RAG-Based Financial Document Analysis System: Methodology & Reasoning

Project: Conceptualization and implementation of a RAG-based information retrieval system for querying SEC filing document

Date: September 3, 2025

Table of Contents

---

1. Executive Summary

This document outlines the architecture, methods, and design rationale for a sophisticated Retrieval-Augmented Generation (RAG) system built to answer complex queries about dense financial reports. Standard RAG pipelines often struggle with the specific, keyword-driven nature of financial and legal documents, where semantic similarity alone is insufficient. Our system addresses this challenge by implementing a multi-stage, hybrid retrieval process that combines semantic search with advanced keyword analysis and a cross-encoder reranker. The goal is to deliver highly accurate, context-aware, and verifiable answers by intelligently filtering, scoring, and synthesizing information from source PDFs.

---

2. System Architecture Overview

The system is divided into two primary phases: **Offline Indexing** (a one-time process per document set) and **Online Querying** (executed for each user question).

- **Offline Indexing Pipeline**: Load PDFs → Extract Text & Metadata → Chunk Documents → Embed Chunks → Store in VectorDB.
- **Online Querying Pipeline**: User Query → Initial Vector Search → **Advanced 3-Stage Reranking** → Augment Context → LLM Generation → Final Answer & Sources.

---

3. Core Reasoning & Design Philosophy 🧠

This system is built on the philosophy that for specialized domains like finance, neither simple keyword search nor pure semantic search is sufficient. Financial documents are dense, jargon-heavy, and require high precision. Our architecture is designed to create a "funnel" that intelligently filters information to find the best possible context.

- **Context is King (SEC Item Extraction)**
  - **Reasoning**: Simply chunking a PDF loses its structure. By extracting SEC Item numbers (e.g., "Item 1A. Risk Factors," "Item 7. Management's Discussion"), we provide crucial **structural context** to every chunk. This helps the system differentiate between forward-looking statements, risk analysis, and historical financial data.
- **From Broad to Specific (The Retrieval Funnel)**
  - **Reasoning**: Our 3-stage reranking is a deliberate funneling process.
    1. We start with a **wide-net semantic search** to ensure we capture all potentially relevant documents, even if they use different wording.
    2. Then, we apply a **lexical filter (keyword scoring)** because financial queries often depend on specific terms, numbers, and dates that semantic search alone can miss.
    3. Finally, we use the powerful **cross-encoder** on a much smaller, pre-qualified list. This is a computationally smart way to apply the most accurate model only where it matters most, ensuring both speed and precision.
- **Trust and Verifiability (Grounded Generation)**
  - **Reasoning**: An answer from an AI about financial data is useless if it can't be trusted. We strictly limit the LLM's role to **synthesizing answers from provided sources**, not using its general knowledge. The enforced JSON output with source `ref_ids` is the final step in this chain of trust, making every claim **fully verifiable** by tracing it back to the original document.

---

4. Detailed Methodology & Implementation

4.1. Data Ingestion and Preprocessing

- **Method**: We use `PyPDFLoader` to extract raw text, which is then split using `RecursiveCharacterTextSplitter`. A set of regular expressions

(`_sec_item_patterns`) identifies and extracts SEC filing item numbers and titles from the text, which are attached to each chunk as metadata.
- **Rationale**: The recursive splitter is chosen to maintain semantic coherence by breaking text along natural boundaries (paragraphs, lines). The SEC Item metadata is the implementation of our "Context is King" philosophy, enriching each chunk with vital structural information.

4.2. Embedding and Vector Storage

- **Method**: I used Google's `text-embedding-004` model to generate vector representations of the text chunks. These embeddings are stored in `ChromaDB`, a local, persistent vector database.
- **Rationale**: Google's models provide state-of-the-art semantic understanding for the initial retrieval stage. ChromaDB offers a lightweight, fast, and easy-to-integrate solution for storing and querying these embeddings.

4.3. The Advanced Reranking Pipeline

This is the core of our "Broad to Specific" retrieval funnel.

- **Stage 1: Keyword-Based Filtering and Scoring**
  - **Method**: Documents are scored based on lexical overlap, with a custom list of financial stopwords removed and crucial terms (e.g., "revenue," "net income") protected. It also specifically looks for patterns like dollar amounts (`$`), percentages (`%`), and years.
  - **Rationale**: This stage acts as a high-recall filter, ensuring that documents with precise keyword matches are promoted, counteracting the "fuzziness" of pure semantic search.
- **Stage 2: Cross-Encoder Semantic Reranking**
  - **Method**: The filtered documents are reranked using `FlashRank`, which wraps a **cross-encoder model**. Unlike the initial embedding model (a bi-encoder), a cross-encoder evaluates the query and a document *together*.
  - **Rationale**: This method provides far more accurate relevance scoring by analyzing the direct interaction between query and context. We apply it only to the pre-filtered candidates to maintain high performance.
- **Stage 3: Hybrid Scoring**
  - **Method**: The final score is a weighted average of the keyword score and the cross-encoder score. $$ \text{Final Score} = (1 - w_{\text{hybrid}}) \times \text{Score}*{\text{Cross-Encoder}} + w*{\text{hybrid}} \times \text{Score}_{\text{Keyword}} $$
  - **Rationale**: This **hybrid approach** creates a robust balance, combining the nuanced understanding of the cross-encoder with the precision of keyword matching.

4.4. Generation and Prompt Engineering

- **Method**: The top-ranked chunks are formatted into a context block for a **Google Gemini LLM**. A strong `SYSTEM_PROMPT` instructs the model on its persona, its constraints (use context only), and its output format: a structured JSON object.
- **Rationale**: This implements our "Trust and Verifiability" principle.
    - **Preventing Hallucinations**: Strictly grounding the LLM in the provided text is non-negotiable for factual domains like finance.

Ensuring Verifiability: The required JSON output ensures every answer can be programmatically parsed and tied back to its source chunks, making the system's claims transparent and auditable.
JSON
{"answer": "...", "ref_ids": ["doc_name|p34|c2", ...]}

    -

---

5. Future Improvements

- **Query Transformation**: Add logic to break down complex questions (e.g., "Compare the revenue growth and net income margin over the last two years") into smaller, answerable sub-queries.
- **Visual Document Understanding**: Use multi-modal models or techniques to extract and understand information from tables, charts, and highlighted text within the PDFs, which currently go unprocessed.

---

6. Conclusion

This RAG system represents a significant step beyond generic implementations by tailoring its retrieval and generation processes to the specific challenges of financial document analysis. By combining a multi-stage hybrid retrieval funnel with a strong emphasis on verifiable, grounded generation, the system delivers precise, trustworthy, and contextually aware answers.