

## Lab2\_GEN\_AI:

### Part 1:

1. Do a Simple practical application using LangChain - Practical Mode only.
2. Kaggle - Explore Kaggle Opensource and Download any dataset of your Own and Perform Simple Data Analysis. You can directly link the dataset link from the Kaggle to your Code. - Practical Mode only.

### Part 2:

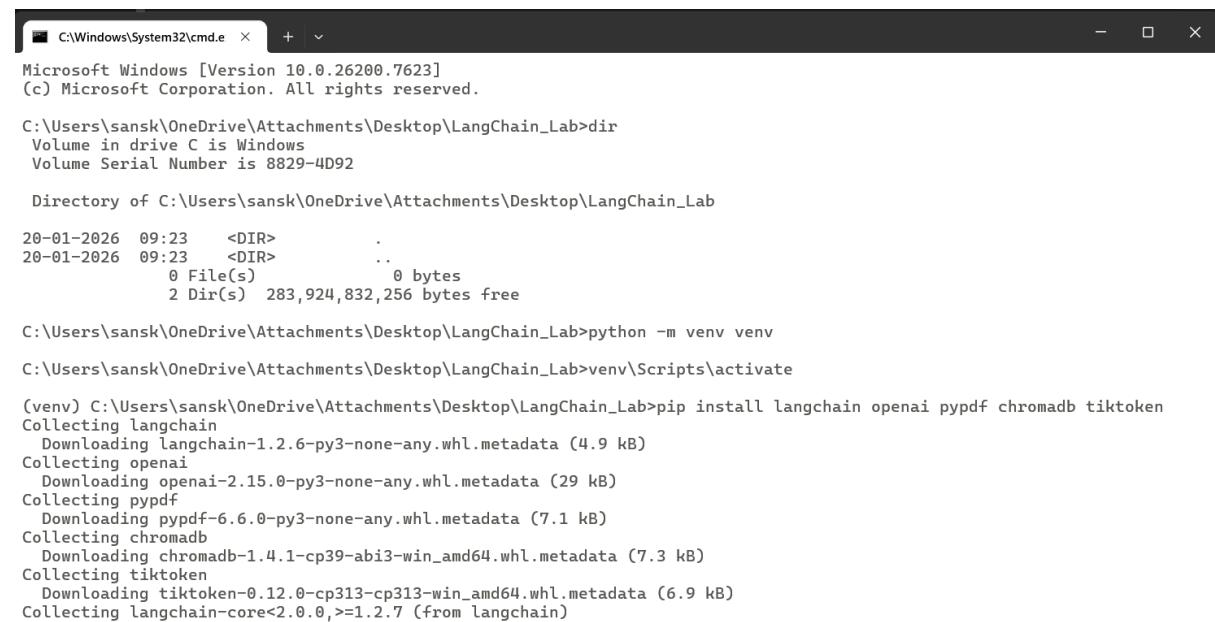
(Installation of Transformer Module, Sentiment analysis, NLP, Translation and summarization and sentence embedding)

3. Fast AI - A simple application on your own to illustrate Fast AI - Practical Mode
4. Hugging Face AI - A simple application on your own to illustrate Hugging Face AI - Practical Mode

All the above should be in Practical Mode only.

### Part1-

LangChain- Is an open-source framework that simplifies building applications powered by Large Language models (LLMs) like ChatGPT by providing tools to connect them with code, data and other services.



```
C:\Windows\System32\cmd.e × + ▾ Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sansk\OneDrive\Attachments\Desktop\LangChain_Lab>dir
Volume in drive C is Windows
Volume Serial Number is 8829-4D92

Directory of C:\Users\sansk\OneDrive\Attachments\Desktop\LangChain_Lab

20-01-2026 09:23    <DIR>
20-01-2026 09:23    <DIR>        ..
0 File(s)           0 bytes
2 Dir(s)   283,924,832,256 bytes free

C:\Users\sansk\OneDrive\Attachments\Desktop\LangChain_Lab>python -m venv venv
C:\Users\sansk\OneDrive\Attachments\Desktop\LangChain_Lab>venv\Scripts\activate
(venv) C:\Users\sansk\OneDrive\Attachments\Desktop\LangChain_Lab>pip install langchain openai pypdf chromadb tiktoken
Collecting langchain
  Downloading langchain-1.2.6-py3-none-any.whl.metadata (4.9 kB)
Collecting openai
  Downloading openai-2.15.0-py3-none-any.whl.metadata (29 kB)
Collecting pypdf
  Downloading pypdf-6.6.0-py3-none-any.whl.metadata (7.1 kB)
Collecting chromadb
  Downloading chromadb-1.4.1-cp39-abi3-win_amd64.whl.metadata (7.3 kB)
Collecting tiktoken
  Downloading tiktoken-0.12.0-cp313-cp313-win_amd64.whl.metadata (6.9 kB)
Collecting langchain-core<2.0.0,>=1.2.7 (from langchain)
```

