List of Required Libraries

To run your code successfully, you'll need to install the following libraries. Some of these libraries may be available on PyPI, while others might be part of specialized or in-house packages. In your Python environment, you can install the commonly available packages via pip. For example, create a **`requirements.txt`** file with the following content:

```
streamlit
python-dotenv
pandas
plotly
langchain-community
langchain-text-splitters
langchain-huggingface
langchain-groq
langchain-core
```

If any of the "langchain_*" or "langchain-…" packages are not available on PyPI, please consult your project documentation or install them directly from their respective GitHub repositories. You can install the libraries by running:

```
pip install streamlit python-dotenv pandas plotly langchain-community
langchain-text-splitters langchain-huggingface langchain-groq
langchain-core
```

Note:

- The built-in `os` module is part of Python's standard library and does not require installation.
- If you encounter issues with any of these packages (e.g., naming or version differences), verify the correct package names from your project documentation or repository source.

Steps to run the file:

1. Download the **Assignment_2_Part_2_Sayya.py** file and open it in any IDE
2. In the powershell / terminal:
   `streamlit run Assignment_2_Part_2_Sayya.py`

Now Upload pdf or csv or docx file to analyze

Observations on Techniques in Data Analysis Assistant: Chunking & Prompting

1. Document Chunking

- The `RecursiveCharacterTextSplitter` is used to split documents into chunks of 2000 characters with an overlap of 200 characters before vector storage.
- This ensures that each chunk is large enough to capture meaningful context while preventing important information from getting lost at chunk boundaries.
- After chunking, the text embeddings are generated using `sentence-transformers/all-mpnet-base-v2`, and the documents are stored in a Chroma vector database.

Observations:

Improved Retrieval Accuracy: Smaller, focused chunks enhance the relevance of retrieved information.
Context Retention: Overlapping text ensures continuity between chunks, reducing fragmented responses.
Challenge: Adjusting chunk size dynamically based on content structure could further optimize retrieval.

---

2. Advanced Prompting Techniques

The system applies Chain-of-Thought prompting to structure model responses logically and improve reasoning over retrieved data.

- **Step 1:** Context Injection – The retrieved chunks from ChromaDB are passed as `{context}` into the prompt template.
- **Step 2:** Structured Breakdown of Insights – The prompt explicitly directs the model to provide:
    - Key statistics
    - Trends/patterns
    - Data limitations
    - Visualization suggestions
- **Step 3:** Markdown Formatting – Enforces structured output, ensuring better readability in Streamlit.

Observations:

Enhanced Interpretability: The CoT-style prompt improves structured responses instead of generic answers.
Consistency in Outputs: The breakdown of key insights ensures that all aspects of the query are addressed.
Challenge: The model sometimes struggles when the retrieved context is too broad or lacks numerical data for statistical insights.