

Rise & Shine

project documentation

Prepared, designed and programmed
by Katarina Miksche

as the final project at
CFGdegree SW Development summer 2022

Table of content

Main objectives	2
Functionality	3
GUI	3
Start screen	3
First screen after start and successful load of data	4
Portfolios - top view	4
Portfolio overview	5
Portfolio performance with graph	5
Search for ticker tool	6
Ticker performance with graph	6
Buy / Sell tool	7
Start / Close portfolio	7
Wallet - Deposit / Withdrawal, see records	8
API - button will open API file and close GUI	9
API	10
Home - http://127.0.0.1:5000/	10
/portfolios - for list of portfolios, their names and current values	10
/portfolio_detail/<int:id> - for detail of portfolio	11
/stock/<ticker>/<int:interval> - for performance data about stock	12
/stock_current/<ticker> - for actual price of ticker	12
Technical specifications, used technologies & architecture	13
Application system requirements	13
Installation	13
Other used technologies during project	13
Application architecture	14
SQL architecture	15
GUI design	15
Example of function Show portfolio:	16
Development process	17
Sprint 1	17
Sprint 2	18
Sprint 3	19
Sprint 4	20
Final words	20

Main objectives

Rise & Shine is a stock market application. It allows users to track and trade stocks, manage multiple portfolios and see the gains in a friendly graphic interface. Results are available online through API as well.

Full functionality available through the graphic interface includes an overview of portfolios including their current values, more detailed view of each portfolio and its historic performance including graph, search tool for tickers and stock performance view, buy/sell functionality, monetary operations in wallet, start of new portfolio and option to close it fast as well.

Limited functionality available online via own API includes an overview of portfolios including their current values, more detailed view of each portfolio and its historic performance, stock performance view and most recent stock price.

