Assignment for Engineers (Advanced)

This project demonstrates a full-stack AI application using Retrieval Augmented Generation. Instead of retraining a large language model, the system enhances model responses by retrieving relevant document context using embeddings and similarity search. The architecture ensures accuracy, scalability, and cost efficiency while providing a modern ChatGPT-style user interface.

Project Architecture Summary

Frontend (React)
⬇
FastAPI Backend
⬇
Embedding Model
⬇
Vector Store (pickle)
⬇
Similarity Search
⬇
Gemini LLM
⬇
Response Returned

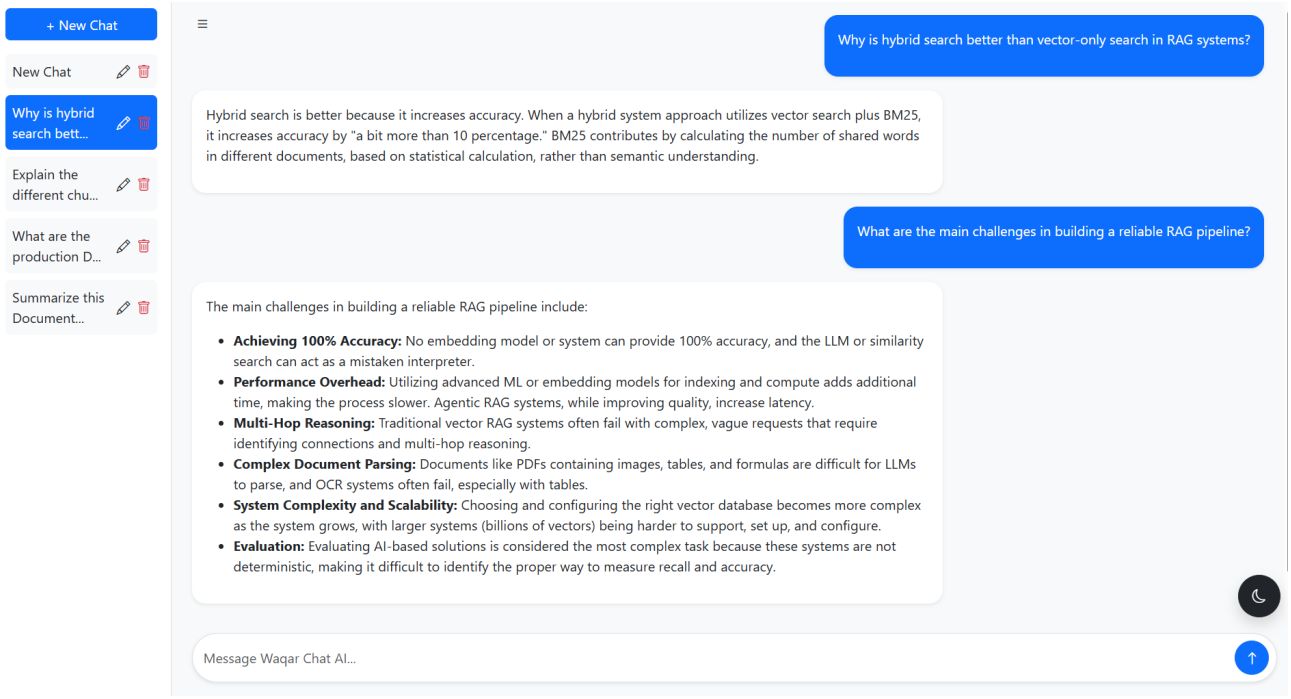
GitHub Link for the Project

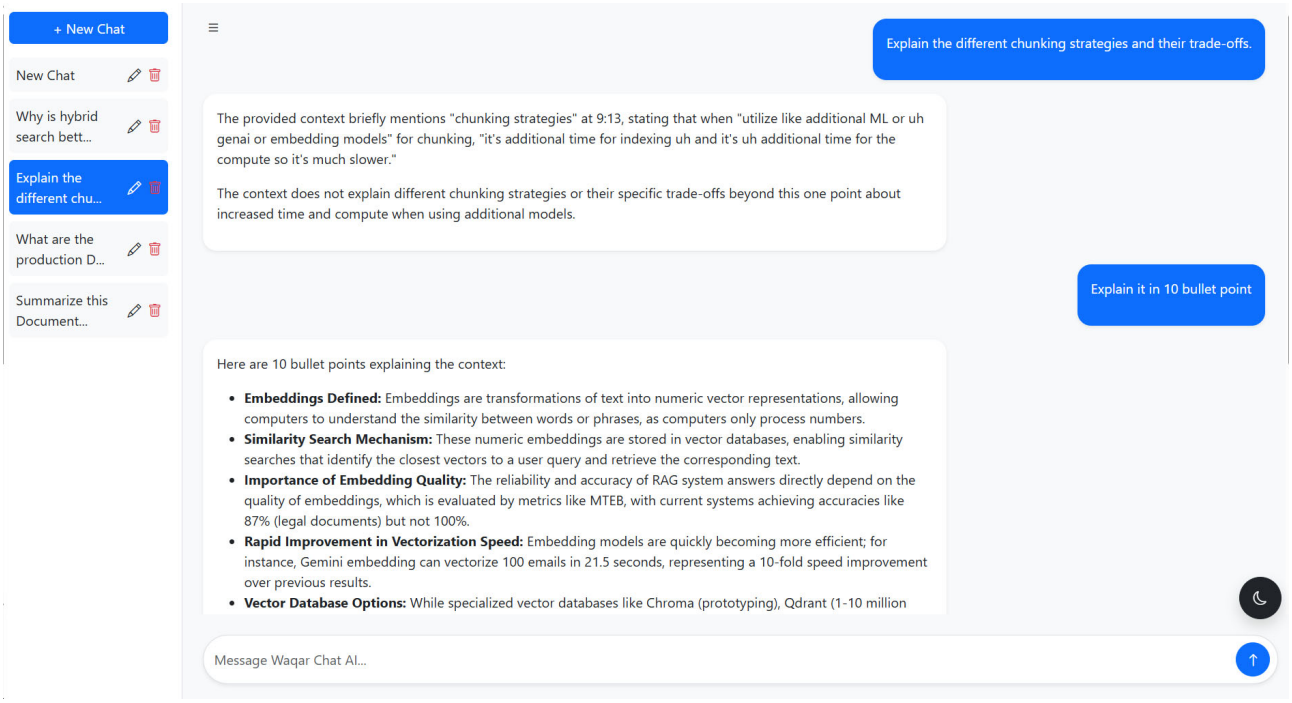
https://github.com/waqarahmad3626/Module_4_Home_Work

