LAB No. 14

Document Loading using LangChain for Retrieval-Augmented Generation (RAG)

This lab introduces students to Document Loaders in LangChain, a key component of **Retrieval-Augmented Generation (RAG)** systems. Students will learn how to load, preprocess, and structure data from different document formats such as text and PDF files. By converting documents into LangChain's Document objects, students will understand how external knowledge can be prepared and supplied to large language models for improved, context-aware responses.

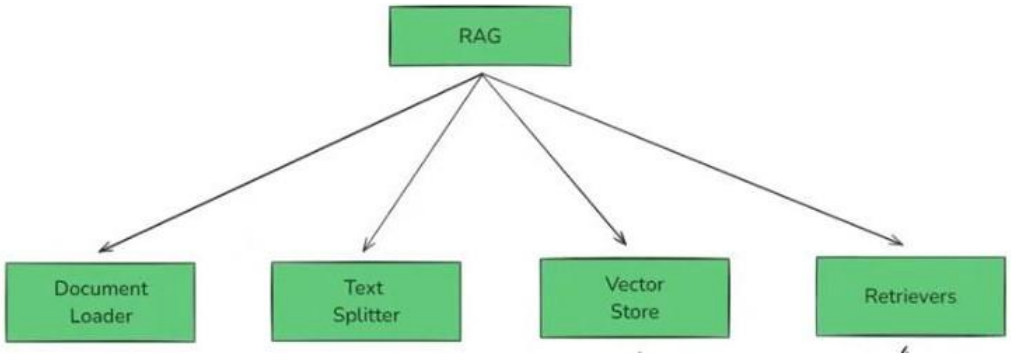
LAB Objectives

- Understand the role of document loaders in RAG
- Load data from multiple file formats using LangChain
- Inspect document metadata and content
- Prepare documents for downstream tasks like chunking and retrieval

