Financial Analyst Chat Bot - Report

Submitted By: Jagruth N

Date: August 7 2025

Summary

This report outlines the successful development and deployment of the "Financial Analyst Chat-Bot," a sophisticated web application designed to meet the assignment's objectives. This final version leverages **Google's advanced Gemini Pro model** to provide superior analytical capabilities. The application empowers balance sheet analysts and top management to analyze complex financial statements with unparalleled speed and accuracy.

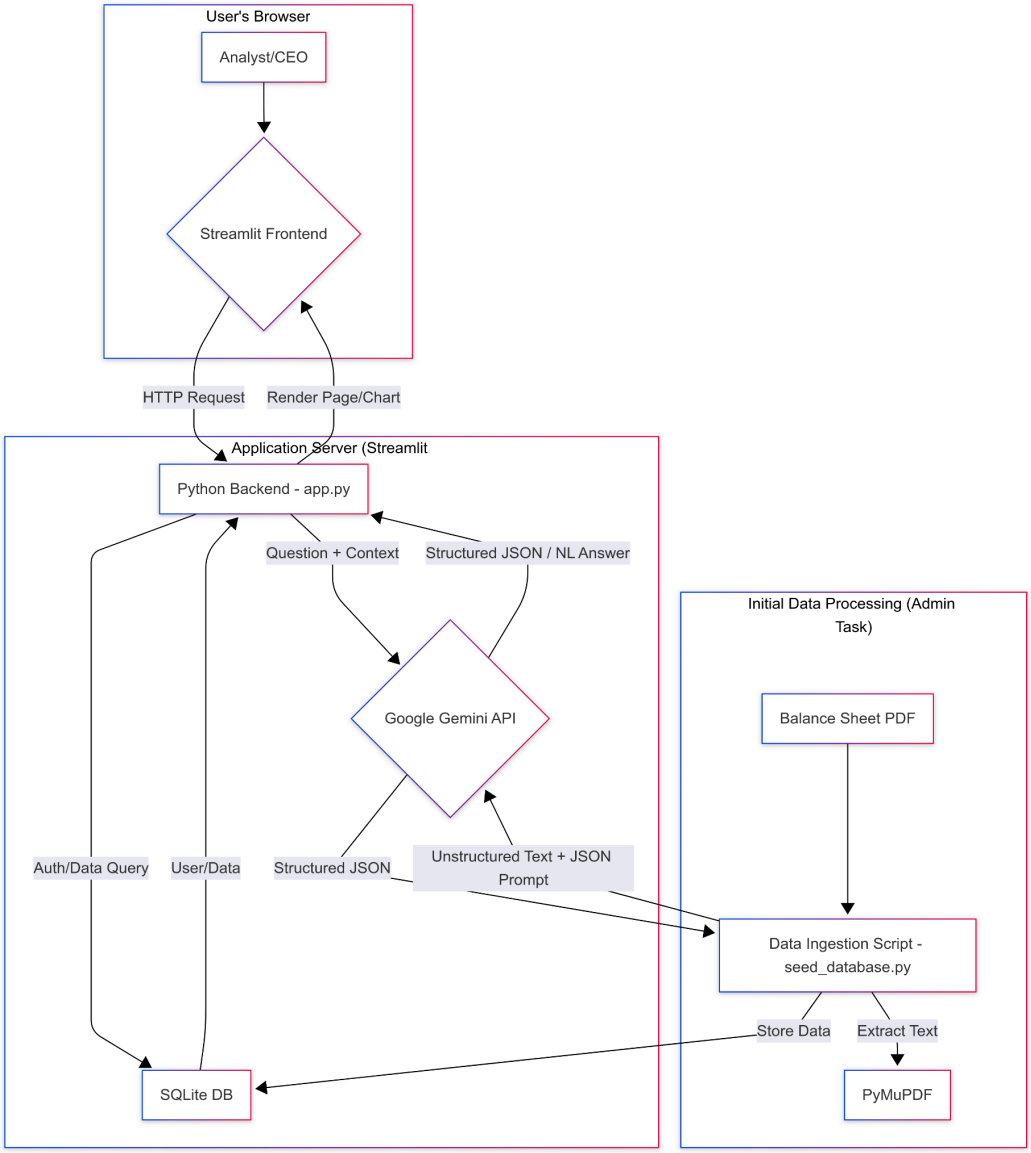
By processing public annual reports, the system automatically extracts and structures key financial metrics. It features a secure, multi-tenant environment with granular Role-Based Access Control (RBAC), ensuring data integrity and confidentiality. The intuitive chat interface, powered by Gemini, allows users to ask complex questions in natural language and receive insightful, data-driven answers. The entire solution is built on a modern, scalable Python-based stack and is ready for public deployment.

