The University of Hong Kong

Department of Computer Science

FITE4801 Final Year Project (2024-2025)

# FYP24095: A Central Hub to Facilitate Organization of Financial Information to Improve Financial Decision

Interim Report

MOK TSZ HIM SOLOMON (3035835296)

Under the supervision of Prof. Chow, Kam Pui

Date of Submission: 26th January 2025

# Abstract

With the current positive bull market in both the stock and cryptocurrency market, it incentives novice and experienced investors alike to get into the trading market and improve their own portfolio. This project aims to create a portfolio management app to allow investors to build, check and track their portfolio at a central hub. By focusing on an intuitive interface, a high flexibility design and product comparison function, it aims to accustomed to new investors and investors who wishes to manage their portfolio during off-trading hours. Currently, the project has finished its research and design on API, GUI and libraries. Next, it will begin to work on a program prototype and apply improvements to account for more functionality in the program.

# Acknowledgement

I would like to express my deepest gratitude to Prof. Chow, Kam Pui for supervising this project.

# Table of Contents

# List of Figures

# List of Tables

# Abbreviations

| Abbreviations | Definitions |
|---|---|
| API | Application Programming Interface |
| GUI | Graphical User Interface |

# 1 Introduction

## 1.1 Project Background

Nowadays, there are a lot of different types of financial products out in the market, for example, stocks, ETFs and cryptocurrencies. Hence, there are different types of programs for investors to manage their investment portfolio, each tailored to a different purpose.

However, most of the current managing apps provide the investors a real-time trading platform, where it includes different complex functions that not only help investors to manage their portfolio by building, checking and tracking it, but are also able to trade alongside it at the same app. These solutions are aimed at experienced investors where they have a substantial knowledge on the investment industry.

Therefore, the project aims to provide a central hub, allowing users to build, check and track their portfolio at a single program. The program focuses on 3 aspects, by having an intuitive interface, high flexibility, and product comparison functions. The central hub allows the user to effectively organize, customize and analyze their portfolio, facilitating better financial decisions in the process. The central hub also allows beginners investors to get into the investment industry easier given its nature.

## 1.2 Literature Review

Most of the current solutions as a portfolio management app focuses on different aspects of portfolio management, namely building a portfolio, checking the information of a portfolio and tracking the portfolio.

The first type of solution is a portfolio builder, i.e. InteractiveBroker. It aims to help the user to create investment strategies based on the fundamental data and research, it also

allows back-test for the user to adjust their investment strategies (ibkr, 2018). This type of solution allows the user to build their portfolio from the ground-up, with no prior portfolio. Furthermore, it also provides the user with investment strategies when building their portfolio. For example, the percentage weighting for each industry sector of their portfolio can be adjusted according to the investment direction of the user. This helps the user to create a balanced portfolio suitable for their needs.

The second type of solution is a portfolio checker, i.e. Yahoo!Finance. It aims to allow the user to check information regarding the products in their portfolio, it also allows the user to view their portfolio performance and key metrics (Yahoo!Finance, 2024). This type of solution allows the user to view product information of products, like price history, day change, dividends payout. Furthermore, it allows the user to have an idea on the performance of the stock, thus allowing them to analyze and decide on whether to include or remove it from the portfolio.

The third type of solution is portfolio tracker, i.e. sharesight. It aims to help the user to track the dividends of the products in their portfolio (sharesight, 2024). For example, whether the dividends are kept separately are re-invested into purchasing more shares, when will be the next dividend payout, or the dividend declaration date. This is especially useful when managing a large portfolio where a lot of products have dividends payouts, as their dates tends to differ by a small margin.

## 1.3 Motivation

However, most of the current solutions have two issues.

First, most of the solutions has a steep learning curve to it. Those solutions provide a comprehensive set of functions for the users to use, allowing the user to monitor different stocks, their price history, price change etc. This would be a good application for a

experienced trader as the presented information helps inform the user of the current market adequately.

However, the overwhelming amount of information in the portfolio manager may make potential new traders to be reluctant in getting into finance. As the user now not only has to learn about the knowledge of the financial market, but they would also have to learn to navigate through the abundant functions and information provided by the portfolio manager.