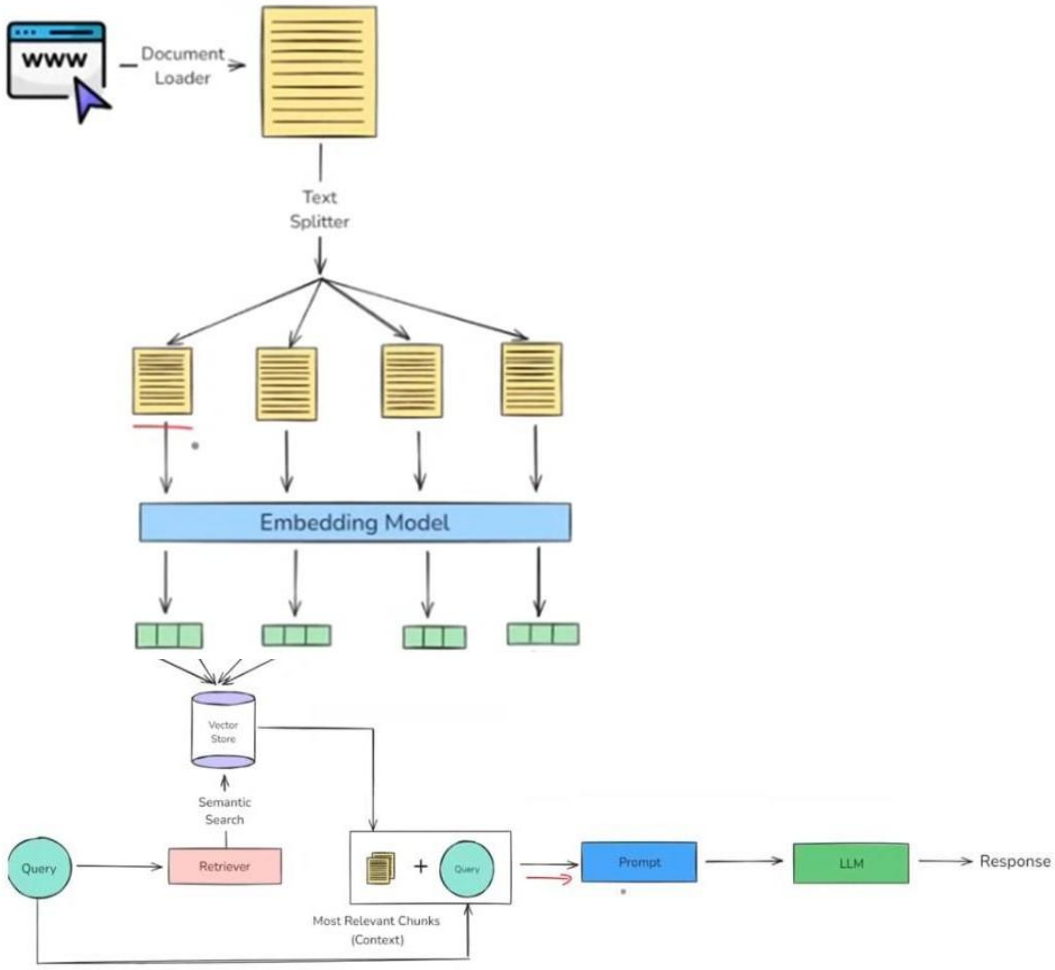
Tools s Libraries

- Python 3.6+
- Required libraries:
    - langchain
    - langchain-community
    - pypdf
    - unstructured

Flow Methodology of RAG



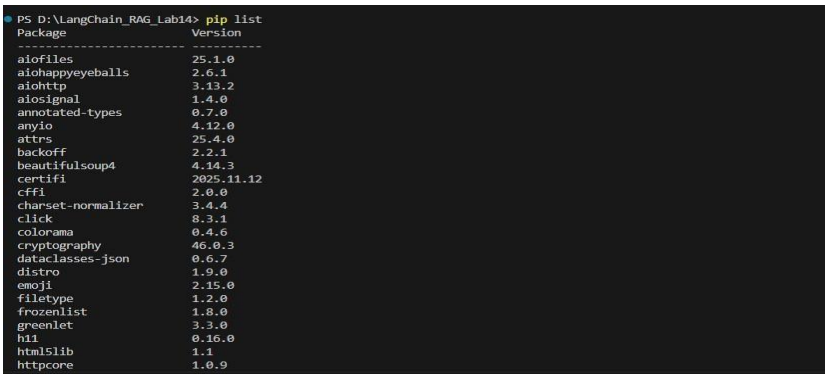
Complete Flow diagram of RAG



Lab Tasks (Practice Steps)

1: Environment Setup

- Create a virtual environment
- Install required LangChain libraries
- Verify installation



```
PS D:\LangChain_RAG_Lab14> pip list
Package                  Version
------------------------ ----------
aiofiles                 25.1.0
aiohappyeyeballs         2.6.1
aiohttp                  3.13.2
aiosignal                1.4.0
annotated-types          0.7.0
anyio                    4.12.0
attrs                    25.4.0
backoff                  2.2.1
beautifulsoup4           4.14.3
certifi                  2025.11.12
cffi                     2.0.0
charset-normalizer       3.4.4
click                    8.3.1
colorama                 0.4.6
cryptography             46.0.3
dataclasses-json         0.6.7
distro                   1.9.0
emoji                    2.15.0
filetype                 1.2.0
frozenlist               1.8.0
greenlet                 3.3.0
h11                      0.16.0
html5lib                 1.1
httpcore                 1.0.9
```

Task 2: Understand the Main Concept – Document Loaders

- Study the role of **Document Loaders** in RAG
- Explain how loaders convert raw data into LangChain Document objects

Document loaders in LangChain are used to load data from different sources such as text files, PDF documents, CSV files, and web pages. In this lab, different loaders like TextLoader, PyPDFLoader, CSVLoader, WebBaseLoader, and DirectoryLoader are used. These loaders convert raw data into LangChain Document objects. Each Document object contains page_content, which stores the extracted text, and metadata, which stores information such as file name, page number, or URL. In Retrieval-Augmented Generation (RAG), document loaders prepare external knowledge so that it can be retrieved and supplied to large language models for generating accurate and context-aware response

Task 3: Load PDF Data (PyPDFLoader)

