BearWatch

By: Aisha Ahmad, Jared Bergenstock, Owen Byron, Joel David, Irene Paul, Emilia Szynwald

1. Introduction

BearWatch is a real-time trading information application. An end-user specifies a ticker for which a quotation, trend chart, and relevant fundamental data are displayed. A real-time data feed is provided, which the application uses to reflect changes in price and graphical data to be displayed. News related to the selected ticker is displayed in reverse chronological order. The application is designed with Flask, Python, and Typescript to provide an intuitive interface for users to explore various financial markets with a focus on the U.S. financial market. The application integrates with Yfinance API and News API to fetch real-time updates on the financial market, including price, news, and market indicators. This app is designed to help U.S. users stay informed about any financial market.

1.1. Purpose:

This paper is a guide for the technical requirements and specifications needed to navigate the BearWatch financial market application. This project serves as an application to allow users to retrieve real-time information from the Yfinance and News API.

1.2. Project Scope:

The project scope is building a user-friendly online application that provides an interface for the Yfinance API to receive and display real-time financial market data. The News API will allow users to see relevant updates and headlines that may impact the financial market (politics, policy changes, etc.).

1.3. Target Audience:

The target audience is intended to be U.S. customers with a primary interest in the U.S. financial market.

2. Technology and the Operating Environment

2.1. Product Features

BearWatch provides several features that allow users to obtain information about the stock market.

2.1.1. **News Page:** Users can access news articles regarding up-to-date information about the stock market through the Yahoo Finance (Yfinance) API. News articles are sorted in reverse chronological order for easier viewing.

2.1.2. **Trend Charts:** Users can view trend charts of the stocks they are interested in. These charts will also include important financial data such as: previous close, day range, year range, market cap, avg. volume, P/E ratio, dividend yield, and the primary exchange.

2.1.3. **Search Engine:** Users can search for stocks of their choosing with the search bar feature. The search bar executes an app callback, which prompts Dash to interact with the necessary APIs to update the dashboard.

2.1.4. **Similar Stock Recommendations:** BearWatch will recommend similar stocks to the stock being currently viewed based on data from the Yahoo Finance API.

2.1.5. **Accessibility Features:** To improve usability for colorblind users, trend charts can be viewed in colorblind mode for better visibility.

2.2. Product Description

2.2.1. Front-End

The front-end of the BearWatch app is created using Dash, a framework for building web applications. The layout of the dashboard is created with HTML components through Dash. The visualization of the trend charts is created with Plotly. The front-end user interface consists of a search bar where the user can enter a stock ticker symbol or name, and real-time stock data is displayed, such as the stock's trend chart, key stock metrics, and news regarding the specific stock and the stock market as a whole.

2.2.2. Back-End

The back-end of the BearWatch app is created using Flask and Dash. Flask operates as the web server which manages server requests and runs the application. Dash serves as the application's layout, as well as the primary way of interaction between users and the server. Yahoo Finance API is used in conjunction with Pandas to acquire and subsequently process stock data.

2.2.3. API

The Yahoo Finance API is utilized by the application to fetch both real-time and historical stock market data based on the user's input. Data to be retrieved by the Yahoo Finance API include:

stock trend data, key financial metrics, market news, and similar stock recommendations. The News API will also be utilized to fetch supplementary stock market news.

2.3. User Classes and Characteristics

The application is intended solely for unregistered users, with full functionality of the application granted to these users without requiring account registration or authentication.

2.3.1. **Unregistered Users:** Users can browse through information regarding every stock available through Yahoo Finance. Users will be able to access these stocks' trend charts, key financial metrics, news, and similar stocks without requiring account creation.

2.4. Operating Environment

2.4.1. **Supported Platforms:** The application is a web-based platform designed to run on all modern internet browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.

2.4.2. **Hardware Requirements:** The application is designed to run on all devices that can run their internet browser of choice with internet connection.

2.4.3. **Network Requirements:** Stable internet connection is required to access the application and to communicate with the Yahoo Finance API.

2.4.4. **Server Environment:** The application’s server can be hosted on any system running Linux, macOS, or Windows with Python installed and a stable internet connection. It can be hosted on a local machine or a cloud platform.

