FinSentio: A Modular LLM-Powered Financial Assistant for Investment Guidance, Risk Analysis, and Education

Yavuz Selim Sever *TED University*
Dept of Computer Engineering
Ankara, Türkiye
yavuz.sever@tedu.edu.tr

Başak Nehir Artan
TED University
Dept of Computer Engineering
Ankara, Türkiye
basaknehir.artan@tedu.edu.tr

Berna Danışman
TED University
Dept of Computer Engineering
Ankara, Türkiye
berna.danisman@tedu.edu.tr

Nuran Er
TED University
Dept of Computer Engineering
Ankara, Türkiye
nuran.er@tedu.edu.tr

Murat Karakaya
TED University
Dept of Computer Engineering
Ankara, Türkiye
murat.karakaya@tedu.edu.tr

2025 10th International Conference on Computer Science and Engineering (UBMK) | 979-8-3315-9975-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/UBMK67458.2025.11207001

***Abstract*—This paper introduces FinSentio, a modular AIpowered financial assistant designed to deliver investment guidance, risk analysis, and financial education tailored to user profile. Leveraging large language models (LLMs), FinSentio integrates real-time market data, retrieval-augmented generation (RAG), sentiment analysis, and structured tool-based reasoning to provide users with accessible and explainable financial insights. The platform features a financial educator powered by an embedded knowledge base, and an advisor module capable of answering complex investment queries. The proposed system is evaluated using an LLM-as-a-judge methodology based on ninety human-generated prompts and demonstrates strong performance across relevance, correctness, and completeness.**

Keywords—financial technology; large language models; sentiment analysis; retrieval-augmented generation; risk analysis; function calling.

I. INTRODUCTION

As personal responsibility for financial decision-making continues to grow, individuals, especially non-experts, face increasing difficulty navigating complex financial systems. Investment planning, risk assessment, and understanding financial news require not only access to up-to-date data but also the ability to interpret that data within a personal context. Despite the availability of various finance-related tools and platforms, most lack adaptability, real-time intelligence, and personalized support, often leaving users overwhelmed or misinformed.

This challenge falls within the broader study area of financial decision support systems, where the aim is to develop tools that assist users in making informed economic choices. Within this context, FinSentio specifically intersects with subfields such as financial technology (FinTech), natural language processing (NLP), and LLM-based intelligent assistants. These technologies, when combined, offer the potential to revolutionize the way individuals interact with complex financial data, making it more understandable, adaptive, and actionable. This study presents FinSentio, an AI-supported financial assistant designed to assist users in three key areas: investment advising, risk analysis, and financial education. The system integrates retrieval-augmented generation (RAG), function calling, sentiment analysis and is based on a modular, multi-agent architecture that integrates several external APIs and tools to deliver timely and explainable outputs. FinSentio utilizes financial news sentiment, macroeconomic indicators, real-time stock and commodity data, and user profile information to provide informed recommendations. Moreover, the system employs structured output and function-calling capabilities to generate consistent and reliable responses across modules.

II. RELATED WORK

LLMs in finance have inspired diverse applications, including sentiment analysis, RAG, and advisory systems. Zhang et al. [1] proposed FinLlama, a finance-tuned LLaMA 2 model excelling in sentiment strength and valence, surpassing FinBERT in portfolio tasks but focusing on trading rather than user-centric guidance like FinSentio. A retrieval-augmented sentiment framework [5] improved LLM performance by 15–48% using contextual expansion. FinSentio similarly integrates FinBERT and VADER within a broader sentiment module.

Multi-agent systems are also prevalent. A trading-oriented multimodal framework [2] combined agents processing text, visuals, and signals, while ElliottAgents [3] used inter-agent dialogue and wave theory. FinSentio differs by focusing on user profiling, document QA, and personalized advice.

RAG-enhanced systems like the LLM-RAG model [4] achieved 78.6%accuracy and 34.8% latency reduction. FinSentio's Education Module shares this foundation but expands it with interactive and personalized learning. Stock-

Chain [4] combines AlphaFin data and RAG to reduce hallucinations, a goal shared by FinSentio's advisory tools.

