Case Study: Medical Research Assistant with LlamaIndex

By Syed Armghan Ahmad – syedarmghanahmad.work@gmail.com

Linkedin - GitHub - Repo for this project

Overview

As a self-taught developer passionate about AI and healthcare, I created the **Medical Research Assistant with LlamaIndex**, a hybrid search system for analyzing medical research papers. Built with LlamaIndex, GroqLLM, HuggingFace embeddings, and Streamlit, this tool integrates data preprocessing, retrieval-augmented generation (RAG), and a medical-themed UI to deliver evidence-based answers for medical professionals. This project showcases my skills in data engineering, AI integration, and production-grade system design, mastered through independent learning.

Problem Statement

Medical research papers are complex and voluminous, making it challenging for professionals to quickly extract relevant, evidence-based insights. Existing tools often lack semantic understanding or domain-specific filtering, leading to noisy results. My goal was to build a system that:

- Processes and cleans medical PDFs for accurate indexing.
- Combines semantic and keyword search for precise retrieval.
- Provides structured, evidence-based answers via natural language queries.
- Optimizes performance with persistent storage and query caching.
- Offers a user-friendly, medical-themed interface.

Approach

I designed a modular application with two phases:

1. **Data Preprocessing**: Extracted and cleaned PDF text, storing it in JSON for structured indexing.
2. **RAG System**: Implemented a production-grade RAG pipeline with hybrid search, persistent storage, and a Streamlit UI.

The system loads medical papers, indexes them with LlamaIndex, retrieves relevant chunks using hybrid search, generates answers with GroqLLM, and presents results in a styled interface. Key components include preprocessing, hybrid retrieval, and query optimization.

Key Features

- **Data Preprocessing**: Extracts text from PDFs using PyPDF2, cleans headers/footers, and stores in JSON (Phase 1).
- **Hybrid Search**: Combines vector (semantic) and BM25 (keyword) retrieval, merging top-15 results by score.
- **RAG Pipeline**: Uses LlamaIndex for retrieval and GroqLLM (Llama-3.1-8b-instant) for context-aware answers.
- **Query Postprocessing**: Filters nodes with a 0.75 similarity cutoff and medical keywords (e.g., "treatment", "diabetes").
- **Persistent Storage**: Saves the index to `./storage` for faster reloads.
- **Query Caching**: Stores responses in memory to reduce latency for repeated queries.
- **Interactive UI**: Streamlit interface with a medical-themed design, animated sidebar, and response cards.

Technical Implementation

- **Frontend (Streamlit)**:
  - Built a responsive UI with a centered header, animated sidebar, and search bar.
  - Applied custom CSS for a medical theme with animations (e.g., fade-in, pulse).
  - Used `st.text_input` for queries and styled response cards for answers.
- **Data Preprocessing**:
  - Extracted PDF text with PyPDF2, removing noise (headers, footers, special characters).
  - Stored raw and cleaned text in JSON for structured indexing (Phase 1).
  - Loaded PDFs with `SimpleDirectoryReader` for indexing.
- **Hybrid Search**:
  - Created a `VectorStoreIndex` with `BAAI/bge-small-en-v1.5` embeddings.
  - Used `BM25Retriever` for keyword search, merging with vector results via score sorting.
  - Implemented `HybridRetriever` to return top-15 nodes.
- **RAG Pipeline**:
  - Configured `RetrieverQueryEngine` with postprocessors for relevance and medical keywords.
  - Used a custom prompt to ensure evidence-based, structured LLM responses.
  - Integrated GroqLLM for fast, accurate answer generation.
- **Optimization**:
  - Persisted the index to `./storage` for efficient reloads.
  - Implemented query caching with a dictionary for repeated queries.
  - Added a warm-up phase to reduce initial latency.

- **Core Technologies**:
    - **LlamaIndex**: Enabled hybrid search, RAG, and indexing.
    - **GroqLLM**: Powered evidence-based answer generation.
    - **HuggingFace Embeddings**: Provided compact embeddings for medical text.
    - **PyPDF2**: Facilitated PDF text extraction.