2.5. Design and Implementation Dependencies, Constraints, and Assumptions

2.5.1. Dependencies:

2.5.1.1. **Third-Party API Dependency:** The application depends on the Yahoo Finance API to fetch stock market data. If the API is unavailable, then users may experience missing data.

2.5.2. Constraints:

2.5.2.1. **Read-Only Access:** The application provides read-only access, which means users cannot bookmark stocks they are interested in or receive recommendations personalized to a specific user.

2.5.2.2. **Device and Browser Performance:** The performance of the application may vary based on the user's device and browser used to access the application. Older browsers and devices may provide a suboptimal experience.

2.5.3. Assumptions:

2.5.3.1. **Internet Connectivity:** Users will have stable internet connection while accessing the application to receive accurate stock market data.

2.5.3.2. **API Availability:** The Yahoo Finance API will remain accessible and provide up-to-date stock market data and news.

2.5.3.3. **User Knowledge:** The user will have basic knowledge of the stock market, as the application does not provide explanations or financial advice.

3. System Users and Features:

3.1. Home Page:

3.1.1. *Description:*

The home page is presented to all users and serves as a central page for easy navigation through the different stock pages and news tab. It is the starting point for using and accessing all functionalities of the program. To navigate back to the home page, a user must click on the application name "BearWatch." It has the following key features:

- **Search Bar:** Users can search using a name or ticker symbol for the stock they are looking for. The feature will lead the user to another page which holds further information regarding their chosen stock. (Described in Section 3.2)
- **Trending Stocks:** Users will see a list of the current trending stocks. (Described in Section 3.3)
- **Current News:** Users will see a list of the current news in reverse chronological order regarding stocks. The feature, if clicked on the title, will lead the user to another page that holds further information regarding current news. (Described in Section 3.4)

- **Main Page Graph:** Users will see a graph alternating between the S&P 500, Nasdaq, and Dow Jones. (Described in Section 3.5)

3.1.2. *System-User Interaction:*

3.1.2.1. The user clicks the application name, "BearWatch," from any page.

3.1.2.2. The system brings the user to the home page.

3.1.3. *Functional Requirements:*

3.1.3.1. R-1: Ensure the ability to click "BearWatch" on the secondary main page or Current News page to lead to the Home Page.

3.2. <u>Search Bar:</u>

3.2.1. *Description:*

This feature is a high-priority functionality of the application. It appears on all the pages and allows the users to search for a particular stock by name or ticker symbol. The auto-complete feature suggests which stock is being typed by the user based on partial input. Once the user has searched for a particular stock and clicked on the name/ticker of their chosen stock, our secondary main page will open containing further information about said stock (further information regarding secondary main page in Section 3.6).

3.2.2. *System-User Interaction:*

3.2.2.1. The user enters the stock name or ticker symbol into the search bar.

3.2.2.2. The system uses the Yahoo Finance API to retrieve the information regarding that stock and displays the appropriate information on its secondary main page.

3.2.3. *Functional Requirements:*

3.2.3.1. R-2: Create a search bar that allows users to search by name or ticker symbol.

3.2.3.2. R-3: Create an auto-complete feature that suggests which stock is being input.

3.2.3.3. R-4: Create a secondary main page for each stock that includes a graph, statistics, news, recommendations for other stocks, and current trends for the chosen stock.

3.3. <u>Trending Stocks:</u>

3.3.1. *Description:*

This feature displays the current trending stocks, as well as their key information.

3.3.2. *System-User Interaction:*

3.3.2.1. N/A

3.3.3. *Functional Requirements:*

3.3.3.1. R-5: Display the proper trending stocks and their key information.

3.4. <u>Current News:</u>

3.4.1. *Description:*

This feature will showcase the current news in reverse chronological order. If the user clicks on the title “Current News,” it will lead the user to a page containing purely current news articles.

3.4.2. *System-User Interaction:*

3.4.2.1. The user clicks the “Current News” title.

3.4.2.2. The system redirects the user to the Current News page.

3.4.3. *Functional Requirements:*

3.4.3.1. R-6: Create a Current News page.

3.4.3.2. R-7: Ensure proper news display on the Home Page and secondary main pages.

3.4.3.3. R-8: Ensure the ability to click "Current News" on the Home Page and secondary main pages to lead to the Current News page.