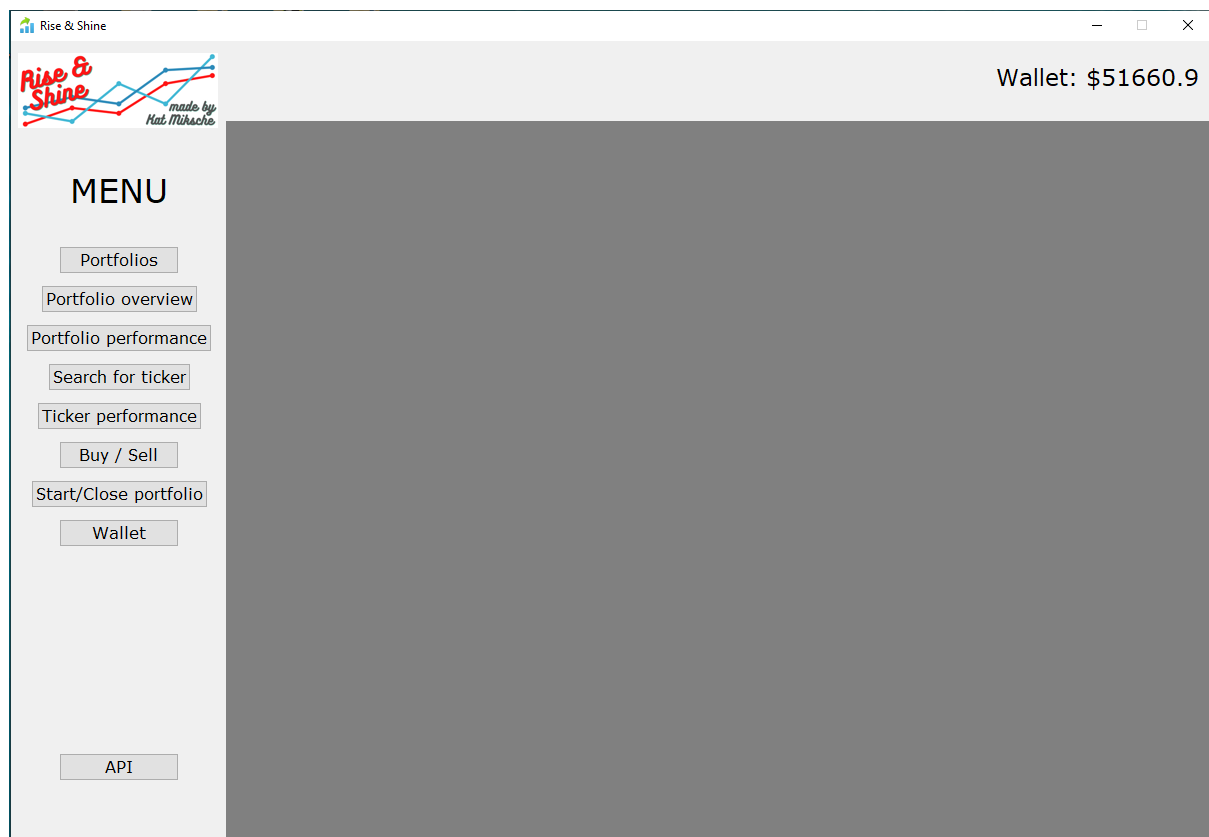
Functionality

GUI

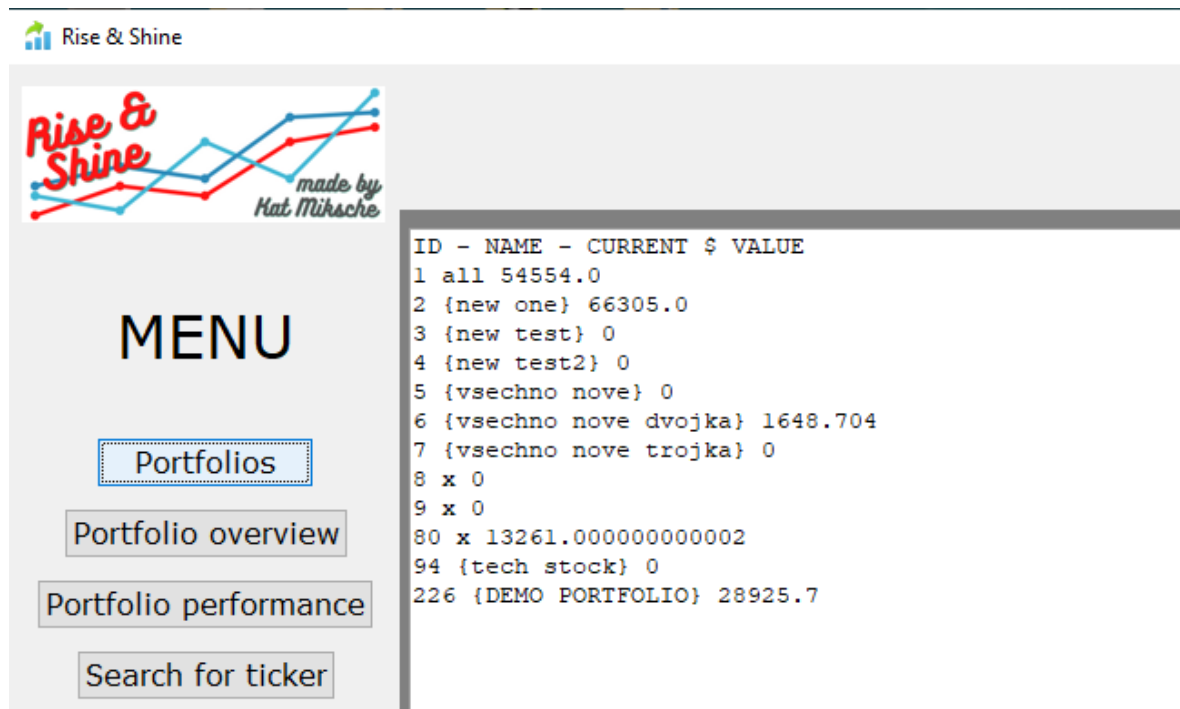
