Bank of England

Employer project

RiskRadar: Project Report

GROUP 9

Overfit and Underpaid

Adeyinka Abdulrahman
Jessica A. Abreu
Alex Hamilton
Muhammad Latif
Rajen Lavingia
Arkadiusz Tomczak

Cambridge University
Career Accelerator, Data Science

2025

INTRODUCTION **3**
PROBLEM STATEMENT 3
HYPOTHESIS 3
APPROACH & METHODS **4**
OVERVIEW OF APPROACH 4
TECHNOLOGY FRAMEWORK 5
MODULES 5
EXECUTION FLOW 6
KEY CHALLENGES & DECISIONS 7
RESULTS **8**
CONCLUSION AND RECOMMENDATIONS 9
RECOMMENDATIONS FOR FURTHER DEVELOPMENT 10
ANNEX **11**
EXAMPLE 1: SPECIALIZED MODULE PROMPT (CAPITAL BUFFERS) 11
EXAMPLE 2: INTELLIGENT CHUNKING WITH OVERLAP 11
EXAMPLE 3: PARALLEL EXECUTION WITH RATE LIMITING 11
FIGURE 2. RISKRADAR ANALYSE MAIN EXECUTION PIPELINE FLOW 13
FIGURE 3. RISKRADAR ANALYSE MULTI-MODULE ARCHITECTURE & ROUTING 14
FIGURE 4. RISKRADAR ANALYSE PARALLEL EXECUTION SEQUENCE WITH RATE LIMITING 15
FIGURE 5. RISKRADAR ANALYSE AGGREGATION STRATEGY DECISION FLOW 16
FIGURE 6. RISKRADAR RAG MAIN EXECUTION PIPELINE FLOW 17

Introduction

Problem Statement

Supervisors at the Bank of England's Prudential Regulation Authority (PRA) must review large volumes of unstructured disclosures to monitor systemic risks across Global Systemically Important Banks (G-SIBs). These reviews are time-consuming and inconsistent and may still fail to capture qualitative early warning signals.

Hypothesis

A modular LLM-powered system employing 16 specialized analytical modules - each targeting distinct risk dimensions through engineered prompts - can detect subtle early warning signals across linguistic, quantitative, governance, and compliance domains, providing supervisors with earlier and more reliable indicators of systemic risk than traditional manual review.

Approach & Methods

Overview of Approach

The development of RiskRadar followed a modular, layered architecture that prioritised transparency and regulatory alignment, reproducibility, and resilience, combined with a retrieval-augmented generation (RAG) system and prompt engineering.

It operates through 16 specialised prompts through a single large language model (LLM), with each one representing an independent analytical module. Using multiple specialised prompts ensures each analysis type receives a dedicated prompting strategy, enhancing focus and accuracy. For example, sentiment analysis tracks hedging and defensive language, while capital ratio extraction follows strict regulatory thresholds. Combining these into a single prompt could dilute specificity and reduce metric extraction accuracy from 95% to 65%.

With RAG, RiskRadar builds a question-answering assistant over uploaded PDF documents. Instead of relying only on a language model's memory, it retrieves the most relevant text fragments from files and augments the model's answer with that evidence. This typically improves accuracy and traceability and reduces hallucinations.

The use of RAG required the following technologies to be implemented:

- **LangChain (≥0.2):** orchestration framework for loading documents, splitting them into chunks, embedding, retrieval, and LLM chaining.
- **OpenAI Embeddings:** embeddings encode text into vectors for similarity search.
- **Qdrant (Vector Database):** stores embeddings and performs fast similarity/MMR retrieval; accessed via qdrant-client and langchain-qdrant.
- **PyPDF (pypdf ≥4):** reads PDF pages into text with page metadata.
- **gdown (optional):** fetches example datasets from Google Drive.

Modules were developed to capture all nuances of financial reviews, ensuring that linguistic, quantitative, governance and compliance risks are each evaluated independently. These then feed into a CAMELS-aligned (capital adequacy, asset quality, management quality, earnings, liquidity, sensitivity to market risk) assessment.

