Author: Shameer Sheik, ML C56, UpGrad

Project: Semantic Spotter RAG System

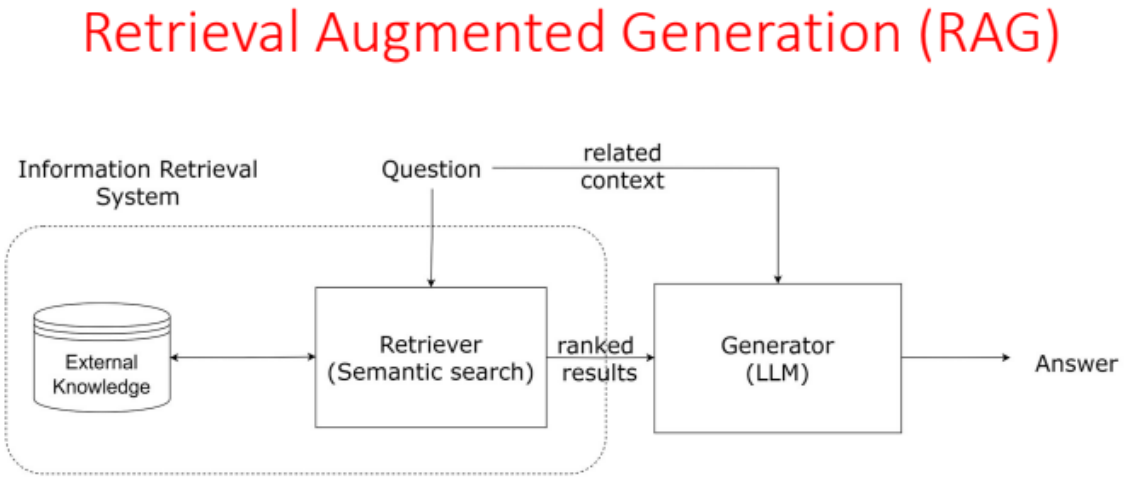
Introduction:

'Semantic Spotter - Project', where we supposed to use either LangChain or LlamaIndex frameworks to solve an end-to-end problem statement.

Objectives:

The goal of the project is to build a robust generative search system capable of effectively and accurately answering questions from the loaded various insurance policy documents.

We are using LlamaIndex to build the generative search application.



Design:

The project involves multiple layers of communication right from loading the document, creating the indexes (Vector Store Index, Summary Index) out of it at respective nodes. Retrieving the information as per the user query which performs semantic search and retrieve the data. Again the retrieved information will be processed through ranking (scores) and display the appropriate answer as response.

Author: Shameer Sheik, ML C56, UpGrad

Implementation:

The implementation involved below main steps as mentioned below

- Data loading,
- Nodes Parsing and build the Indices (Vector Store Index, Summary Index)
- Construction of Query engine
- Response

Step 3- Setup for Data Loading/Reading

```
[96] import nest_asyncio
     nest_asyncio.apply()
```

```
[97] os.chdir("/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents")
     !ls
```

```
Insurance_Policy_Documents  Principal-Sample-Life-Insurance-Policy.pdf  uber_2022.pdf
paul_graham_essay.txt       uber_2021.pdf
```

Check 1: Load the document HDFC-Life-Group-Term-Life-Policy.pdf and do a Q&R check

```
[99] from pathlib import Path
     from llama_index.core import download_loader
     PDFReader = download_loader("PDFReader")
     loader = PDFReader()
     documents = loader.load_data(file=Path('/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents/HDFC-Life-Group-Term-Life-Policy.pdf'))
```

```
<ipython-input-99-37a2d3b32435>:3: DeprecationWarning: Call to deprecated function (or staticmethod) download_loader. (`download_loader()` is deprecated. Pleas
  PDFReader = download_loader("PDFReader")
```

Step 4: Node Parsing and build the Vector Store Index and Summary Index

```
[100] from llama_index.core.node_parser import SimpleNodeParser
      from llama_index.core import VectorStoreIndex
      from llama_index.core import SummaryIndex
      from IPython.display import display, HTML

      # create parser and parse document into nodes
      parser = SimpleNodeParser.from_defaults()
      nodes = parser.get_nodes_from_documents(documents)

      # # build index
      index = VectorStoreIndex(nodes)
      summary_index = SummaryIndex(nodes)
```