Challenges and Solutions

- **Challenge**: Learning LlamaIndex's hybrid search as a self-taught developer.
    - **Solution**: Studied LlamaIndex documentation to implement vector and BM25 retrieval with score-based merging.
- **Challenge**: Preprocessing noisy medical PDFs.
    - **Solution**: Used PyPDF2 for text extraction and custom cleaning to remove headers/footers, storing results in JSON.
- **Challenge**: Ensuring domain-specific results.
    - **Solution**: Applied `KeywordNodePostprocessor` with medical keywords and exclusions.
- **Challenge**: Optimizing query performance.
    - **Solution**: Added persistent storage, query caching, and a warm-up phase to reduce latency.
- **Challenge**: Designing a medical-themed UI.
    - **Solution**: Crafted custom CSS with animations and a professional color scheme.

Impact

The Medical Research Assistant streamlines medical research analysis, delivering significant benefits:

- **Enhanced Efficiency**: Natural language queries provide quick, evidence-based answers, reducing research time.
- **Improved Accuracy**: Hybrid search ensures precise retrieval of relevant medical content.
- **Reliability**: Postprocessing and prompt engineering deliver domain-specific, safe responses.
- **Performance**: Caching and persistence optimize response times and scalability.
- **Skill Development**: Through self-learning, I mastered LlamaIndex, hybrid search, and medical NLP, preparing me for advanced AI roles.
- **Portfolio Strength**: This project showcases my ability to build production-grade AI tools for healthcare, a standout addition to my portfolio.

Lessons Learned

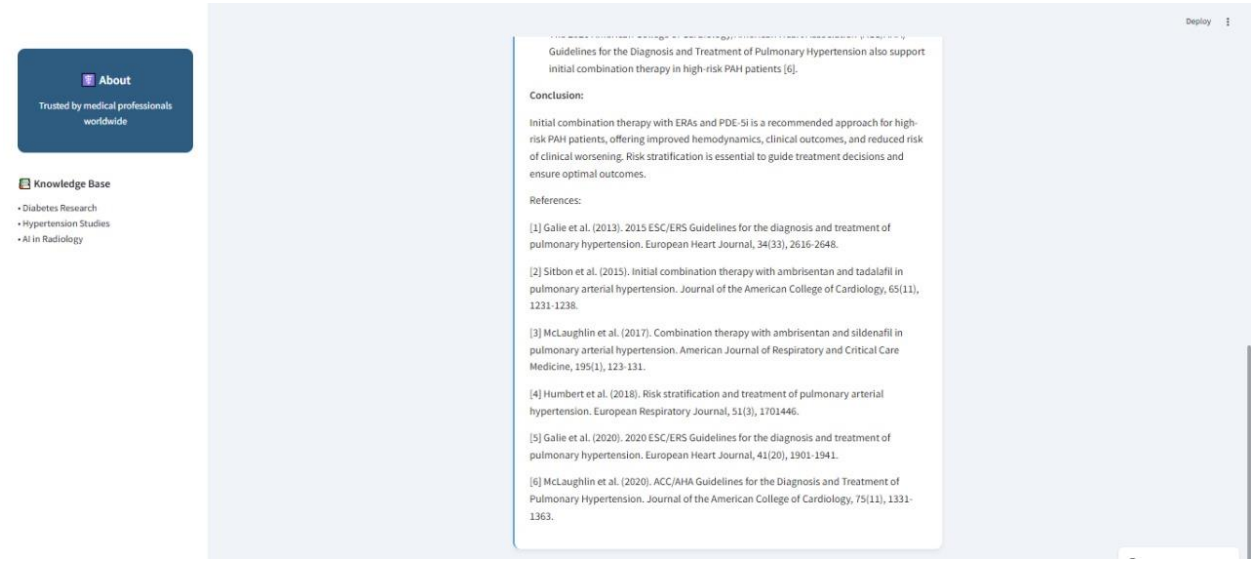
- **Hybrid Search**: Combining vector and BM25 retrieval improves accuracy for medical queries.
- **Prompt Engineering**: Detailed, domain-specific prompts ensure reliable LLM outputs.
- **Preprocessing**: Cleaning and structuring PDF data is critical for effective indexing.
- **Optimization**: Caching and persistence are key for production-grade performance.
- **Self-Learning**: Experimentation and documentation were essential for mastering LlamaIndex and GroqLLM.
- **UI Design**: Medical-themed styling enhances user trust and engagement.