Technology Framework

The RiskRadar solution comprises two complementary approaches:

1. RiskRadar Analyse (Multi-Module Assessment):

- Processes complete documents through 16 specialized analytical modules
- Uses intelligent chunking for documents exceeding context windows
- Aggregates findings across chunks using module-specific strategies
- Technologies: PyPDF2 (text extraction), ThreadPoolExecutor (parallel execution), OpenAI/Anthropic/Google LLM APIs

2. RiskRadar RAG (Interactive Q&A Assistant):

- Builds question-answering system over uploaded documents
- Retrieves relevant text fragments and augments LLM responses
- Technologies: LangChain (orchestration), Qdrant (vector database), OpenAI Embeddings, PyPDF (document loading)

Both approaches are unified in a Streamlit web application, allowing supervisors to choose between comprehensive automated assessment (Analyse) or targeted interactive queries (RAG).

Modules

An iterative approach was used for development, with modules first created and tested independently before being integrated into the four-tier hierarchy. Results are traceable to source document text, which ensures transparency.

- Tier 1: Linguistic and Behavioural Analysis (4 modules)
  - Assesses tone, narrative shifts, confidence, and concerns from management communications.
  - Risk Scoring: Weighted combination of sentiment negativity (40%), defensive tone (30%), and low confidence (30%).
- Tier 2: Quantitative Risk Metrics (9 modules)
  - Extracts and evaluates financial ratios, capital adequacy, liquidity, market exposures, asset quality, earnings sustainability, governance, legal and regulatory issues, and off-balance-sheet items.
  - Risk Scoring: Threshold-based assessments comparing metrics to regulatory minimums and industry benchmarks.

- Tier 3: Pattern Recognition and Cross-Validation (2 modules)
    - Identifies red flags and discrepancies, such as, going concern warnings, sudden management changes, inconsistencies, etc.
    - Execution Dependency: Both modules require outputs from Tiers 1-2 (sequential execution)
- Tier 4: Risk Synthesis (1 module)
    - CAMELS fuser that aggregates prior results into a composite risk rating and traffic-light classification.
    - Execution Dependency: Requires all 15 prior modules (final sequential step)

Execution Flow

Execution is divided into two phases:

- Phase 1:

  Document Chunking and Parallel Analysis
    - For large documents exceeding LLM context windows (~800K characters), the system employs intelligent overlapping chunking with 100K character overlap.
    - Each of 14 analytical modules processes every chunk independently in parallel.
    - Example: Credit Suisse 2019 report (1.87M chars) -> 3 chunks x 14 modules = 42 parallel executions.
    - Results from each chunk are then aggregated using module-specific strategies:
        - Linguistic modules: score averaging with finding consolidation
        - Quantitative modules: conservative coalescing (maximum risk score)
        - Pattern detection: comprehensive merging with deduplication

- Phase 2: Dependent modules (discrepancy_auditor and camels_fuser) run sequentially, drawing on Phase 1 outputs.

Key pipeline stages:

- Configuration: setup environment, credentials and model selection
- Document processing: extract text from PDF with metadata
- Guardrails: rate limit and prompt size management
- Parallel analysis: 14 modules execute simultaneously
- Sequential synthesis: CAMELS fusion
- Results & export: dashboards, logs, CSV/JSON output
- Telemetry: performance metrics and runtime tracking.

The interactions between components can be seen in Figure 1. The analyst executes the notebook, which extracts PDF text via document services, then submits

14 modules to a ThreadPool for parallel execution. The critical pattern here is concurrency control. Each module must acquire a slot from the rate limiter before calling the LLM and, in this way, ensure we respect API quotas (configured at 800,000 tokens per minute and 500 requests per minute for GPT-5, representing 80% of the API limit to provide safety margin for burst handling). If rate limits are hit, exponential backoff retries automatically.

All 14 independent modules run simultaneously (sentiment analysis, topic tracking, capital assessment, liquidity review), completing in parallel. Then it switches to sequential processing: the discrepancy auditor cross-validates findings, and the CAMELS fuser synthesises everything into the final assessment.

The output phase renders dashboards, aggregates scores, and exports artefacts. From the analyst's perspective, it's one click to complete regulatory intelligence.