Step 5: Query and Response Check with the above loaded document.

```
[101] # Construct Query Engine
      query_engine = index.as_query_engine()
      query_engine1 = summary_index.as_query_engine()

      # Query the engine.
      response = query_engine.query("Who is the insurance provider")

      # print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

      # Query the engine.
      response = query_engine.query("what is the maximum premium to be paid for the people of age more than 60 years?")

      # print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

      # Query the engine.
      response = query_engine1.query("What is the Summary of the document")
```

Author: Shameer Sheik, ML C56, UpGrad

Model Integration:

Use the LLAMAIndex libraries and functions to parse the nodes, build the Indexes and used them as input for query search engines.

Query Response Generation:

Here are the few documents which are loaded and verified the Query and Response check

#**Check 1**:

Load the document HDFC-Life-Group-Term-Life-Policy.pdf and do a Q&R check

Check 1: Load the document HDFC-Life-Group-Term-Life-Policy.pdf and do a Q&R check

```
[99] from pathlib import Path
     from llama_index.core import download_loader
     PDFReader = download_loader("PDFReader")
     loader = PDFReader()
     documents = loader.load_data(file=Path('/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents/HDFC-Life-Group-Term-Life-Policy.pdf'))
```

```
<ipython-input-99-37a2d3b32435>:3: DeprecationWarning: Call to deprecated function (or staticmethod) download_loader. (`download_loader()` is deprecated. Please install tool using pip install directly i
  PDFReader = download_loader("PDFReader")
```

Step 4: Node Parsing and build the Vector Store Index and Summary Index

```
[100] from llama_index.core.node_parser import SimpleNodeParser
      from llama_index.core import VectorStoreIndex
      from llama_index.core import SummaryIndex
      from IPython.display import display, HTML

      # create parser and parse document into nodes
      parser = SimpleNodeParser.from_defaults()
      nodes = parser.get_nodes_from_documents(documents)

      # # build index
      index = VectorStoreIndex(nodes)
      summary_index = SummaryIndex(nodes)
```

Query & Response Check:

Step 5: Query and Response Check with the above loaded document.

```
[101] # Construct Query Engine
      query_engine = index.as_query_engine()
      query_engine1 = summary_index.as_query_engine()

      # Query the engine.
      response = query_engine.query("Who is the insurance provider")

      # print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

      # Query the engine.
      response = query_engine.query("what is the maximum premium to be paid for the people of age more than 60 years?")

      # print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

      # Query the engine.
      response = query_engine1.query("What is the Summary of the document")

      # print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))
```

HDFC Life Insurance Company Limited.

The maximum premium to be paid for individuals above 60 years of age is not specified in the provided context information.

The document is a Group Term Life Insurance Policy issued by HDFC Life Insurance Company Limited. It outlines the terms and conditions for providing life insurance coverage to Insured Members under different groups, such as Employer-Employee and Non-Employer-Employee groups. The policy covers aspects like Premium payment, Grace Period, Cover Cessation, Claims process, Premium Rates, Taxes, Assignment, Nomination, Renewal Privilege, Surrender Value, and Exclusions like the Suicide clause and exclusions for Accidental Death Benefit. The document also includes information on Eligibility Criteria, Policyholder's obligations, Governing Law and Jurisdiction, and the process for issuance of a Duplicate Policy/Certificate of Insurance in case of loss or damage.

Author: Shameer Sheik, ML C56, UpGrad

#**Check 2**:

Load paul graham essay text document and do a Query Response Check.

Data Loading and Q&R check

```
[102] from llama_index.core import SimpleDirectoryReader
      # Create a new document reader for a text file.
      reader = SimpleDirectoryReader(input_files=["/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/paul_graham_essay.txt"])

      # Load data from the new document.
      documents = reader.load_data()

      # Create new nodes from the new documents.
      new_nodes = parser.get_nodes_from_documents(documents)

      # Insert new nodes into the existing index.
      index.insert_nodes(new_nodes)

      # Reconstruct the query engine.
      query_engine = index.as_query_engine()

      # Query the engine with a new question.
      response = query_engine.query("Why did Paul Graham start YC.")

      # Print the synthesized response.
      display(HTML(f'<p style="font-size:20px">{response.response}</p>'))
```

Paul Graham started Y Combinator (YC) to help and support early-stage startups by providing them with funding, mentorship, and resources to grow and succeed in the competitive business landscape.