Conversational agents have addressed trust and personalization: a user study [6] showed LLMs match humans in preference alignment but falter with conflicting goals; FinPersona [7] provided novice-focused dialogue support. FinSentio builds on this by integrating multi-agent flows, document grounding, and dual education–advice modules.

Finally, FinRobot [8] presents a layered, open-source, multi-agent platform enabling financial CoT reasoning and multi-LLM orchestration. Like FinRobot, FinSentio emphasizes modularity and accessibility, using prompt-based routing and integrated CoT for transparent financial advisory.

III. System Design

A. Modular Architecture

Fig.1. presents the multi-agent architecture of the Financial Education module, which enables users to upload financial documents and receive grounded, explainable responses based on their content. This module is designed to handle complex user interactions such as document summarization and context-aware question answering, by leveraging a RAG pipeline and a sequence of coordinated agents.

The interaction begins when a user submits a prompt, typically a question regarding an uploaded financial document. This prompt is initially processed by a central agent, which delegates the request to a Safety Checker to filter unsafe or policy-violating inputs. Once verified, the prompt is passed to the Intent Classifier, which determines whether the user request is related to file-based analysis or web-based search.

If the intent is classified as "file analysis", the system routes the query to the rag check agent, which is responsible for orchestrating the RAG pipeline. As shown in the diagram, this agent interfaces with a pre-trained Embedding Model that converts the uploaded financial document into vector representations. These vectors are stored in a Vector Database (Vector DB), which serves as the retrieval backbone of the RAG system.

When a user asks a question, the rag check agent retrieves the most relevant text chunks from the vector database. Before these chunks are used in the response, they are validated through the check relevance tool to ensure that only accurate and contextually appropriate segments are passed to the LLM. This ensures source-grounded generation and helps avoid hallucinations.

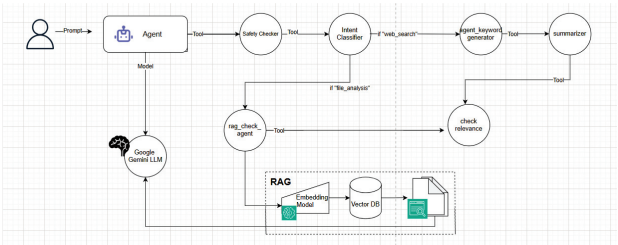

Fig. 1. Multi-agent architecture of Financial Education Module.

In parallel, if the user prompt is classified as "web search", the system engages a different agent sequence. The agent keyword generator formulates appropriate search terms, which are then used by a summarizer tool to produce a concise, human-readable answer based on external web sources.

Finally, the Google Gemini LLM (gemini-2.0-flash) [10] serves as the core generative model, synthesizing responses based on retrieved document chunks or summarized web content. The modular design of this architecture allows the system to answer questions strictly grounded in the uploaded documents or switch to external sources when necessary.

Fig.2. illustrates the multi-agent architecture of the Financial Advisor module within our system. The workflow begins when a user submits a natural language prompt, which is received by a central agent. This agent acts as the core orchestrator, routing the prompt to the appropriate submodules or tools based on its content.

The first stage in the pipeline involves the Safety Checker, which evaluates the prompt for harmful, unsafe, or policyviolating content. If the input passes the safety check, it is forwarded to the Intent Classifier, which determines the user's intent. Depending on the classification, the system proceeds in one of three directions. If the intent is identified as "risk", the prompt is sent to the RiskAnalyzerAgent. If the intent is "data", the system invokes the appropriate financial functions via function calling, using the Google Gemini LLM (gemini-1.5-flash-latest) [10] as the execution layer. If the intent is neither "risk" nor "data", it falls back to "web" and is redirected to the WebSearchAgent.

