-packages (from torch)
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.11/dist-packages (from torch) (0
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist-packages (from torch) (2.21.5
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch) (12.4
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch) (
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-packages (from torch) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-packages (from torch) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch
Requirement already satisfied: huggingface-hub<1.0,>=0.26.0 in /usr/local/lib/python3.11/dist-packages (from transforme
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.1
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers) (0
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (1.
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (1.14.1)
Requirement already satisfied: Pillow in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (11.1.0)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.70.16
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datasets) (3.10.11)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datase
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (25.3.
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (
Requirement already satisfied: yarl<2.0,>=1.12.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->trans
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (3
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->transforme
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->transforme
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from jinja2->torch) (3.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2025.
```

```python
from langchain_community.document_loaders import PyMuPDFLoader
from langchain_text_splitters import TokenTextSplitter
from langchain.text_splitter import RecursiveCharacterTextSplitter
import pdfplumber
from uuid import uuid4
from langchain_core.documents import Document
from langchain_huggingface import HuggingFaceEmbeddings
from keybert import KeyBERT
import re
import pymupdf
import datetime
from pinecone import Pinecone
from pinecone import Pinecone, ServerlessSpec
from langchain_pinecone import PineconeVectorStore
import os
import openai
import json
from langchain_openai import ChatOpenAI,OpenAI
from langgraph.graph import StateGraph,MessagesState
from typing import List, Optional
from langchain.prompts import ChatPromptTemplate,SystemMessagePromptTemplate,AIMessagePromptTemplate,HumanMessagePromptTempla
from pydantic import BaseModel, Field
from langchain.output_parsers import PydanticOutputParser
from langgraph.graph import StateGraph, START, END
```

```
import re
```

```
    Requirement already satisfied: cffi≥1.12 in /usr/local/lib/python3.11/dist-packages (from cryptography≥36.0.0→pdfmin
```

```
import os

os.environ["SERPAPI_API_KEY"] = "650421ed0daa29addfe17bed6601a809a70d0aacea2a8c804397b4c61020ff29"
os.environ['PINECONE_API_KEY'] = "pcsk_5qAp4G_QyLtWLYcjNsy23CDhhgkH4cAvesEhYHMKWTFkC32i8SyzjrDLsWhnLPcWS97PUm"

serpapi_key = os.getenv("SERPAPI_API_KEY")
pinecone_api_key = os.getenv("PINECONE_API_KEY")
```

```
    Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch
```

```
from pinecone import Pinecone, ServerlessSpec

pinecone_api_key = os.getenv("PINECONE_API_KEY")

pc = Pinecone(api_key=pinecone_api_key)
pc.create_index(
    name="profilestore",
    dimension=768 , # Replace with your model dimensions
    metric="cosine", # Replace with your model metric
    spec=ServerlessSpec(
        cloud="aws",
        region="us-east-1"
    )
)
```

```
    Requirement already satisfied: safetensors≥0.4.3 in /usr/local/lib/python3.11/dist-packages (from transformers<5.0.0,≥
```

```
from huggingface_hub import login

token = "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"
login(token=token, add_to_git_credential=True)  # ADD YOUR TOKEN HERE
```

```
    Requirement already satisfied: langchain_community in /usr/local/lib/python3.11/dist-packages (0.3.20)
```

```
pc= Pinecone(api_key= pinecone_api_key)
index = pc.Index("profilestore")
embedding_model = HuggingFaceEmbeddings(model_name = 'hshashank06/final-regulatory-policy')
vector_store = PineconeVectorStore(index=index, embedding= embedding_model)
key_model = KeyBERT();
```