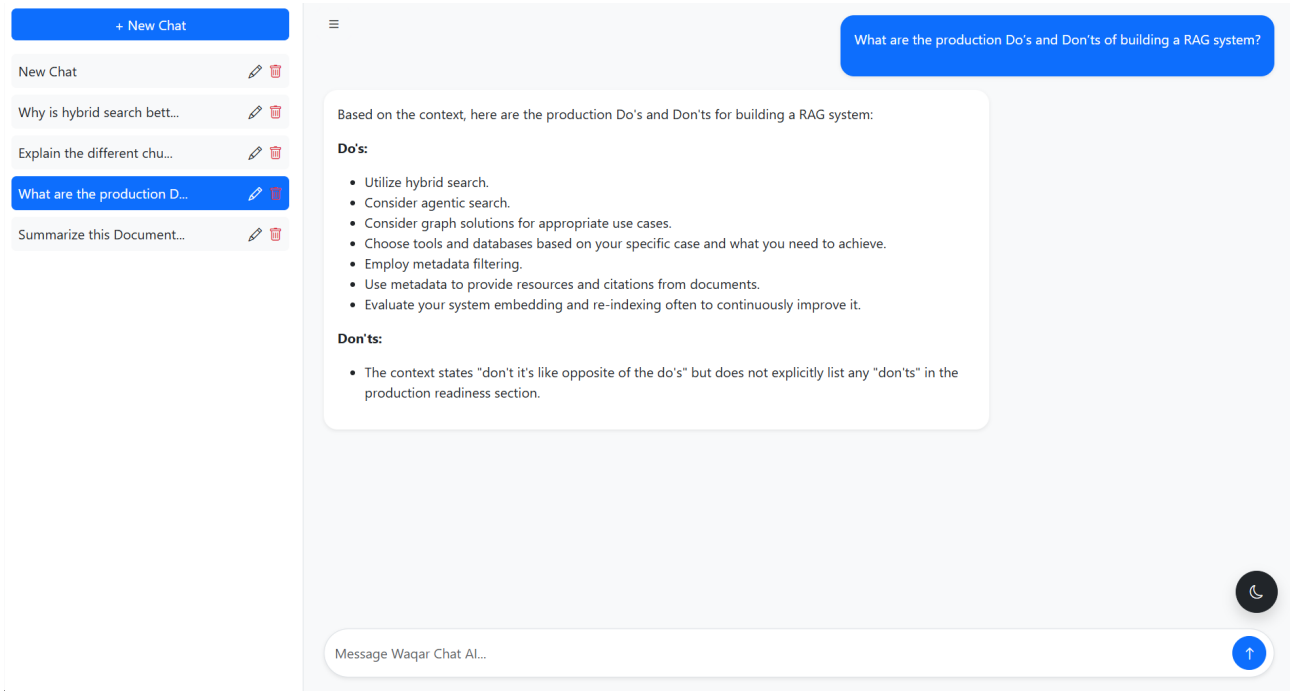
Reflection

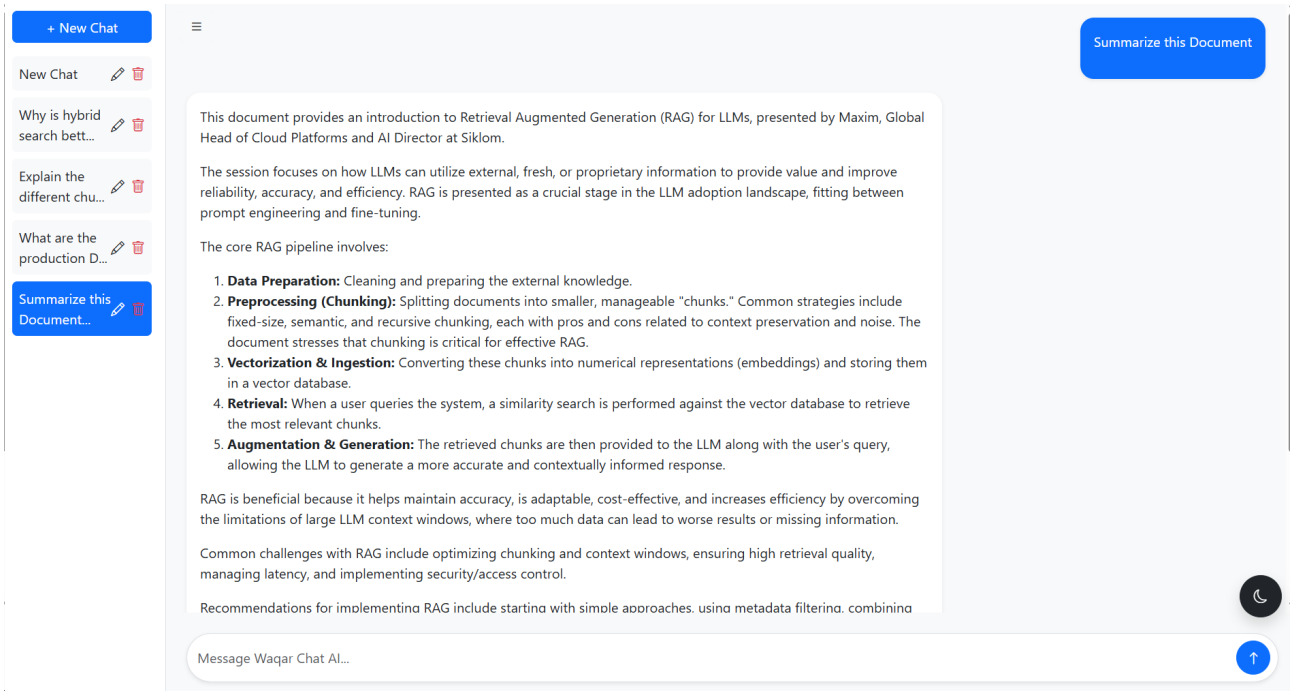
The most challenging part of this project was designing a reliable chunking and retrieval strategy. While extracting text from PDFs and transcribing audio were technical hurdles, the real complexity emerged in balancing chunk size, overlap, and retrieval precision. Small chunks improved relevance but sometimes lost context, while larger chunks preserved meaning but reduced retrieval accuracy.