3.5. <u>Main Page Graph:</u>

3.5.1. *Description:*

This feature will display real-time graphs of the S&P 500, Nasdaq, and Dow Jones alternating. The graphs will be interactive for the user to view the key information, such as date, close, open, high, low, and volume. The graphs will also allow the user to look at the 1 day, 5 day, 1 month, 6 months, 1 year, and 5 year history.

3.5.2. *System-User Interaction:*

3.5.2.1. The user hovers over the graph at any chosen location.

3.5.2.2. The system informs the user of the key information.

3.5.3. *Functional Requirements:*

3.5.3.1. R-9: Create the real-time graphs of the S&P 500, Nasdaq, and Dow Jones.

3.5.3.2. R-10: Ensure the interactiveness of all three graphs.

3.5.3.3. R-11: Ensure proper key information is displayed when hovered over.

3.5.3.4. R-12: Ensure the ability for the graph to alternate between the three stock options.

3.5.3.5. R-13: Ensure the ability for the graph to show different day/month/year history.

3.6. Secondary Main Page:

3.6.1. *Description:*

This page is presented to all users once they have searched for a stock name/ticker symbol, it serves as the central page regarding that particular stock. It has the following key features:

- **Stock Graph:** Users will see a graph regarding their chosen stock. (Described in Section 3.7)
- **Stock Information:** Users will see information regarding their chosen stock, such as information such as who founded it, when it was founded, etc. (Described in Section 3.8)
- **Stock Statistics:** Users will see trends regarding their chosen stock, such as previous close, day range, year range, market cap, avg. volume, P/E ratio, dividend yield, and the primary exchange. (Described in Section 3.9)
- **Stock Relevant News:** Users will see a list of the current news related to the stock in reverse chronological order. The feature, if clicked on the title, will lead the user to another page that holds further information regarding general current news. (Described in Section 3.10)

- **Current News:** Users will see a list of the current news in reverse chronological order regarding stocks. The feature, if clicked on the title, will lead the user to another page that holds further information regarding current news. (Described in Section 3.4)
- **Recommended Stocks:** Users will see a list of recommended stocks that lie in the same industry as the one they've searched. (Described in Section 3.11)

3.6.2. *System-User Interaction:*

3.6.2.1. The user searches for a stock by stock name/ticker symbol.

3.6.2.2. The system brings the user to the secondary main page.

3.6.3. *Functional Requirements:*

3.6.3.1. R-14: Ensure the ability to search a stock to lead to the secondary main page.

3.7. <u>Stock Graph:</u>

3.7.1. *Description:*

This feature will display real-time graphs of the chosen stock. The graphs will be interactive for the user to view the key information, such as date, close, open, high, low, and volume. The graphs will also allow the user to look at the 1 day, 5 day, 1 month, 6 months, 1 year, and 5 year history.

3.7.2. *System-User Interaction:*

3.7.2.1. The user hovers over the graph at any chosen location.

3.7.2.2. The system informs the user of the key information.

3.7.3. *Functional Requirements:*

3.7.3.1. R-15: Create the real-time graphs of the chosen stock.

3.7.3.2. R-16: Ensure the interactiveness of the graph.

3.7.3.3. R-17: Ensure proper key information is displayed when hovered over.

3.7.3.4. R-18: Ensure the ability for the graph to show different day/month/year history.

3.8. <u>Stock Information:</u>

3.8.1. *Description:*

This feature will display information such as who founded the chosen stock, when it was founded, etc.

3.8.2. *System-User Interaction:*

3.8.2.1. N/A

3.8.3. *Functional Requirements:*

3.8.3.1. R-19: Ensure the ability to display proper information regarding the stock.

3.9. <u>Stock Statistics:</u>

3.9.1. *Description:*

3.9.1.1. This feature will display trends within the stock, such as previous close, day range, year range, market cap, avg. volume, P/E ratio, dividend yield, and the primary exchange.

3.9.2. *System-User Interaction:*

3.9.2.1. N/A

3.9.3. *Functional Requirements:*

3.9.3.1. R-20: Ensure the ability to display proper trends regarding the stock.

3.10. <u>Stock Relevant News:</u>

3.10.1. *Description:*

3.10.1.1. This feature will showcase the current news related to the stock in reverse chronological order.

3.10.2. *System-User Interaction:*

3.10.2.1. N/A

3.10.3. *Functional Requirements:*

3.10.3.1. R-21: Ensure proper relevant news display on secondary main pages.

3.11. <u>Recommended Stocks:</u>

3.11.1. *Description:*

This feature will display a list of recommended stocks that lie in the same industry as the one the user has searched for.

