

MediChat Pro: Your Intelligent Medical Document Assistant

MediChat Pro is a powerful Streamlit application that acts as a **Medical Document Assistant**. It uses the **Retrieval-Augmented Generation (RAG)** pattern to allow users to upload PDF medical documents, search the content, and get answers grounded strictly in the provided files.

✦ Features

- **Multi-PDF Upload:** Easily upload one or more PDF medical documents via the sidebar.
 - **RAG Implementation:** Uses LangChain components for text extraction, chunking, vector storage, and context-aware querying.
 - **FAISS Vector Store:** Efficiently indexes document content for fast, relevant retrieval.
 - **Euri AI Integration:** Utilizes the `gpt-4.1-nano` model (via Euri AI) to generate answers based on retrieved context.
 - **Chat Interface:** Provides a dynamic chat experience to interact with the processed documents.
 - **Custom Styling:** Enhanced UI/UX using Streamlit's custom markdown and CSS.
-

Getting Started

Follow these steps to set up and run MediChat Pro locally.

Prerequisites

- Python 3.8+
- An **Euri AI API Key**
- The necessary libraries specified in the `requirements.txt` (see next section).

Installation

1. Clone the Repository:

```
git clone <your-repo-link>
cd <your-repo-name>
```

2. Create `requirements.txt`:

Based on the imports, your project requires the following libraries. Create a `requirements.txt` file:

```
streamlit
langchain
python-dotenv
# Libraries likely used in your utility files (app/...)
```

```
pypdf
faiss-cpu
```

3. Install Dependencies:

```
pip install -r requirements.txt
```

4. Setup Environment Variables:

Create a file named **.ENV** in your project root directory and add your Euri AI API key:

```
# .ENV
EURI_API_KEY="your_euri_ai_api_key_here"
```




Running the Application

1. **Ensure the utility files** (**app/ui.py**, **app/pdf_utils.py**, etc.) **are correctly implemented in an app directory.**
2. **Run the Streamlit App:**

```
streamlit run main_app.py
# Replace main_app.py with the name of your main Python file (where your
provided code lives).
```

3. The application will open in your web browser, typically at **http://localhost:8501**.

Usage Guide



1. **Upload Documents:** Use the  **Document Upload** section in the sidebar to browse and upload one or more medical PDF files.
2. **Process Documents:** Click the  **Process Documents** button. The application will:
 - Extract text from all uploaded PDFs.
 - Split the combined text into chunks using the **RecursiveCharacterTextSplitter**.
 - Create a **FAISS vector store** from these chunks.
 - Initialize the **gpt-4.1-nano** chat model.
 - A  **Documents processed successfully!** message will confirm readiness.
3. **Chat with Documents:** In the main chat area, type your questions related to the content of your medical documents (e.g., "What does my latest lab report say about my glucose level?", "What is the dosage of the prescribed medication?").
4. **Get Contextual Answers:** The assistant will retrieve the most relevant information from your documents, use it as context, and provide an accurate answer. If the information is not present, it will inform you.

Customization

Component	Default Value	File/Location	Description
Model	"gpt-4.1-nano"	main_app.py	The specific Euri AI chat model used for Q&A.
Chunk Size	1000	main_app.py	The maximum size of text segments for vector storage.
Chunk Overlap	200	main_app.py	The overlap between consecutive text segments.
API Key	(Value from .ENV)	.ENV	Authentication key for Euri AI service.

Disclaimer

MediChat Pro is an **AI assistant** built to help summarize and retrieve information from your documents. **It is not a substitute for professional medical advice, diagnosis, or treatment.** Always seek the advice of a qualified healthcare provider with any questions you may have regarding a medical condition.

 Powered by Euri AI & LangChain |  Medical Document Intelligence