The image shows two windows side-by-side. The left window is a code editor displaying the Python script `app.py`. The right window is a file browser showing the directory structure of the project.

**Code Editor (app.py):**

```

1  from langchain_community.document_loaders import PyPDFLoader
2  from langchain.text_splitter import RecursiveCharacterTextSplitter
3  from langchain.openai import OpenAIEmbeddings, ChatOpenAI
4  from langchain_community.vectorstores import Chroma
5  from langchain.chains import RetrievalQA
6
7  # Load PDF
8  loader = PyPDFLoader("Sample.pdf.pdf")
9  documents = loader.load()
10
11 # Split text
12 text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=50)
13 docs = text_splitter.split_documents(documents)
14
15 # Create embeddings
16 embeddings = OpenAIEmbeddings()
17
18 # Store in vector DB
19 db = Chroma.from_documents(docs, embeddings)
20
21 # Create QA system
22 llm = ChatOpenAI()
23 qa = RetrievalQA.from_chain_type(llm=llm, retriever=db.as_retriever())
24
25 print("PDF Chatbot Ready! Type 'exit' to quit.\n")
26
27 while True:
28     query = input("Question: ")
29     if query.lower() == "exit":
30         break
31
32     result = qa.run(query)
33     print("Answer:", result)
34
35

```

**File Browser:**

Name	Status	Date modified	Type	Size
venv	🕒	20-01-2026 09:30	File folder	
app.py	🕒	20-01-2026 09:52	Python Source File	1 KB
Sample.pdf.pdf	🕒	19-01-2026 08:13	Microsoft Edge PD...	557 KB

Online version requires API key - which requires credits and for that billing-

```
vocab.txt: 232kB [00:00, 8.89MB/s]
tokenizer.json: 466kB [00:00, 20.1MB/s]
special_tokens_map.json: 100%|██████████| 112/112 [00:00<00:00, 556kB/s]
config.json: 100%|██████████| 190/190 [00:00<00:00, 905kB/s]
✓ PDF Chatbot Ready! Type 'exit' to quit.

Question: what is the pdf about?
Traceback (most recent call last):
  File "C:/Users/sansk/OneDrive/Attachments/Desktop/LangChain_Lab/app.py", line 51, in <module>
    result = chain.invoke(q)
  File "C:/Users/sansk/OneDrive/Attachments/Desktop/LangChain_Lab/venv\Lib\site-packages\langchain_core\runnables\base.py", line 3151, in invoke
    input_ = context.run(step.invoke, input_, config)
  File "C:/Users/sansk/OneDrive/Attachments/Desktop/LangChain_Lab/venv\Lib\site-packages\langchain_core\language_models\chat_models.py", line 402, in invoke
    self.generate_prompt()
~~~~~
    [self._convert_input(input)],
    ^^^^^^^^^^^^^^^^^^^^^^^^^^
...<6 lines>...
    **kwargs,
    ^^^^^^^^^^
).generations[0][0],
^
File "C:/Users/sansk/OneDrive/Attachments/Desktop/LangChain_Lab/venv\Lib\site-packages\langchain_core\language_models\chat_models.py", line 1121, in generate_prompt
    return self.generate(prompt_messages, stop=stop, callbacks=callbacks, **kwargs)
~~~~~
File "C:/Users/sansk/OneDrive/Attachments/Desktop/LangChain_Lab/venv\Lib\site-packages\langchain_core\language_models\chat_models.py", line 931, in generate
```

Therefore switching to OFFLINE version for the LangChain-  
Using Ollama-