3.11.2. *System-User Interaction:*

3.11.2.1. N/A

3.11.3. *Functional Requirements:*

3.11.3.1. R-22: Ensure that stocks of the same industry will appear.

3.12. <u>Current News Page:</u>

3.12.1. *Description:*

3.12.1.1. This feature will showcase the current news in reverse chronological order.

3.12.2. *System-User Interaction:*

3.12.2.1. The user clicks the “Current News” title on either the Home Page or a secondary main page.

3.12.2.2. The system redirects the user to the Current News page.

3.12.3. *Functional Requirements:*

3.12.3.1. R- 23: Ensure proper news display on the Current News page.

3.13. <u>Accessibility:</u>

3.13.1. *Description:*

This feature is of high priority, and users can navigate through various visual effects specifically to make it more accessible to those who have difficulty discerning between certain colors.

3.13.2. *System-User Interaction:*

3.13.2.1. The user clicks on the “Accessibility button” found on all pages

3.13.2.2. The system provides a dropdown menu that allows users to select the color-blindness option that best applies to them.

3.13.2.3. The user selects their option.

3.13.2.4. The system modifies the user interface to reflect the changes.

3.13.3. *Functional Requirements:*

3.13.3.1. R-24: Create an Accessibility button that all users can access.

3.13.3.2. R-25: Create a dropdown option once the button has been clicked.

3.13.3.3. R-26: Include an option to adjust the page colors for better accessibility, accommodating color blindness and other visual needs.

4. External Interface Requirements:

4.1. <u>Example Interfaces for Users (UI):</u>

The *BearWatch* interface is designed to be adaptive and responsive on desktop devices, providing users with seamless access to stock market data and financial news. Some important UI components consist of:

4.1.1. Home Page

4.1.1.1. Provides a clear path regarding all the information stated within popular trends and stocks within the market. The center of the page features an interactive graph displaying trending stocks. Users can toggle between the top five trending stocks and adjust the displayed timeline using buttons above the graph. Below the graph, an overview section provides real-time updates on trending stocks, ensuring users stay informed of market fluctuations. The right section contains a list of trending stocks labeled by their ticker symbols, with a search bar above allowing users to look up specific stocks quickly. The left section displays stock-related news headlines, offering users a glance at market-relevant news updates.