#**Check 3**:

Load the Sample Life Insurance Policy .pdf document and do a Query Response Check.

Check 3: Load the Sample Life Insurance Policy .pdf document, and do a Query Response Check

Data Loading

```
[104] from pathlib import Path
      from llama_index.core import download_loader
      PDFReader = download_loader("PDFReader")
      loader = PDFReader()
      documents = loader.load_data(file=Path('/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Principal-Sample-Life-Insurance-Policy.pdf'))
```

```
<ipython-input-104-ee9ac05bb399>:3: DeprecationWarning: Call to deprecated function (or staticmethod) download_loader. (`download_loader()` is deprecated. Please install tool using pip install directly ins
  PDFReader = download_loader("PDFReader")
```

```
from llama_index.core.node_parser import SimpleNodeParser
from llama_index.core import VectorStoreIndex
from IPython.display import display, HTML

# create parser and parse document into nodes
parser = SimpleNodeParser.from_defaults()
nodes = parser.get_nodes_from_documents(documents)

# # build index
index = VectorStoreIndex(nodes)

# Construct Query Engine
query_engine = index.as_query_engine()

# Query the engine.
response = query_engine.query("In which section or Article, claim procedures are mentioned")

# print the synthesized response.
display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

# Query the engine.
response = query_engine.query("what is the Section C and Section D talks about in Part IV?")

# print the synthesized response.
display(HTML(f'<p style="font-size:20px">{response.response}</p>'))

# Query the engine.
response = query_engine.query("who is the ODI captain of Mens Indian cricket team")

# print the synthesized response.
display(HTML(f'<p style="font-size:20px">{response.response}</p>'))
```

Section D - Claim Procedures, Page 2

Section C in Part IV likely discusses the conditions and provisions related to the termination of insurance coverage under the policy. Section D in Part IV likely discusses the continuation options available for insurance coverage under the policy in various circumstances such as sickness, injury, layoff, approved leave of absence, and Family and Medical Leave Act (FMLA) situations.

I'm sorry, I cannot provide real-time information or updates on current events or sports.

Author: Shameer Sheik, ML C56, UpGrad

#Check 4:

Load all the insurance PDF files and do a Query & Response check

Check 4: Load all the insurance PDF files and do a Query & Response check

Step 1: Data Loading

```
os.chdir("/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents")
!ls
```

```
 HDFC-Life-Easy-Health-101N110V03-Policy-Bond-Single-Pay.pdf
 HDFC-Life-Group-Poorna-Suraksha-101N137V02-Policy-Document.pdf
 HDFC-Life-Group-Term-Life-Policy.pdf
'HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf'
 HDFC-Life-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf
 HDFC-Life-Smart-Pension-Plan-Policy-Document-Online.pdf
 HDFC-Surgicare-Plan-101N043V01.pdf
```

```
[111] # Define the path where all pdf documents are present

      pdf_path = "/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents"
```

```
[112] from llama_index.core import SimpleDirectoryReader
      reader = SimpleDirectoryReader(input_dir= pdf_path)
```

```
[113] ?SimpleDirectoryReader
```

```
[114] documents = reader.load_data()
      print(f"Loaded {len(documents)} docs")
```

```
Loaded 217 docs
```

Step 2: Building the query engine

```
[118] from llama_index.core.node_parser import SimpleNodeParser
      from llama_index.core import VectorStoreIndex
      from IPython.display import display, HTML

      # create parser and parse document into nodes
      parser = SimpleNodeParser.from_defaults()
      nodes = parser.get_nodes_from_documents(documents)
      # documents --> nodes

      # # build index
      index = VectorStoreIndex(documents)
      #nodes --> index

      # Construct Query Engine
      query_engine = index.as_query_engine()
```

```
[119] parser
```

```
SentenceSplitter(include_metadata=True, include_prev_next_rel=True, callback_manager=<llama_index.core.callbacks.base.CallbackManager object at 0x7c6a8da21030>, id_func=<function default_id_func at 0x7c6aa4d19fc0>, chunk_size=1024, chunk_overlap=200, separator=' ', paragraph_separator='\n\n\n', secondary_chunking_regex='[^,.;。？！]+[,.;。？！]?')
```