Start screen



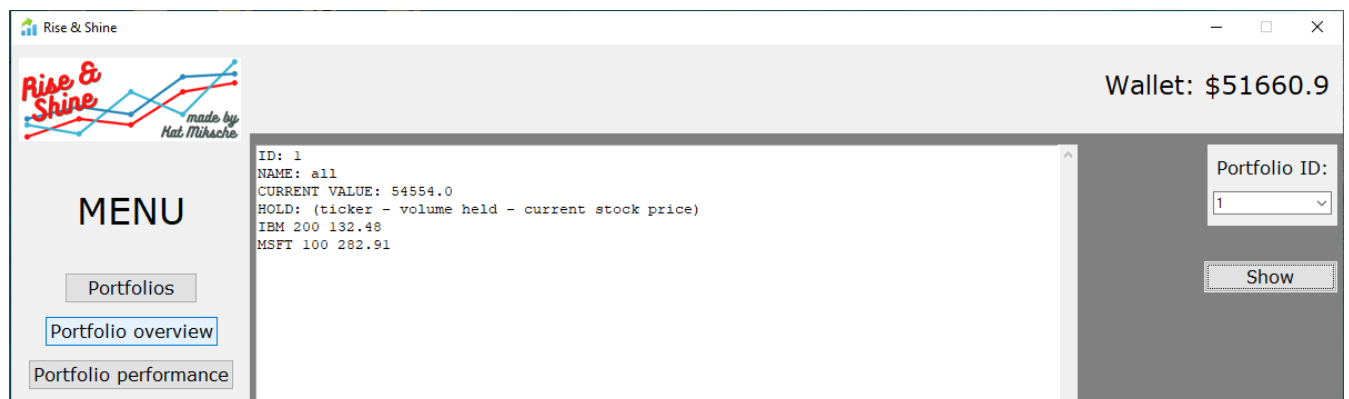
First screen after start and successful load of data