Second, most of the solutions are “active” in nature. Take InteractiveBroker’s portfolio builder for example, it’s function includes live stock prices and recent news (ibkr, 2018). These functions allow trader to effectively trade during live hours, where the market has constant transactions and news. The layout and access of functions are also catered towards usage during live trading hours.

However, it would also mean that the function presented are heavily live trading focused and lack catering for the non-live trading customers. Although the functionality is similar, as at base both are portfolio manager, the layout, presentation and selection of functions would be different.

## 1.4 Objectives

Therefore, the project objective is to tackle the shortcomings by focusing on 3 focuses. Having an intuitive interface, a high flexibility design and a product comparison function.

By going for these 3 focuses, this portfolio app will serve as a central hub allowing the building, checking and tracking of a financial portfolio to be better adopted by any level of investors, including novice investors. By a better facilitation of the organization of all the

financial information at hand, it aims to allow the user to improve and make better financial decisions.

This portfolio manager project is desirable as it allows newcomers to financial trading to get into finance trading more easily with its simplified layout. As they can focus on getting grasp of the knowledge of the financial market, while greatly lowering the need to learn about the portfolio manager itself early on. Furthermore, the project also caters for the non-active traders and more casual traders, providing them with a manger that allow to organize and improve their financial portfolio even at off-trading hours.

## 1.5 Outline

In the following sections, the Methodologies explores the three focuses of the project, then go into the Project Architecture, including the Programming language, API, Libraries and Modules, the front-end interface and back-end development design. Current Progress explores the additional progress done up-till now, and the difficulties encountered and mitigation, following up with a current progress summary. Project Schedule includes to improvement planned to be implemented over the course of the second semester. Conclusion then acts as a conclusion to the report.

# 2 Methodologies

## 2.1 Overview

The following Methodologies section will go through of the three focuses of the portfolio management app, then the project architecture of the program followed by the programming language, API, Libraries and Modules chosen for this project. It further explores the front-end interface and back-end development design.

## 2.2 Focuses

As mentioned, this project develops based on 3 focuses. An intuitive interface, a high flexibility design, a product comparison function.

First, an intuitive interface is categorized as “allowing the user to easily find out what they are looking for”. This means that the user would be able to quickly access their desired information intuitively. This is done by a GUI design of having a simple layout upfront with the crucial information displayed, where the more advanced information can be displayed by expanding into more detailed functionality at the user command. This is done to tackle the issue of a steep learning curve of the program itself. While the program does consist of more in-depth and advanced functions, they will not be displayed at the front at default. This allows novice traders to focus on managing the important functions when they first get started with the program, and only when they become more advanced in the field, they can call out the different previously ‘hidden’ functions. Presenting only the crucial function at default effectively eliminates the need to extensively learn about the program itself at first.

Second, a high flexibility design is adopted for the portfolio manager. The layout of the program would be highly customizable. This is done by using a window-ed GUI, allowing the user to move and resize the displayed information according to their needs.

Furthermore, it also allows the user to save and recall their own layout presets, allowing them to create different presets of the layout according to their own different needs. This can further tackle the issue of a steep learning curve of the program itself. As the user will be having a personalized layout of the program themselves, which is customized to their own preferences and usage, they would be get comfortable using the program very quickly as they have designed the layout themselves. While they do have to understand the information displayed in the individual windows, using the program will be more intuitive and better suits their own needs.

Third, a product comparison function is implemented as one of the functions in the program. This function allows a direct comparison on the information of different financial products, i.e. the Open price, Close price. The differences will also be colorized as to better indicate which product is performing better at each section. This helps tackle the issue of the "active" nature of the program. The product comparison function allows and more in-depth look and understanding of the products, which would not be usually done during active hours of trading. This helps cater the needs for off-active hours portfolio management for the investors using the program.