As shown under the if "data" path in Fig.2., the system incorporates a suite of ten domain-specific function calls, each designed to retrieve a distinct type of financial data. These functions are exposed to the language model via the function calling interface and are dynamically invoked by the LLM based on user queries. This approach enables realtime retrieval of structured financial information to support informed decision-making. The supported functions include:

- fetch_dividend_data(): retrieves dividend-related metrics for a given stock,
- fetch_macro_data(): provides key macroeconomic indicators,
- fetch_stock_data(): returns the latest market data for equities,
- fetch_market_sentiment(): analyzes sentiment from recent financial news,
- analyze_trend(): assesses stock price trends over time,
- calculate_volatility(): computes historical volatility metrics,
- fetch_crypto_quote(): retrieves current cryptocurrency market quotes,
- fetch forex data(): accesses foreign exchange rates,
- fetch commodity data(): fetches data for commodities such as gold or oil,
- fetch_historical_data(): provides historical pricing for financial instruments.

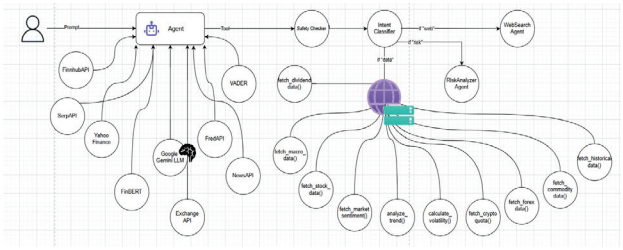

Fig. 2. Multi-agent architecture of Financial Advisor Module.

This modular, multi-agent design enables scalability, separation of concerns, and clearer interpretability. By decomposing complex queries into specialized function calls and agents, the system can provide users with personalized, accurate, and context-aware financial advice in real time.

B. Data Sources and Security

This subsection describes the origins of all data used by the system and the security controls in place to protect sensitive information. The system integrates real-time and static data sources for financial analysis. Real-time market quotes are retrieved via the Finnhub API [11], while up-to-date and historical exchange rates are obtained from the ExchangeRate API [12]. Macroeconomic indicators such as gross domestic product, interest rates, and inflation are fetched from the Federal Reserve Economic Data (FRED) API [13]. End-of-day price series for equities, indices, and commodities are downloaded through the yfinance library (Yahoo Finance Data) [14]. To enrich contextual understanding, user-uploaded PDF reports are parsed with pypdf [15] and web search results are obtained via SerpAPI [16].

Sentiment is evaluated in two stages. In order to retrieve recent news headlines related to a company or topic in real time, NewsAPI [17] is used. For the headlines in Turkish language, VADER's SentimentIntensityAnalyzer [18] is used to compute a compound polarity score over the sentiment of those headlines related to the user's query. For the headlines in English language, FinBERT model (yiyanghkust/finberttone) [19] and its tokenizer via Hugging Face Transformers are used to perform domain-specific sentiment classification on up to twenty English news headlines. Because FinBERT is pre-trained on U.S. financial corpora, it only accepts English inputs, making it ideally suited for market news analysis. The softmax function from PyTorch [20] normalizes output logits into sentiment probabilities, and we aggregate counts of positive, neutral, and negative labels to determine the prevailing market mood.

Inside the Financial Education Module as summarized in Table I., once the uploaded document is parsed and loaded into memory, its textual content undergoes semantic segmentation. This is achieved using LangChain's RecursiveCharacterTextSplitter [21] and SentenceTransformersTokenTextSplitter [22], which divide the documents into semantically coherent chunks. Chunks are embedded via the sentence-transformers model "paraphrasemultilingual-mpnet-base-v2" [23]. Embeddings and their associated metadata such as document source, page number, and category are persistently stored in ChromaDB's PersistentClient.