Portfolios - top view

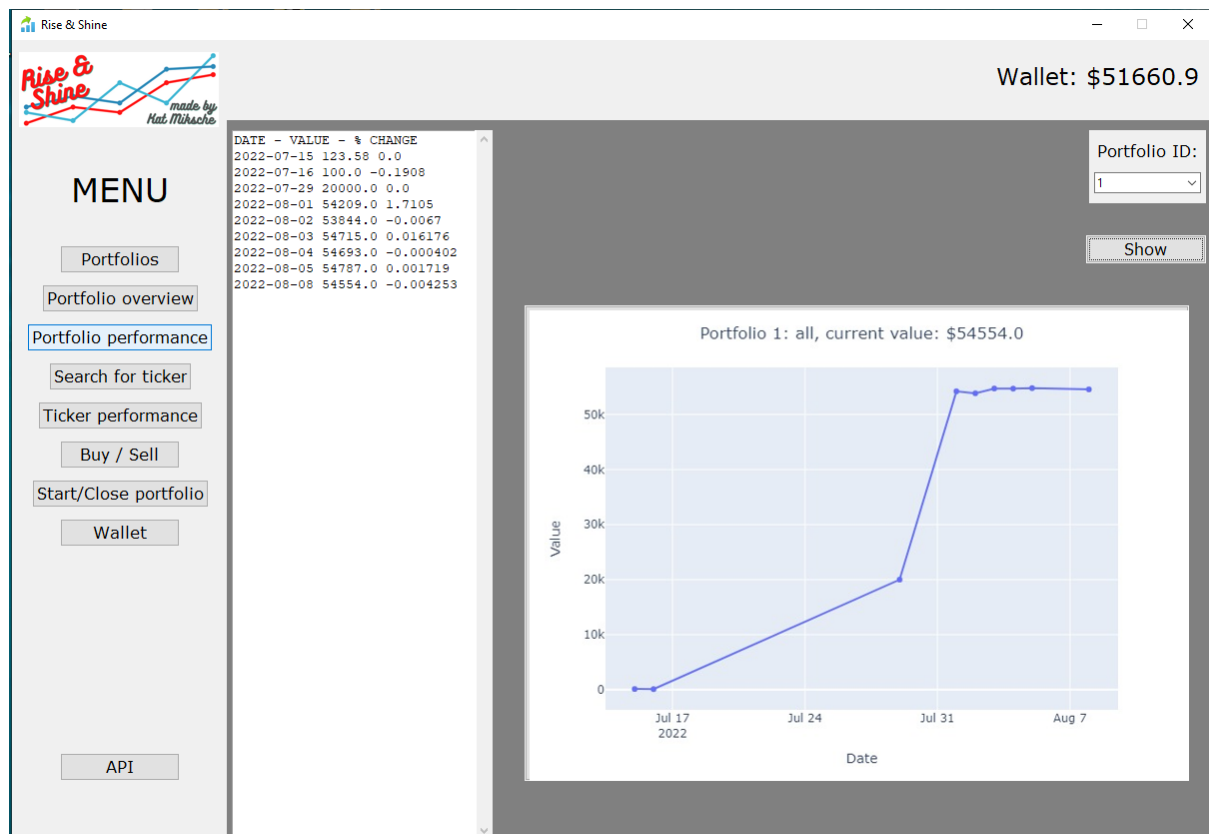


Portfolio overview

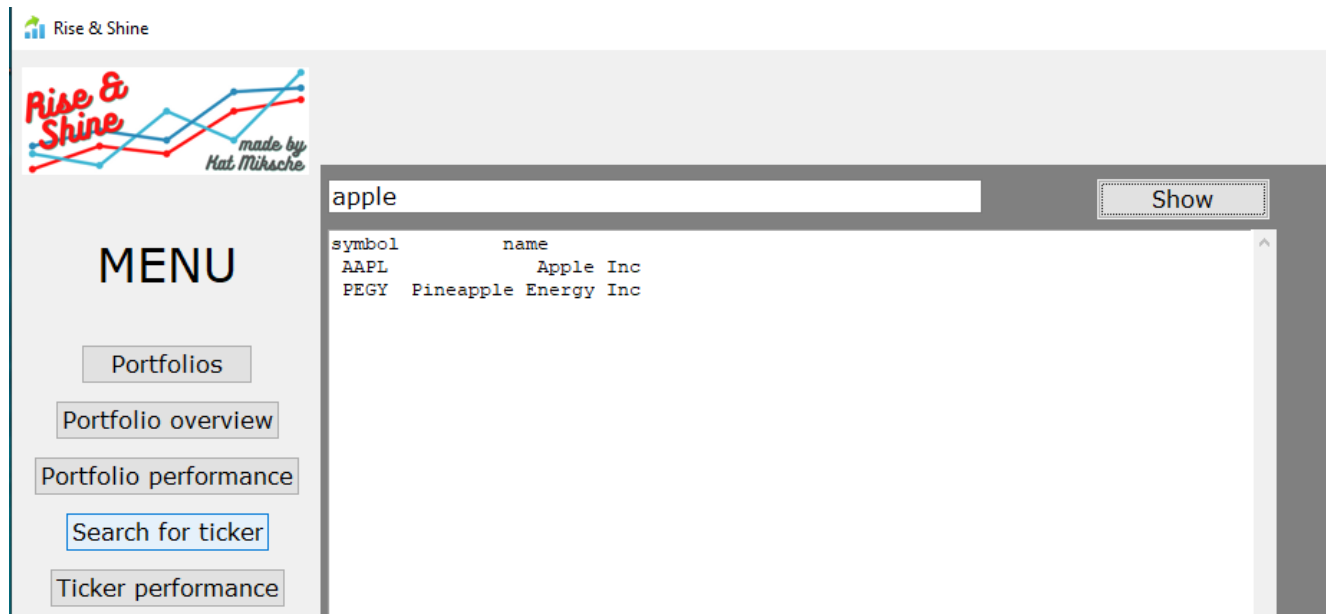


Portfolio performance with graph

(graph is clickable, opens as an active object in a browser)