## 2.3 Project Architecture

The program is divided into the front-end user interface and the back-end interface. At the front-end, the user will communicate with the program from the Graphical User Interface (GUI). When the user requests data, the program first checks the data base to see if the requested data is in the database. If the data is not in the database or there are additional data need to be collected to fulfil the request, the program calls the API to get data from the respective sites, then save the data in the database and display it accordingly to the user

through the GUI interface. The checking step with the database allows the reduce of repeated API calls for the same set of data.

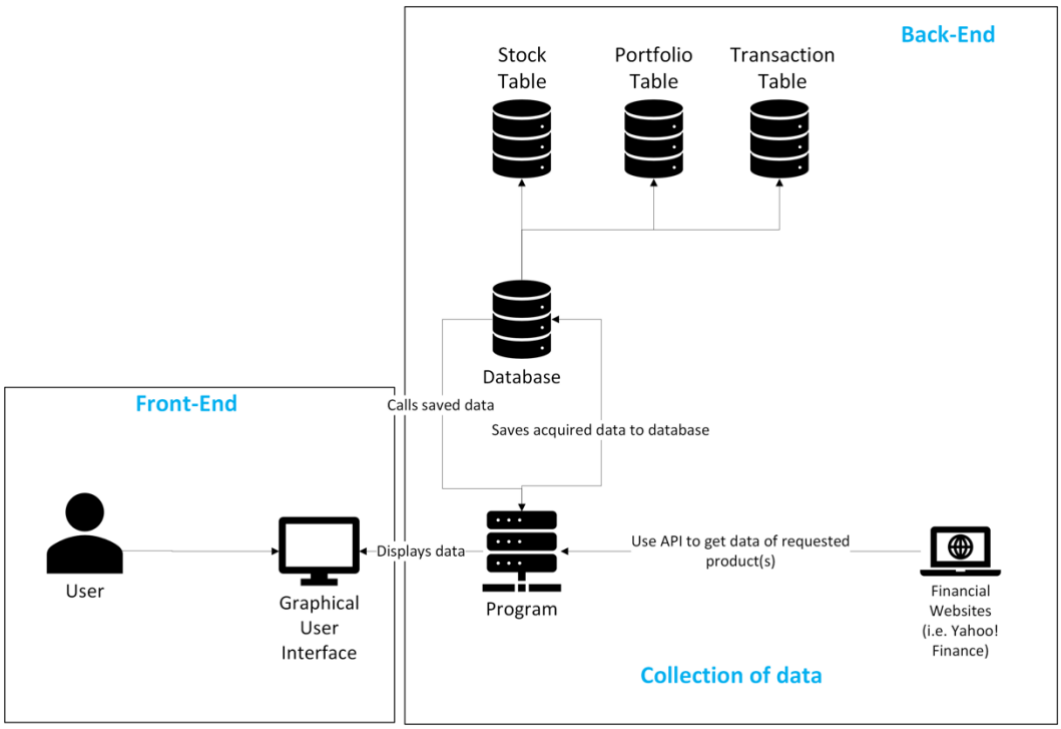


*Figure 1.1: The System Architecture Design*

## 2.4 Programming language, API, Libraries and Modules

Python is the main programming language used for the project as it is excellent at data manipulation. It offers an extensive range of different libraries to be used (Academy, 2024), including the adaption of different data-manipulation libraries, effectively allowing the data of the financial product to be adjusted to the project's needs.

yFinance is Yahoo!Finance's API, used to download the market data from it. This API offers to fetch the available financial and market data from Yahoo!Finance in a pythonic way (Ranaroussi, 2025). This is the main source of financial data obtained by the project.

pandas is one of the libraries used for this project, is a fast and easy to use open source data analysis tool, which allow flexible and powerful data manipulation, it is also built on top of the Python programming language (pandas, 2024). This would be used after

acquiring the information with the API, allowing arranging and organizing the information returned from it.

Matplotlib is another library used for this project, it is a comprehensive library allowing static, animated and interactive visualization within Python (Matplotlib, 2024). This allows the visualization of the financial products information acquired with the API, i.e. the price history. It also allows the presentation of the information of the user portfolio, like the sector percentage of it.