```
    Requirement already satisfied: SQLAlchemy<3,≥1.4 in /usr/local/lib/python3.11/dist-packages (from langchain) (2.0.39)
    Requirement al                          /usr/local/lib/python3.11/dist-packages (from langchain) (2.32.3)
    modules.json: 100%                       349/349 [00:00<00:00, 34.7kB/s]
    Requirement already satisfied: PyYAML≥5.3 in /usr/local/lib/python3.11/dist-packages (from langchain) (6.0.2)
    Requirement already satisfied                      10 in /usr/local/lib/python3.11/dist-packages (from lang
    config_sentence_transformers.json: 100%             116/116 [00:00<00:00, 13.5kB/s]
    Requirement already satisfied: langgraph-prebuilt<0.2,≥0.1.1 in /usr/local/lib/python3.11/dist-packages (from langgrap
    Requirement                              0.1.42 in /usr/local/lib/python3.11/dist-packages (from langgraph)
    README.md: 100%                          10.5k/10.5k [00:00<00:00, 250kB/s]
    Requirement already satisfied: xxhash<4.0.0,≥3.5.0 in /usr/local/lib/python3.11/dist-packages (from langgraph) (3.5.0)
    sentence_bert_config.json: 100%                     53.0/53.0 [00:00<00:00, 1.78kB/s]
    Requirement already sat                             sr/local/lib/python3.11/dist-packages (from langchain_communi
    Requirement already satisfied: tenacity≠8.4.0,<10,≥8.1.0 in /usr/local/lib/python3.11/dist-packages (from langchain_c
    config.json: 100%                         612/612 [00:00<00:00, 9.71kB/s]
    Requirement a                            0.7,≥0.5.7 in /usr/local/lib/python3.11/dist-packages (from langchain_
    Requirement already satisfied: pydantic-settings<3.0.0,≥2.4.0 in /usr/local/lib/python3.11/dist-packages (from langcha
    model.safetensors: 100%                   90.9M/90.9M [00:00<00:00, 229MB/s]
    Requirement already                       .0 in /usr/local/lib/python3.11/dist-packages (from langchain_commu
    Requirement already                          /local/lib/python3.11/dist-packages (from langchain_community) (
    tokenizer_config.json: 100%                   350/350 [00:00<00:00, 29.8kB/s]
    Requirement already satisfied: pinecone<6.0.0,≥5.4.0 in /usr/local/lib/python3.11/dist-packages (from langchain_pineco
    Requirement                              <1.0.0,≥0.3.7 in /usr/local/lib/python3.11/dist-packages (from langchain_
    vocab.txt: 100%                           232k/232k [00:00<00:00, 533kB/s]
    Requirement already satisfied: huggingface-hub≥0.23.0 in /usr/local/lib/python3.11/dist-packages (from langchain_huggi
    Requirement                              ers≥2.6.0 in /usr/local/lib/python3.11/dist-packages (from langchain_
    tokenizer.json: 100%                      466k/466k [00:00<00:00, 17.1MB/s]
    Requirement already satisfied: tokenizers≥0.19.1 in /usr/local/lib/python3.11/dist-packages (from langchain_huggingfac
    special_tokens_map.json: 100%                     112/112 [00:00<00:00, 12.9kB/s]
    Requirement already sa                            sr/local/lib/python3.11/dist-packages (from langchain_huggingf
    Requirement already satisfied: aiohappyeyeballs≥2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,≥
    config.json: 100%                         190/190 [00:00<00:00, 21.8kB/s]
    Requirement a                            in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,≥3.8.3→
    Requirement already satisfied: attrs≥17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,≥3.8.3→la
    Requirement already satisfied: frozenlist≥1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,≥3.8.3
```

```
os.environ["OPENAI_API_KEY"] = 'sk-proj-KP0_jRkwjRJVGF5cm4cvIzysl-6sdf0d3QxoN2pEZYqk6SeUMM3VA-ROsZTnKwF-vjxmvPgZTZT3BlbkFJhoI

openai.api_key = os.getenv("OPENAI_API_KEY")
```

```
    Requirement already satisfied: fsspec≥2023.5.0 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub≥0.23.
```

```
text = """GENERAL INSTRUCTIONS ......................................................................................
WHO MUST REPORT ..................................................................................................
WHERE TO SUBMIT THE REPORTS ......................................................................................
WHEN TO SUBMIT THE REPORTS .......................................................................................
HOW TO PREPARE THE REPORTS: ......................................................................................
Schedule A – Retail ..............................................................................................
A.1 – INTERNATIONAL AUTO LOAN .....................................................................................
A.2 – US AUTO LOAN ................................................................................................
```

```
A.3 – INTERNATIONAL CREDIT CARD ........................................................................................
A.4 – INTERNATIONAL HOME EQUITY ........................................................................................
A.5 – INTERNATIONAL FIRST LIEN MORTGAGE ................................................................................
A.6 – INTERNATIONAL OTHER CONSUMER SCHEDULE ..........................................................................
A.7 – US OTHER CONSUMER ................................................................................................
A.8 – INTERNATIONAL SMALL BUSINESS .....................................................................................
A.9 – US SMALL BUSINESS.................................................................................................
A.10 – STUDENT LOAN ....................................................................................................
Schedule B—Securities ....................................................................................................
B.1—SECURITIES 1 (“MAIN SCHEDULE”) .....................................................................................
B.2—SECURITIES 2 (“INVESTMENT SECURITIES WITH DESIGNATED ACCOUNTING HEDGES”) ..............................................
Schedule C—Regulatory Capital Instruments ................................................................................
C.1—REGULATORY CAPITAL AND SUBORDINATED DEBT INSTRUMENTS AS OF QUARTER END ...............................................
C.2—REGULATORY CAPITAL AND SUBORDINATED DEBT INSTRUMENT REPURCHASES/REDEMPTIONS DURING QUARTER ...................... 63
C.3 – REGULATORY CAPITAL AND SUBORDINATED DEBT INSTRUMENTS ISSUANCES DURING QUARTER .......................................
Schedule D—Regulatory Capital .............................................................................................
Schedule E—Operational Risk ...............................................................................................
E.1—OPERATIONAL LOSS HISTORY ...............................................................................................
E.2.  INTERNAL BUSINESS LINE ...............................................................................................
E.3.  UNIT-OF-MEASURE (UOM) ...............................................................................................
E.4.  THRESHOLD INFORMATION ...............................................................................................
E.5—LEGAL RESERVES FREQUENCY ...............................................................................................
Schedule F—Trading .........................................................................................................
GLOSSARY ...................................................................................................................
REGIONAL GROUPINGS .........................................................................................................
F.1—EQUITY BY GEOGRAPHY ......................................................................................................
F.2—EQUITY SPOT-VOL GRID .....................................................................................................
F.3—OTHER EQUITY .............................................................................................................
F.4—FX SPOT SENSITIVITIES ....................................................................................................
F.5—FX VEGA ..................................................................................................................
F.6—RATES DV01 ...............................................................................................................
F.7—RATES VEGA ...............................................................................................................
F.8—OTHER RATES ..............................................................................................................
F.9—ENERGY ...................................................................................................................
F.10—METALS ..................................................................................................................
F.11—AGS & SOFTS .............................................................................................................
F.12—COMMODITY INDICES ........................................................................................................

