AI Product Design Handbook

This handbook is a fictional but realistic internal document created to simulate how AI teams design, evaluate, and deploy AI-powered products in an organization. It is intended for testing Retrieval-Augmented Generation (RAG) systems with deep, multi-page, context-rich content.This handbook is a fictional but realistic internal document created to simulate how AI teams design, evaluate, and deploy AI-powered products in an organization. It is intended for testing Retrieval-Augmented Generation (RAG) systems with deep, multi-page, context-rich content.This handbook is a fictional but realistic internal document created to simulate how AI teams design, evaluate, and deploy AI-powered products in an organization. It is intended for testing Retrieval-Augmented Generation (RAG) systems with deep, multi-page, context-rich content.

1. Project Overview

Project Athena is an internal AI assistant designed to help product managers, developers, and support teams query internal documentation. The system uses a Retrieval-Augmented Generation architecture to ensure responses are grounded in verified company documents.Project Athena is an internal AI assistant designed to help product managers, developers, and support teams query internal documentation. The system uses a Retrieval-Augmented Generation architecture to ensure responses are grounded in verified company documents.Project Athena is an internal AI assistant designed to help product managers, developers, and support teams query internal documentation. The system uses a Retrieval-Augmented Generation architecture to ensure responses are grounded in verified company documents.

2. Target Users

Primary users include junior product managers, senior engineers, data scientists, and customer support leads. Each user group interacts with the system differently and expects varying levels of detail and technical depth.Primary users include junior product managers, senior engineers, data scientists, and customer support leads. Each user group interacts with the system differently and expects varying levels of detail and technical depth.Primary users include junior product managers, senior engineers, data scientists, and customer support leads. Each user group interacts with the system differently and expects varying levels of detail and technical depth.

3. Data Sources

The system ingests PDFs, internal wikis, architecture diagrams, and API specifications. PDFs are prioritized because they contain finalized and approved information.The system ingests PDFs, internal wikis, architecture diagrams, and API specifications. PDFs are prioritized because they contain finalized and approved information.The system ingests PDFs, internal wikis, architecture diagrams, and API specifications. PDFs are prioritized because they contain finalized and approved information.

4. Chunking Strategy

Documents are split into chunks of approximately 800 to 1200 characters with an overlap of 150 to 200 characters. This ensures semantic continuity across chunk boundaries.Documents are split into chunks of approximately 800 to 1200 characters with an overlap of 150 to 200 characters. This ensures semantic continuity across chunk boundaries.Documents are split into chunks of approximately 800 to 1200 characters with an overlap of 150 to 200 characters. This ensures semantic continuity across chunk boundaries.

5. Embedding Model Selection

SentenceTransformer models such as all-MiniLM-L6-v2 are selected due to their balance between performance and computational efficiency.SentenceTransformer models such as all-MiniLM-L6-v2 are selected due to their balance between performance and computational efficiency.SentenceTransformer models such as all-MiniLM-L6-v2 are selected due to their balance between performance and computational efficiency.

6. Vector Database

ChromaDB is used as the vector database because of its lightweight setup, persistence support, and developer-friendly API. Each chunk is stored along with metadata including source document, page number, and ingestion timestamp.ChromaDB is used as the vector database because of its lightweight setup, persistence support, and developer-friendly API. Each chunk is stored along with metadata including source document, page number, and ingestion timestamp.ChromaDB is used as the vector database because of its lightweight setup, persistence support, and developer-friendly API. Each chunk is stored along with metadata including source document, page number, and ingestion timestamp.

7. Retrieval Logic

At query time, the user query is embedded and compared against stored vectors using cosine similarity. The top-k most relevant chunks are retrieved and passed to the language model.At query time, the user query is embedded and compared against stored vectors using cosine similarity. The top-k most relevant chunks are retrieved and passed to the language model.At query time, the user query is embedded and compared against stored vectors using cosine similarity. The top-k most relevant chunks are retrieved and passed to the language model.

8. Prompt Engineering

The prompt explicitly instructs the LLM to answer only using the provided context and to say 'I do not know' if the answer is not present in the documents.The prompt explicitly instructs the LLM to answer only using the provided context and to say 'I do not know' if the answer is not present in the documents.The prompt explicitly instructs the LLM to answer only using the provided context and to say 'I do not know' if the answer is not present in the documents.

9. Failure Modes

Common failure modes include retrieving irrelevant chunks, partial answers due to insufficient context, and hallucinations when prompts are poorly structured.Common failure modes include retrieving irrelevant chunks, partial answers due to insufficient context, and hallucinations when prompts are poorly structured.Common failure modes include retrieving irrelevant chunks, partial answers due to insufficient context, and hallucinations when prompts are poorly structured.

10. Evaluation Metrics

The system is evaluated using qualitative feedback, answer groundedness, retrieval precision, and latency measurements.The system is evaluated using qualitative feedback, answer groundedness, retrieval precision, and latency measurements.The system is evaluated using qualitative feedback, answer groundedness, retrieval precision, and latency measurements.

11. Security Considerations

Sensitive documents are access-controlled, and embeddings are stored in isolated collections per department.Sensitive documents are access-controlled, and embeddings are stored in isolated collections per department.Sensitive documents are access-controlled, and embeddings are stored in isolated collections per department.

12. Future Improvements

Planned improvements include hybrid search, document-level reranking, conversation memory, and user feedback loops.Planned improvements include hybrid search, document-level reranking, conversation memory, and user feedback loops.Planned improvements include hybrid search, document-level reranking, conversation memory, and user feedback loops.

Conclusion

This document serves as a comprehensive reference for building and evaluating AI-powered document question-answering systems using Retrieval-Augmented Generation.This document serves as a comprehensive reference for building and evaluating AI-powered document question-answering systems using Retrieval-Augmented Generation.This document serves as a comprehensive reference for building and evaluating AI-powered document question-answering systems using Retrieval-Augmented Generation.