IST 688 Project Report

Team Members:
Jill Karia
Anjaneya Padwal
Sudhanshu Pawar
Aakanksha Maheshwari

Project Overview:

The goal of the Project is to create a MAS (Multi-Agent System) that can recommend a user investment strategy based on stocks, company trends, and risk assessment done by AI Agents. The way the project achieves this goal is by first generation a newsletter based on financial news and the stock trends. By using a chatbot we also wanted to make the system more user friendly. The Northstar Goal is to augment the efforts of a larger team putting in human efforts to get the right data and context to give investment recommendations

The Project consists of 4 features. First, the newsletter newsletter generation uses 4 separate agents.The second feature that we have in our project is that we used bespoke API to fact-check the newsletter. Then one can even upload in their portfolio and it can give insights on the companies they have invested in based on their portfolio. Now finally we also have a chatbot that gives ticker information and answers about the newsletter or any news information that is stored in the rag. If the information is not stored in our rag it is also capable of answering out-of-scope questions. The data for the RAG (Retrieval Augmented Generation) comes from two API calls that extract highly active stock data of the day and news relevant to the financial world onto the system. The data is then sent to the 4 Agents work in a structure such that the output is provided by the last bot which is then summarized in a newsletter and displayed when it is triggered with a button on the Frontend. For the chatbot, it has three functionalities. First, it uses a function to call the day's stock data for a certain ticker. It is also able to query the RAG so that the chatbot can answer any question about the newsletter. Bespoke is used to clarify the accuracy of the insights provided in the newsletter in terms of an accuracy score displayed post-clicking the “Check Accuracy” button. Lastly we can input into the portfolio and it uses the same function from the chatbot to get the ticker information for the companies and recommends what to do with the investments.

Strengths of the project:

- Innovative Multi-Agent System (MAS): The project uses a well-structured MAS approach with multiple agents collaborating for newsletter generation, fact-checking, and providing investment insights..
- User-Friendly Chatbot: The chatbot adds an interactive layer, enabling users to query financial data, newsletter content, and general investment-related information.
- Portfolio Analysis: The ability to upload a portfolio and get tailored insights is a valuable feature for personalized investment guidance.
- Fact-Checking with Bespoke API: Enhances credibility and trustworthiness by providing an accuracy score for newsletter content.
- RAG Implementation for Dynamic Responses: Retrieval-augmented generation ensures that the system provides contextually relevant answers from pre-existing financial data.
- Seamless Integration of APIs: Pulls real-time stock and news data, keeping the system updated and relevant.

Weaknesses of the project:

- Dependency on External APIs: Heavily reliant on APIs for data; service disruptions could impact functionality.
- Limited Fact-Checking Scope: Fact-checking is confined to Bespoke API's capabilities, which may not cover all types of misinformation or inaccuracies.
- Scalability Concerns for Large Portfolios: The system might face performance issues when analyzing and recommending actions for very large portfolios.
- Lack of User-Centric Risk Profiling: Recommendations might not account for individual user risk tolerance, which is critical for investment strategies.
- Absence of Historical Analysis: The system focuses on current trends and does not provide historical trend analysis, which could add depth to investment insights.
- Security and Privacy Risks: Handling sensitive portfolio data might raise security and compliance concerns if robust measures are not implemented.

Technical Details:

API Usage

- The APIs used for this project are as follows:

API Name	Task	Link to Documentation
OpenAI	The OpenAI API has been used as the LLM in this project	https://platform.openai.com/docs/api-reference/introduction
AlphaVantage Stocks	The AlphaVantage Stocks API is used for getting real-time data on stocks	https://www.alphavantage.co/documentation/
AlphaVantage News	The News API provided by Alpha Vantage delivers the news to the RAG	https://www.alphavantage.co/documentation/
AlphaVantage Daily Stocks	The AlphaVantage daily stocks API is used for getting stock daily data from ticker	https://www.alphavantage.co/documentation/
BeSpoke API	The bespoke API is used for fact checking the newsletter	https://docs.bespokelabs.ai/

- Alpha Vantage News Sentiment API (news_url):
    - Purpose: Fetch real-time news sentiment data for financial markets.
    - Functionality: Provides sentiment scores, article relevance,ticker information and related metadata for up to 50 news articles at a time.
    - Use Case in the Project: Enables the company_analyst_agent to analyze and summarize recent news insights about companies
- Alpha Vantage Market Trends API (tickers_url):

