**!Bears**

!Bears is a one stop shop AI integrated stock tracker application. Users of !Bears will be able to add funds to their account, purchase and sell stocks and manage a portfolio. Within their portfolio an analysis will be visible to the user which will show a daily overall gain/loss. Users will also be able to view relevant news such as market highlights and notable events. AI integration will equip a user with a personalized AI assistant which will act as financial advisor. Finally, users will be able to view a table view of their portfolio as well as a live price tracker of any stock within their portfolio.

**Requirements** (functional and non-functional)

The functional requirements of the app are that it must allow its users to search for a specific stock by entering its ticker, the user should be able to create an account and log in to access their personal portfolio, the app should allow users to add and remove stocks from their portfolio, the app would use AI to let users ask questions regarding stocks and finance, and its users should be able to see financial news related to the stocks owned.  
 The non-functional requirements of the app is that the API calls should be managed efficiently and staying within the 5 calls per minute limit, all user data must be protected and secure, the application should be responsive with an easy to navigate interface that’s user friendly and must consistently return accurate data.

**Specifications** (detailed system behavior)

The application will be working with a Finnhub.io API that will be initialized following a user's successful sign in. Assuming a user has tickers associated with their account, the initial API call will produce a JSON object that accounts for all tickers associated with the user account. This object will be parsed and will produce an average profit/loss analysis for the account on the "main portfolio" page. Along with this average, the main portfolio page will use an AI api for a chatbot that will be prompted as a financial advisor with specialties in each ticker associated with the user account. The main page may also incorporate relevant news for the stocks on the users account. Additionally, a user will also be able to access a particular ticker in the form of a "snapshot". This will be a live price using an API provided web socket. This is where a user will be able to "buy" or "sell" shares of the respective stock. Due to the limitations, users may only add funds of a mock currency.

**World assumptions** (environment, users, constraints)

The world is composed of the real-life financial world and the users interacting with it. It will cater to seasoned and novice investors alike with tools such as portfolio and PnL trackers, and an AI question box interface that can be used to answer financial questions. These tools will rely on external APIs such as Finnhub.io and OpenAI, which influence system behavior but are outside the system boundary. The application will be accessed as a Java desktop application, installed and run directly on the user's device. Users' interactions with it will depend on their own device and internet connection capabilities and constraints, which are also outside the system boundary. Users will be able view the performance of a stock by inputting its ticker symbol, and they can track their own performance by adding their portfolio information and stock holdings. Performance data can span days/months, dependent on economic data gathered from daily market closes. Limited API calls could be a possible constraint, however there are ways around this such as alternative providers or paying for more, if needed.

**Program** (implementation approach)

The program will be a Java desktop application that helps users lookup, track, and analyze stock performance through both live and historical data. The backend will be Firebase, which will hold user and portfolio data. It will calculate profit/loss by using the close data from Finnhub.io API calls. JSON parsing will be used to pull the data from the API calls. The application will also include a ChatGPT API interface via the OpenAI API. This will allow users to ask questions about their portfolio, stocks, financial terms, etc. This will provide users with more real world information to make better informed decisions. Multithreading will most likely be used to separate different users/portfolios from each other, as this will be a better user experience because it will no longer block threads. In order to manage the 5 API call limit per minute, an entire user portfolio will use the API call so all data is returned at once, only using one call per portfolio for better API use management.

**Machine** (hardware/platform requirements)

The program will be in Java and run on the JVM, or Java Virtual Machine. The team is using intelliJ IDEA on Windows 10 or 11. The backend will be developed using Firebase.The API’s being used will be Finnhub.io and OpenAI ChatGPT API. There may be other external services needed as they arise.