Search for ticker tool



Rise & Shine

MENU

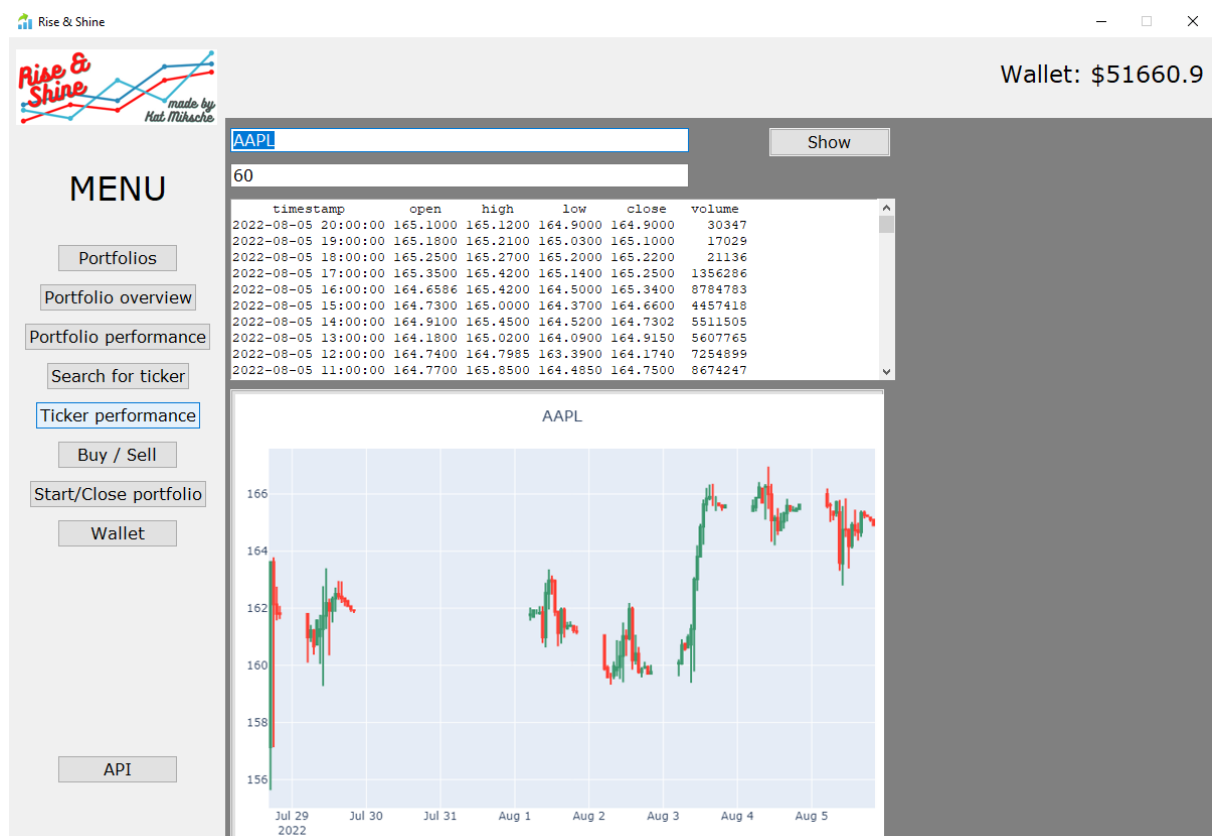
- Portfolios
- Portfolio overview
- Portfolio performance
- Search for ticker**
- Ticker performance

apple Show

symbol	name
AAPL	Apple Inc
PEGY	Pineapple Energy Inc

Ticker performance with graph

(graph is clickable, opens as an active object in a browser)



Rise & Shine

MENU

- Portfolios
- Portfolio overview
- Portfolio performance
- Search for ticker
- Ticker performance**
- Buy / Sell
- Start/Close portfolio
- Wallet
- API

Wallet: \$51660.9

AAPL Show


60

timestamp	open	high	low	close	volume
2022-08-05 20:00:00	165.1000	165.1200	164.9000	164.9000	30347
2022-08-05 19:00:00	165.1800	165.2100	165.0300	165.1000	17029
2022-08-05 18:00:00	165.2500	165.2700	165.2000	165.2200	21136
2022-08-05 17:00:00	165.3500	165.4200	165.1400	165.2500	1356286
2022-08-05 16:00:00	164.6586	165.4200	164.5000	165.3400	8784783
2022-08-05 15:00:00	164.7300	165.0000	164.3700	164.6600	4457418
2022-08-05 14:00:00	164.9100	165.4500	164.5200	164.7302	5511505
2022-08-05 13:00:00	164.1800	165.0200	164.0900	164.9150	5607765
2022-08-05 12:00:00	164.7400	164.7985	163.3900	164.1740	7254899
2022-08-05 11:00:00	164.7700	165.8500	164.4850	164.7500	8674247