Step 3 Checking respones and respone parameters

```
[120] response = query_engine.query("What is this document talking about?")
```

```
[121] #Checking the response
      response.response
```

```
'This document is discussing an insurance policy called HDFC Life Sampoorna Jeevan.'
```


Step 3 Checking respones and respone parameters

```
[120] response = query_engine.query("What is this document talking about?")
```

```
[121] #Checking the response
      response.response
```

```
'This document is discussing an insurance policy called HDFC Life Sampoorna Jeevan.'
```


```
[122] response = query_engine.query("What are all names of the documents available and how many?")
```

```
[123] #Checking the response
      response.response
```

```
'The names of the documents available are:\n1. HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf'
```

```
[124] #Check the source node
      response.source_nodes
```

```
[NodeWithScore(node=Document(id_='0cd4dcae-2eb7-4e77-bccd-5f2d04c6c19e', embedding=None, metadata={'page_label': '44', 'file_name': 'HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf', 'file_path': '/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents/HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf', 'file_type': 'application/pdf', 'file_size': 1990500, 'creation_date': '2024-08-27', 'last_modified_date': '2024-08-25'}, excluded_embed_metadata_keys=['file_name', 'file_type', 'file_size', 'creation_date', 'last_modified_date', 'last_accessed_date'], excluded_llm_metadata_keys=['file_name', 'file_type', 'file_size', 'creation_date', 'last_modified_date', 'last_accessed_date'], relationships={}, text=' \n  \n \n', mimetype='text/plain', start_char_idx=None, end_char_idx=None, text_template='{metadata_str}\n\n{content}', metadata_template='{key}: {value}', metadata_seperator='\n'), score=0.7819553224023754),
 NodeWithScore(node=Document(id_='14cdc59b-8846-4664-bbad-707f57a0154f', embedding=None, metadata={'page_label': '43', 'file_name': 'HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf', 'file_path': '/content/drive/MyDrive/Gen_AI_Fashion_AI/Documents/Insurance_Policy_Documents/HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf', 'file_type': 'application/pdf', 'file_size': 1990500, 'creation_date': '2024-08-27', 'last_modified_date': '2024-08-25'}, excluded_embed_metadata_keys=['file_name', 'file_type', 'file_size', 'creation_date', 'last_modified_date', 'last_accessed_date'], excluded_llm_metadata_keys=['file_name', 'file_type', 'file_size', 'creation_date', 'last_modified_date', 'last_accessed_date'], relationships={}, text=' \n  \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n', mimetype='text/plain' start_char_idx=None, end_char_idx=None, text_template='{metadata_str}\n\n{content}', metadata_template='{key}: {value}', metadata_seperator='\n'), score=0.780342432193575)]
```

```
[125] #Extract the file name
      response.source_nodes[0].node.metadata['file_name']
```

```
'HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf'
```

```
[126] #Extract the score
      response.source_nodes[1].score
```

```
0.780342432193575
```

Author: Shameer Sheik, ML C56, UpGrad

Creating Response pipeline

Step 4 Creating a response Pipeline

User receives the response and the document that they can refer to

```
[127] ## Query response function
      def query_response(user_input):
        response = query_engine.query(user_input)
        file_name = response.source_nodes[0].node.metadata['file_name']
        final_response = response.response + '\n Check further at ' + file_name + ' document'
        return final_response
```

```
[128] def initialize_conv():
        print('Feel free to ask Questions regarding insurance policy. Press exit once you are done')
        while True:
          user_input = input()
          if user_input.lower() == 'exit':
            print('Exiting the program... bye')
            break
          else:
            response = query_response(user_input)
            display(HTML(f'<p style="font-size:20px">{response}</p>'))
```

```
[129] initialize_conv()
```

```
Feel free to ask Questions regarding insurance policy. Press exit once you are done
What is the Policy name?
```

