Financial AI Assistant with ReAct Architecture

Abstract

The objective of this project is to develop an autonomous "Financial Analyst Agent" capable of gathering real-time financial data, analyzing it, and making logical inferences. The project utilizes the **ReAct (Reasoning + Acting)** architecture, employing the **Qwen 2.5 3B** model running locally. The system possesses capabilities such as fetching live stock market data, performing technical analysis, scanning current news from the internet, and interpreting CSV files statistically.

Key Objectives

1. **Thought:** Interpreting user queries and formulating necessary action plans.
2. **Action:** Selecting and executing the correct tool.
3. **Observation & Final Answer:** Interpreting the results and reporting them to the user.

System Architecture and Technologies

ReAct (Reason + Act) Loop

The system uses a cyclical thought structure instead of a classical Question-Answer approach:

- **Thought:** "The user asked for the THY price, I must find the ticker symbol first."
- **Action:** Call the `get_ticker_symbol('THY')` tool.
- **Observation:** Symbol found as `THYAO.IS`.
- **Thought:** "Now I must fetch the price using this symbol."
- **Action:** Call `get_stock_price('THYAO.IS')`...

Technologies Used

- **LLM:** Qwen 2.5 3B Instruct (Local).
- **Interface:** Streamlit.
- **Data Sources:**
    - `yfinance`: For stock and crypto data.
    - `Google Serper API`: For current news and macroeconomic data.
    - `Pandas`: For CSV file analysis.

Developed Tools

7 fundamental capabilities have been defined for the agent:

1. **`search_general_info`:** Performs real-time news and information scans via Google (e.g., "What was the CBRT interest rate decision?").
2. **`get_ticker_symbol`:** Converts company names into stock exchange symbols (e.g., "Turkish Airlines" -> "THYAO.IS").
3. **`get_stock_price`:** Fetches delayed or real-time stock/crypto prices.
4. **`analyze_technical_data`:** Calculates RSI (Relative Strength Index) and SMA (Simple Moving Average) for stocks.
5. **`analyze_full_csv`:** Reads user-uploaded datasets to perform trend, volatility, and correlation analysis.
6. **`read_csv_preview`:** Reads the first 5 rows to understand the structure of the uploaded file.
7. **`query_knowledge_base`:** (Optional) A dictionary for financial terms.

Challenges and Solutions

Three main bottlenecks were encountered and resolved during the development process:

1. Search Engine Issue (DuckDuckGo vs. Google)

- **Problem:** The `DuckDuckGo` library used in the initial version frequently returned "Rate Limit" errors and failed in local queries (e.g., "Who is the CEO of THY?").
- **Solution:** Switched to **Google Serper API** integration. This resolved the agent's "blindness" issue and increased information access speed.

2. Hallucination and Format Errors

- **Problem:** The model sometimes invented tools not present in the Python code (like `calculate_currency`) or broke the JSON format.
- **Solution:** The **System Prompt** was strengthened. Commands such as "ONLY use these tools" and "Perform mathematical operations mentally" were added.

3. Symbol Mapping

- **Problem:** The model was retrieving the `GBKB.SG` code (Singapore exchange) for "Garanti Bankası".
- **Solution:** A manual mapping layer was added to the `get_ticker_symbol` function for popular Turkish stocks.

Benchmark and Performance Tests

Performance Summary

In a 50-question benchmark test, the model's success in answering questions using live data sources was measured.

- **General Accuracy: 84%** (42/50 Successful)
- **Incorrect/Incomplete Answers:** 16% (8/50 - Generally caused by complex tax regulations or momentary data latency).
- **Average Response Time:** 22.4 Seconds.

Benchmark Comparison

The table below compares the performance of our Local ReAct Agent against other leading models on the same financial dataset:

Model / Architecture	Accuracy	Success Rate	Notes
Financial AI Agent (This Project)	**84%**	**42/50**	Uses Qwen 2.5 (Local) + Tools
Google Gemini (Advanced)	96%	48/50	High reasoning capability
Groq (Llama 3 70B)	94%	47/50	Extremely fast inference
Meta Llama 3 (Base)	90%	45/50	Good performance but slower

Conclusion

The developed "Financial AI Assistant" has proven that a local LLM can transform into a powerful analyst when equipped with the right tools. In particular, its CSV file analysis and technical indicator interpretation capabilities distinguish the system from standard chatbots. With the Google API integration, the system has achieved a stable structure capable of adapting to real-time market conditions.