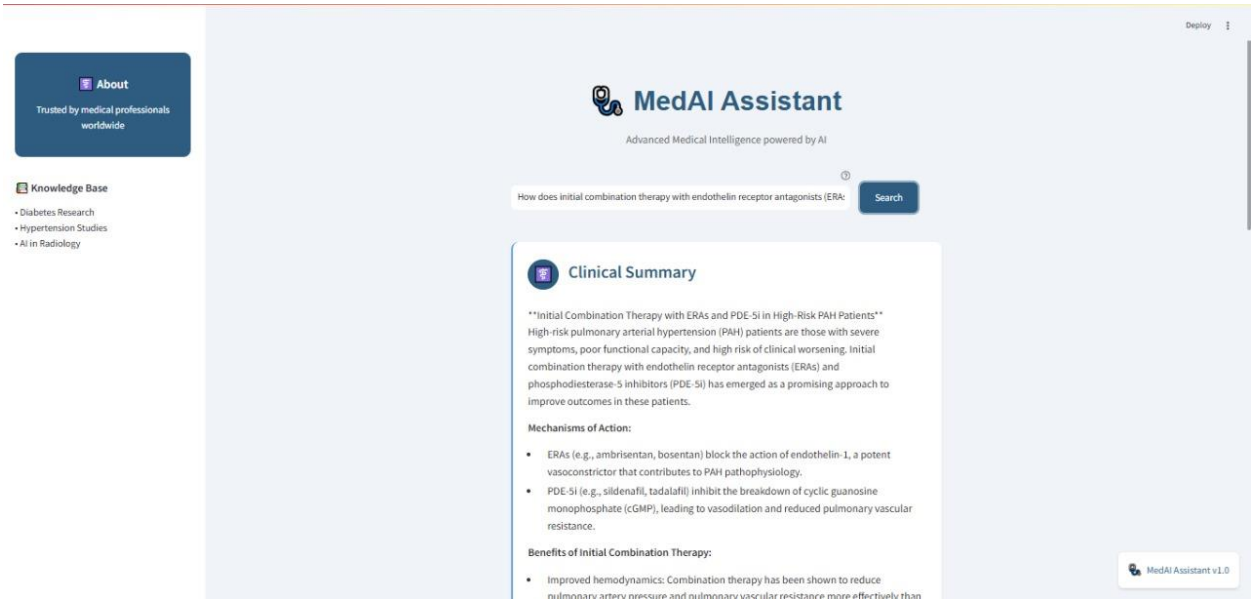
Future Enhancements

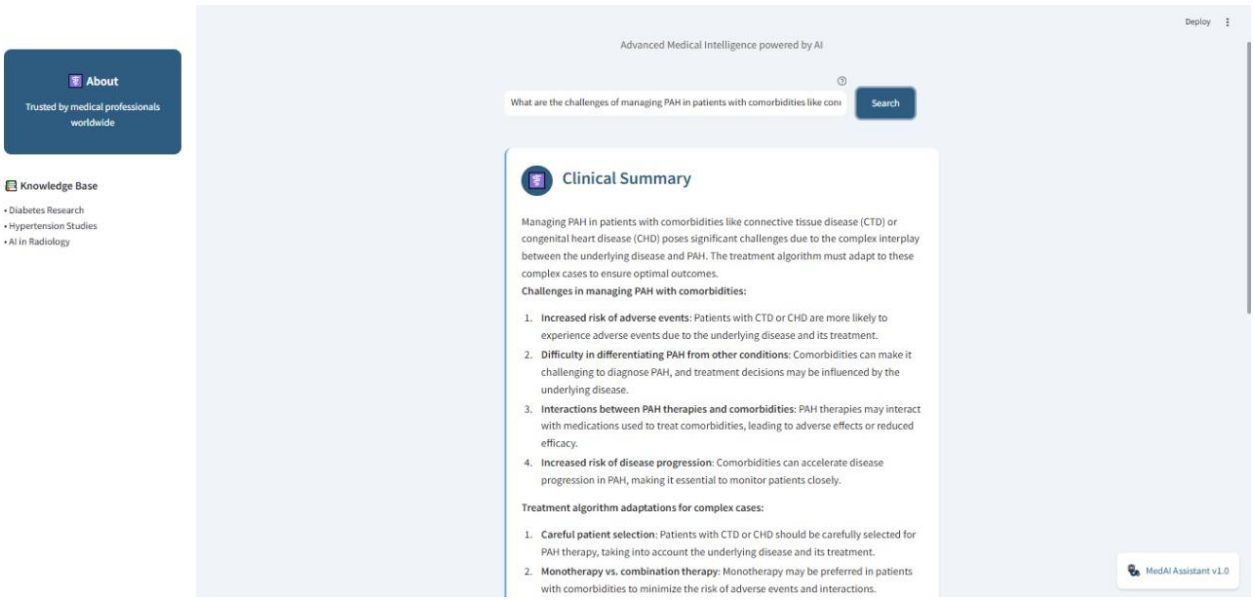
To productionize the application, I would:

