Engineering AI-Powered Chatbots with RAG and LLMs

👩‍💻 Done by: Malak Albahri

📧 Email: malakbaderalbahri@gmail.com

🔗 LinkedIn: Malak Albahri

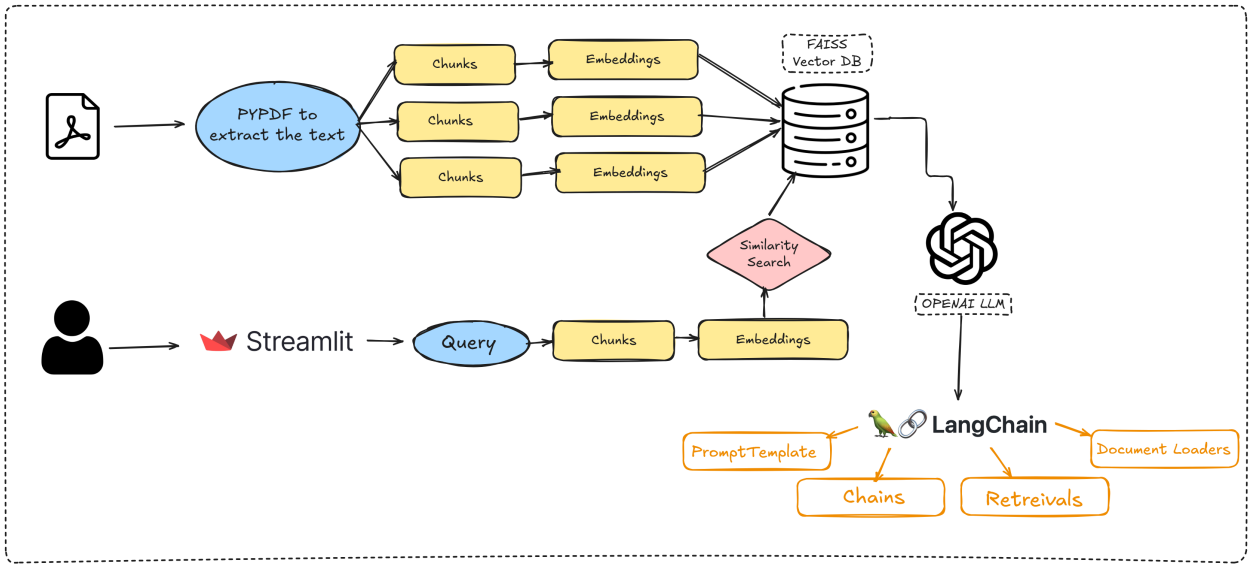
🐙 GitHub: MALAKBADER00

Welcome to this workshop summary on Building an AI chatbot using **Retrieval-Augmented Generation (RAG)**! 🚀

- **What is RAG?** A powerful AI technique that combines **retrieval** (fetching relevant data) with **generation** (LLM-powered responses).
- **Why use it?** To enhance responses by grounding them in real data.
- **What's an LLM?** A **Large Language Model** like OpenAI's GPT that can generate human-like text.

Let's dive in! 🔎

RAG Structure



❓ How do we extract PDFs?

✅ We use **LangChain document loaders**, which work on top of **PyPDF** to read and process PDFs seamlessly.

❓ Why do we need chunking?

🪚 Context windows in LLMs are limited! Chunking helps break long texts into digestible pieces, making retrieval more efficient.

❓ What are embeddings?

🧠 Converting human language into vectors! Just like **linear algebra**, we map words into numerical space. We use **OpenAI embeddings** for this.

❓ What's a Vector Database? Why FAISS?

🔍 A database optimized for **fast similarity searches**! **FAISS** is easy to use and efficient.

❓ How do users interact with our system?

💻 **Streamlit UI!** A simple yet powerful way to create interactive web apps in Python.

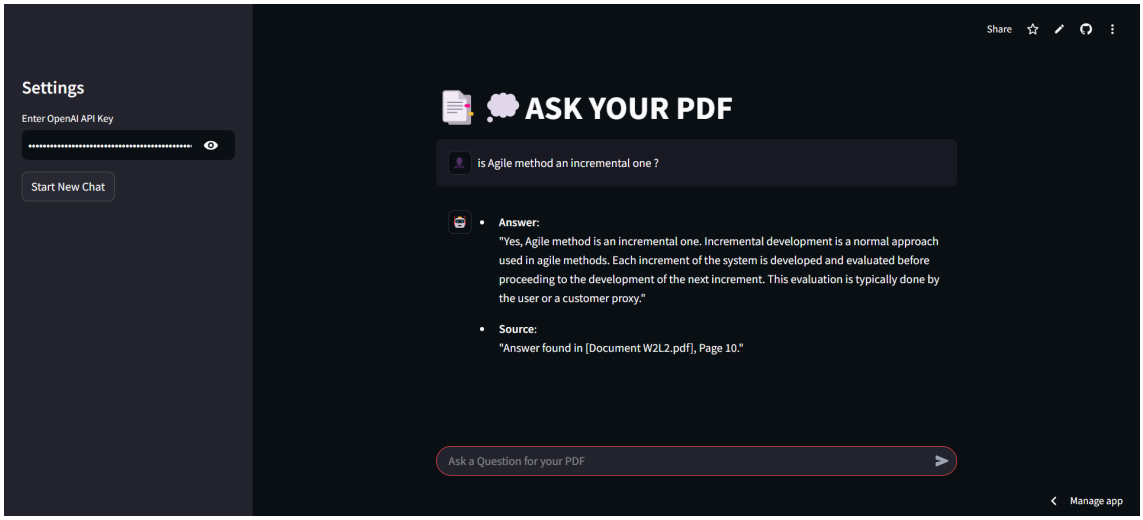
❓ What does LangChain offer us?

