AI and Machine Learning Boot Camp Graduation Project

Sprints

Market Eye AI-Powered Stock Analysis System

1. General Description of the Project

You are part of the AI Development Team at VisionEdge, a financial consulting company. In December 2024, your manager, Mr. Alex Carter, calls a meeting and gives you this assignment: "We need a smarter way to understand the market and guide our clients with real insights. I want you to build an AI system that collects stock data, forecasts prices for January 2025, generates recommendations through an LLM, and evaluates the forecasts against real market results. This will help us and our clients make data-driven investment decisions and position our company ahead of the competition."

---

2. Project Set-up

- **Recommended Team Size:** 3–5 members
- **Estimated Time:** About 1 week.

---

3. Objectives

- Build a full-stack AI system that:
- Collects real-world stock data.
- Analyzes historical stock performance.
- Forecasts stock prices for January 2025.
- Compares the predicted vs real stock prices after January.
- Uses Gemini LLM to generate professional investment recommendations.
- Provides results through a Streamlit web app.
- Allows users to download analysis reports as PDFs.

---

4. Tools to be Used

- **Frontend:** Streamlit.
- **Backend:** FastAPI **[Bonus]**
- **Agentic Framework:** CrewAI
- **Authentication & Database:** SQLite.
- **LLM:** LLM models to be called with an API such as Gemini using Google AI Studio.
- **Deployment:** Streamlit community or any open-source alternative. **[Bonus]**
- **Version Control:** GitHub(for source code management).

5. Functional Requirements

1- Database & User Management

Steps:

- Implement user authentication: sign-up and login pages.
- Securely hash and store passwords.
- Define and create two tables:
  - `users(user_id, username, password_hashed, registration_date)`
  - `activity_logs(user_id, action, timestamp)`
- Log every user action (registration, login, app usage) into `activity_logs`.
- **Deliverable:**
  Working user authentication module with `users` and `activity_logs` tables fully implemented and logging enabled.

2- CrewAI Agents

- Steps:

- **Agent 1 – Data Collector:**
  - Load “World Stock Prices” data from Kaggle for specified tickers.
  - Clean and prepare Pandas DataFrames.
- **Agent 2 – Data Processor:**
  - Compute high, low, and 2020 growth% for each ticker.
  - Compare sectors: Tech vs. Finance vs. Sportswear.
  - Train and use an LSTM/MLP model on historical “Close” prices.
  - Predict January 2025 prices and (once available) calculate MSE & RMSE.
- **Agent 3 – LLM Recommendation Generator:**

- Build prompts dynamically (company + analysis + forecast + errors).
- Call the Gemini API to generate market-trend summaries and Buy/Hold/Sell recommendations.

- **Deliverable:**
 Three deployed CrewAI agents: Data Collector, Data Processor, and LLM Recommendation Generator, all tested end-to-end.

3- **Dataset Integration**

- Steps:

- Ingest the Kaggle "World Stock Prices – Daily Updating" dataset.
- Automate daily updates through December 2024.
- **Deliverable:**
 A scheduled pipeline that fetches and updates the stock-price dataset daily up to Dec 31, 2024.

4- **Analysis & Forecasting**

- Steps:

- Compute analytics: highest price, lowest price, and annual growth for each ticker.
- Generate cross-company and cross-sector comparisons.
- Train forecasting model on "Close" prices.
- Produce January 2025 forecasts.
- Evaluate forecasts by calculating MSE and RMSE against real Jan 2025 data.
- **Deliverable:**
 Analytical reports, comparison charts, forecast outputs, and error metrics (MSE, RMSE).

5- **Streamlit Front-End**

- Steps:

- Build authentication UI: sign-up and sign-in forms.
- Add a ticker selector (single or multi-select dropdown).
- Create dashboard pages for: analytics tables, forecast results, and real vs. forecast charts.
- Implement a PDF-report download button.
- Display LLM-generated recommendations.
- **Deliverable:**
  Deployed Streamlit app with all pages, selector, charts, PDF export, and recommendation panel.

6- **PDF Report Generation**

- Steps:

- Use ReportLab or FPDF to assemble:
  - Analytical tables and figures.
  - Forecast vs. real-data comparisons.
  - Error metrics (MSE, RMSE).
  - LLM-generated text.
- **Deliverable:**
  A PDF-generation module producing comprehensive, formatted stock analysis reports.

7- **LLM Integration**

- Steps:

- Integrate Google Gemini API.
- Dynamically assemble prompts from agent outputs.
- Ensure responses are fully AI-generated (no static templates).
- **Deliverable:**
  Functional Gemini API integration that returns contextual market insights and recommendations.

8- **GitHub & Version Control**

- Steps:

- Organize repository with clear folder structure (agents, models, frontend, backend).
- Enforce frequent commits with descriptive messages.
- Include a `.gitignore` and contribute guidelines.
- **Deliverable:**
  A public GitHub repo with logical structure, documented commit history, and contribution instructions.

```
/backend
    /agents
    /models
    /database
/frontend
    /pages
/docs
    /report_templates
requirements.txt
README.md
```

9- **Final Presentation**

- Steps:

- Record a short video covering:

1. Business problem overview
2. System architecture
3. Live Streamlit demo
4. Code walkthrough highlights
5. Lessons learned and challenges

- **Deliverable:**
  A recorded presentation video (and accompanying slides) encapsulating the entire project.