TABLE I RAG PIPELINE PARAMETERS

Category	**Pipeline Component**	
	Parameter	***Value***
Document Preprocessing	Document Loader	pypdf.PdfReader
	Chunking Strategy	RecursiveCharacterTextSplitter
	Chunk Size	500 tokens
	Chunk Overlap	50 tokens
Embedding	Embedding Model	paraphrase-multilingual-mpnet-base-v2
	Embedding Generator	sentence-transformers
Storage	Vector Store	ChromaDB with PersistentClient storage
	Stored Metadata	Source, page number, category
	Identifier Hashing	UUID + SHA-256
Retrieval & Inference	Retrieval Method	Similarity search with ChromaDB (cosine similarity)
	Top-k Retrieved Chunks	4
	RAG Trigger Mechanism	Activated upon PDF upload
	Fallback Strategy	Web search via SerpAPI

To preserve user privacy and prevent direct linkage to original filenames or personal identifiers, every document and text chunk is assigned a unique, non-reversible identifier generated via UUID [24] and SHA-256 hashing [25].

Robust security controls protect credentials and data at runtime. All API keys reside in an external .env file and are loaded at execution time using python-dotenv, ensuring that no secrets are hard-coded in the source repository.

All external communications employ HTTPS with TLS encryption to secure data in transit. Access to the ChromaDB instance and custom database modules is governed by rolebased permissions, and detailed audit logs capture every API call and data retrieval event for forensic analysis. Future work includes encrypting embeddings and metadata at rest for enhanced data protection.

C. Application Development

The application is a single-page web interface built with Gradio's Blocks API [9], structured into three layers: persistence (user data/profiles), business logic (risk analysis and educational chat), and presentation (UI rendering and styling). Upon launch, users encounter a Login/Registration form (Fig.3.) that validates credentials against a local SQLite store. Passwords are salted and hashed with PBKDF2-HMAC-SHA256, and all secrets (API keys, database paths) remain outside the source code in environment variables loaded at runtime. Login requires accepting Terms & Conditions via checkbox and modal before submission.

After login, the interface transitions to a Dashboard with three tabs. In the Profile tab (Fig.4.), ten questions gather a user's risk preferences. Answers are stored in JSON format, and previous selections are reloaded for returning users.

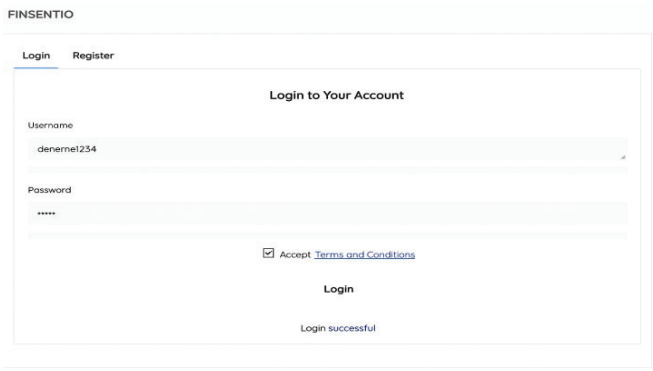

Fig. 3. User interface for login.

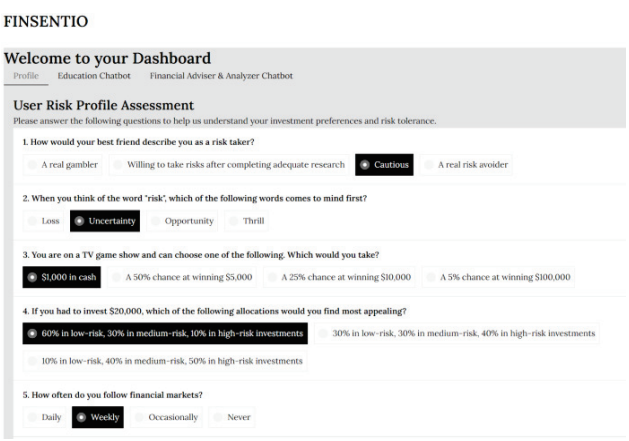

Fig. 4. User risk profile assessment (first five questions shown as example).

The Education Chatbot tab (Fig. 5) allows users to ask questions and optionally upload PDF documents. The system uses an LLM to answer these questions and, if a document is provided, activates the RAG pipeline to give answers based on the file's content.

