Retail Insights Assistant (GenAI + Scalable Data System)

Overview

You are tasked with designing and developing an intelligent Retail Insights Assistant — a GenAI-powered solution capable of analyzing large-scale retail sales data, generating automated business insights, and answering ad-hoc analytical questions in natural language.
This assignment evaluates your ability to combine data engineering, LLM integration, and scalable system design.

Problem Statement

Retail organizations often deal with large volumes of sales data across regions, products, and time periods. Executives and analysts want to query this data conversationally (e.g., *"Which category saw the highest YoY growth in Q3 in the North region?"*) and receive instant summaries and insights.
You will build a GenAI-driven assistant that can:

- Interpret natural language queries about sales performance.
- Summarize key insights from a structured dataset.
- Scale efficiently to 100GB+ of historical data.

Core Requirements

1. Functional Scope

Your solution should:

- Accept either:
    - A sales CSV dataset, or
    - A summarized sales report (Excel, JSON, or text).
- Support two primary use cases:
    - Summarization Mode: Generate a concise, human-readable summary of performance (e.g., "Overall sales grew 12% YoY, led by the West region.")
    - Conversational Q&A Mode: Answer ad-hoc business questions from the user (e.g., "Which product line underperformed in Q4?")

2. Technical Implementation

Use Python as the primary language, and leverage one or more of the following frameworks:

- LLM Integration: Gemini API / OpenAI API / LangChain / LlamaIndex / or any other LLM of your choice
- Multi-agent implementation with at least following 3 agents – Language to query resolution agent, data extraction agent and validation agent. Feel free to add more as you feel necessary. You can use LangGraph / AutoGen / Crew.ai / or any other agentic framework of your choice.
- Data Layer: Pandas, DuckDB, SQL or similar for structured querying
- Optional UI: Streamlit, Gradio or similar as per your comfort, for interaction
- Optional Vector Indexing (as required for Solution): FAISS, Pinecone, ChromaDB or similar

Include a prompt-engineering layer to ensure:

- Consistent, contextual responses
- Instruction-following behavior
- Ability to maintain conversation context (memory or retrieval-based)

3. Scalability Challenge (100GB+ Dataset)

You must design and present architecture for scaling the assistant when the dataset grows beyond 100GB.

Your solution should address:

A. Data Engineering & Preprocessing

- How you will ingest, clean, and preprocess massive CSV or transactional data (batch or streaming).
- Use of distributed or cloud-native processing (e.g., PySpark, Dask, BigQuery, Databricks).

B. Storage & Indexing

- Propose how and where to store large-scale data efficiently:
    - Cloud data warehouse (e.g., BigQuery, Snowflake)
    - Data lake (e.g., AWS S3, Azure Data Lake, GCS)
    - Analytical layer (DuckDB, Delta Lake, or Parquet-based querying)

C. Retrieval & Query Efficiency

- How you will retrieve only relevant subsets of data for each query.
- Consider:
    - Semantic or metadata-based filtering
    - Vector embeddings for similarity search (e.g., using FAISS, Pinecone)
    - RAG (Retrieval-Augmented Generation) pattern with LLMs

D. Model Orchestration

- Handling LLM queries at scale:
    - Prompt templates and caching

    - Chaining queries through LangChain or LlamaIndex
    - Cost optimization and latency control

E. Monitoring & Evaluation

- Metrics for evaluating response accuracy, latency, and cost.
- Error handling or fallback strategies when LLM confidence is low.

4. Deliverables

Candidates must submit the following:

1. Code Implementation
    - Working multi-agent chatbot or summarization script.
    - Should run on sample sales data (CSV).
    - Include all dependencies and setup instructions.
    - Format : share the code base as zip or GIT repo link
2. Architecture Presentation (Mandatory)
    - Prepare a presentation few slides for small covering:
        - System architecture and data flow
        - LLM integration strategy
        - Data storage, indexing, and retrieval design for 100GB scale
        - Example query-response pipeline
        - Optional: cost and performance considerations
    - Format: PowerPoint or PDF.
3. Screenshots / Demo Evidence. Testing from 1st one
    - Screenshot(s) of the chatbot or Streamlit UI.
    - Example Q&A interactions.
    - Example summary output.
4. README / Technical Notes
    - Setup and execution guide.
    - Assumptions, limitations, and possible improvements.

Dataset

📂 **Access the dataset:** which is in zip file