F.13—COMMODITY SPOT-VOL GRIDS ...................................................................................................
F.14—SECURITIZED PRODUCTS ......................................................................................................
F.15—AGENCIES .................................................................................................................
F.16—MUNIS ....................................................................................................................
F.17—AUCTION RATE SECURITIES (ARS) ..............................................................................................
F.18—CORPORATE CREDIT-ADVANCED ...................................................................................................
F.19—CORPORATE CREDIT-EMERGING MARKETS.............................................................................................
F.20—SOVEREIGN CREDIT .........................................................................................................
F.21—CREDIT CORRELATION .......................................................................................................
F.22—IDR-CORPORATE CREDIT .....................................................................................................
F.23—IDR-JUMP TO DEFAULT ......................................................................................................
F.24—PRIVATE EQUITY ...........................................................................................................
F.25—OTHER FAIR VALUE ASSETS ...................................................................................................
Schedule G—PPNR ...............................................................................................................
G.1—PPNR SUBMISSION WORKSHEET ...................................................................................................
G.2—PPNR NET INTEREST INCOME (NII) WORKSHEET .......................................................................................
G.3—PPNR METRICS ..................................................................................................................
Schedule H—Wholesale Risk .......................................................................................................
H.1 - CORPORATE LOAN DATA SCHEDULE ..................................................................................................
H.2 – COMMERCIAL REAL ESTATE SCHEDULE ................................................................................................
H.3 – LINE OF BUSINESS SCHEDULE ......................................................................................................
H.4 – INTERNAL RISK RATING SCHEDULE ..................................................................................................
Schedule J – Retail Fair Value Option/Held for Sale (FVO/HFS) .........................................................................
Schedule K - Supplemental ...............................................................................................................
Schedule L - Counterparty ...............................................................................................................
Schedule M—Balances ....................................................................................................................
Appendix A:  FR Y-14Q Supporting Documentation ...........................................................................................
SUPPORTING DOCUMENTATION FOR SCHEDULE C – REGULATORY CAPITAL INSTRUMENTS ..................................................................
SUPPORTING DOCUMENTATION FOR SCHEDULE D – REGULATORY CAPITAL .................................................................................
SUPPORTING DOCUMENTATION FOR SCHEDULE L – COUNTERPARTY ......................................................................................
 """
```

```
from google.colab import drive
drive.mount('/content/drive')

folder_path = "/content/drive/My Drive/GenAI/data/"
```

```
Mounted at /content/drive
```

```
# Function to extract headings and page numbers
def extract_headings(text):
    pattern = r"(Schedule [A-Z]—?.*?)\s+(\d+)"  # Match "Schedule A—Retail" and its page number
    matches = re.findall(pattern, text)

    headings = {}
    for i in range(len(matches) - 1):  # Pair each heading with its page range
        heading, start_page = matches[i]
        heading = heading.split(" .")[0].strip()
        next_start_page = matches[i + 1][1]  # Next heading's page
        headings[heading] = (int(start_page), int(next_start_page) - 1)  # Define range

    # Add the last heading separately (it goes till the end of the document)
    last_heading, last_page = matches[-1]
    last_heading = last_heading.split(" ,")[0].strip()
    headings[last_heading] = (int(last_page), None)  # Until the end

    return headings

def generate_meta_data(doc, heading, key_words,content_type):
    metadata = {
        "heading": heading,
        "author": doc.metadata.get("author", "Unknown"),
        "creation_date": doc.metadata.get("creationDate", "Unknown"),
        "modification_date": doc.metadata.get("modDate", "Unknown"),
        "key_words": key_words,
        "content_type":content_type

    }
    return metadata

def key_word_extractor(chunk, num_keywords = 5):
    keywords = key_model.extract_keywords(chunk, keyphrase_ngram_range=(1,2),stop_words="english",top_n=num_keywords)
    return [kw[0] for kw in keywords]