Adviser & Analyzer Chatbot tab (Fig.6.) provides both financial recommendations and risk assessments, allowing users to ask investment-related questions, request stock or sector insights, and receive real-time evaluations of financial risk. Free-form questions trigger a risk-analysis engine that computes factors such as volatility and sector performance, formats them into a summary table, and generates a PDF report for download which includes user profile, risk factors table, and visualization of volatility & trend insights, and summary.

For both modules, responses are sanitized to plain text, appended to the chat history, and accompanied by a dynamically updated cost panel that displays token usage and each interaction also displays intent-routing details (e.g., safety check, intent classifier) and real-time LLM cost metrics in a sidebar.

Styling relies on a comprehensive CSS layer that defines color variables, typography scales, and utility classes for larger text, prominent inputs, and chat bubbles. Dark-mode overrides ensure high contrast for readability, and form controls are customized for accessibility and a consistent brand aesthetic.

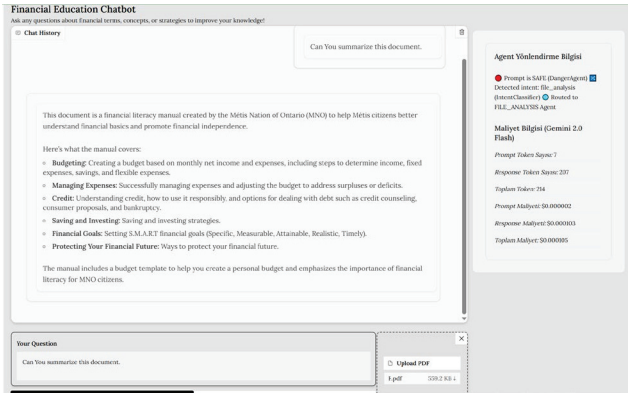

Fig. 5. Financial Education Chatbot tab.

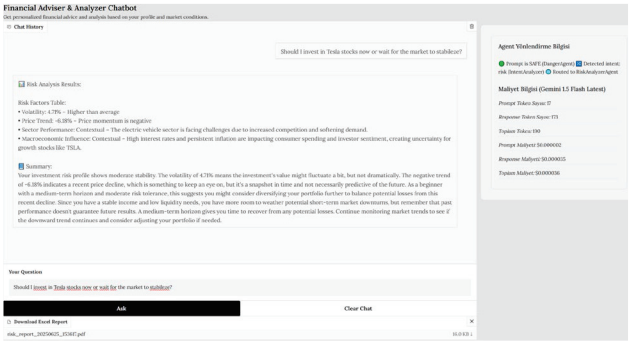

Fig. 6. Financial Adviser & Analyzer Chatbot tab.

D. *Financial Feature Utilization*

To support its financial reasoning capabilities, FinSentio relies on a diverse set of structured financial inputs retrieved via function calls. These include both market-level indicators and asset-specific metrics. The RiskAnalyzerAgent uses features such as historical price volatility, trend slope, dividend yield, and sector classification to generate risk summaries. Macroeconomic context is provided through indicators such as interest rates, GDP, and inflation, which are retrieved via the FRED API. The Market Sentiment module analyzes recent news headlines to compute sentiment polarity scores using FinBERT and VADER. The Financial Data Tools module exposes ten callable functions to the LLM, offering real-time access to stock prices, cryptocurrency quotes, foreign exchange rates, and commodity prices. Together, these features provide a domain-specific reasoning layer that informs the system's personalized insights, risk assessments, and educational outputs.

IV. TESTING AND EVALUATION

The performance was evaluated using a rubric-based framework and the LLM-as-a-Judge method, where responses were scored by GPT-4o based on predefined criteria. The quality of system responses, covering Risk Analysis, Financial Education, and data retrieval tools, was assessed based on custom rubrics. Rubrics were tailored to each module: Risk Analyzer responses were judged on risk classification accuracy, quantitative data use, user profile alignment, and clarity. Data Retrieval Tools were evaluated for correct function use, value fidelity, relevance, and explanation. Financial Education Module was scored on

document/web source accuracy, completeness, reasoning, and clarity.