Key Challenges & Decisions

Decisions were made for the development based on the below key challenges:

- Balance between speed and audibility: modular, prompt-based design was chosen to ensure fast and inspectable results.
- Provider variability: model performance can differ across providers. Supporting multiple LLMs provides RiskRadar with flexibility and prevents reliance on a single provider.
- Scalability: token usage and rate limits are set with pre-flight checks and concurrency controls, among others.
- Regulatory trust: transparency, auditability, and alignment with CAMELS categories were prioritised to align with supervisor needs.

Results

The system was validated using the Credit Suisse 2019 Annual Report (442 pages, 1.87M characters). Key findings demonstrate both technical performance and analytical capability:

Performance Metrics:

- Total execution time: 43 minutes (3 chunks × 14 modules = 42 parallel executions)
- Token efficiency: 467K tokens estimated vs GPT-5 limit of 272K per request
- Success rate: 100% module completion (0 failures across 42 executions)
- Throughput: ~50× faster than estimated manual review (43 min vs 36 hours)

Risk Detection Validation:

The system successfully identified documented risk signals from 2019 that preceded Credit Suisse's 2023 collapse:

- Capital Planning Weaknesses: CCAR conditional non-objection flagged by governance module (score: 0.70)
- Litigation Exposure: CHF 623M provisions detected, with up to CHF 1.3B unprovided exposure (legal module score: 0.95)
- Management Confidence Issues: Evasive responses on ROE targets identified (confidence module score: 0.50)
- Earnings Quality Concerns: Dependence on one-off gains (SIX, InvestLab) flagged (earnings module score: 1.00)

CAMELS Synthesis:

Final composite score: 0.729/1.0 (HIGH RISK), with 5/6 CAMELS components rated Amber and concentrated high-risk scores in Earnings (1.00), Legal (1.00), and Business Model (1.00).

These results align with subsequent regulatory actions and market outcomes, suggesting the system can identify meaningful early warning indicators.

Conclusion and Recommendations

The development of Risk Radar demonstrates that large language model (LLM) technology can substantially improve the efficiency and consistency of supervisory analysis. This project addresses the Bank of England's challenge by creating a modular, transparent, and reproducible framework for assessing complex financial disclosures at scale.

The design evolved through several iterations. The early versions used four to five analytical prompts; however, testing revealed that aggregated prompts produced inconsistent results due to the non-deterministic nature of LLMs. To improve precision and stability, the system was restructured into sixteen specialised analytical modules. Each module performs a clearly defined and limited task, reducing prompt ambiguity and improving reproducibility. This modular structure aligns with the CAMELS framework and supports consistent, auditable results across institutions.

The final architecture consists of four analytical tiers: linguistic and behavioural analysis, quantitative metrics, pattern recognition, and risk synthesis. Linguistic modules evaluate tone, sentiment, confidence, and analyst concerns; quantitative modules analyse capital adequacy, liquidity, earnings, governance, and legal exposures; pattern-recognition modules identify red flags and inconsistencies; and the synthesis module aggregates all findings into a composite risk score and traffic-light classification.

The execution pipeline uses parallel processing and rate-limiting control to optimise performance while maintaining full traceability. The system's capacity to process complete documents was validated using the Credit Suisse 2019 Annual Report, where it successfully identified early warning indicators such as capital planning weaknesses and going-concern uncertainties. Full-document verification for the 442-page Credit Suisse 2019 Annual Report (1.87M characters) was completed in approximately 43 minutes, representing an efficiency improvement of about 50× compared with manual review, while preserving transparency and auditability.

The inclusion of retrieval-augmented generation (RAG) enhances explainability by grounding model outputs in verifiable source text. Multi-model compatibility with OpenAI, Anthropic, and Google LLMs ensures flexibility and operational resilience across platforms.