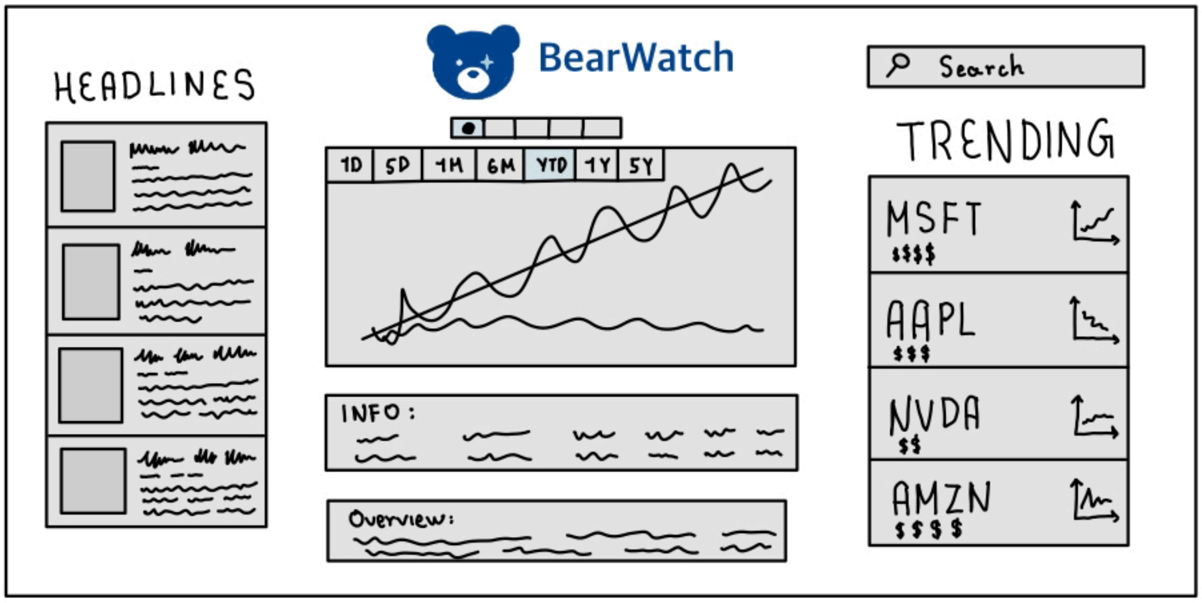

Figure 1: Home Page

4.1.2. News Page

4.1.2.1. When you click on the *Headlines* section on the *Home Page*, the News Page shows up. This page presents trending financial news articles in a block structure, each featuring a summary and an accompanying image. The news layout ensures easy navigation, allowing users to quickly browse through multiple articles. A centered logo at the top of the page serves as a navigation element—clicking it will redirect users back to the *Home Page*.

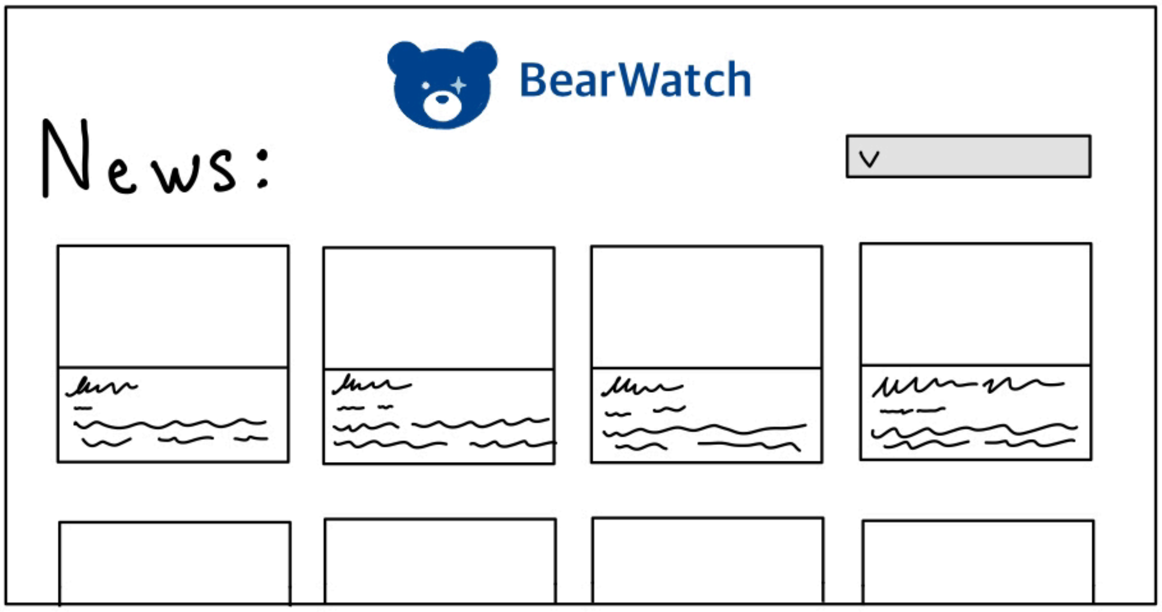

Figure 2: News Page

4.1.3. Stock Information Page

4.1.3.1. When a user searches for a stock by entering either the ticker symbol or company name in the search engine—or by clicking on any of the trending stocks—the web application navigates to the *Stock Information Page*, providing detailed insights into the selected stock. The graph from the Home Page will still contain buttons that change the timeline of how the stock is displayed more information and statistics are listed below. Below the search bar in the top right corner, there are two sections containing stocks that people also bought in comparison to the stock you are looking at and trending stock as seen on the Home Page. Below this first half are two sections of news articles, one related to the specific stock and the other carrying the headlines seen on the Home Page. The logo will follow the same procedure as the News Page, when you click, it will lead back to the Home Page.

