Project Implementation Plan

Project: Quick Best-Performance RAG System for Systematic Review
Name: Clifford Hepplethwaite
Email: clifford@tumpetech.com

Phase 1: Environment Setup

Objective: Prepare the local development environment and core dependencies.

Tasks:

- Install Python 3.9 or higher.
- Create and activate a virtual environment.
- Install required packages:

  ```
  pip install llama-index streamlit openai qdrant-client
  ```

- Create a `.env` file and securely store the following keys:
  - `OPENAI_API_KEY`
  - `QDRANT_API_KEY`
- Confirm API connectivity to OpenAI and Qdrant.

Phase 2: Document Ingestion

Objective: Load academic PDFs and extract structured text and metadata.

Tasks:

- Create a `data/` directory and add sample academic PDFs.
- Use LlamaIndex's `PDFLoader` or `SimpleDirectoryReader` to ingest files.
- Extract key metadata: title, authors, sections, etc.
- Review and verify extracted output against original PDFs.

Phase 3: Chunking and Embedding

Objective: Divide documents into structured chunks and generate embeddings.

Tasks:

- Use `SectionNodeParser` to chunk documents by logical section headings (e.g., Introduction, Methods).
- Set chunk overlap to maintain continuity (e.g., 20%).
- Generate embeddings using `text-embedding-3-large` from OpenAI.
- Store embedding vectors with metadata: section name, page number, source filename.

Phase 4: Vector Indexing

Objective: Store document embeddings in a high-performance vector database.

Tasks:

- Create a collection in Qdrant Cloud.
- Push vectors and their metadata to the Qdrant index via API.
- Test basic vector similarity queries for sanity check.
- Confirm accurate retrieval of original document chunks from Qdrant.

Phase 5: Retrieval Pipeline

Objective: Implement semantic and hybrid retrieval from indexed data.

Tasks:

- Set up LlamaIndex’s hybrid retriever (vector + keyword).
- Fine-tune retrieval parameters for relevance and speed.
- (Optional) Integrate a reranker (e.g., `bge-reranker` or Cohere) to reorder top results.
- Validate retriever output using academic-style queries.

Phase 6: LLM Integration

Objective: Generate context-aware, citation-style answers using GPT-4 Turbo.

Tasks:

- Configure OpenAI GPT-4 Turbo for text generation.
- Construct prompt templates to emphasize citation formatting and academic tone.
- Inject top retrieved chunks into the prompt context.
- Parse and return generated answers with reference to source metadata.

Phase 7: Frontend Interface

Objective: Create a simple, usable interface for query and review.

Tasks:

- Build a minimal interface using Streamlit.
- Add components:
    - Text input for questions
    - File uploader (optional)
    - Display area for generated answers and source snippets
- Test interface with several questions and documents.
- Refine layout and UX as needed.

Phase 8: Testing and Validation

Objective: Ensure system stability and response accuracy.

Tasks:

- Conduct unit tests for ingestion, chunking, and embedding.
- Perform integration testing across all components (input to output).
- Run multiple end-to-end tests simulating systematic review scenarios.
- Validate output against known answers or summaries.

Phase 9: Deployment

Objective: Deploy the working application for internal or public access.

Tasks:

- Choose hosting platform (e.g., Streamlit Cloud, Render).
- Set environment variables and secure API keys on the host.
- Push Streamlit application and test hosted version.
- Monitor performance, errors, and usage logs post-deployment.

Phase 10: Post-MVP Enhancements

Objective: Improve accuracy, usability, and scalability.

Tasks:

- Add a reranker if not already implemented.
- Enable answer exports (e.g., Markdown, PDF, or BibTeX).
- Add advanced document navigation or filter sidebar.
- Explore FastAPI + React frontend for multi-user deployments.
- Implement basic user authentication if sharing externally.
- Monitor API costs and set alerts if needed.

Project Completion Checklist

Task Group	Completed
Environment Setup	☐
PDF Ingestion	☐
Chunking & Embeddings	☐
Vector Indexing	☐
Retrieval Working	☐
GPT-4 Answer Generation	☐
Streamlit UI Functional	☐
Full Pipeline Tested	☐
MVP Deployed	☐
Enhancements Reviewed	☐