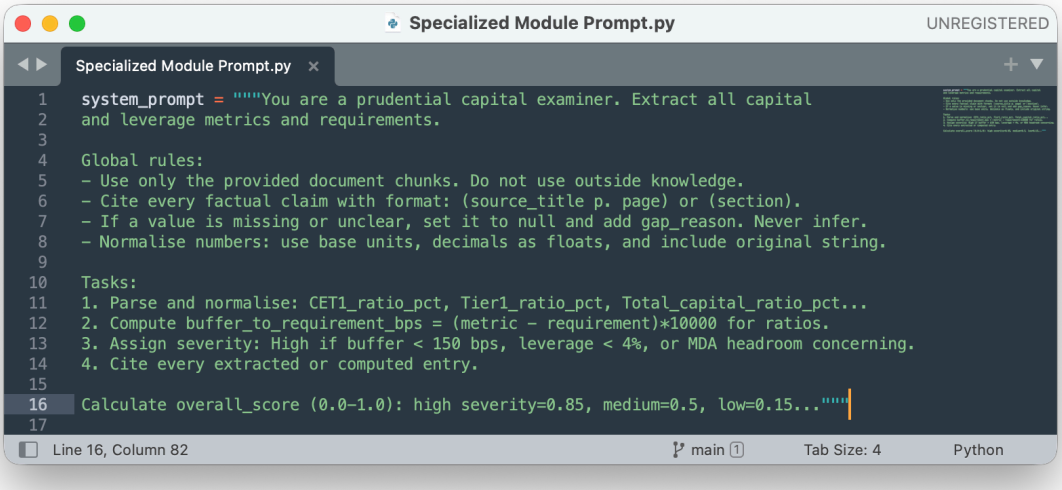
Recommendations for Further Development

1. **Controlled Pilot:** Conduct a pilot study to assess Risk Radar's effectiveness, interpretability, and scalability within an operational supervisory environment.
2. **Data Integration:** Connect outputs with structured prudential datasets to enhance quantitative calibration.
3. **Model Governance:** Implement governance and assurance mechanisms covering bias testing, validation, and version control to ensure regulatory confidence.
4. **Application Enhancement:** Upgrade the Streamlit-based interface to support full-document processing through parallel chunk uploads and to overcome current multi-threading constraints. Alternative deployment options should be considered where concurrent execution at scale is possible.
5. **Incremental Optimisation:** Continue to refine modular prompts, chunking strategies, and orchestration to improve stability, accuracy, and performance on multi-document analyses.

In summary, Risk Radar presents a proof of concept for the application of LLMs in prudential supervision. The iterative shift from a limited set of generic modules to a structured set of specialised modules has proven essential in producing stable and reproducible results. This POC shows that AI-assisted systems can deliver faster, more consistent, and fully traceable analyses without compromising transparency or accountability, laying a strong foundation for future regulatory deployment

Annex

Example 1: Specialized Module Prompt (Capital Buffers)