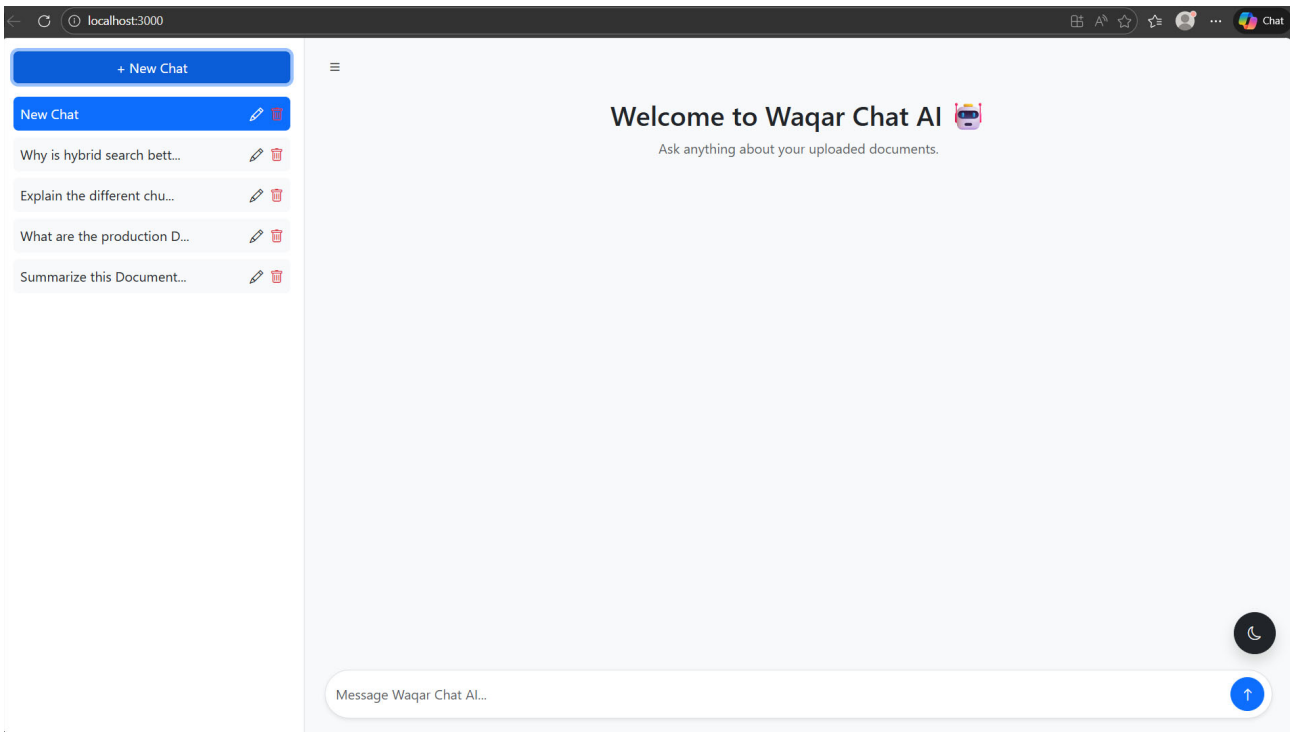
Through this challenge, I gained a deeper understanding of how embeddings power similarity search and how hybrid search significantly improves performance over vector-only approaches. I also learned that evaluation is one of the hardest aspects of RAG systems because results are probabilistic rather than deterministic. This project strengthened my understanding of production-level AI system design beyond just calling an LLM API.


+ New Chat
New Chat
Why is hybrid search bett...
Explain the different chu...
What are the production D...
Summarize this Document...
Why is hybrid search better than vector-only search in RAG systems?
Hybrid search is better because it increases accuracy. When a hybrid system approach utilizes vector search plus BM25, it increases accuracy by "a bit more than 10 percentage." BM25 contributes by calculating the number of shared words in different documents, based on statistical calculation, rather than semantic understanding.
What are the main challenges in building a reliable RAG pipeline?
The main challenges in building a reliable RAG pipeline include:
Achieving 100% Accuracy: No embedding model or system can provide 100% accuracy, and the LLM or similarity search can act as a mistaken interpreter.
Performance Overhead: Utilizing advanced ML or embedding models for indexing and compute adds additional time, making the process slower. Agentic RAG systems, while improving quality, increase latency.
Multi-Hop Reasoning: Traditional vector RAG systems often fail with complex, vague requests that require identifying connections and multi-hop reasoning.
Complex Document Parsing: Documents like PDFs containing images, tables, and formulas are difficult for LLMs to parse, and OCR systems often fail, especially with tables.
System Complexity and Scalability: Choosing and configuring the right vector database becomes more complex as the system grows, with larger systems (billions of vectors) being harder to support, set up, and configure.
Evaluation: Evaluating AI-based solutions is considered the most complex task because these systems are not deterministic, making it difficult to identify the proper way to measure recall and accuracy.
Message Waqar Chat AI...