    - Purpose: Fetch the list of top gainers and losers in the market, along with associated trend data.
    - Functionality: Retrieves dynamic market performance data to highlight prominent movers and shakers.
    - Use Case in the Project: Powers the market_trends_analyst_agent to extract actionable market trends for newsletter inclusion
- Alpha Vantage Daily Stock Data API(function = fetch_ticker_price)
    - Purpose: Fetch daily stock data from ticker data for the chatbot and the portfolio recommendation.
    - Functionality: Takes in a ticker and gives back the daily stock data for the ticker
    - Use case in the project: A function is used to call this ticker which then is used for chatbot and portfolio and recommendation.
- OpenAI GPT-4o-mini API (call_openai_gpt4):
    - Purpose: Generate natural language summaries, risk analyses, and newsletters from structured financial data.
    - Functionality: Processes user-defined prompts, generates cohesive content, and integrates insights from various data sources.
    - Use Case in the Project: Provides language modeling capabilities for summarization, risk evaluation, and newsletter creation.
- Bespoke Labs API (assess_accuracy_with_bespoke):
    - Purpose: Evaluate the factual accuracy of the generated newsletter against the provided dataset.
    - Functionality: Uses fact-checking algorithms to analyze the alignment between the newsletter's claims and underlying data.
    - Use Case in the Project: Ensures the reliability of generated content by quantifying its accuracy as a percentage.

Data Used and Data Handling Processes

- Data Sources:
    - Financial data (news sentiment, stock prices, market trends) is fetched from APIs like Alpha Vantage.
    - CSV files uploaded by users (e.g., investor portfolio data with columns like Investor Name, Company, Ticker).
- Data Handling Processes:
    - Preprocessing: Uploaded CSV data is parsed using pandas, validated for required fields, and analyzed for ticker symbols.
    - RAG (Retrieval-Augmented Generation): Combines retrieved data from vector databases (ChromaDB) with OpenAI's GPT for context-based responses.

Vector Database Usage

- Database:
    - ChromaDB is used as the vector database for RAG.
    - Collections like news_sentiment_data and ticker_trends_data are created/updated to store and retrieve financial data embeddings.
- Operations:

- Querying: Retrieves top n results from ChromaDB collections based on user queries.
- Updating: Adds new financial data (e.g., market trends, news articles) to ChromaDB collections for future retrieval.

Functions/Tools Used:

- Tools:
  - The fetch_ticker_price tool is an essential part of the application, allowing users to get real-time financial data for a specific stock ticker symbol (like AAPL or MSFT). It's set up with a clear name, description, and parameters to make it easy to use. The tool requires the user to provide a valid stock ticker symbol as input, which it uses to fetch data from the Alpha Vantage API. It then returns important details like the stock's opening, high, low, and closing prices, along with trading volume. This tool makes it simple for users to access accurate and up-to-date stock information, streamlining the process and improving the app's overall financial analysis features.

Roles of Agents in the Project

1. Researcher:
   - Role: The Researcher is responsible for analyzing news sentiment data to extract relevant insights.
   - Tasks:
     - Retrieve news sentiment data from ChromaDB or APIs.
     - Identify key trends, patterns, and themes in financial news.
     - Provide summarized insights that can guide decision-making.
2. Market Analyst:
   - Role: The Market Analyst focuses on analyzing market trends and stock data to identify potential investment opportunities or risks.
   - Tasks:
     - Fetch and process market trend data from ChromaDB.
     - Analyze stock performance metrics and trends (e.g., gainers, losers).
     - Deliver actionable insights for portfolio optimization or market strategies.
3. Risk Analyst:
   - Role: The Risk Analyst evaluates potential risks based on market trends, financial news, and portfolio data.
   - Tasks:
     - Combine insights from the Researcher and Market Analyst.
     - Assess financial and investment risks.
     - Generate reports highlighting risk factors and mitigation strategies.
4. Writer:
   - Role: The Writer is responsible for synthesizing outputs from the other agents into a coherent and professional financial newsletter.
   - Tasks:

        - Gather context and insights from the Researcher, Market Analyst, and Risk Analyst.
        - Create a well-structured newsletter using financial jargon.
        - Ensure the newsletter is concise, actionable, and based entirely on data provided by other agents.

Third-Party Libraries and Tools

- Libraries:
    - chromadb: For vector database operations (RAG implementation).
    - openai: For GPT-4-based response generation and context augmentation.
    - pandas: For data manipulation and CSV processing.
    - streamlit: For creating the interactive user interface.
    - bespokelabs: For fact-checking newsletter claims using Bespoke Labs' APIs.
    - requests: For API communication (e.g., Alpha Vantage).
- APIs and Tools:
    - Alpha Vantage: Provides financial market data, including stock prices and news sentiment.
    - Bespoke Labs: Used for fact-checking generated content.
    - Streamlit Cloud: Deployed for hosting the interactive application.

