**First step build\_index.ipynb**

* **End-to-end RAG pipeline (chunking, embedding, vector DB, LLM integration)**
* **Use Python frameworks like Haystack, LlamaIndex, or LangChain**
* **Try different vector DBs: ChromaDB, FAISS, Weaviate**
* **Use local LLMs (Llama.cpp, Ollama, GPT4All) and/or OpenAI APIs**

**Use Option A (RAG):**

* **Use Colab only once to:**
  + **Extract your Word file**
  + **Create FAISS vector index**
  + **Download a .gguf model**

| **Feature** | **RAG** | **Fine-tuning** | **Prompt-only** |
| --- | --- | --- | --- |
| **AI memorizes doc?** | **❌ No** | **✅ Yes** | **❌ No** |
| **Updates easy?** | **✅ Yes, just re-embed** | **❌ No, retrain needed** | **✅ Yes, just re-paste** |
| **Handles big docs?** | **✅ Yes** | **❌ Not practical** | **❌ Only tiny text** |
| **Data privacy** | **✅ Yes (local possible)** | **✅ Yes (if local)** | **✅ Yes** |
| **Setup required** | **⚡ Medium (1 time)** | **🐢 High (training!)** | **🌱 None** |
| **Fast on CPU?** | **⚡ Fast** | **🐢 Slower (large model)** | **⚡ Fast** |
| **Good for Q&A?** | **✅ Yes** | **✅ If trained well** | **❌ Only small Qs** |
| **Example** | **“What are my skills?”** | **“What are my skills?”** | **“Summarize this text:”** |

**As a Specialist: What Should You Learn?**

1. **RAG Architecture**
   * **Understand vector search, chunking, retrieval, LLM prompt engineering**
2. **Vector Databases**
   * **Try FAISS, ChromaDB, Weaviate, Milvus, Qdrant**
3. **Embedding Models**
   * **SentenceTransformers, HuggingFace transformers, BGE, E5, GTE, etc.**
4. **Open Source LLMs**
   * **Llama (Meta), Mistral, Phi-2, Zephyr, Gemma, Mixtral**
   * **Learn how to run them locally (llama.cpp, ollama, GPT4All, etc.)**
5. **Frameworks**
   * **Haystack, LlamaIndex, LangChain, Gradio/Streamlit (for UI)**
6. **Deployment**
   * **Docker, local server, desktop app, etc.**
7. **Performance Tuning**
   * **Chunking strategy, reranking, memory vs speed, etc.**

**Python, RAG, LLM, vector database, basic UI**

**install langchain openai chromadb streamlit python-docx**