Product Requirements Document (PRD)

Product Name: Systematic Review AI Assistant
Prepared by: Clifford Hepplethwaite
Email: clifford@tumpetech.com
Date: 2nd July, 2025
Version: 1.0

---

1. Purpose

This product enables academic and research professionals to query large sets of academic PDFs, such as systematic reviews and technical papers, and receive accurate, citation-backed answers using Retrieval-Augmented Generation (RAG). It simplifies literature analysis, speeds up systematic reviews, and improves knowledge accessibility.

---

2. Scope

The app focuses on supporting:

- Researchers conducting systematic literature reviews
- Policy analysts synthesizing complex reports
- Educators and students analyzing academic texts

It will support:

- Uploading of PDFs
- Semantic search through RAG
- Context-rich, cited responses generated by GPT-4 Turbo
- A simple, accessible web interface (Streamlit)

---

3. Target Users

User Type	Needs
Academic Researchers	Speed up literature synthesis, cite sources accurately
Research Assistants	Reduce manual review effort
NGOs/Consultants	Analyze large policy or donor documents
Postgraduate Students	Answer questions based on their thesis material

---

4. Assumptions

- Users will have access to internet to connect to OpenAI and Qdrant APIs.
- PDFs will be in English and contain structured academic content.
- GPT-4 Turbo and OpenAI embeddings will be available through paid API keys.
- Users are familiar with basic research workflows but not necessarily with AI tools.

---

5. Features

5.1 Document Upload & Ingestion

- Upload PDF files through the UI or automatically from a folder
- Extract full text and metadata (e.g., title, author, sections)

5.2 Chunking

- Split documents into sections using `SectionNodeParser`
- Maintain context with overlapping chunking

5.3 Embedding & Indexing

- Convert chunks into semantic vectors using `text-embedding-3-large`
- Store vectors and metadata in Qdrant Cloud

5.4 Retrieval

- Retrieve relevant chunks using semantic + keyword (hybrid) search
- Optional: Add reranker to boost relevance

5.5 Answer Generation

- Use GPT-4 Turbo to answer user queries using retrieved context
- Cite sources clearly (e.g., author, page, section)

5.6 Frontend Interface

- Simple, clean interface using Streamlit
- Input box for questions
- Output area with answer and citations
- (Optional) Document viewer or sidebar for document selection

5.7 System Settings

- Manage API keys via environment variables
- Show cost estimation per request (future enhancement)

---

6. User Flows

6.1 Query Flow

1. User uploads documents
2. System processes and indexes documents
3. User enters question
4. App retrieves top-k chunks
5. GPT-4 Turbo generates response with citations
6. User sees answer and source breakdown

---

7. Non-Functional Requirements

Category	Requirement
Performance	Response time under 5 seconds (cached preferred)
Usability	Simple interface; requires no installation knowledge
Security	API keys stored securely in environment variables
Scalability	Indexing and retrieval should handle >100 documents
Reliability	System must gracefully handle API failures or timeouts

---

8. Dependencies

Component	Description
OpenAI API	Embedding and LLM (GPT-4 Turbo)
Qdrant Cloud	Vector storage
LlamaIndex	Chunking, embedding, and retrieval library
Streamlit	Frontend for end users

---

9. Success Metrics

Goal	Metric
Useful Answers	80%+ user satisfaction (manual review)
Time Saved per Review	>10 hours saved per researcher
Accuracy of Citations	95% citation traceability
Query Latency	<5 seconds for 80% of queries
Uptime	>99% in production

---

10. Future Enhancements

- Reranker integration (e.g., Cohere or bge-reranker)
- Long-term memory / user history
- Export responses (PDF, Word, BibTeX)
- Multi-user dashboard with authentication
- Multilingual document support
- Document filter by date, topic, or source