The overall evaluation results, including test coverage and methods used for each module, are summarized in Table II. All results were scored automatically using GPT-4o, with fallback logic and error handling verified during testing. Average response times remained below 5 seconds, excluding external API delays.

Prompts were crafted to reflect a range of real-world user intents, covering beginner to advanced investor profiles, document comprehension depth, and volatility vs. sentiment-driven queries.

As shown in Fig.7., core function tools such as fetch stock data, fetch crypto quote, and fetch macro data demonstrated high average scores, indicating reliable performance in both data fidelity and language clarity. Slightly lower scores in functions such as analyzing trend and fetching dividend data highlight areas for potential refinement, especially in output explanation and relevance handling.

As illustrated in Fig.8., the model maintains consistent performance across difficulty levels, with only a slight decrease in explanation quality and personalization for more complex prompts. These results suggest that while the Risk Analyzer handles straightforward queries well, further refinement is needed to enhance reasoning depth and user alignment in difficult scenarios.

As shown in Fig.9., the Web Search QA feature achieved perfect average scores, reflecting its ability to retrieve relevant and well-sourced content. In contrast, Document QA had slightly lower performance due to chunking limitations in longer PDFs, affecting accuracy in final-page queries.

TABLE II TEST RESULTS OF MODULES AND TOOLS

Metric	Module		
	Risk Analyzer	***Data Tools***	***Education Module***
Test Cases	30 prompts (3 levels)	10 functions *3 Qs +5 sentiment Qs	4 PDFs* 3 Qs +12 web Qs
Evaluation Method	Rubric + LLM-as-a-Judge	Rubric + LLM-as-a-Judge	Rubric + Manual + LLM-as-a-Judge
Avg. Score (out of 10)	8,17	8,63	9,25

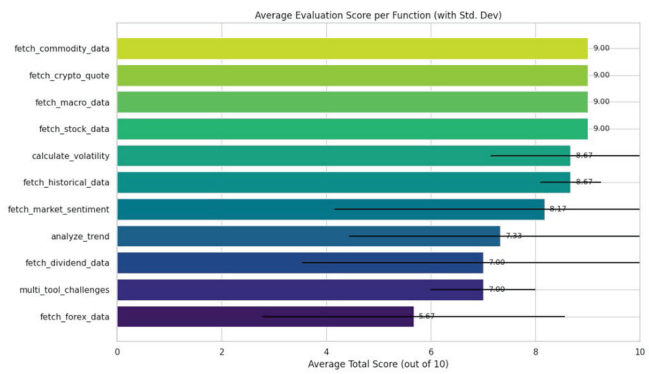

Fig. 7. Evaluation scores of data retrieval tools.

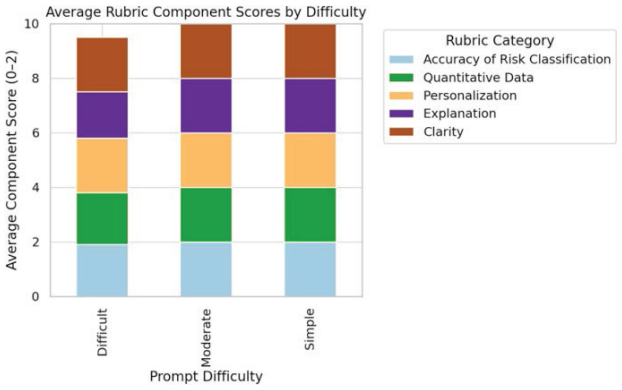

Fig. 8. Average rubric component scores across difficulty levels (Simple, Moderate, Difficult) for Risk Analyzer prompts.