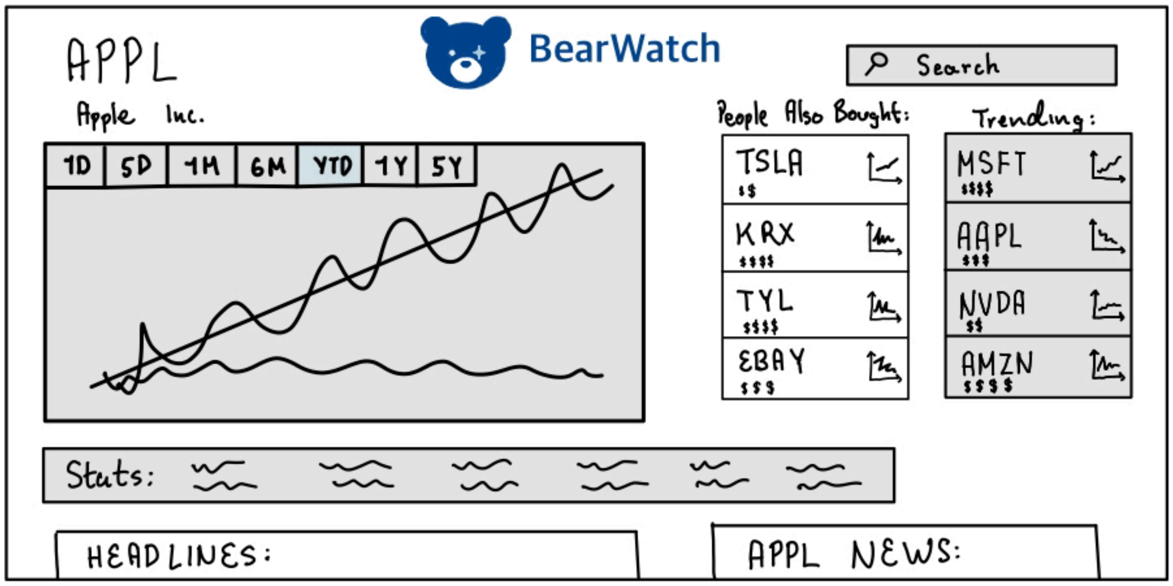

Figure 3: Stock Information Page

4.2. Interfaces for API

The implementations using YFinance and News API allow for a steady stream of information along

4.2.1. YFinance - Providing most of the data, including stock market data, historical prices, real-time stock updates, financial news, and more.

4.2.2. News API - Provides information specifically for the Headline section and the News Page to fetch financial and business news from multiple sources to stay updated on relevant information.

4.3. Interfaces for Hardware

With this web application specifically, there is no need for any hardware components. However, this web page is compatible with:

4.3.1.1. Desktops (Windows, macOS, Linux) or laptop

4.3.1.2. Web Browsers (Google Chrome, Safari, Microsoft Edge)

4.4. Interfaces for Software

Within the software components, this includes:

4.4.1.1. Dash - A Python framework that allows for building an interactive web application and rendering UI elements dynamically.

4.4.1.2. Flask - A web framework that powers the backend by handling the API and user interactions.

4.4.1.3. YFinance API - the main component is retracting financial data, timelines, ticker names, and more to use as the foundational link to power through the pages.

4.4.1.4. News API - used to attain information specifically for trending news to adapt throughout the different pages.

4.4.1.5. Pandas - an analysis library to process and structure the data from the API

4.4.1.6. Plotly - A graphing library integrated with Dash to create a more interactive user experience and provide visuals.

5. Non-Functional Requirements:

5.1. Performance Requirements:

5.1.1. **Response Time:** API requests to the real-time trading data feed(Yfinance) must be processed within 200 milliseconds to ensure a smooth user experience. Rendering pages and updating price quotes/trend charts will occur in under one second.

5.1.2. **Scalability:** The system must accommodate at least 100 users at the same time without noticeable performance issues. The backend infrastructure needs to handle increased data traffic and user activity efficiently.

5.1.3. **Throughput:** The application will support the processing of at least 10,000 market data updates per minute without downtime or significant latency.

5.2. Usability:

5.2.1. **User-Friendly Design:** The interface will allow users to effortlessly search for a stock ticker and stock name and view real-time price quotes, trend charts, and other fundamental data.

5.2.2. **Layout**: The user interface will be optimized for usage on desktop devices and mobile websites.

5.2.3. **Accessibility:** The system will comply with accessibility guidelines, ensuring keyboard navigation and appropriate contrast ratios for readability for people with certain disabilities.