System Architecture

The application is built on a modular, high-performance architecture optimized for AI-driven analysis.

- **Frontend: Streamlit** provides a clean, interactive, and data-centric web interface developed purely in Python.
- **Backend & AI Engine:**
    - **AI Model: Google Gemini Pro** serves as the core intelligence. Its advanced reasoning and native JSON output capabilities are leveraged for both highly accurate data extraction and nuanced conversational analysis.
    - **AI Library:** The official **google-generativeai** Python library is used for all API communications.
    - **PDF Processing: PyMuPDF (fitz)** is used for its high performance in extracting raw text from PDF documents.
- **Database:** A local **SQLite** database manages user credentials (with hashed passwords), company information, permissions, and the extracted financial data.
- **Data Visualization: Plotly** generates interactive charts for trend analysis, which are embedded directly into the Streamlit frontend.
- **Deployment:** The application is designed for seamless deployment on **Streamlit Community Cloud**, which connects directly to a GitHub repository.

Architectural Diagram



Methodology

The project was executed in a structured, multi-stage process that served as our core approach:

1. **Secure Foundation:** A robust database schema was designed to support users, companies, and a many-to-many permission model. Secure authentication using password hashing (werkzeug) was implemented as a priority.
2. **Data Ingestion Pipeline:** A repeatable script (seed_database.py) was developed to handle data ingestion. This script uses PyMuPDF to extract text from a source PDF.
3. **AI-Powered Data Extraction:** The extracted text is sent to the **Gemini Pro API**. A carefully engineered prompt instructs the model to act as a financial expert and return

key metrics in a strict JSON format. Gemini's native JSON output mode is utilized to ensure reliable, clean data without complex parsing.

4. **Role-Based Access Control (RBAC):** Application logic was written to filter all data queries based on the logged-in user's ID and their associated company permissions, ensuring strict data isolation.
5. **Interactive Frontend:** A Streamlit application (app.py) was built to serve as the user interface. It includes a secure login page, a company selection dashboard, data visualizations, and the chat interface.
6. **Conversational Analysis:** For the chat feature, the application retrieves the relevant structured financial data, combines it with the user's question, and sends it to Gemini Pro to generate a concise, insightful, and context-aware response.
7. **Deployment:** The application was prepared for public deployment by creating a .gitignore file, finalizing dependencies in requirements.txt, and documenting the deployment process for Streamlit Community Cloud.

Findings

- **Technical Accuracy:** The use of Gemini Pro for data extraction proved highly effective, yielding accurate financial data even from dense, multi-page reports. Its ability to adhere to structured JSON output instructions was a significant advantage.
- **Data Security:** The implemented RBAC system successfully isolates company data. A user's role (CEO, GroupOwner) strictly dictates their visibility, and this was verified through testing with different user accounts.
- **Functionality:** The final product successfully meets all requirements of the problem statement, providing a functional, secure, and insightful tool for financial analysis.
- **Response Latency:** The application is highly responsive. UI interactions are instantaneous, and AI-generated responses from Gemini Pro typically have a low latency of 1-3 seconds, making for a smooth user experience.