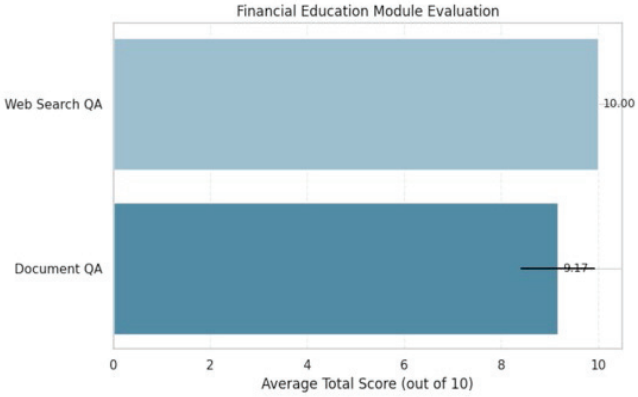

Fig. 9. Average evaluation scores for the Financial Education module across two functionalities: Web Search QA and Document-based QA.

V. DISCUSSION AND LIMITATIONS

The FinSentio project exhibited stable performance across its modules. The Risk Analysis component effectively integrates user profile data with market indicators, while Data Retrieval tools provide low-latency access to real-time financial information. The Financial Education module performed well both for web-based informational queries and documentbased questions using a RAG pipeline. When PDFs were uploaded, the system largely restricted its answers to content from the document, confirming the RAG setup works as intended.

Nonetheless, some limitations were identified. Long PDFs caused minor performance drops due to chunking and segmentation challenges. Dependence on third-party APIs (e.g., Finnhub, ExchangeRate) introduced occasionallatency and inconsistency. Use of Gemini's free-tier models led to ratelimit errors (429) under load, reducing response reliability. Additionally, multi-intent queries complicated module routing, occasionally causing suboptimal responses. Finally, while "LLM-as-a-Judge" was used for evaluation, its potential bias suggests future assessments should combine human judgment for better reliability.

VI. CONCLUSION AND FUTURE WORK

FinSentio is a modular LLM-based assistant offering personalized investment guidance, risk analysis, and financial education through real-time data, RAG, sentiment

analysis, and function-based reasoning. Each module was evaluated using targeted scenarios to ensure intent-aware response accuracy.

FinSentio contributes to the international research community by demonstrating how LLMs can be used not only for general-purpose tasks, but also for personalized financial analysis when combined with real-time market data, sentiment evaluation, and retrieval-augmented reasoning. The system's modular, multi-agent architecture can serve as a reusable framework for future research in finance-focused AI assistants. In particular, our approach to grounding LLM responses using user profiles, financial features, and sourceaware document QA offers new directions for building explainable and adaptive decision support systems.

Future work will focus on improving accuracy, adaptability, and personalization. Planned enhancements include advanced intent handling for multi-intent queries, better long-document processing via hierarchical segmentation and metadata, and a feedback loop for learning. Additional goals include mobile support and integration of ethical investment options to expand real-world use.

ACKNOWLEDGMENT

This study was conducted as part of a course (SENG 472- Building AI Applications with Large Language Models) project under the supervision of Prof. Dr. Murat Karakaya, at TED University.

REFERENCES

[1] G. Iacovides, T. Konstantinidis, M. Xu, and D. Mandic, "FinLlama: A finance-specific LLM framework for sentiment analysis and portfolio management," in Proc. 29th Int. Conf. on Intelligent User Interfaces (IUI '24), ACM, 2024. Available: https://dl.acm.org/doi/pdf/10.1145/3677052.3698696

[2] S. Fatemi and Y. Hu, "FinVision: A multi-agent framework for stock market prediction," in Proc. 29th Int. Conf. on Intelligent User Interfaces (IUI '24), ACM, 2024. Available: https://dl.acm.org/doi/pdf/10. 1145/3677052.3698688

[3] W. Zhang, Y. Yang, and L. Chen, "ElliottAgents: Multi-agent system for Elliott wave analysis," in Proc. 38th Pacific Asia Conf. on Language, Information and Computation (PACLIC), 2024. Available: https://aclanthology.org/2024.paclic-1.91.pdf

[4] Y. Zhang, X. Wang, and J. Li, "LLM-RAG financial data analysis system and Stock-Chain framework," arXiv preprint, arXiv:2403.12582, 2024. Available: https://arxiv.org/pdf/2403.12582

