Build a Retrieval-Augmented Generation (RAG) system to answer queries about a company’s financial performance using financial reports (e.g., Meta’s Q1 2024 PDF report). The project consists of 3 steps with increasing complexity. Use any open-source tools or low-cost APIs and document your process.

Deliverables per Step

● Source code (Jupyter notebook/scripts)

● Brief report (Markdown or PDF) covering:

○ Approach & tools

○ Challenges & results

● Sample outputs for test queries

Step 1: Basic RAG Pipeline

Objective: Build a simple RAG pipeline for factual QA from a single financial report.

Tasks

● Preprocessing: Extract and clean text from PDF.

● Chunking & Embedding: Split into chunks; generate embeddings with an open-source model.

● Retrieval: Use vector similarity (e.g., cosine) to retrieve top-3 relevant chunks.

Generation: Answer queries using an open-source LLM with prompt like: Based on the following context: {context}

Answer the query: {query}

●

Test Queries

● “What was Meta’s revenue in Q1 2024?”

● “What were the key financial highlights for Meta in Q1 2024?”

Evaluation

● Understanding of RAG basics

● Tool selection

● Answer correctness

Step 2: Structured Data Integration

Objective: Integrate structured data (e.g., tables) into the RAG pipeline.

Tasks

● Table Extraction: Parse tables into structured formats (e.g., DataFrame, JSON).

● Hybrid Retrieval: Combine vector search (text) + keyword/SQL-like search (structured).

Prompt Update: Text context: {text\_context}

Structured data: {structured\_data}

Answer the query: {query}

Test Queries

● “What was Meta’s net income in Q1 2024 compared to Q1 2023?”

● “Summarize Meta’s operating expenses in Q1 2024.”

Evaluation

● Structured data handling

● Hybrid search effectiveness

● Numerical answer accuracy

Step 3: Query Optimization & Advanced RAG

Objective: Enhance pipeline relevance and accuracy.

Tasks

1. Query Optimization: Use LLMs or rules to rewrite/improve queries.

2. Advanced Retrieval:

○ Rerank results using cross-encoder/relevance model

○ Experiment with chunk sizes

○ Optional: Iterative retrieval

Evaluation Framework:

○ Retrieval: Precision@k, Recall@k, MRR

○ Answer: BLEU, ROUGE, or factual accuracy

○ End-to-End: Manual rubric or user scoring

4. Test Set: 15 diverse queries (factual, comparative, multi-step)

5. Performance Analysis: Analyze failure cases & compare configurations

6. Ablation Study: Remove one component (e.g., reranking) and measure impact

7. Improvement Proposals: Suggest at least 2 enhancements with justification

Evaluation

● Evaluation framework quality

● Analysis depth

● Use of advanced retrieval methods

● Research-backed improvements

Submission Guidelines

● Code (GitHub or ZIP) with comments

● Single report covering all steps

● Sample outputs in a folder or notebook