5.3. Availability:

5.3.1. **System Uptime:** The application will maintain 99.99% uptime, excluding scheduled maintenance, which will be announced in advance.

5.3.2. **Fault Tolerance:** Automatic failover mechanisms will be implemented to ensure continuous service in case of server failures or data provider disruptions.

5.3.3. **Error Handling:** If the data feed becomes unavailable, users will receive a clear error message, and cached data will be displayed until service is restored.

5.4. Maintainability:

5.4.1. **Code Readability:** The project will follow industry best practices for code documentation, adhering to general code conventions for readability.

5.4.2. **Modularity**: A modular design approach will be utilized to ensure that updates or changes in data feeds do not impact other system functionalities.

5.5. Accuracy of Data:

5.5.1. **Data Consistency:** Market data updates must be synchronized across all system components, with a maximum discrepancy of one second.

5.5.2. **Data Integrity:** The system must validate and verify incoming market data to prevent incorrect or misleading information from being displayed.

5.6. Reliability:

5.6.1. **System Stability:** The application must be designed to operate continuously without unexpected crashes or disruptions.

5.7. Adherence to Rules and Regulations:

5.7.1. **Financial Compliance:** The system must comply with relevant financial regulations and requirements for handling market data.

5.7.2. **ACM Code of Ethics Compliance**: The system will adhere to ethical standards, ensuring transparency, fairness, and accessibility for all users.

6. Project Management

6.1. Methodology:

6.1.1. The team operates by completing a working portion of the project every week. Weekly meetings, either in person or via Zoom, are attended to make sure any challenges or problems anyone may have are addressed and to make sure progress is on track.

6.2. Team Structure:

6.2.1. Leadership is shared equally among the group, with each person having an equal say and the ability to take charge as needed. Discord is used for day-to-day communication, along with the weekly meetings used to check in on progress. Additionally, meetings will be scheduled as necessary, with everyone's daily schedule being kept in mind to avoid time conflicts. Responsibilities are divided equally, keeping in mind each member's experience and expertise.

6.3. Tools and Technology:

6.3.1. For the backend of the project, Python will be used due to its readability, vast support of libraries, and easy integration with many front-end frameworks. Such frameworks that will be used include Dash and Flask, which are both easy to use and flexible. TypeScript will also be used due to its practical front-end application. GitHub will be used for code review and team collaboration on the code.

6.4. Quality Assurance:

6.4.1. The code is reviewed by members of the team to maintain code quality. This is done by a member who has not worked on said code before updating GitHub. These reviews are done weekly to catch mistakes early and make sure poor code is not updated to GitHub. Also, multiple people have been assigned to each task to maintain accountability and code quality.

6.5. Deliverables and Milestones:

6.5.1. For each task, the number of lines of code needed is evaluated to assess the time needed concretely. The amount of lines of code per day that each team member can write has also been assessed to evaluate how long it will take a given member to complete a given task. These metrics will help assess whether or not the member and the team are on schedule. As we continue with the project, we will evaluate prior tasks to improve our time estimates and thus have a better idea of how to manage our time properly.

6.6. Change Management:

6.6.1. Changes are inevitable as the project progresses and will be dealt with through team meetings and discussions. During each team meeting, any changes within reason can be proposed. The effect these changes have on the workload and timeline of the project will be discussed and taken into consideration. Members will collectively decide whether or not the change will be implemented.

<u>Appendix A:</u> References and Resources

1) YFinance API
    a) "API Reference#." *API Reference - Yfinance*, ranaroussi.github.io/yfinance/reference/index.html. Accessed 12 Feb. 2025.
2) Dash
    a) "Dash Documentation & User Guide." *Plotly*, dash.plotly.com/. Accessed 12 Feb. 2025.
3) News API
    a) "News API – Search News and Blog Articles on the Web." *News API Â" Search News and Blog Articles on the Web*, newsapi.org/. Accessed 12 Feb. 2025.
4) Pandas
    a) "Pandas Documentation#." *Pandas Documentation - Pandas 2.2.3 Documentation*, pandas.pydata.org/docs/. Accessed 12 Feb. 2025.
5) Flask
    a) "Welcome to Flask¶." *Welcome to Flask - Flask Documentation (3.1.x)*, flask.palletsprojects.com/en/stable/. Accessed 12 Feb. 2025.