6. Non-Functional Requirements

1. Performance

- **Response Time**: The system should have a response time of less than 10 seconds for user interactions under normal load conditions to ensure smooth, real-time conversations.

2. Usability

- **User Interface**: The interface should be intuitive and accessible to users of varying technical skill levels, ensuring ease of use for both candidates and recruiters.
- **Accessibility**: Adherence to WCAG 2.1 guidelines to ensure the application is accessible to users with disabilities.

3. Security

- **Authentication and Authorization**: Implement strong authentication mechanisms and ensure that authorization practices follow the principle of least privilege.

7. Dependencies

- **Python 3.x** (for all core logic & CrewAI Framework)
- **SQLite** (bundled with Python for user-auth and activity logs)
- **CrewAI SDK** (agent orchestration & execution)
- **TensorFlow** or **PyTorch** (deep-learning forecasting models)
- **Streamlit** (interactive web frontend)
- **Gemini API client** (to call Google's Gemini LLM)
- **ReportLab** or **FPDF** (PDF report generation)
- **Git/GitHub** (version control and collaboration)

---

8. Final Deliverables

- ☐ User Authentication Module
- ☐ CrewAI Agents Suite
- ☐ Automated Dataset Pipeline
- ☐ Analytics & Forecasting Reports
- ☐ Streamlit Web App
- ☐ PDF Report Generator
- ☐ Gemini API Integration
- ☐ Deployment Artifacts (Streamlit & FastAPI URLs)
- ☐ GitHub Repository (structure, docs, commits)
- ☐ Final Presentation Package (video + slides)

9. Conclusion

In wrapping up the Market Eye project, you've built a full-stack, AI-driven stock analysis platform that brings together:

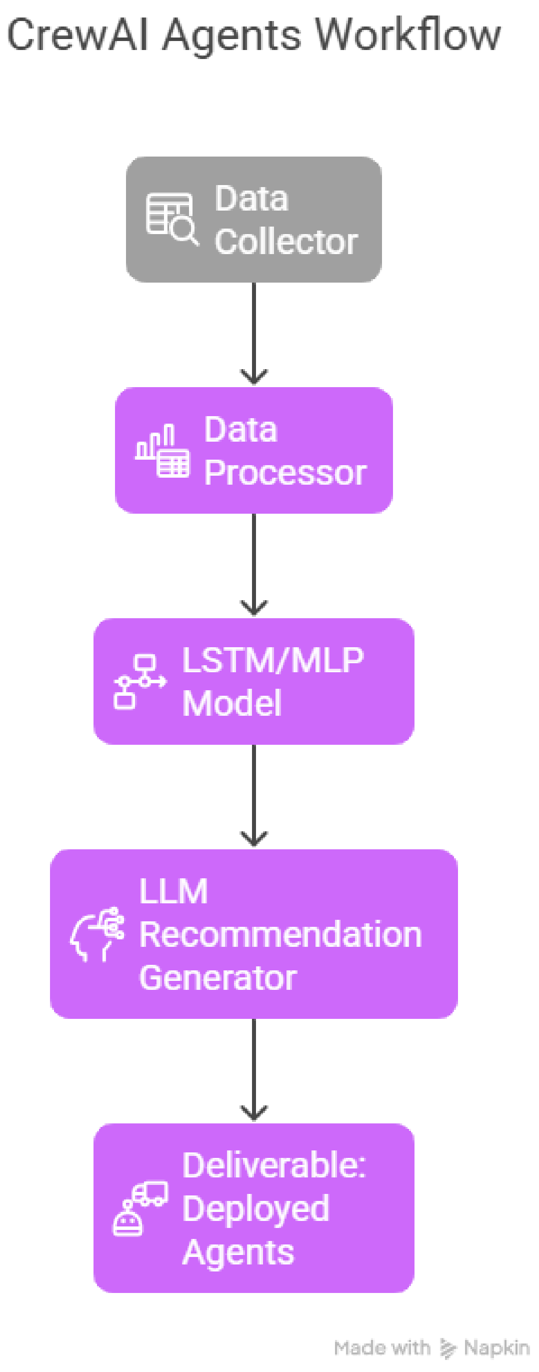
- **Secure user management**, ensuring each analyst's journey is tracked and protected
- **Automated data pipelines** that keep your price history fresh through December 2024
- **CrewAI agents** that collect, process, forecast, and then translate raw numbers into clear, actionable insights
- A **Streamlit interface** where anyone can sign in, pick tickers, explore analytics and forecasts, download PDF reports—and even get AI-powered buy/hold/sell guidance
- A robust **FastAPI backend**, flexible PDF-generation modules, and clean GitHub versioning for maintainability
- **Gemini LLM integration** to turn your models' outputs into human-readable market commentary

- And finally, a **polished presentation package** (video + slides) to showcase how all these pieces fit together.

Together, these deliverables don’t just demonstrate technical prowess—they empower traders, analysts, and decision-makers with an end-to-end system that’s as informative as it is engaging.

Looking ahead, you could deepen the platform by adding real-time data feeds, expanding to other asset classes, or integrating advanced memory layers so the chatbot “remembers” past conversations even longer. But as it stands, Market Eye lays a solid, scalable foundation for turning market data into confident, data-driven decisions—congratulations on bringing it all together!

10. System Diagram


CrewAI Agents Workflow
Data Collector
Data Processor
LSTM/MLP Model
LLM Recommendation Generator
Deliverable: Deployed Agents
Made with Napkin