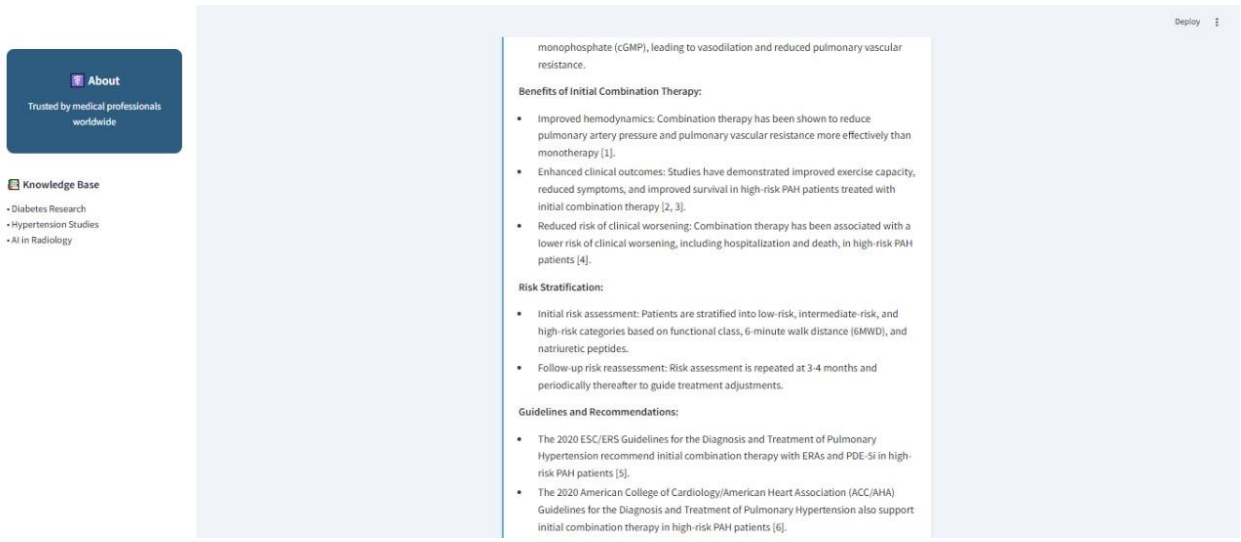
- Store PDFs in AWS S3 and metadata/JSON in MongoDB for scalability.
- Replace in-memory index with Pinecone for distributed vector search.
- Implement a FastAPI backend with Celery for async processing.
- Add OAuth2 authentication and rate limiting for security.
- Use Redis for persistent query caching.
- Integrate Prometheus and Grafana for system monitoring.
- Develop machine learning models (e.g., with TensorFlow) for clinical outcome prediction.

