🔹 RAG Workflow Summary

1. Data Ingestion (collecting knowledge)

- Load files from different sources:

```
PDFs / Word Docs / Text Files / Websites / Databases / APIs
```

- Convert them into raw text using loaders.

---

2. Preprocessing (splitting + cleaning)

- Large docs are **split into smaller chunks** (e.g., 500–1000 words).
- Each chunk is treated as an independent **document unit**.
- Why? Because embeddings and retrieval work best on small, meaningful pieces.

---

3. Embedding (turning text into numbers)

- Each chunk → **embedding model** → high-dimensional vector.
- Example: OpenAI `text-embedding-ada-002` → 1536-dim vector.
- These vectors **capture meaning**, not exact words.
  (e.g., "dog" and "puppy" vectors are close together).

---

4. Store in Vector Database

- All embeddings + original text + metadata stored in:

```
FAISS / Pinecone / Chroma / Weaviate / Milvus
```

- Acts like a "search engine" for semantic meaning.

---

5. User Query

- User types a question (e.g., *"What is Python used for?"*).
- Query is also **embedded** into a vector (same model).
- Vector DB finds **top-k similar document chunks** (semantic search).

---

6. Retrieval + LLM

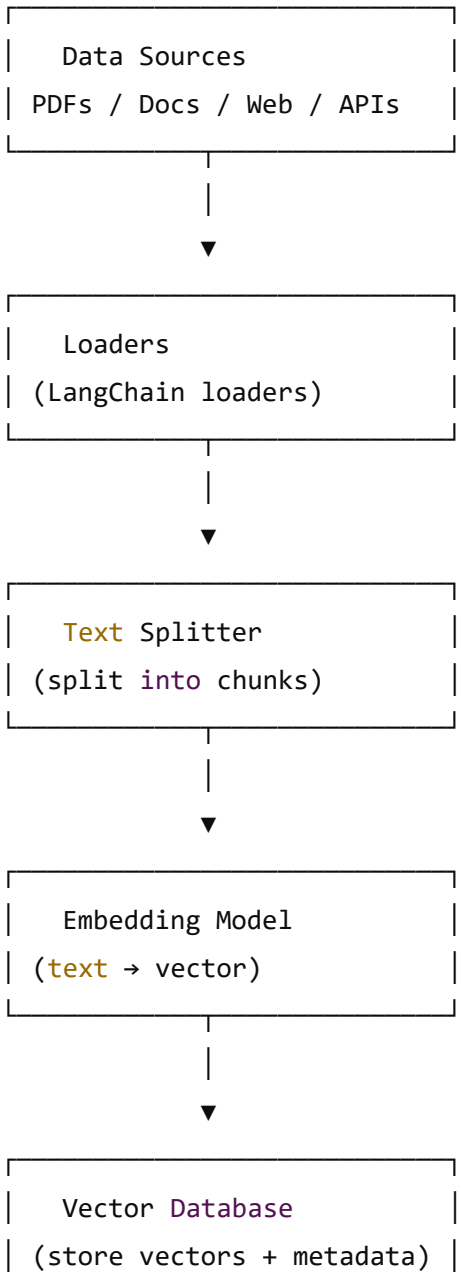
- Retrieved chunks are sent **along with the query** to the LLM.
- The LLM uses these chunks as **context** to generate an answer.
- This avoids hallucination and grounds the model in your data.

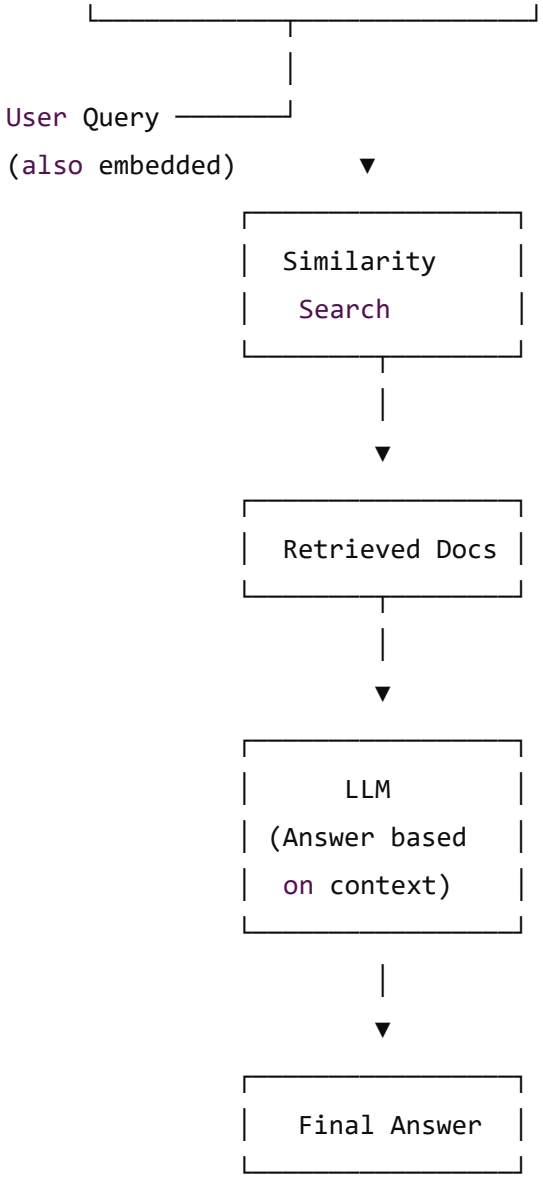
---

7. Final Answer

- User sees an **accurate, context-aware answer** built from their data.

🔹 Visualization (ASCII Flowchart)





🔹 Key Notes

- **Splitting** = prevents long docs from being unsearchable.
- **Embeddings** = create numerical meaning space.
- **Vector DB** = remembers all your knowledge.
- **Query embedding** = lets your question "live" in the same space as docs.
- **RAG = Retrieval + Generation** → retrieval makes LLMs reliable.