Sqlite3 is the modules chosen for the database of this project, it is a module helping the integrate of the SQLite database when using Python, it allows straightforward interface for interacting with SQLite database (GeeksforGeeks, 2024). It also comes along with Python after 2.5x version. This allows the python program easily to access the database, allowing it to create, add and modify the database and it's table, which would be crucial for an application like this current one where an extensive database management is needed.

## 2.5 Front-end User Interface

The front-end user interface is displayed in the form of a graphical user interface (GUI). This allows the program to adapt fulfil the focus of having an intuitive interface and a high flexibility design**.** The chosen preferred library is CustomTkinter and Tkinter. CustomTkinter is a modernized version and customizable python UI-library based on Tkinter (Schimansky, 2024). The modernized aesthetic will align with the modern expectations of a program, allowing the user to understand that this program is a modern application. Furthermore, since the CustomTkinter is built upon the Tkinter GUI toolkit, the original Tkinter would also be used for its basic functionality functions.

When designing the front-end user interface, it is important to take note of the display hierarchy in this program. Deciding on which of the data should be displayed up front and which should be hidden away would be crucial for the focus of institutive interface.

## 2.6 Back-end User development

The back-end development includes the design of the architecture of the database. Currently, the database consists of 3 main tables. The Stock Table, which includes the data of the stock. The Portfolio table, which includes the data of the user portfolio. And the Transaction table, which includes all the transaction done by the user. Each of the table has its own schema. The database is managed with the SQLite3 module as mentioned in section 2.4 Programming language, API, Libraries and Modules.

The Stock table consists of the following schema. As all the stock data will be put in this table, there is an additional index of the symbol of the stock, this allows quick access to a specific stock when needed. Each row of the data will represent the data of a specific stock on a specific date. For example, the data of the stock AAPL on date 20241226.

*Table 1.1 Stock Table Schema*

| **Column Name** | **Datatype** | **Explanation** |
|---|---|---|
| Ticker | TEXT | Unique symbol of the stock |
| Date | DATE | Date of the data it is from |
| Open | REAL | Opening Price |
| Close | REAL | Closing Price |
| High | REAL | Highest price traded |
| Low | REAL | Lowest price traded |
| Dividends | REAL | Dividends payout |
| StockSplits | INTEGER | Stock Splits |

| | | |
|---|---|---|
| Volume | INTEGER | Total number of shares traded |

The Portfolio table consists of the following schema. It is mainly used to store the user's current portfolio. As it is within the same database as the Stock table and Transaction table, it can also access the latest price info of the stocks in the portfolio from it. Each row of the data will represent the holdings of a stock. For example, the holdings of stock AAPL.

*Table 2.2 Portfolio Table Schema*

| **Column Name** | **Datatype** | **Explanation** |
|---|---|---|
| Ticker | TEXT | Unique symbol of the stock |
| AvgPrice | REAL | Avg Buy-in Price |
| Quantity | REAL | Quantity |
| TotalInvestment | REAL | Total amount invested |
| CurrentValue | REAL | Current Value of stock |

The Transaction table consists of the following schema. It is mainly used to document all the transaction made by the user, including buying and selling of any asset. This helps keep track of the total trading history of the user for reference. It also allows to see their previous performance, as assets that has been bought and sold will no longer exist in the Portfolio table. As this table is also in the same database with Portfolio table, it can also update the Portfolio table accordingly when there is a new transaction. Each row of the data will represent a specific buy/sell transaction done by the user.

*Table 3.1 Transaction Table Schema*

| **Column Name** | **Datatype** | **Explanation** |
|---|---|---|
| Ticker | TEXT | Unique symbol of the stock |

| | | |
|---|---|---|
| Date | DATE | Date of the data it is from |
| Type | TEXT | Type of transaction (bought/sold) |
| Quantity | INTEGER | Quantity bought / sold |
| Price | REAL | Price at the time transaction occurred |

# 3 Current Progress

## 3.1 Additional Progress

Aside from the above-mentioned progress, the testing of the compatibility of the libraries have been tested and there is no found errors.