Ethical Implications

- Bias in the Model
    - Models to the likes of these are at risk of creating bias in Historical Markets
    - These models trained on historical stock and company data might overemphasize well-established companies, disadvantaging newer or underrepresented firms.
    - If the training data reflects historical inequities or market trends that favor certain groups or sectors, recommendations might perpetuate these biases.
    - This bias can be around industries where recommendations may unintentionally favor certain industries, sectors, or geographical areas
    - These biases if not fixed can lead to skewed results
- Impact on Society
    - The impact on society according to our assessment will largely be positive given the capability of the product
    - The MAS can democratize access to high-quality investment advice after integrating human insights and AI capabilities.
    - The system can empower underrepresented and underprivileged groups with tailored recommendations
    - This system will also provide opportunities to brokers and finance professionals who could previously afford to hire qualified teams or even afford sophisticated software like Blackrock Aladdin or Bloomberg Terminal
- Data Security
    - Although our system does not use personal data to create any implications on Data Security and Data Privacy, there are some concerns about the Data being used

- The Data used by the Model via the API if skewed can give biased and wrong results which can result in favoring a certain company, type of business, or even geographic location
- Aggregated data can sometimes be reverse-engineered to reveal sensitive patterns

Mitigation of Ethical Issues:

- The first mitigation step that has already been taken while developing this system has been to not store Personal Data of any kind eliminating the risks of Data Privacy
- The Bias in the Model can be mitigated with strategies such as:
  - Choosing a diverse set of datasets for data sourcing ensures that any skewness in the data can be managed
  - Creating a Maker Checker Model in the initial stages to ensure that the output of the model displayed to the user is checked by a human and only then approved
  - A Feedback Loop can be created which can ensure that any mistakes found in the output of the model will be used for correcting the working of the model
  - Active Skew Checking can be implemented in the RAG where the Data is stored to ensure that no skewness indicating bias exists
- Data Security
  - Strict Data Security Standards on the Pipeline can be used to ensure no data leaks or attacks that occur
  - Audits can be regularly done both on data and outputs to ensure that no patterns or unnecessary information is leaked

Individual Section:

Anjaneya Padwal

- My contribution was more towards working on the Multi-Agent System as well as creating the documentation for the project which was 35% of the project and at the same time 25% of the entire delivery for the semester
- The first challenge I faced was creating Multi-Agent Systems without the usage of CrewAI
- Jill and I struggled with implementing it and then integrating that with the existing Streamlit
- We looked at some Github repos and some examples on Medium to understand how these implementations have been done in the past which gave us some strong ideas on how it could be done
- This project personally for me was a lesson in Managing Projects and building AI Products
- I learned that traditional constructs of Scrum or Integrating different components frontend, backend, and logic do not work in the age of AI where the product's expected output is more probabilistic and less deterministic
- I also understood the interoperability of different AI Tools like Streamlit, ChromaDB, CrewAI, and OpenAI API as well as Bespoke
- Although I do mention these things in my resume, I believe something that will help me in the years to come will be the skill that I understand how all of these things can be glued together to create amazing products

Jill Karia:

My primary contribution to the project was designing and implementing the Multi-Agent System and integrating it with the front end and the Vector Database (VectorDB) to manage the operation end-to-end, accounting for approximately 40% of the project's work. This involved building the architecture for task delegation among agents and ensuring seamless interaction between the agents and the VectorDB for data retrieval and updates. I also integrated Bespoke Labs' fact-checking functionality to enhance the reliability and credibility of the newsletter generated by the system.

One major challenge was integrating a Multi-Agent System with a custom Retrieval-Augmented Generation (RAG) design, particularly when attempting to use Crew AI. The available documentation for Crew AI was not well-aligned with our custom RAG setup, which focused on structured financial data rather than scraped documents or PDFs typically used in RAG implementations. This created significant gaps in understanding how to adapt Crew AI to our use case. Additionally, handling the dynamic interaction between agents and ensuring reliable communication with the VectorDB required meticulous debugging and testing.

After evaluating the limitations and inefficiencies of Crew AI for our specific use case, we decided to bypass it entirely and focus on custom solutions tailored to our system's architecture. By leveraging the flexibility of our in-house RAG design and combining it with clear task breakdowns among agents, we achieved the desired functionality. Collaboration with team members and focused discussions helped clarify integration issues and align our approach to ensure smooth system operations.

Through this project, I learned the importance of clear communication and adaptability when working in a team. Collaboration was key in brainstorming solutions to overcome technical obstacles, especially when integrating new technologies. Regular updates and feedback loops helped ensure that everyone was aligned, and tasks were progressing as expected. Additionally, I learned the value of recognizing when to pivot from an approach that isn’t working and focus on solutions that are more practical for the team’s goals.