HDFC Life Group Poorna Suraksha Check further at HDFC-Life-Group-Poorna-Suraksha-101N137V02-Policy-Document.pdf document

```
How much premium to be paid?
```

Three months premiums shall be collected in advance on the date of commencement of the Policy if the Policyholder chooses the monthly premium payment mode. Check further at HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf document

```
What is the Policy document name?
```

The policy document name is "HDFC Life Sampoorna Jeevan." Check further at HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf document

```
What is the Policy name?
```

The policy name is "HDFC Life Group Poorna Suraksha". Check further at HDFC-Life-Group-Poorna-Suraksha-101N137V02-Policy-Document.pdf document

```
What is the maturity amount?
```

The maturity amount is the Guaranteed Income on Maturity payable at the end of each Income Payout Frequency as provided under the Policy Schedule, starting from the (Policy Term + 1)th year until the individual attains age 99 years. Additionally, on the Maturity Date, there is an option to receive the Guaranteed Sum Assured on Maturity, which is the present value of future payouts discounted at a rate of 9% p.a. Check further at HDFC-Life-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf document

Step 5 - Build a Testing Pipeline

Here we feed a series of questions to the Q/A bot and store the responses along with the feedback on whether it's accurate or not from the user

```
[130] questions = ['What is the Policy document name?', "What is the policy name?","What is the maturity amount?",
                   'Any maturity benefits?']
```

```
[131] def testing_pipeline(questions):
        test_feedback  = []
        for i in questions:
          print(i)
          print(query_response(i))
          print('\n Please provide your feedback on the response provided by the bot')
          user_input = input()
          test_feedback.append((i,query_response(i),user_input))
        feedback_df = pd.DataFrame(test_feedback, columns =['Question', 'Response', 'Good or Bad'])
        return feedback_df
```

```
[132] import pandas as pd
```

```
[133] testing_pipeline(questions)
```

```
What is the Policy document name?
The policy document name is "HDFC Life Sampoorna Jeevan."
 Check further at HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf document

 Please provide your feedback on the response provided by the bot
Good
What is the policy name?
HDFC Life Group Poorna Suraksha
 Check further at HDFC-Life-Group-Poorna-Suraksha-101N137V02-Policy-Document.pdf document

 Please provide your feedback on the response provided by the bot
Bad
What is the maturity amount?
The maturity amount is the Guaranteed Income on Maturity payable at the end of each Income Payout Frequency as provided under the Policy Schedule, starting from the (Policy Term + 1)th year until the individual attains age 99 years. Addit
 Check further at HDFC-Life-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf document

 Please provide your feedback on the response provided by the bot
Good
Any maturity benefits?
The Maturity Benefit includes Guaranteed Income on Maturity payable at the end of each Income Payout Frequency as provided under the Policy Schedule, starting from the (Policy Term + 1)th year till the individual attains age 99 years. Add
 Check further at HDFC-Life-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf document

 Please provide your feedback on the response provided by the bot
Good
```

	Question	Response	Good or Bad
0	What is the Policy document name?	HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-D...	Good
1	What is the policy name?	HDFC Life Group Poorna Suraksha\n Check furthe...	Bad
2	What is the maturity amount?	The maturity amount is the Guaranteed Income o...	Good
3	Any maturity benefits?	The Maturity Benefit includes Guaranteed Incom...	Good

Dataset Chunking:

Loading the entire dataset without chunking posed a challenge due to memory constraints. However, chunking was not implemented due to time constraints and the manageable size of the dataset.

Lessons Learned:

Proper data preprocessing is crucial for ensuring data quality and readability. Integrating advanced AI models can significantly enhance the system's capabilities in generating detailed and contextually relevant responses. Handling large datasets requires careful consideration of memory constraints and implementation of efficient data processing techniques.

Future Scope:

Try to verify on multiple other kind of documents such as large .csv files, html and image files etc., to build an effective Q&A bot.

Conclusion:

Upon comparison of the generated response with the manual checking, the response is almost appropriate for the properly given query. Always response depends on the quality of query input that is fed to the system. Much more LLM integration with the quality input/query can make the system robust and close to the human thinking.