# Function to extract text for each heading
def extract_text_by_heading(pdf_path, headings):
    doc = pymupdf.open(pdf_path)
    heading_text = {}

    with pdfplumber.open(pdf_path) as pdf:
        for heading, (start_page, end_page) in headings.items():
            inner_text = ""

            # Convert 1-based to 0-based index
            for page_num in range(start_page - 1, end_page if end_page else len(pdf.pages)):
                page = pdf.pages[page_num]
                extracted_text = page.extract_text()  # Use extract_text() instead of get_text("text")
                extracted_table = page.extract_tables()
                if extracted_text:  # Handle cases where no text is found
                    inner_text += extracted_text + "\n"
                if extracted_table:
                    first_row = True
                    for table_id, table in enumerate(extracted_table):
                        if not table or len(table) < 2:  # Ensure there's at least a header and one row
                            continue


                        headers = [cell for cell in table[0][:3]]
                        for row_id, row in enumerate(table[1:]):
                            if row:
                                row_text = {headers[i]: row[i] for i in range(len(headers)) if i < len(row) and row[i] is not
                                key_words = key_word_extractor(str(row_text))
                                meta_data = generate_meta_data(doc, heading, key_words, "Table Row")
                                print("TABLE Adding data " + str(row_text) + " With Heading as " + heading + " And keywords a
                                vector_store.add_documents([Document(page_content=str(row_text), metadata=meta_data)],  ids=[


            heading_text[heading] = inner_text

            text_splitter = RecursiveCharacterTextSplitter(chunk_size = 500,chunk_overlap =80 , length_function = len )
            split_pdf_content = text_splitter.create_documents([inner_text])

            for t in split_pdf_content:
                key_words = key_word_extractor(t.page_content)
                meta_data = generate_meta_data(doc, heading,key_words, "Paragraphs")
                print("TEXT PART Adding data " + t.page_content + " With Heading as " + heading + " And keywords are " + str(
                vector_store.add_documents([Document(page_content=t.page_content, metadata=meta_data)], ids = [str(uuid4())])



    return heading_text


#pdf_path = folder_path + "/policy docs/FR_Y-14Q20240331_i.pdf"  # Replace with your PDF path
headings = extract_headings(text)
extract_text_by_heading("test_data_excel.pdf",headings)

# Print results
print(headings)
```

Streaming output truncated to the last 5000 lines.

```
nalID) If the credit facility is not
guaranteed, enter ‘NA’. With Heading as Schedule H—Wholesale Risk And keywords are ['guarantor clcgm300', 'guarantor ide
TEXT PART Adding data nalID) If the credit facility is not
guaranteed, enter ‘NA’.
46 Guarantor Name CLCG9017 Report the guarantor name on the credit facility. Full legal Must not contain a carriage
corporate name is desirable. If the guarantor is an individual(s) return, line feed, comma or any
(GuarantorNam
(Natural Person (s)), do not report the name; instead substitute unprintable character.
e)
with the text: "Individual."
If the credit facility With Heading as Schedule H—Wholesale Risk And keywords are ['guarantor credit', 'guarantor clcg9(
TEXT PART Adding data e)
with the text: "Individual."
If the credit facility
For facilities with multiple guarantors, provide the guarantor name is not guaranteed, enter ‘NA’
for the primary or most substantial guarantor.
47 Guarantor TIN CLCG6191 Report the Taxpayer Identification Number (TIN) assigned to the The 9 digit identification
guarantor by the U.S. Internal Revenue Service (IRS) in the assigned by the Internal
(GuarantorTIN) With Heading as Schedule H—Wholesale Risk And keywords are ['guarantor tin', 'identification guarantor',
TEXT PART Adding data (GuarantorTIN)
administration of tax laws. If the guarantor is an individual(s) Revenue Service for the
(Natural Person(s)), do not report Social Security Number; instead guarantor identified in Field
enter ‘NA’. If, the guarantor does not have a TIN, enter ‘NA’. 45. Allowable forms are either
##-#######,
For facilities with multiple guarantors, provide the TIN assigned to
#########, or
the primary or most substantial guarantor.
‘NA’.
If the credit facility
is not guaranteed, enter ‘NA’ With Heading as Schedule H—Wholesale Risk And keywords are ['guarantor individual', 'guara
TEXT PART Adding data ‘NA’.
If the credit facility
is not guaranteed, enter ‘NA’
Field Name;
Field
(Technical Field MDRM Description Allowable Values
No.
Name)
48 Guarantor CLCGG080 Report the guarantor rating grade from the reporting entity’s Free text indicating the obligor
Internal Risk internal risk rating system. rating grade.
Rating
This is the reporting entity’s probability of default (PD) rating. If the If the credit facility is not With Heading as
TEXT PART Adding data (GuarantorInter reporting entity uses a one-dimensional risk rating system, record guaranteed or i
nalRiskRating) that rating here. does not have a rating, enter
‘NA’
For facilities with multiple guarantors, provide the guarantor rating
grade for the primary or most substantial guarantor.
49 Entity Internal CLCEM300 Report the reporting BHC’s or IHC’s or SLHC’s unique internal Must not contain a carriage Wi
TEXT PART Adding data ID identifier for the entity that is the primary source of repayment for return, line feed, comma
the facility in Field 15 unprintable character.
(EntityInternalI
D) Leave blank if the entity is the
same as the Obligor identified
in Field 2.
50 Entity Name CLCE9017 Report the name of the entity that is the primary source of Must not contain a carriage
repayment for the facility in Field 15. Full legal corporate name is return, line feed, comma or any
```