- Load lecture_notes.pdf
- Count total pages
- Display content of first page
- Attach code with output screenshot

```
from langchain_community.document_loaders import PyPDFLoader

loader = PyPDFLoader('dl-curriculum.pdf')

docs = loader.load()

print(len(docs))

print(docs[0].page_content)

print(docs[1].metadata)
```

```
  from pydantic.v1.fields import FieldInfo as FieldInfoV1
23
CampusXDeepLearningCurriculum
A.ArtificialNeuralNetworkandhowtoimprovethem
1.BiologicalInspiration
● Understandingtheneuronstructure● Synapsesandsignaltransmission● Howbiologicalconceptstranslatetoartificialneur
2.HistoryofNeuralNetworks
● Earlymodels(Perceptron)● BackpropagationandMLPs● The"AIWinter"andresurgenceofneuralnetworks● Emergenceofdeeplearning
3.PerceptronandMultilayerPerceptrons(MLP)
● Single-layerperceptronlimitations● XORproblemandtheneedforhiddenlayers● MLParchitecture
4. LayersandTheirFunctions
● InputLayero Acceptinginputdata● HiddenLayerso Featureextraction● OutputLayero Producingfinalpredictions
5.ActivationFunctions
{'producer': 'Skia/PDF m131 Google Docs Renderer', 'creator': 'PyPDF', 'creationdate': '', 'title': 'Deep Learning Curriculum', 'sourc
e': 'dl-curriculum.pdf', 'total_pages': 23, 'page': 1, 'page_label': '2'}
```

Task : 4 Structured Data (CSVLoader)

- Load students.csv
- Inspect how rows are converted into documents
- Print one document sample
- Attach code with output screenshot

```
from langchain_community.document_loaders import CSVLoader

loader = CSVLoader(file_path='Social_Network_Ads.csv')

docs = loader.load()


print(len(docs))

print(docs[1])
```

```
400
page_content='User ID: 15810944
Gender: Male
Age: 35
EstimatedSalary: 20000
Purchased: 0' metadata={'source': 'Social_Network_Ads.csv', 'row': 1}
```

Task 6: Compare All Loaders

Students must compare:

- Content format
- Metadata fields
- Attach code with output screenshot

```
from langchain_community.document_loaders import PyPDFLoader, CSVLoader,
WebBaseLoader, TextLoader

loaders = {
    "PDF": PyPDFLoader("dl-curriculum.pdf"),
    "CSV": CSVLoader("Social_Network_Ads.csv"),
    "WEB": WebBaseLoader("https://www.langchain.com"),
    "TEXT": TextLoader("cricket.txt")
}

for name, loader in loaders.items():
    docs = loader.load()
    print(f"\n===== {name} Document Loader =====")
    print("Sample text:", docs[0].page_content[:200])
    print("Metadata:", docs[0].metadata)
```

```
===== CSV Document Loader =====
Sample text: User ID: 15624510
Gender: Male
Age: 19
EstimatedSalary: 19000
Purchased: 0
Metadata: {'source': 'Social_Network_Ads.csv', 'row': 0}

===== WEB Document Loader =====
Sample text: LangChain
```

COMPARISON TABLE:

Loader	Content Format	Metadata
PyPDFLoader	Page-wise text	page number, source
WebBaseLoader	Clean web text	URL
CSVLoader	Row-wise text	file path, row index

Lab Questions

1. What is the role of document loaders in RAG?

Answer:

They load external data and convert it into **Document objects** so LLMs can retrieve and use knowledge

2. Why is metadata important in LangChain documents?

Answer:

Metadata helps track **source, page number, URL**, improving accuracy and traceability.

3. Difference between TextLoader and PyPDFLoader?

Answer:

TextLoader: Loads plain text files

PyPDFLoader: Loads PDF files page by page

4. What happens if a PDF has scanned images instead of text?

Answer:

Text cannot be extracted without **OCR**, so content will be empty or unreadable.

5. Why is directory-based loading useful in real applications?

Answer:

It allows loading **multiple files automatically**, useful for large document collections.

6. How does document quality affect RAG performance?

Answer:

Clean, accurate documents give **better retrieval and more correct answers**.