🛠️ **Prompt templates, chains, retrievers, document loaders**—all built-in!

❓ How do we retrieve relevant information?

🔎 By default, **similarity search** is used to find the closest matches in our vector database.

Final Look - The Streamlit UI



🛠️ 1. Setting Up Python

📌 **Step 1: Download & Install Python**

🔗 Download Python

Ensure you check the option **"Add Python to PATH"** during installation.

📌 Step 2: Verify Python Installation

Run the following command:

OS	Command
Windows	`python --version`
macOS	`python3 --version`

🌍 2. Creating a Virtual Environment

📌 Step 1: Create a Virtual Environment

OS	Command
Windows	`python -m venv venv`
macOS	`python3 -m venv venv`

📌 Step 2: Activate the Virtual Environment

OS	Command
Windows	`venv\Scripts\activate`
macOS	`source venv/bin/activate`

📦 3. Installing Dependencies

📌 Step 1: Create a `requirements.txt` File

Inside your project folder, create a file called `requirements.txt` and add the following:

```
langchain
langchain_openai
langchain_community
streamlit
python_dotenv
ipykernel
pypdf
faiss-cpu
```

📌 Step 2: Install Dependencies

OS	Command
Windows	`pip install -r requirements.txt`
macOS	`pip3 install -r requirements.txt`

📂 4. Organizing Project Structure

Your project should be structured as follows:

```
/chatbot-project
│── /venv              # Virtual Environment
│── /data              # Folder to store PDFs
│── /faiss_index       # Folder to store embeddings (Auto-created)
│── requirements.txt   # Dependencies
│── app.py             # Streamlit UI
│── rag.py             # RAG pipeline
```

💻 5. Writing `app.py` and `rag.py`

The core of our project consists of two key files:

📌 `rag.py` – Handles the Retrieval-Augmented Generation (RAG) pipeline.

📌 `app.py` – A Streamlit-powered UI for user interaction.

💡 **Good News!** The complete code has been **uploaded to GitHub**, so you can check it out and use it directly:

🔗 **GitHub Repository:** githubRepo

✅ You can download or clone the repo and use `rag.py` and `app.py` as a reference to build your own chatbot! 🚀

🎨 6. Running the Chatbot UI

📌 **Run Streamlit**

OS	Command
Windows	`streamlit run app.py`
macOS	`streamlit run app.py`

🔑 7. OpenAi API Key Extraction

- 🔗 **Get Your API Key:** OpenAI API Keys
- 📺 **Watch This Guide:** YouTube Tutorial

🚀 Pushing Code to GitHub - Step by Step

1️⃣ Install Git

- Download Git from here.
- During installation, **check "Add Git to PATH"** before clicking Next.

2️⃣ Create a GitHub Repository

- Go to GitHub and create a new repo.
- Name it, add a description, set it to **Public**, and click **Create**.

3️⃣ Set Up Git in VS Code

- Copy the GitHub setup commands after repo creation.
- Open **VS Code Terminal (** `Ctrl + ~` **)** and paste the commands.
- Refresh GitHub, and you'll see the repo with only a README file.

4️⃣ Add a `.gitignore` File

- On GitHub, go to **Add file → Create new file**.
- Name it `.gitignore`, select **Python**, and commit changes.
- In VS Code, run:

```
git pull
```

- `.gitignore` prevents pushing unnecessary/sensitive files.

5️⃣ Push Your Code to GitHub

Run these commands in VS Code Terminal:

```
git add .
git commit -m "Initial commit"
git push -u origin main
```

- Refresh GitHub → Your code is now uploaded. 🎉

🌍 Deploying the Streamlit App

- Run the App in locally using `streamlit run app.py`
- Click the **Deploy** button and set your desired web URL name.

📚
Explore My App: It features **Software Engineering course content** – check it out here: comp3401chatbot.streamlit.app

---

📚 Resources & Documentation

- 🔗 OpenAI API Docs
- 🔗 LangChain Docs
- 🔗 Streamlit Docs