```
import difflib

def get_best_match(user_input, options):
    user_input = user_input.strip()  # Remove leading/trailing spaces
    best_match = difflib.get_close_matches(user_input, options, n=1, cutoff=0.1)  # Low cutoff for flexibility
    return best_match[0] if best_match else None
```

```
from sentence_transformers import CrossEncoder, SentenceTransformer

def get_context_from_db(user_query, section_heading, content_type):
    model = SentenceTransformer("hshashank06/final-regulatory-policy")
    query_vector = model.encode(user_query, normalize_embeddings = True).tolist()

    top_k = 400 if section_heading != "" else 800

    print("Params passed to index.query" + user_query + section_heading + content_type)

    section_heading = get_best_match(section_heading, headings)
    filter = {}

    if section_heading != "":
      filter["heading"] = { "$eq" : section_heading }
    if content_type != "":
      filter["content_type"] = { "$eq" : content_type }

    pinecone_results = index.query(vector=query_vector, top_k=top_k, include_metadata=True, filter=filter)


    if len(pinecone_results["matches"]) is 0:
      return -1

    # Cross Encoder
    cross_encoder_path = folder_path + 'CrossEncoderModel'
    print('Cross Encoder model exists', os.path.exists(cross_encoder_path))

    cross_encoder = CrossEncoder(cross_encoder_path)
    query_chunk_pairs = [(user_query, doc["metadata"]["text"]) for doc in pinecone_results["matches"]]
    cross_encoder_scores = cross_encoder.predict(query_chunk_pairs)


    # Vector scores
    vector_scores = [doc["score"] for doc in pinecone_results["matches"]]

    final_scores = [cross * 0.5 + vector * 0.5 for cross, vector in zip(cross_encoder_scores, vector_scores)]

    final_results = sorted(zip(pinecone_results["matches"], final_scores), key=lambda x: x[1], reverse=True)[:500]

    return final_results
```

```
<>:22: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:22: SyntaxWarning: "is" with a literal. Did you mean "=="?
<ipython-input-23-5c995a5cd49b>:22: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if len(pinecone_results["matches"]) is 0:
```

```
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate
from langchain_openai import ChatOpenAI,OpenAI


# plan prompt that allows the LLM to plan it's output
plan_prompt = """
I need you to analyze the following chunks of text and generate a structured content plan. Your task is to:

1. **Reconstruct meaningful content** from the provided chunks.
2. **Organize the content into structured sections** with clear headings and subheadings.
3. **Ensure logical flow** so that the extracted chunks form a coherent piece.
4. **Assign word count guidelines** for each section.
5. **No context must be left out at any cost.

Use the following random chunks of text:

{context}

Use the following instructions to expand more on content matching it

{instructions}

### **Content Planning Instructions**
- Extract **relevant themes** from the given chunks.
- Group similar ideas under **clear, structured headings**.
- Ensure **at least 8+ subheadings** for better organization.
- Avoid overly fine splitting; **each section must be meaningful**.
- No context must be left out at any cost.

### **Expected Output Format**
**Heading 1** - Main Point: [Describe the core idea in detail]
🔹 **Sub Heading 1**:  [1000 words]
   - Point 1: [Detailed explanation]
   - Point 2: [Detailed explanation]

🔹 **Sub Heading 2**:  [1000 words]
   - Point 1: [Detailed explanation]
   - Point 2: [Detailed explanation]
…


Each sub heading (Sub Heading 1,2 etc.) must have atleast 1000 words worth of content.
There should be minimum 15 sub headings.
Ensure that the **entire content is structured meaningfully** based on the provided chunks.
The final output should resemble a **well-organized content plan** ready for detailed writing.
"""

write_prompt = """
You are an expert in **rule extraction and content structuring**. Your task is to:
1. **Analyze** the structured content plan.
2. **Extract explicit rules and regulations** from the content structure.
3. **Ensure completeness** — all important regulations must be covered.
4. **Format the extracted rules and regulations properly** for easy implementation.

### **Input Data**
- **Structured Content Plan**:
{plan}

### **Your Task**
1. Identify all **underlying rules, regulations and constraints** present in the content structure.
2. **Break them down** into precise, easy-to-follow rules and policies.
3. Organize the rules and regulatory policies into **clear categories** for better readability.
4. No context must be left out at any cost

### **Expected Output Format**
**Rule / Policy 1**: [Brief Description]
🔹 **Explanation**: [Detailed breakdown of the rule or policy]
🔹 **Example (if applicable)**: [Provide an example if needed]

**Rule / Policy 2**: [Brief Description]
🔹 **Explanation**: [Detailed breakdown of the rule or policy]
🔹 **Example (if applicable)**: [Provide an example if needed]

…

🚀 Ensure that the final rules provide **clear guidance** based on the structured content plan.

"""
llm2 = ChatOpenAI(model="gpt-4", temperature=0.3)
plan_prompt = ChatPromptTemplate([("user", plan_prompt)])
plan_chain = plan_prompt | llm2 | StrOutputParser()

write_prompt = ChatPromptTemplate([("user",write_prompt)])
write_chain = write_prompt | llm2 | StrOutputParser()


def generate_rules(query, context):
    print("Generating rules for given query", query)
    # ═══════════════ PLANNING AGENT ══════════════════
    plan = plan_chain.invoke({"instructions": query,"context": context})

    print("Plan agent executed with plan", plan)

    # ═══════════════ WRITING AGENT ═══════════════════
    plan = plan.strip().replace('\n\n', '\n')
    planning_steps = plan.split('\n')

    print("Writing the rules")
    # text = ""
    # responses = []
    # for idx,step in enumerate(planning_steps):
```

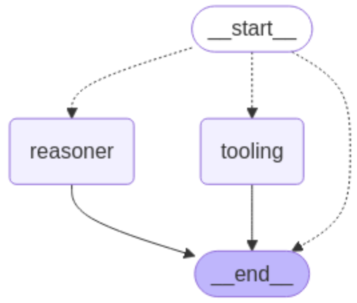
```
    #     # Invoke the write_chain
    #     result = write_chain.invoke({
    #         "instructions": query,
    #         "plan": plan,
    #         "text": text,
    #         "STEP": step
    #     })
    #     responses.append(result)
    #     text += result + '\n\n'

    final_doc = write_chain.invoke({"plan": plan})

    print("Final rules doc generated as", final_doc)
    return final_doc



# OpenAI
import openai
import json
from langchain_openai import ChatOpenAI,OpenAI
from langgraph.graph import StateGraph,MessagesState
from typing import List, Optional
from pydantic import BaseModel
import json
import os
from langchain.prompts import ChatPromptTemplate,SystemMessagePromptTemplate,AIMessagePromptTemplate,HumanMessagePromptTempla
from pydantic import BaseModel, Field
from langchain.output_parsers import PydanticOutputParser
from langgraph.graph import StateGraph, START, END
import json
import re


# Define JSON Structure using Pydantic
class SearchQuery(BaseModel):
    search_heading: Optional[str] = ""
    search_content: Optional[str] = ""
    user_wants_heading: Optional[str] = ""
    user_wants_search_content: Optional[str] = ""
    llm_interactive_output: Optional[str] = ""
    final_check: Optional[str] = ""

class MemoryState(BaseModel):
    messages: list[str]
    query: SearchQuery
    user_input: str
    rules_generated: str


# Initialize LLM (Replace with your API key)
llm1 = ChatOpenAI(model="gpt-4", temperature=0.3)

parser = PydanticOutputParser(pydantic_object=SearchQuery)


def node_1(state : MemoryState) → SearchQuery:
  prompt = f"""\ You are an helpful Financial AI Assistant. You basically have two jobs.
    Given below is an example conversation. Understand the logical intent of the conversation, and act in a manner according
   * When user asks for an input, you must identify if the user is asking something about  **COMPLEX FINANCE, REGULATIONS,PO
   * Example json when the user comes for the first time

    Given below is only an example, you don't need to use the same sentences, but the JSON updates explained below must be fo
    User Query - I want some rules pertaining to regulations for commerical banking
    {{
    "search_heading": ""
    "search_content": ""
    "user_wants_heading": ""
    "user_wants_search_content": ""
    "llm_interactive_output": "Sure, are you looking for documents related to complex finance or policies?"
    "final_check":""
    }}

    (Don't ask the question again and again)
    If the User answers 'YES' or any similar form of yes

    Now you will work with the user and add your messages to llm_interactive_output and fill the JSON eg.
```

```
 User Query - Yes or ( anything similar )
  {{
 "search_heading": ""
 "search_content": ""
 "user_wants_heading": ""
 "user_wants_search_content": ""
 "llm_interactive_output": "Sure Do you have any Search Content or Search Heading in mind? This will help me filter better
 "final_check":""
 }}

 If user say's yes and tells some headings:
 User query - Yes, I want the rules under Schedule A for Auto Loan.
 {{
 "search_heading": "Schedule A"
 "search_content": ""
 "user_wants_heading": "Yes"
 "user_wants_search_content": ""
 "llm_interactive_output": "Sure Do you have any Search Content mind? This will help me filter better. "
 "final_check":""
 }}

 The user Did not specify anything about search_content, so ask the user again. And if the user says no for example, then
 {{
 "search_heading": "Schedule A"
 "search_content": ""
 "user_wants_heading": "Yes"
 "user_wants_search_content": "No"
 "llm_interactive_output": "Okay, so you want to fetch rules from Schedule A for Auto Loan and you don't have any particul
 "final_check":""
 }}

