Policy Search AI: Semantic Spotter on HDFC Policy Documents

---

1. Problem Statement

Create a generative search system using LlamaIndex capable of searching a collection of HDFC policy descriptions and recommending appropriate policies based on user queries. The system should ensure accurate, reliable, and citation-supported answers.

---

2. Starter Code Overview

- **Library Installations:** Installed `llama-index`, `openai`, `chromadb`, and related packages.
- **API Setup:** Mounted Google Drive for data access and set up OpenAI API key.
- **Data Loading:** Loaded HDFC policy documents using `SimpleDirectoryReader`.
- **Vector Storage:** Created a ChromaDB collection to store document vectors.
- **Query Engine:** Built a query engine from the vector index.
- **Response Handling:** Implemented an interactive Q&A system.

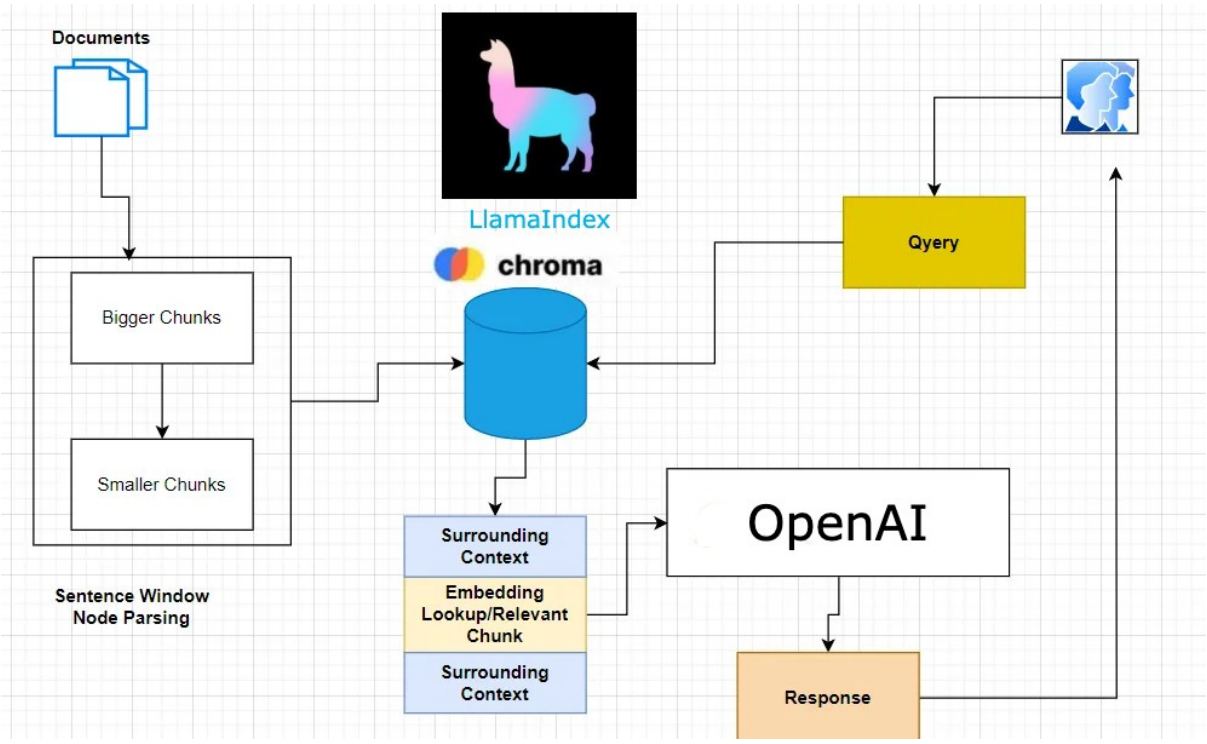
---

3. Product Specifications

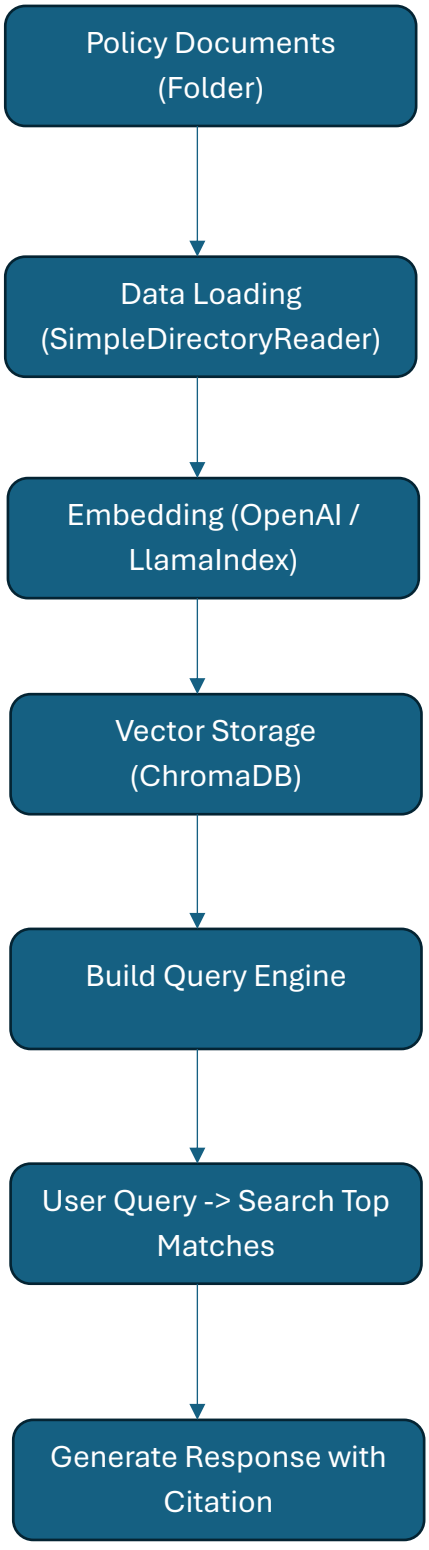
- **Dataset:** HDFC policy documents stored in a local folder.
- **Response:** Retrieve and summarize content relevant to user queries.
- **Citation:** Include source file names and page numbers with responses.
- **Evaluation:** Use Faithfulness and Relevancy Evaluators.
- **Tools:** LlamaIndex, ChromaDB, OpenAI API.

4. Solution Strategy

1. Load policy documents.
2. Create vector embeddings and store in ChromaDB.
3. Build a semantic query engine.
4. Implement an agent for enhanced reasoning and complex query handling.
5. Evaluate responses for faithfulness and relevancy.
6. Provide citations for transparency.



5. Flow Diagram



6. Implementation Details

Step 1: Install Libraries

```
!pip install llama-index openai chromadb -U
```

Step 2: Setup Drive and API Key

```
from google.colab import drive
from google.colab import userdata
import os
import openai

drive.mount('/content/drive')
openai.api_key = userdata.get('OPENAI_KEY')
os.environ['OPENAI_API_KEY'] = openai.api_key
```

Step 3: Load Documents

```
from llama_index.core import SimpleDirectoryReader
reader = SimpleDirectoryReader(input_dir="/path_to_policy_docs")
documents = reader.load_data()
```

Step 4: Create Vector Index

```
import chromadb
from llama_index.vector_stores.chroma import ChromaVectorStore
from llama_index.core import StorageContext, VectorStoreIndex

db = chromadb.PersistentClient(path="./chroma_db")
collection = db.get_or_create_collection("HDFC_policy")
vector_store = ChromaVectorStore(chroma_collection=collection)
storage_context = StorageContext.from_defaults(vector_store=vector_store)
index = VectorStoreIndex.from_documents(documents,
storage_context=storage_context)
```

Step 5: Build Query Engine

```
query_engine = index.as_query_engine()
```

Step 6: Setup Agent

```
from llama_index.agent.openai import OpenAIAgent
from llama_index.core.tools import QueryEngineTool

search_tool = QueryEngineTool.from_defaults(
    query_engine=query_engine,
    name="policy_document_search",
    description="Useful for answering HDFC Policies related queries"
)

agent = OpenAIAgent.from_tools(tools=[search_tool], llm=OpenAI(model="gpt-
3.5-turbo", temperature=0.0), verbose=True)
```

Step 7: Evaluate Responses

```
from llama_index.core.evaluation import FaithfulnessEvaluator,
RelevancyEvaluator

evaluators = {
    "faithfulness": FaithfulnessEvaluator(llm=OpenAI(model="gpt-3.5-turbo",
temperature=0.0)),
    "relevancy": RelevancyEvaluator(llm=OpenAI(model="gpt-3.5-turbo",
temperature=0.0))
}

results = {}
for name, evaluator in evaluators.items():
    results[name] = evaluator.evaluate_response(query=query,
response=response)
```

---

7. Next Steps and Enhancements

- **Data Quality:** Clean policy documents for better retrieval quality.
- **Advanced Node Parsing:** Use customized sentence splitting and chunking.
- **Model Improvements:** Integrate newer models like GPT-4 for improved performance.
- **UI Development:** Build a web-based user interface for easier access.
- **Multi-Agent Collaboration:** Implement multiple specialized agents (e.g., health, accident, investment policies).

---

8. References

- LlamaIndex Documentation
- OpenAI API Documentation
- ChromaDB Documentation