AAPL

Jul 29 2022 Jul 30 Jul 31 Aug 1 Aug 2 Aug 3 Aug 4 Aug 5

Buy / Sell tool



MENU

- Portfolios
- Portfolio overview
- Portfolio performance
- Search for ticker
- Ticker performance
- Buy / Sell**
- Start/Close portfolio
- Wallet

Portfolio ID:


Pick portfolio ID ▾

Insert required ticker

Insert required volume

Buy Sell

Start / Close portfolio



MENU

- Portfolios
- Portfolio overview
- Portfolio performance
- Search for ticker
- Ticker performance
- Buy / Sell
- Start/Close portfolio**
- Wallet

Insert required name of the new portfolio


Start

Portfolio ID:

Pick portfolio ID ▾

Close

Wallet - Deposit / Withdrawal, see records

 Rise & Shine



MENU

Portfolios

Portfolio overview

Portfolio performance

Search for ticker

Ticker performance

Buy / Sell

Start/Close portfolio

Wallet

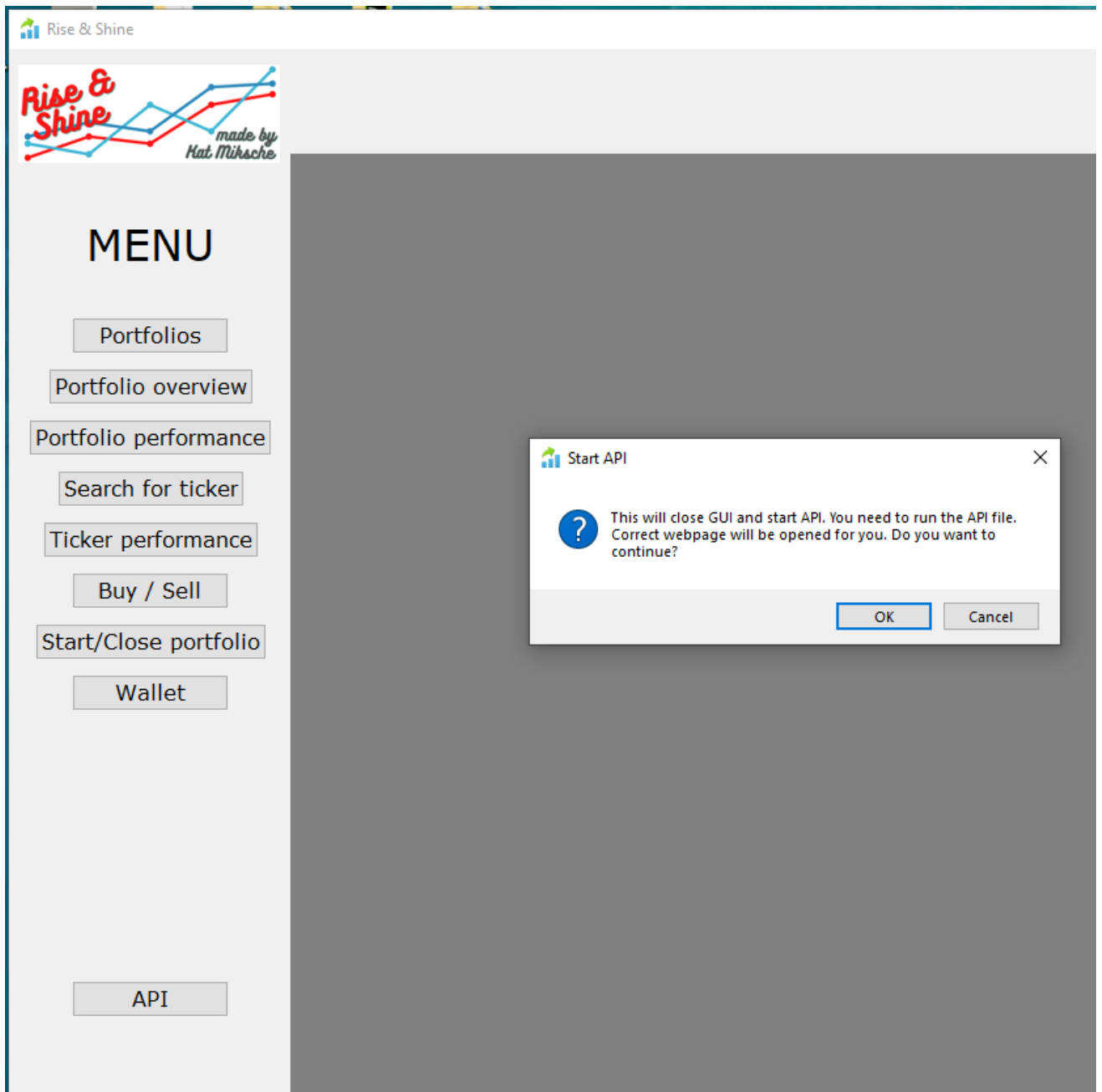
Insert required amount

Deposit

Withdraw

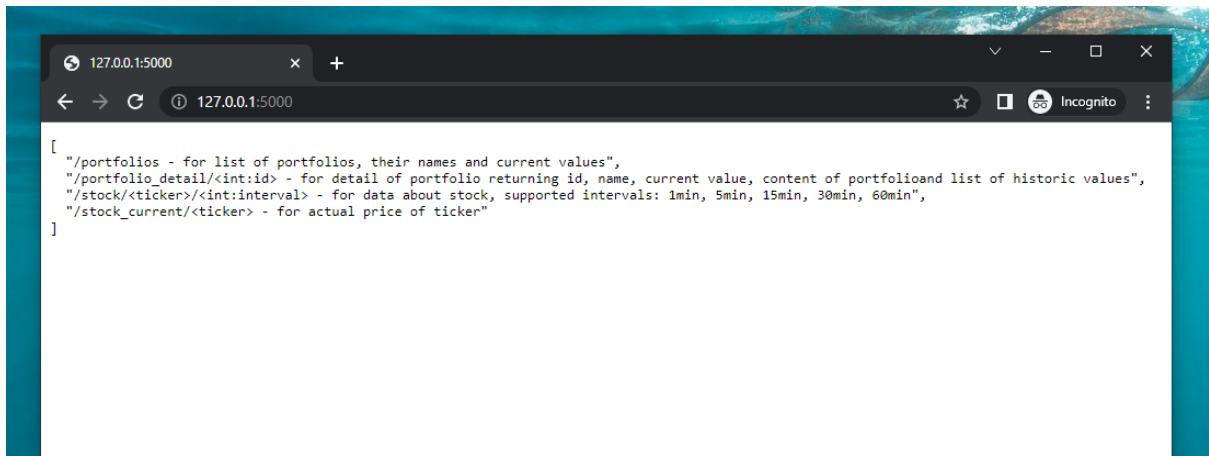
Value	Change	Timestamp	Comment
51660.90	0.00	2022-08-08 21:35:20	reset
20000.00	10000.00	2022-08-08 21:34:24	DEMO MONEY
10000.00	10000.00	2022-08-08 17:22:51	DEMO MONEY
51660.90	-1600.10	2022-08-03 17:24:21	Purchase of 10 stocks of AAPL
53261.00	0.00	2022-08-03 17:18:19	Purchase of 10 stocks of AAPL
53261.00	0.00	2022-08-03 17:15:05	Purchase of 10 stocks of AAPL
53261.00	20188.75	2022-08-02 18:17:32	Sale of 125 stocks of AAPL
33072.25	13204.00	2022-08-02 18:17:31	Sale of 100 stocks of IBM
19868.25	-4037.75	2022-08-02 18:00:13	Purchase of 25 stocks of AAPL
23906.00	-4037.75	2022-08-02 17:59:21	Purchase of 25 stocks of AAPL
27943.75	-4037.75	2022-08-02 17:59:01	Purchase of 25 stocks of AAPL
31981.50	10000.00	2022-08-02 17:59:00	personal investment
21981.50	10000.00	2022-08-02 17:58:46	personal investment
11981.50	-4037.75	2022-08-02 17:58:34	Purchase of 25 stocks of AAPL
16019.25	10000.00	2022-08-02 17:58:33	personal investment
6019.25	-4037.75	2022-08-02 17:58:24	Purchase of 25 stocks of AAPL
10057.00	10000.00	2022-08-02 17:58:23	personal investment
57.00	-4037.75	2022-08-02 17:56:15	Purchase of 25 stocks of AAPL