 The when user says "Yes" or something similar
 {{
 "search_heading": "Schedule A"
 "search_content": ""
 "user_wants_heading": "Yes"
 "user_wants_search_content": "No"
 "llm_interactive_output": "Okay, Let me fetch you documents"
 "final_check":"Yes" [Always Yes or No]
 }}

 The search_headings should match the following headings {headings} alteast 70% else you may leave it.
 When asking for the search_headings , display the headings {headings} in a good format

 If the user Says No, update the corresponding variable with No, and dont ask the same question again.
 Make sure you fill user_wants_heading and user_wants_search_content variables in the json.
 This End your conversation, and then a new conversation can start. When a new conversation states always
 1. Check if it is the same user that is conversating with you, based on history
 2. If a new user, then do not let the history mislead you.

 NOTE Search Content Can only be Table Row, Or Paragraph. So if user says Table, Table Row, or Row, the json must have it
* When user asks for something else, you are free to answer according to the user input, and put your interactive outputs
Example :
User - Hi How are you ?
Your output -
{{
 "search_heading": "Schedule A"
 "search_content": ""
 "user_wants_heading": "Yes"
 "user_wants_search_content": "No"
 "llm_interactive_output": "Hello! How can I help you?"
 }}

 ALWAYS GIVE your output as a json BASED on the instructions below. If you have something to say, put it inside the llm_in

The output Instruction are :
{parser.get_format_instructions()}


REQUIRED:
1. You Must always give the output as the JSON specified above
2. You must not ask the same question twice in a row.



"""
print("I am here 1" + str(state.query))
```

```
  messages = [
    {"role": "system", "content": prompt},
    {"role": "user", "content": state.user_input}
  ]

  print("Previous messages" + str(state.messages))

  state.messages = [llm1.invoke(messages + state.messages  + [str({"user":state.user_input})])]
  print("Before before" + str(state.messages[-1].content))
  current_query = json.loads(state.messages[-1].content)
  state.query = current_query
  return state


def check(state: MemoryState):
  print("I am here " + str(state.query))
  searchQuery = state.query
  if(searchQuery.user_wants_heading == "" or searchQuery.user_wants_search_content == ""):
    return "reasoner"
  else:
    return "tooling"


def tooling(state: MemoryState) -> MemoryState:
    print(" Now inside tooling " + str(state))
    requestObject = state.messages[-1]

    match = re.search(r"content='({.*?})'", str(requestObject))
    json_data = {}
    if match:
      json_string = match.group(1)  # Extract JSON string
      json_string = json_string.replace("\\n", "")  # Remove \n
      json_string = json_string.replace("\\'", "'")  # Fix any escaped single quotes

      json_data = json.loads(json_string)  # Convert to dictionary
      print(json.dumps(json_data, indent=2))  # Pretty-print JSON

    query  = json_data["llm_interactive_output"]
    content_type = json_data["search_content"]
    section_heading = json_data["search_heading"]
    print("Tooling node processing:")
    print(f"Query: {query}")
    print(f"Content Type: {content_type}")
    print(f"Section Heading: {section_heading}")

    results = get_context_from_db(query, section_heading, content_type)
    print("Results are " + str(results))
    if results is -1:
      raise Exception("Sorry, some error occured. Will be right back!!!")
    else:
      result = list(results)
      context = ""
      for i in result:
        context = list(i)[0]["metadata"]["text"]
        print(context)
      rules = generate_rules(query, context)
      state.rules_generated = rules
      return state


workflow = StateGraph(MemoryState)

workflow.add_node("reasoner",node_1)
workflow.add_node("tooling",tooling)
workflow.add_conditional_edges(START,check)
workflow.add_edge("tooling",END)
workflow.add_edge("reasoner", END)
```

```
<>:186: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:186: SyntaxWarning: "is" with a literal. Did you mean "=="?
<ipython-input-26-41b7203a4200>:186: SyntaxWarning: "is" with a literal. Did you mean "=="?
  if results is -1:
<langgraph.graph.state.StateGraph at 0x7ba2fdb08cd0>
```

```
final_graph = workflow.compile()
final_graph
```

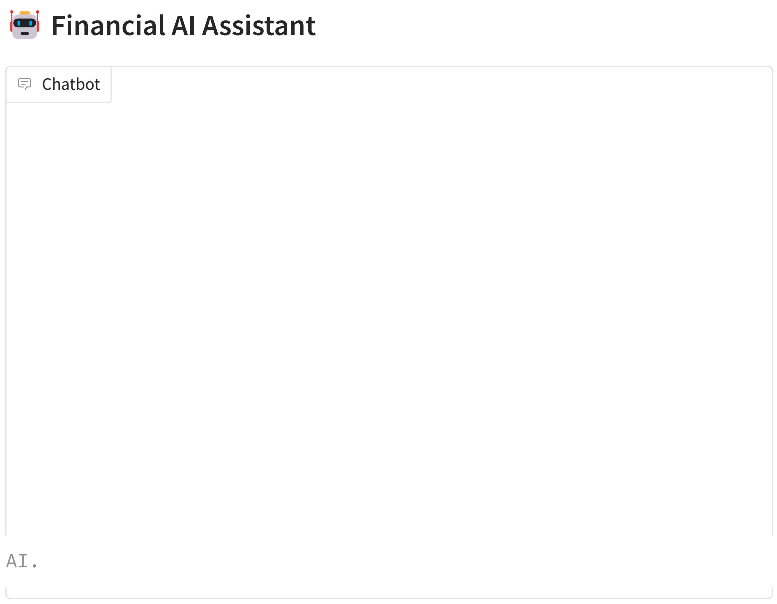