[5] Y. Wang, Z. Liu, and M. Chen, "A retrieval-augmented framework for financial sentiment analysis with instruction-tuned LLMs," in Proc. 2024 ACM Web Conf. (TheWebConf), ACM, 2024. Available: https://dl.acm.org/doi/pdf/10.1145/3604237.3626866

[6] T. Takayanagi, K. Izumi, J. Sanz-Cruzado, and R. McCreadie, "Are generative AI agents effective personalized financial advisors?" arXiv preprint, arXiv:2504.05862v2, Apr. 2025. Available: https://arxiv.org/abs/2504.05862

[7] T. Takayanagi et al., "FinPersona: An LLM-driven conversational agent for personalized financial advising," in C. Hauff et al., Eds., Proc. 46th European Conf. on IR Research (ECIR 2025), vol. 15576, Lecture Notes in Computer Science, Cham: Springer, 2025, pp. 25–37. Available: https://doi.org/10.1007/978-3-031-88720-8 3

[8] H. Yang et al., "FinRobot: An open-source AI agent platform for financial applications using large language models," arXiv preprint, arXiv:2405.14767, May 2024. Available: https://arxiv.org/pdf/2405.14767

[9] Gradio, "Gradio: Build & share delightful machine learning apps." Available: https://www.gradio.app/, accessed Jun. 25, 2025.

[10] Google, "Gemini API: Model documentation." Available: https://ai.google.dev/gemini-api/docs/models, accessed Jun. 26, 2025.

[11] Finnhub, "Free APIs for real-time and historical market data." Available: https://finnhub.io/, accessed Jun. 26, 2025.

[12] ExchangeRate-API, "Currency conversion API." Available: https://www.exchangerate-api.com/, accessed Jun. 26, 2025.

[13] Federal Reserve Bank of St. Louis, "FRED API: Economic Data API."Available: https://fred.stlouisfed.org/docs/api/fred/, accessed Jun. 26, 2025.

[14] Yahoo, "Yahoo Finance Python Package." Available: https://pypi.org/project/yfinance/, accessed Jun. 26, 2025.

[15] PyPDF, "A pure-Python PDF library." Available: https://pypi.org/project/pypdf/, accessed Jun. 26, 2025.

[16] SerpApi, "Real-time Google Search API." Available: https://serpapi.com, accessed Jun. 26, 2025.

[17] NewsAPI, "Worldwide news search API." Available: https://newsapi.org, accessed Jun. 26, 2025.

[18] C. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," in Proc. 8th Int. Conf. on Weblogs and Social Media (ICWSM), Ann Arbor, MI, USA, 2014. Available: https://github.com/cjhutto/vaderSentiment

[19] Y. Yang, "FinBERT: Financial sentiment analysis," Hugging Face. Available: https://huggingface.co/yiyanghkust/finbert-tone, accessed Jun. 26, 2025.

[20] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Proc. NeurIPS 2019, vol. 32, pp. 8024–8035. Available: https://pytorch.org/, accessed Jun. 26, 2025.

[21] LangChain, "RecursiveCharacterTextSplitter: LangChain Document Loaders," LangChain Documentation. Available: https://python.langchain.com/api reference/text splitters/character/langchain text splitters.character.RecursiveCharacterTextSplitter.html, accessed Jun. 26, 2025.

[22] LangChain, "SentenceTransformersTokenTextSplitter," LangChain Documentation. Available: https://python.langchain.com/api reference/ text splitters/sentence transformers/langchain text splitters.sentence transformers.SentenceTransformersTokenTextSplitter.html, accessed Jun. 26, 2025.

[23] Sentence-Transformers, "paraphrase-multilingual-mpnet-basev2: Multilingual model for sentence embeddings," Hugging Face. Available: https://huggingface.co/sentence-transformers/ paraphrase-multilingual-mpnet-base-v2, accessed Jun. 26, 2025.

[24] Python Software Foundation, "uuid: UUID objects according to RFC 4122," Python 3.12.3 Documentation. Available: https://docs.python.org/3/library/uuid.html, accessed Jun. 26, 2025.

[25] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)," FIPS PUB 180-4, Aug. 2015. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf