Criterion E

**Meeting the success criteria**

(All tests below are done in MacBook Pro with macOS 10.14. Python IDE is Python IDLE version 3.6.5.)

(**Link: PyPI, GitHub, Documentation，呈现效果)**

**(test plan follows Criterion B)**

|  |  |  |
| --- | --- | --- |
| **Installation and setup** | Success criteria | **Installation using “pip install” is supported and no other steps are needed.** |
| My result | Succeeded. Evidence:  Execute “pip install JAQK” in terminal (command line in macOS):    Result: successful download    Import in Python IDE: successful |
| Jerry’s feedback | Download is easy and straightforward. |
| Success criteria | **Setup can convert setup data and initialize database.** |
| Result | Implemented. Evidence:  Original module after download. (Red blocks need to be converted in **Setup**)    Calling **setup()**, GUI pops out:  Manually choose a setup path:    Click submit: setup completed at the chosen path    Go to the chosen path, needed data are existing:    Go to each folder, data in sheets are correct (with correct formats):    Original data sheets with “.py” has been deleted: |
| Jerry’s feedback | **Setup** is easy to use and choosing database path is very helpful when using. |
| **Web Spyder** | Success criteria | **An Update function to update data of companies in database, but this should expand on original data rather deleting original ones.** |
| Result | Manually identify the date when database was last updated: |
| Jerry’s feedback | Successfully update data within stock list. |
| Success criteria | **A Get function that can automatically collect ALL the factors and ALL financial data of companies provided by Jerry from Yahoo Finance.** |
| Result | Implemented. |
| Jerry’s feedback | Data collection can be faster. |
| Success criteria | **Progress of functions above should be reported.** |
| Result | Implemented. Progress is reported. |
| Jerry’s feedback | Quite useful to know the progress. More information on progress and on other functions would be helpful. |
| **Calculations** | Success criteria | **Some basic calculations should be available: factors, trending, ranking, etc.** |
| Result | Implemented. |
| Jerry’s feedback | Rankings are very useful in analysis. More calculations will be better, maybe further implementation is needed. Perhaps a scoring system that can support inputting a scoring table will be great. |
| Success criteria | **Calculation don’t take longer than 1 second.** |
| Result | Reached. |
| Jerry’s feedback | It will be helpful to make it faster. More options on input parameters will be helpful. |
| **Operations** | Success criteria | **Open can handle possible errors and open sheets in database.** |
| Result | Implemented. |
| Jerry’s feedback | Useful. |
| Success criteria | **Save can save files to database as well as an chosen directory path.** |
| Result | Implemented. |
| Jerry’s feedback | Saving to a chosen path is very useful. It will be better if it can save multiple sheets at a time. |
| Success criteria | **Translate can translate description of companies into Chinese.** |
| Result | Implemented. |
| Jerry’s feedback | Translation is well done. Maybe it can support more translation other than companies descriptions, such as factors, calculations, and so on. |
| Success criteria | **Folder operations can create and check existence of sheets.** |
| Result | Implemented. |
| Jerry’s feedback | Implemented. |
| Success criteria | **Path control can locate factors in database and can provide path of each module, sub module, and function.** |
| Result | Implemented. Better import methods may need to consider to avoid cyclical import (A import B then B import A). |
| Jerry’s feedback | Well done. |
| Success criteria | **Tools and reports can report database status (file and company counts).** |
| Result | Implemented. |
| Jerry’s feedback | Reports are useful. More reports will be helpful. |
| Success criteria | **Format control can clean characters in numbers (comma, money sign, etc.) and format output of Get data.** |
| Result | Implemented. |
| Jerry’s feedback | Well done. |
| Success criteria | **Get data provides functions to retrieve data in database and give formatted results. This include getting data sheets, factors, and so on.** |
| Result | Implemented. |
| Jerry’s feedback | The most helpful feature. Perhaps more output format will help, maybe including some visualization on output. |

**Client’s feedback (details in Appendix C)**

My client is mostly satisfied with the product and he said he can use it without any difficulties. He said he is implementing my product in his quantitative analysis algorithms, and all important functions are well implemented. He pointed out some well-designed features:

* GUI components successfully speed up and simply the usage of the product;
* **Operations** provide useful connections that he can directly use these operations in his quantitative analysis codes.
* **Get data** provides effective tool to retrieve formatted data from the database;

However, he did point out some flaws:

|  |  |
| --- | --- |
| **Those I can resolve** | The calculations module should contain more algorithms than just ranking. Some more complex algorithms include single factor efficiency check, compound factors return check, etc., should be included; |
| Visualization on outputs are not supported. Some graphs, such as candle plot, should be supported. |
| **Those I can’t resolve now** | The Web Spyder is still too slow (takes almost 10 minutes to finish collecting), and it should be shorter than 1 minute; |
| The calculations take 1 second on average, but it needs to be faster. |

**Recommendation of future improvements**

**Those I can resolve**

Calculations module will contain more algorithms. It may require more efficient database structure and design, so some optimization will be necessary. Calculations will be implemented following the calculation methods.

Visualization is very possible with Python’s powerful modules, such as matplotlib, seaborn, plotly, etc. However, it’s hard to determine chart type and X&Y axis without knowing which trends or input data are needed. I will ask a list of charts from Jerry and do researches on financial charts and implement them accordingly.

**Those I can’t resolve now:**

The Web Spyder can be faster. From calculations, if the product needs to request all data from Yahoo Finance in 1 minute, the size of all requested HTML texts is around 35M / s, which is highly unlikely (below referred to as 35M/s problem). Alternatives include:

* First, since my client is in China and Yahoo Finance’s servers are outside China, deploying a foreign VPN or a proxy server can decrease time wasted in redirections among servers; but using foreign proxies and VPNs introduce extra expenditures and security concerns.
* Second, I can collect and post all data in a server where my client can download at once without frequently requesting from Yahoo Finance; but I’m not familiar with techniques of building a server and data transmission protocools.
* Third, more power frameworks (e.g. Scrapy) may speed up Web Spyder with their internal optimization; but whether such professional frameworks are useful remain unevaluated and unchecked.
* Using multiprocessing is not a viable solution because this doesn’t resolve 35M/s problems, and my Web Spyder already very efficiently fill the internet bandwidth, over 3M/s. Multiprocessing may introduce extra use of computer resources but being not helpful in speed up Web Spyder.

Calculations can be faster in two ways. First, use more complicated algorithms (linear algebra, discrete math, etc.), but I have limited knowledge for them. Second, use more alternative databases (MySQL etc.) that have faster searching, cross sheet operations, and relational operations; however, such database will introduce more installation work for client, and I have no knowledge on these databases.

During my developing, I found that functionalities are not divided enough to conquer. I plan to further defining and designing functionalities needed and merge repetitive functionalities. Another future prospect is implementing codes with concepts of object oriented programming. This gives more extensibilities on functionalities and reduce future maintenance works.