Lab Assessment:

Student Name		**LAB Rubrics**	CLO3 , P5, PLO5
		Total Marks	10
Registration No		**Obtained Marks**	
		Teacher Name	Dr. Syed M Hamedoon
Date		**Signature**	

Laboratory Work Assessment Rubrics

Sr. No.	Performance Indicator	Excellent (5)	Good (4)	Average (3)	Fair (2)	Poor (1)
1	**Theoretical knowledge 10%**	Student knows all the related concepts about the theoretical background of the experiment and rephrase those concepts in written and oral assessments	Student knows most of the related concepts about the theoretical background of the experiment and partially rephrase those concepts in written and oral assessments	Student knows few of the related concepts about the theoretical background of the experiment and partially rephrase those concepts in written and oral assessments	Student knows very little about the related concepts about the theoretical background of the experiment and poorly rephrase those concepts in written and oral assessments	Student has poor understanding of the related concepts about the theoretical background of the experiment and unable to rephrase those concepts in written and oral assessments
2	**Application Functionality 10%**	Application runs smoothly and operation of the application runs efficiently	Application compiles with no warnings. Robust operation of the application, with good recovery.	Application compiles with few or no warnings. Consideration given to unusual conditions with reasonable	Application compiles and runs without crashing. Some attempt at detecting and correcting errors.	Application does not compile or compiles but crashes. Confusing. Little or no error detection or correction.
3	**Specifications 10%**	The program works very efficiently and meets all of the required specifications.	The program works and meets some of the specifications.	The program works and produces the correct results and displays them correctly. It also meets most of the other specifications.	The program produces correct results but does not display them correctly.	The program is producing incorrect results.
4	**Level of understanding of the learned skill 10%**	Provide complete and logical answers based upon accurate technical content to the questions asked by examiner	Provide complete and logical answers based upon accurate technical content to the questions asked by examiner with few errors	Provide partially correct and logical answers based upon minimum technical content to the questions asked by examiner	Provide very few and illogical answers to the questions asked by examiner.	Provide no answer to the questions asked by examiner.
5	**Readability and Reusability 10%**	The code is exceptionally well organized and very easy to follow and reused	The code is fairly easy to read. The code could be reused as a whole or each class could be reused.	Most of the code could be reused in other programs.	Some parts of the code require change before they could be reused in other programs.	The code is poorly organized and very difficult to read and not organized for reusability.

6	**AI System Design 10%**	Well-designed AI models. Code is highly maintainable	Good designed AI models and Little code duplications	Some attempt to make AI models. Code can be maintained with significant effort	Little attempt to design AI models and less understanding of code	Very poor attempt to design AI models and its code
7	**Responsiveness to Questions/ Accuracy 10%**	1. Responds well, quick and very accurate all the time. 2. Effectively uses eye contact, speaks clearly, effectively and confidently using suitable volume	1. Generally Responsive and accurate most of the times. 2. Maintains eye contact, speaks clearly with suitable volume and pace.	1. Generally Responsive and accurate few times. 2. Some eye contact, speaks clearly and unclearly in different portions.	1. Not much Responsive and accurate most of the times. 2. Uses eye contact ineffectively and fails to speak clearly and audibly	. 1. Non Responsive and inaccurate all the times. 2. No eye contact and unable to speak 3. Dress es inapprop riately
8	**Efficiency 10%**	The code is extremely efficient without sacrificing readability and understanding	The code is fairly efficient without sacrificing readability and understanding	Some part of the code is efficient and other part of the code is not understandable and work properly	The code is brute force and unnecessarily long	The code is huge and appears to be patched together
G	**Delivery 10%**	The program was delivered in time during lab.	The program was delivered in Lab before the end time.	The program was delivered within the due date.	The code was delivered within a day after the due date.	The code was delivered more than 2 days overdue.
10	**Awarenessof Safety Guidelines 10%**	Student has sufficient knowledge of the laboratory safety SOPsand protocol and is fully compliant to the guidelines	Student has sufficient knowledge of the laboratory safety SOPs and protocol and is Partially compliant to the guidelines	Student has little knowledge of the laboratory safety SOPs and protocol and is Partially compliant to the guidelines	Student has little knowledge of the laboratory safety SOPs and protocol and is non-compliant to the guidelines	Student has no knowledge of the laboratory safety SOPs and protocol and isnon-compliant to the guidelines