```
import json


# To test the code run this
query = {
    "search_heading": "",
    "search_content": "",
    "user_wants_heading": "",
    "user_wants_search_content": "",
    "llm_interactive_output": "",
    "final_check":""
    }
initial_state = {
        "messages":[],
        "query":query,
        "user_input":"",
        "rules_generated":""
    }

while True:
    user_input = input("You: ")  # Take dynamic input from user
    initial_state["user_input"] = str(user_input)

    response = final_graph.invoke(initial_state)
    print("Response isss " + str(response))


    if(response["rules_generated"] is not None and response["rules_generated"] ≠ ""):
       regulations = response["messages"][-1][-1]
    else:
      initial_state["messages"] = initial_state["messages"] + [ str(response["messages"][0])]
      print("Messages are " + str(initial_state["messages"][-1]))
      initial_state["query"] = json.loads(response["messages"][-1].content)
      print("Content" + str( [response["messages"][-1].content]))
      print(response)
      requestObject = json.loads(response["messages"][0].content)
      if(requestObject["user_wants_search_content"] ≠ "" and requestObject["user_wants_heading"] ≠ "" and requestObject["fi
          final_graph.invoke(initial_state)
```

```
import gradio as gr
import json


# To test the code using UI use this
query = {
    "search_heading": "",
    "search_content": "",
    "user_wants_heading": "",
    "user_wants_search_content": "",
    "llm_interactive_output": "",
    "final_check": ""
}
initial_state = {
    "messages": [],
    "query": query,
    "user_input": "",
```

```python
    "rules_generated": ""
}

def chatbot(user_input, chat_history):
    if chat_history is None:
        chat_history = []

    chat_history.append((user_input, None))  # User messages on right

    if user_input.lower() in ["quit", "exit", "q"]:
        chat_history.append((None, "Goodbye!"))
        return chat_history

    initial_state["user_input"] = user_input
    response = final_graph.invoke(initial_state)

    if(response["rules_generated"] is not None and response["rules_generated"] != ""):
        output = response["rules_generated"]
        chat_history.append((None, output))
    else:
        initial_state["messages"].append(str(response["messages"][0]))
        initial_state["query"] = json.loads(response["messages"][-1].content)

        requestObject = json.loads(response["messages"][0].content)
        if (requestObject["user_wants_search_content"] != ""
                and requestObject["user_wants_heading"] != ""
                and requestObject["final_check"] == "Yes"):
            response = final_graph.invoke(initial_state)
            output = response["messages"][-1][-1]
        else:
            output = initial_state["query"]["llm_interactive_output"]

        chat_history.append((None, output))  # AI messages on left

    return chat_history

# Gradio UI with Custom Styles
with gr.Blocks(css="""
    .gradio-chatbot .message.user {text-align: right; background-color: #dcf8c6; border-radius: 10px; padding: 8px 12px;}
    .gradio-chatbot .message.bot {text-align: left; background-color: #f1f1f1; border-radius: 10px; padding: 8px 12px;}
""") as demo:
    gr.Markdown("## 🤖 Financial AI Assistant")

    chatbot_ui = gr.Chatbot()
    user_input = gr.Textbox(label="Enter your query", placeholder="Type a question...")
    send_button = gr.Button("Send")

    state = gr.State([])

    send_button.click(chatbot, inputs=[user_input, state], outputs=[chatbot_ui])

demo.launch(debug=True)
```

```
<ipython-input-30-1701f5156ec0>:61: UserWarning: You have not specified a value for the `type` parameter. Defaulting to
  chatbot_ui = gr.Chatbot()
Running Gradio in a Colab notebook requires sharing enabled. Automatically setting `share=True` (you can turn this off I

Colab notebook detected. This cell will run indefinitely so that you can see errors and logs. To turn off, set debug=Fa
* Running on public URL: https://84fb920b52a9d0741d.gradio.live

This share link expires in 72 hours. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal
```



```
Start coding or generate with AI.
```

```
I am here search_heading='' search_content='' user_wants_heading='' user_wants_search_content='' llm_interactive_output=
I am here 1search_heading='' search_content='' user_wants_heading='' user_wants_search_content='' llm_interactive_outpu
Previous messages[]
```