These lessons can be applied in future professional settings by fostering open communication within teams and being flexible in adapting to challenges. Knowing when to explore alternative approaches and effectively collaborating with team members to troubleshoot issues will help streamline workflows and ensure successful project delivery. Additionally, the technical knowledge gained from designing and integrating Multi-Agent Systems and RAGs will be invaluable for future projects requiring complex data-driven solutions.

Aakanksha Maheshwari:

My primary contribution to the project was managing the VectorDB and developing the Chatbot functionality for portfolio analysis, accounting for approximately 35% of the project's work. This involved ensuring the VectorDB was optimized for retrieving and storing relevant data, maintaining accurate and up-to-date information for efficient operations. Additionally, I implemented the chatbot, which required integrating multiple tools to handle portfolio analysis queries, providing users with actionable insights. The chatbot was designed to incorporate fallbacks to OpenAI's GPT for enhanced reliability and dynamic responses. I also worked on integrating the RAG system, stock ticker data, and contextual information to ensure seamless and meaningful user interactions.

One of the key challenges was integrating various tools into the chatbot, including the fallback mechanism to OpenAI for queries that couldn't be resolved by the RAG system or ticker data. The complex interaction between the tools, especially ensuring smooth transitions between RAG-based retrieval, stock ticker queries, and OpenAI fallbacks, required significant troubleshooting. Another challenge was understanding how to keep the VectorDB updated with real-time data while ensuring data integrity and relevance, which was critical for accurate and responsive chatbot operations.

To address the challenges, I focused on breaking down each integration step into manageable parts, testing each component thoroughly before combining them. Collaboration with team members helped identify and resolve technical gaps, particularly in setting up reliable fallbacks for the chatbot. For the VectorDB, I established clear procedures for data updates, using structured workflows to add new data while preserving existing context and relevance. Regular debugging sessions and leveraging documentation helped ensure the system met its performance goals.

This project taught me the importance of collaboration and detailed planning when integrating complex systems. Effective communication with teammates allowed us to align on goals and troubleshoot issues more efficiently. I also learned to be patient and persistent when resolving integration challenges and to remain flexible in adopting new approaches when initial solutions didn't work as planned.

These lessons will help me in future professional settings by reinforcing the importance of planning, collaboration, and adaptability when managing complex integrations. The hands-on experience with VectorDB management, tool integration, and chatbot development has strengthened my ability to handle data-driven projects and design reliable, user-centric systems. The problem-solving skills gained during this project will be valuable for tackling similar challenges in future roles.

Sudhanshu Pawar

My primary contribution to the project was creating the VectorDB and developing the Chatbot for portfolio analysis, accounting for approximately 35% of the work. This involved building the VectorDB to store and retrieve data efficiently, ensuring it remained up-to-date with relevant financial information. Additionally, I designed and implemented the Chatbot to handle user queries about portfolio analysis, integrating it seamlessly with the RAG system and stock ticker functionality. The Chatbot also included a fallback mechanism to OpenAI's GPT, providing reliable responses even for complex or unsupported queries.

One of the main challenges was integrating multiple tools and features into the Chatbot. This included handling the dynamic interaction between the RAG system, stock ticker queries, and the fallback mechanism to GPT, which required careful coordination to ensure smooth and accurate responses. Another challenge was managing the VectorDB, particularly understanding how to keep it updated with real-time data while maintaining data integrity and relevance. Ensuring the VectorDB could handle high query loads and consistently deliver relevant results added complexity to the task.

To overcome these challenges, I broke down the development process into smaller, manageable tasks and tackled each integration step by step. For the Chatbot, I worked on debugging the interaction between tools and implemented robust error handling to ensure a smooth user experience. Collaboration with the team helped in identifying solutions for tool compatibility issues and refining the fallback mechanism. For the VectorDB, I designed workflows to automate data updates, ensuring the system always had the latest and most relevant information.

This project taught me the importance of collaboration and adaptability when working in a team. Regular discussions with teammates helped identify potential bottlenecks early and ensured we stayed aligned on the project's goals. I also learned how to communicate technical details clearly, which was essential for solving complex integration issues as a team.

These lessons can be applied in future professional settings by emphasizing the need for collaboration, planning, and adaptability when handling complex tasks. The technical experience gained from creating the VectorDB and integrating it with a Chatbot, along with designing fallback systems, will be invaluable in future roles. Additionally, this project reinforced the importance of building scalable and reliable solutions that can adapt to changing data and user needs.