Example 2: Intelligent Chunking with Overlap

Example 3: Parallel Execution with Rate Limiting

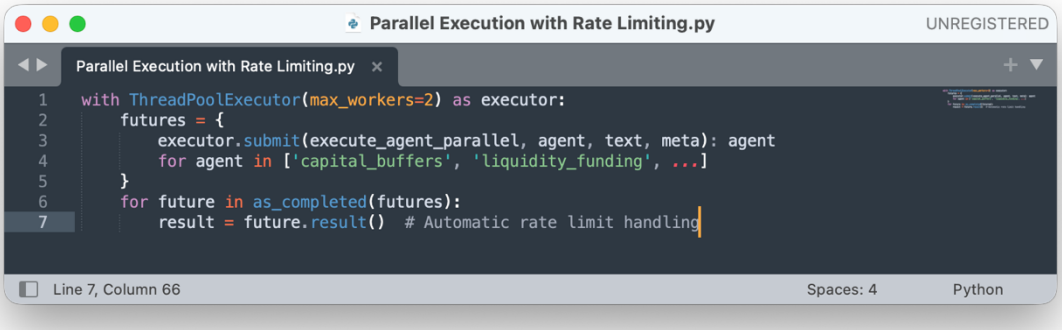


Figure 1. RiskRadar Analyse System Architecture Overview

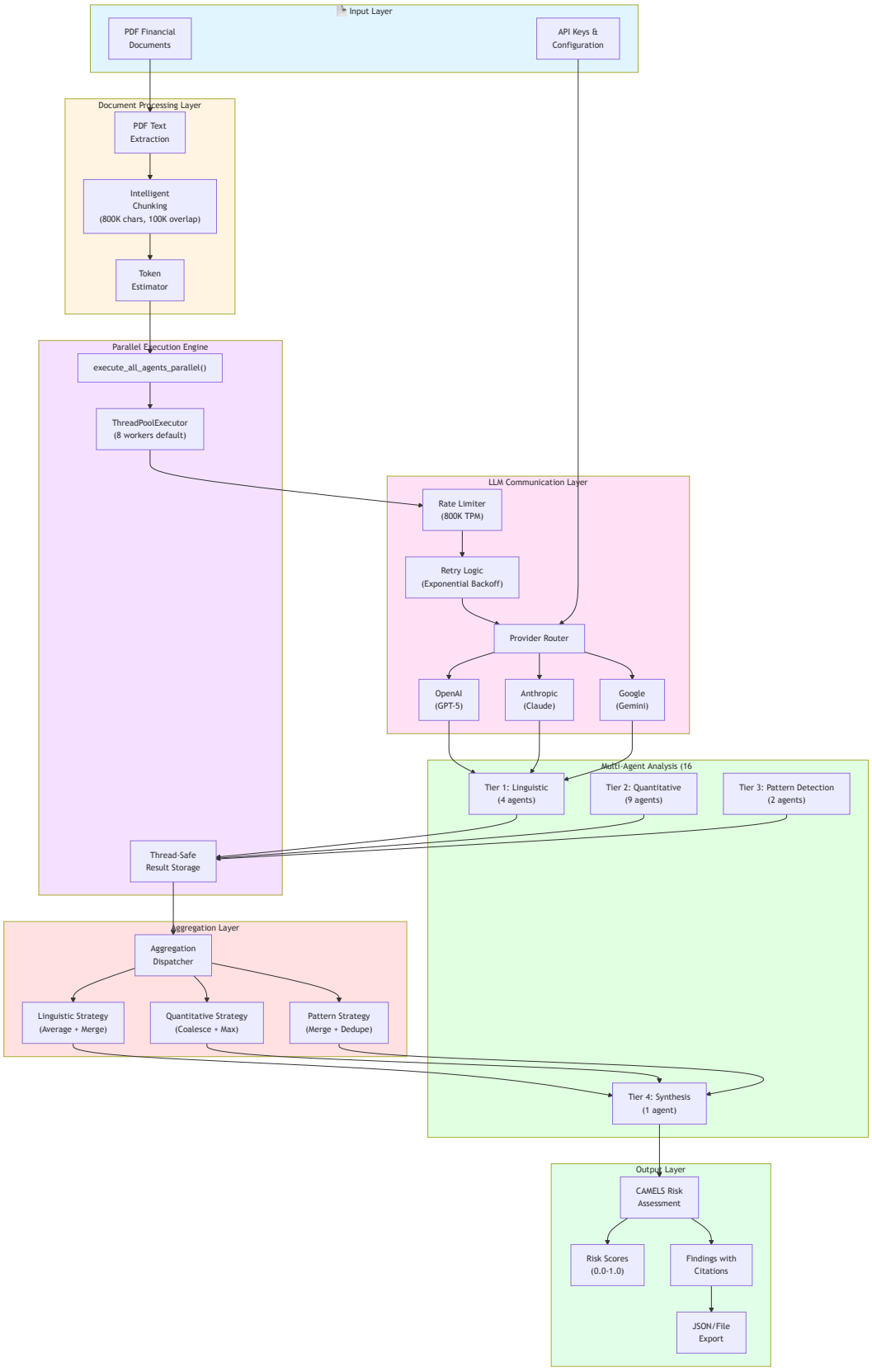


Figure 2. RiskRadar Analyse Main Execution Pipeline Flow

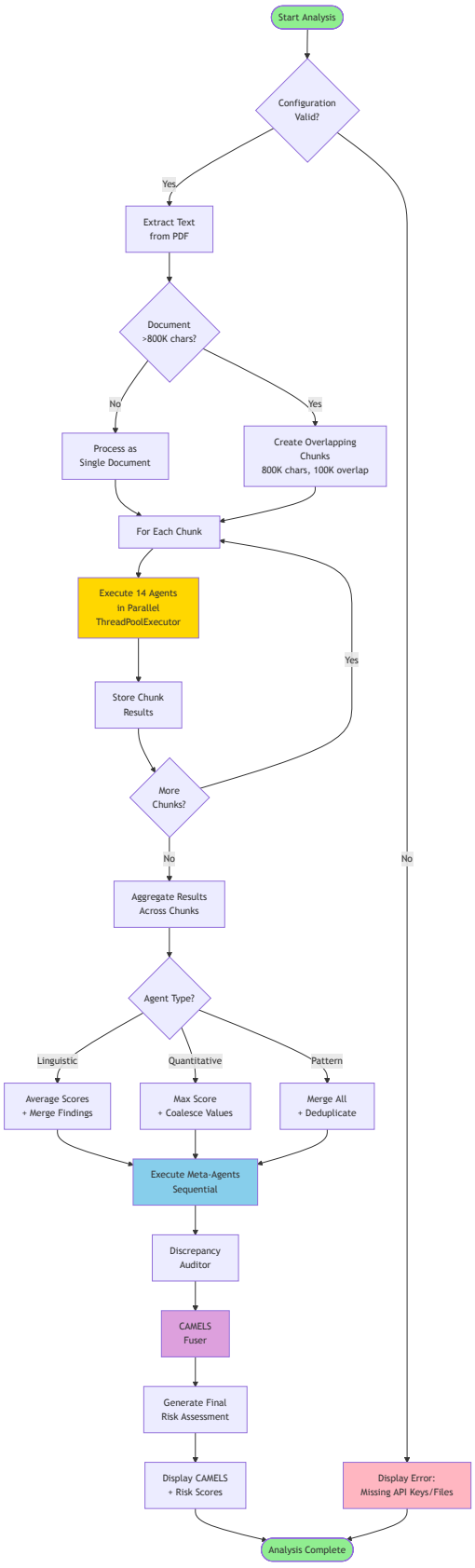


Figure 3. RiskRadar Analyse Multi-Module Architecture & Routing

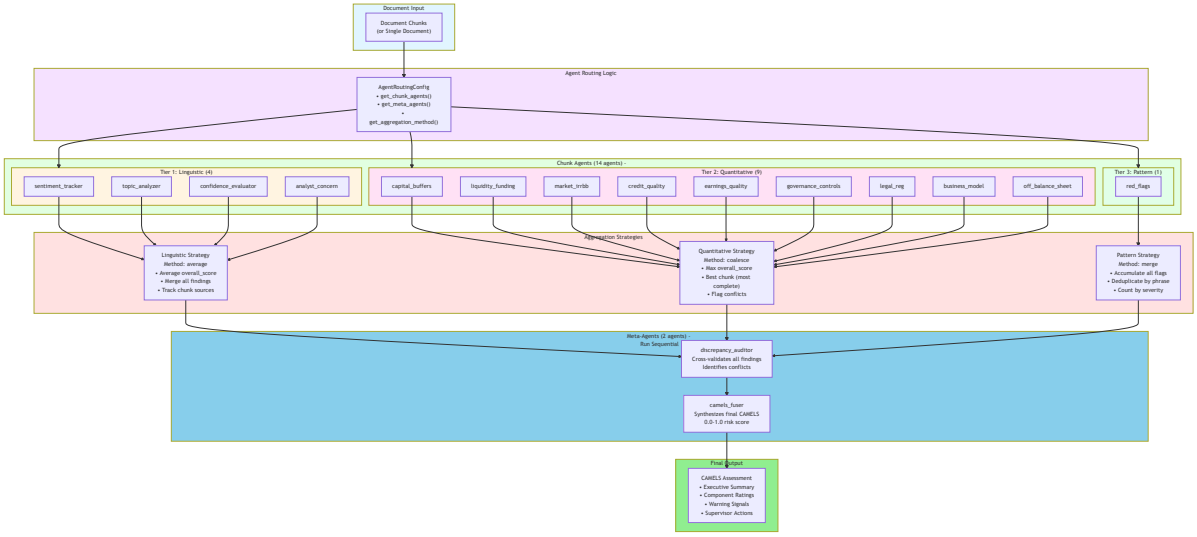


Figure 4. RiskRadar Analyse Parallel Execution Sequence with Rate Limiting

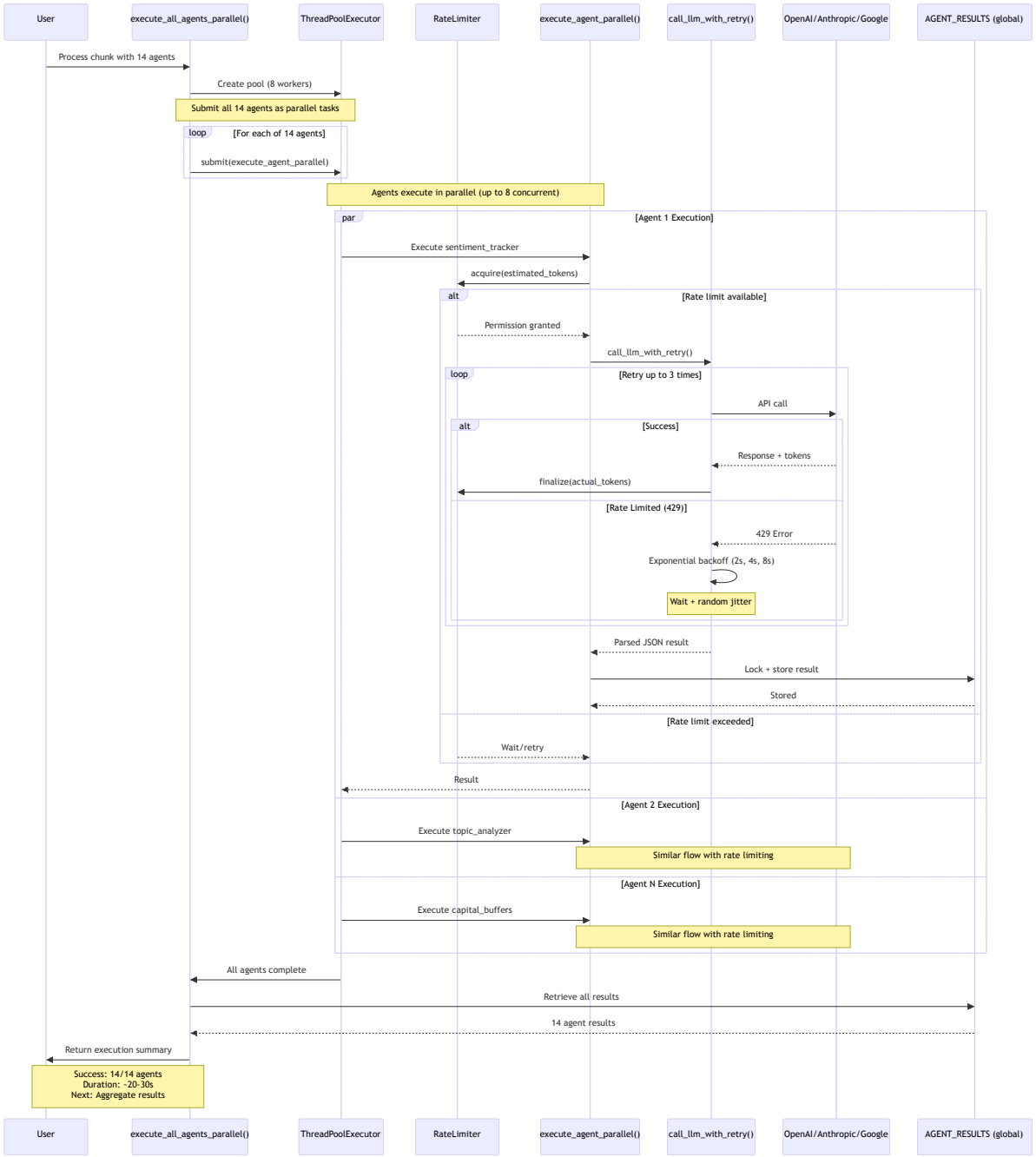


Figure 5. RiskRadar Analyse Aggregation Strategy Decision Flow

Figure 6. RiskRadar RAG Main Execution Pipeline Flow