## 3.2 Difficulties encountered and mitigation

Initially, the project plans to host the program on a server, allowing users to access it from various device they own, thus allowing the flexibility of using the portfolio managing service of the program from a device the user desire. However, this setup would require the user to have a stable internet connection throughout the usage of the service, which would greatly limit the amount of occasion that the user can manage their portfolio.

Therefore, it is instead decided to make the program downloadable. This would greatly reduce the number of occasions where an internet connection is needed. First, when the user downloads the program for the first time. Second, the fetching of the data. Although the once the data is fetched it will be saved into the database, if they user would like to update the data, a internet connection would still be needed. Thirdly, when the program requires an update.

## 3.3 Current Progress Summary

In summary, the initial research was started in August 2024. In September 2024, the project plan was submitted, and the project website was setup. In October, the research and design on API and GUI was done, yFinance, CustomTkinter and Tkinter was chosen. In November 2024, the research on libraries and design was done, pandas, matplotlib and SQLite3. In December 2024, the program development has been initiated, and the

compatibility of the different libraries has been tested to be without conflicts. In January 2025, the front-end design has started, thus. In Mid-January 2025, the First presentation was carried out and the interim report is to be submitted in Late-January 2025.

*Table 4.1 Project Schedule (1)*

| **Time Frame** | **Milestone** |
|---|---|
| AUG 2024 (Early) | Initial Research |
| SEP 2024 (Late) | Project Plan + Project Website setup |
| OCT 2024 | Research on API + GUI and design |
| NOV 2024 | Research on Libraries and design |
| DEC 2024 | Program Development |
| JAN 2025 (Early) | Front-end Design |
| JAN 2025 (Mid) | First Presentation |
| **JAN 2025 (Late)** | **Interim Report (Current stage)** |
| FEB 2025 (Mid) | Back-end Prototype |
| FEB 2025 (Late) | Front-end Prototype |
| MAR 2025 (Early) | Stock Split + Dividend Database Modification |
| MAR 2025 (Mid) | Short Selling Database Modification |
| MAR 2025 (Late) | Crypto-ecosystem Modification |
| APR 2025 (Early) | Fine-tuning |
| APR 2025 (Mid) | Final Report |
| APR 2025 (Late) | Final Presentation |
| APR 2025 (Late) | Project Exhibition |

# 4 Project Schedule

In the second semester, the followings improvements are planned to be tackled.

First, a system to handle stock splits and the resulting dividend is to be implemented, as a stock split would greatly affect the data in the database. For the Portfolio table, all the past holdings of the stock the user has in the portfolio will have to be modified. For the Transaction table, the current architecture will not support stock splits, as a stock split changes the holdings of the user without a buy/sell transaction. Therefore, both the Portfolio table and Transaction table is planned to be updated in the second semester to account for stock splits and its dividends.

Also, the system currently is still yet to support short selling, although short selling may not be a beginner method of trading stocks, it is still a fundamental part of trading stocks. Therefore, it is planned to add functions to accommodates for shorting stocks, namely modifying the Portfolio table and Transaction table. Furthermore, cryptocurrency is also planned to be accounted for in the system during the second semester, as the current system only allows the trading of stocks. This will allow the user to also manage their financial investment in the cryptocurrency ecosystem within this portfolio manager.

In conclusion, below is the project schedule for the second semester. Starting February 2025, it is planned to finish the back-end prototype, confirming and applying the said database structure. During Mid-February 2025 the back-end prototype is to be finished and Late-February 2025 the front-end prototype to be finished. This would result in a usable and testable prototype around Late-February 2025. Through March 2025, the above-mentioned improvements are planned to be implemented I the order of Stock Split and Dividend Database Modification, Short Selling Database Modification and Crypto-ecosystem

Modification. Finally, fine-tuning will be done in early April 2025, where the Final Report, Final Presentation and Project Exhibition will be carried out in the following months.