+ New Chat
New Chat
Why is hybrid search bett...
Explain the different chu...
What are the production D...
Summarize this Document...
Explain the different chunking strategies and their trade-offs.
The provided context briefly mentions "chunking strategies" at 9:13, stating that when "utilize like additional ML or uh genai or embedding models" for chunking, "it's additional time for indexing uh and it's uh additional time for the compute so it's much slower."
The context does not explain different chunking strategies or their specific trade-offs beyond this one point about increased time and compute when using additional models.
Explain it in 10 bullet point
Here are 10 bullet points explaining the context:
Embeddings Defined: Embeddings are transformations of text into numeric vector representations, allowing computers to understand the similarity between words or phrases, as computers only process numbers.
Similarity Search Mechanism: These numeric embeddings are stored in vector databases, enabling similarity searches that identify the closest vectors to a user query and retrieve the corresponding text.
Importance of Embedding Quality: The reliability and accuracy of RAG system answers directly depend on the quality of embeddings, which is evaluated by metrics like MTEB, with current systems achieving accuracies like 87% (legal documents) but not 100%.
Rapid Improvement in Vectorization Speed: Embedding models are quickly becoming more efficient; for instance, Gemini embedding can vectorize 100 emails in 21.5 seconds, representing a 10-fold speed improvement over previous results.
Vector Database Options: While specialized vector databases like Chroma (prototyping), Qdrant (1-10 million
Message Waqar Chat AI...


+ New Chat
New Chat
Why is hybrid search bett...
Explain the different chu...
What are the production D...
Summarize this Document...
What are the production Do's and Don'ts of building a RAG system?
Based on the context, here are the production Do's and Don'ts for building a RAG system:
Do's:
Utilize hybrid search.
Consider agentic search.
Consider graph solutions for appropriate use cases.
Choose tools and databases based on your specific case and what you need to achieve.
Employ metadata filtering.
Use metadata to provide resources and citations from documents.
Evaluate your system embedding and re-indexing often to continuously improve it.
Don'ts:
The context states "don't it's like opposite of the do's" but does not explicitly list any "don'ts" in the production readiness section.
Message Waqar Chat AI...


+ New Chat
New Chat
Why is hybrid search bett...
Explain the different chu...
What are the production D...
Summarize this Document...
Summarize this Document
This document provides an introduction to Retrieval Augmented Generation (RAG) for LLMs, presented by Maxim, Global Head of Cloud Platforms and AI Director at Siklom.
The session focuses on how LLMs can utilize external, fresh, or proprietary information to provide value and improve reliability, accuracy, and efficiency. RAG is presented as a crucial stage in the LLM adoption landscape, fitting between prompt engineering and fine-tuning.
The core RAG pipeline involves:
1. Data Preparation: Cleaning and preparing the external knowledge.
2. Preprocessing (Chunking): Splitting documents into smaller, manageable "chunks." Common strategies include fixed-size, semantic, and recursive chunking, each with pros and cons related to context preservation and noise. The document stresses that chunking is critical for effective RAG.
3. Vectorization & Ingestion: Converting these chunks into numerical representations (embeddings) and storing them in a vector database.
4. Retrieval: When a user queries the system, a similarity search is performed against the vector database to retrieve the most relevant chunks.
5. Augmentation & Generation: The retrieved chunks are then provided to the LLM along with the user's query, allowing the LLM to generate a more accurate and contextually informed response.
RAG is beneficial because it helps maintain accuracy, is adaptable, cost-effective, and increases efficiency by overcoming the limitations of large LLM context windows, where too much data can lead to worse results or missing information.
Common challenges with RAG include optimizing chunking and context windows, ensuring high retrieval quality, managing latency, and implementing security/access control.
Recommendations for implementing RAG include starting with simple approaches, using metadata filtering, combining
Message Waqar Chat AI...


localhost:3000
Chat
+ New Chat
New Chat
Why is hybrid search bett...
Explain the different chu...
What are the production D...
Summarize this Document...
Welcome to Waqar Chat AI
Ask anything about your uploaded documents.
Message Waqar Chat AI...