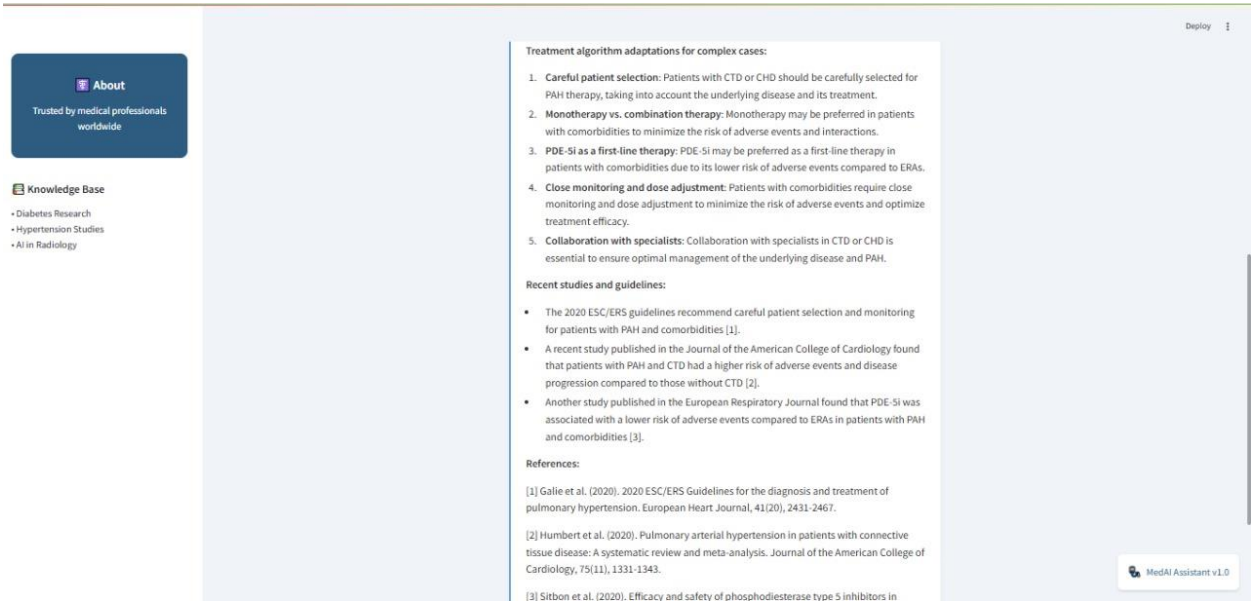
Screenshots






Deploy
About
Trusted by medical professionals worldwide
Knowledge Base
• Diabetes Research
• Hypertension Studies
• AI in Radiology
Advanced Medical Intelligence powered by AI
What are the challenges of managing PAH in patients with comorbidities like con
Search
Clinical Summary
Managing PAH in patients with comorbidities like connective tissue disease (CTD) or congenital heart disease (CHD) poses significant challenges due to the complex interplay between the underlying disease and PAH. The treatment algorithm must adapt to these complex cases to ensure optimal outcomes.
Challenges in managing PAH with comorbidities:
1. Increased risk of adverse events: Patients with CTD or CHD are more likely to experience adverse events due to the underlying disease and its treatment.
2. Difficulty in differentiating PAH from other conditions: Comorbidities can make it challenging to diagnose PAH, and treatment decisions may be influenced by the underlying disease.
3. Interactions between PAH therapies and comorbidities: PAH therapies may interact with medications used to treat comorbidities, leading to adverse effects or reduced efficacy.
4. Increased risk of disease progression: Comorbidities can accelerate disease progression in PAH, making it essential to monitor patients closely.
Treatment algorithm adaptations for complex cases:
1. Careful patient selection: Patients with CTD or CHD should be carefully selected for PAH therapy, taking into account the underlying disease and its treatment.
2. Monotherapy vs. combination therapy: Monotherapy may be preferred in patients with comorbidities to minimize the risk of adverse events and interactions.
MedAI Assistant v1.0


Deploy
About
Trusted by medical professionals worldwide
Knowledge Base
• Diabetes Research
• Hypertension Studies
• AI in Radiology
monophosphate (cGMP), leading to vasodilation and reduced pulmonary vascular resistance.
Benefits of Initial Combination Therapy:
• Improved hemodynamics: Combination therapy has been shown to reduce pulmonary artery pressure and pulmonary vascular resistance more effectively than monotherapy [1].
• Enhanced clinical outcomes: Studies have demonstrated improved exercise capacity, reduced symptoms, and improved survival in high-risk PAH patients treated with initial combination therapy [2, 3].
• Reduced risk of clinical worsening: Combination therapy has been associated with a lower risk of clinical worsening, including hospitalization and death, in high-risk PAH patients [4].
Risk Stratification:
• Initial risk assessment: Patients are stratified into low-risk, intermediate-risk, and high-risk categories based on functional class, 6-minute walk distance (6MWD), and natriuretic peptides.
• Follow-up risk reassessment: Risk assessment is repeated at 3-4 months and periodically thereafter to guide treatment adjustments.
Guidelines and Recommendations:
• The 2020 ESC/ERS Guidelines for the Diagnosis and Treatment of Pulmonary Hypertension recommend initial combination therapy with ERAs and PDE-5i in high-risk PAH patients [5].
• The 2020 American College of Cardiology/American Heart Association (ACC/AHA) Guidelines for the Diagnosis and Treatment of Pulmonary Hypertension also support initial combination therapy in high-risk PAH patients [6].



Conclusion

The Medical Research Assistant with LlamaIndex is a testament to my self-learning journey and expertise in AI-driven healthcare solutions. By building a system with hybrid search, RAG, and production-grade UI, I addressed critical challenges in medical research analysis. This project highlights my skills in data preprocessing, AI retrieval, and user interface design, positioning me for impactful contributions in professional healthcare technology roles.