*Table 5.1 Project Schedule (2)*

| Time Frame | Milestone |
|---|---|
| AUG 2024 (Early) | Initial Research |
| SEP 2024 (Late) | Project Plan + Project Website setup |
| OCT 2024 | Research on API + GUI |
| NOV 2024 | Research on Libraries |
| DEC 2024 | Program Development |
| JAN 2025 (Early) | Front-end Design |
| JAN 2025 (Mid) | First Presentation |
| **JAN 2025 (Late)** | **Interim Report (Current stage)** |
| FEB 2025 (Mid) | Back-end Prototype |
| FEB 2025 (Late) | Front-end Prototype |
| MAR 2025 (Early) | Stock Split + Dividend Database Modification |
| MAR 2025 (Mid) | Short Selling Database Modification |
| MAR 2025 (Late) | Crypto-ecosystem Modification |
| APR 2025 (Early) | Fine-tuning |
| APR 2025 (Mid) | Final Report |
| APR 2025 (Late) | Final Presentation |
| APR 2025 (Late) | Project Exhibition |

# 5 Conclusion

In conclusion, given the positive bull views in the stock market and cryptocurrency market, not only it encourages current investors to re-evaluate their own portfolio, it also incentives others to become an investor and get into the financial market. However, most portfolio manager in the current solution are often built with a comprehensive set of functionalities, meaning new investors not only has to learn about the knowledge of the financial market, but also the portfolio manager app itself. Furthermore, most of the solutions are “active” in nature, with functionalities expecting the user to be using the app at live trading hours. This will not fit the use case for more casual investors or investors wanting to manage and adjust their portfolio in off-trading hours.

Therefore, this project aims to offer a portfolio management app that allows the user to build, check and track their portfolio in a central hub. While focusing on 3 focuses, an intuitive interface, a high flexibility design, a product comparison function. This will effectively reduce the learning curve of the portfolio, as it will be simplified to be intuitive and can be customized by the user to an extent. It also caters for portfolio management in non-live trading hours, allowing investors to manage and check on their portfolio as they would like to.

Currently, the Project Architecture has been designed to be separated into the Front-end and Back-end. The Programming language, API, Libraries and Modules has also been chosen to best fulfil the use case of this management app. The front-end user interface is designed to achieve the two focus of having an intuitive interface and being high flexibility in design. Where addition functions aside from the Project comparison functions are planned to further accommodates for non-active traders. The back-end development is designed to handle stock trading.

Although the hosting the portfolio management app on a server is proven to do more harm than good as it requires a stable connection, it is mitigated by making the program to be downloaded instead.

Furthermore, the immediate next steps are to finish both the back-end and front-end prototype, so as to have a usable prototype of the management app by the end of February 2025. Further improvements such as the addition of catering for stock-split and its dividends, short selling and including the crypto-ecosystem will be implemented throughout the rest of the second semester.

All in all, this project that acts as a portfolio management app which allows investors to build, check and track their portfolio at a central hub will effectively facilitate the organisation of financial information, aiming to improve the financial decisions of the user.

# 6 References

Academy, U. P. (2024). Why do data analysts use Python? | UCD Professional Academy. https://www.ucd.ie/professionalacademy/resources/why-do-data-analysts-use-python/#:~:text=Why%20Do%20Data%20Analysts%20Prefer,Python%20is%20its%20high%20readability.

GeeksforGeeks. (2024). *Python SQLite*. GeeksforGeeks. https://www.geeksforgeeks.org/python-sqlite/

ibkr. (2018). Potfolio Builder. (n.d.). https://www.ibkr.com.hk/en/software/tws/usersguidebook/mosaic/portfolioBuilder.htm

Matplotlib. (2024). *Matplotlib — Visualization with Python*. https://matplotlib.org/

pandas. (2024). *pandas - Python Data Analysis Library*. https://pandas.pydata.org/

Ranaroussi, R. (2025). *yfinance - Download market data from Yahoo! Finance API*. https://pypi.org/project/yfinance/

Schimansky, T. (2024). *CustomTkinter A modern and customizable python UI-library based on Tkinter*. https://customtkinter.tomschimansky.com/

sharesight. (2024). sharesight, The best portfolio and dividend tracker. *n.d.* https://www.sharesight.com/features/

Yahoo!Finance. (2024). Yahoo!Finance One place for your portfolios, metrics and more. *n.d.* https://finance.yahoo.com/portfolios/