API - button will open API file and close GUI

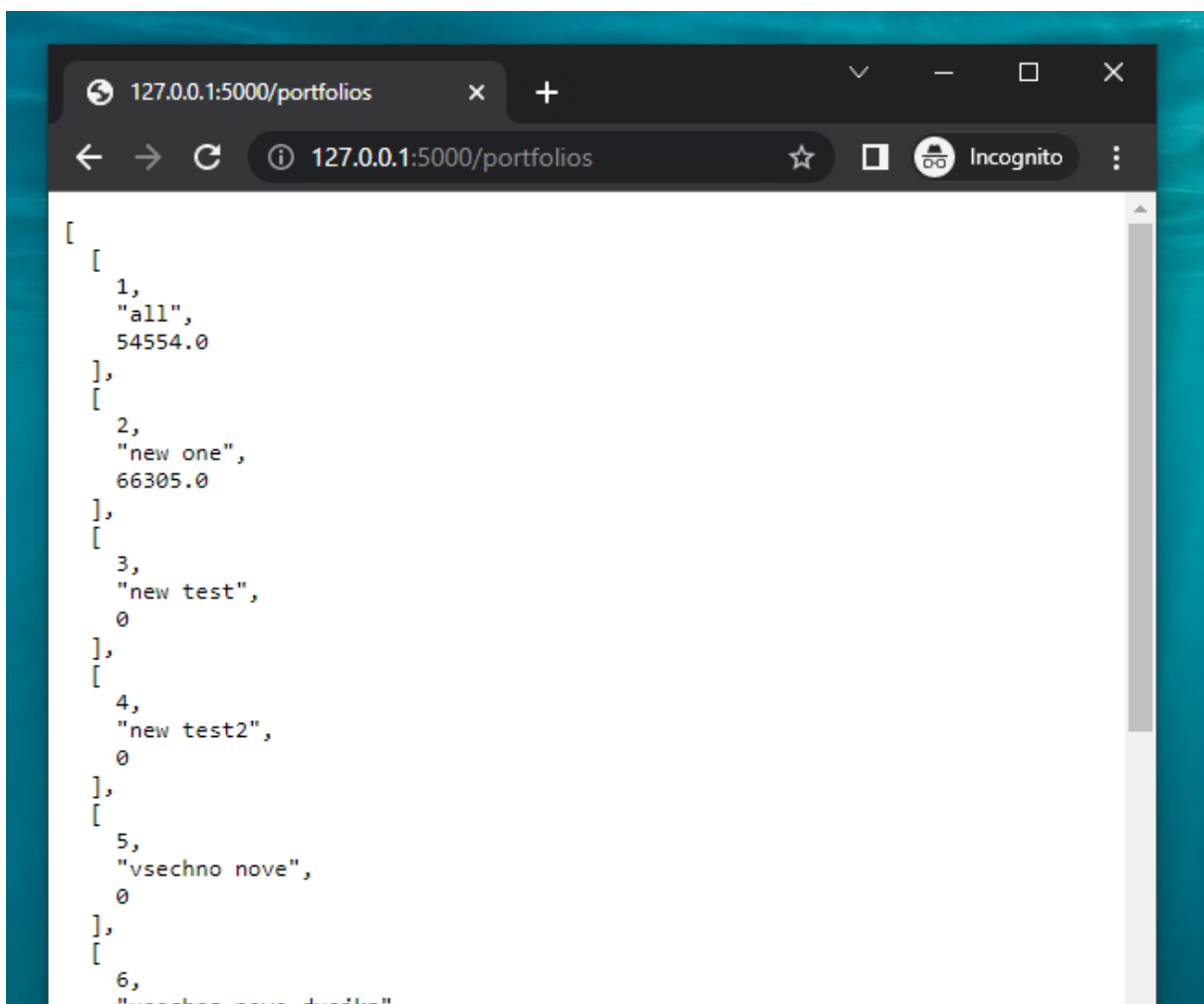


API

Home - <http://127.0.0.1:5000/>

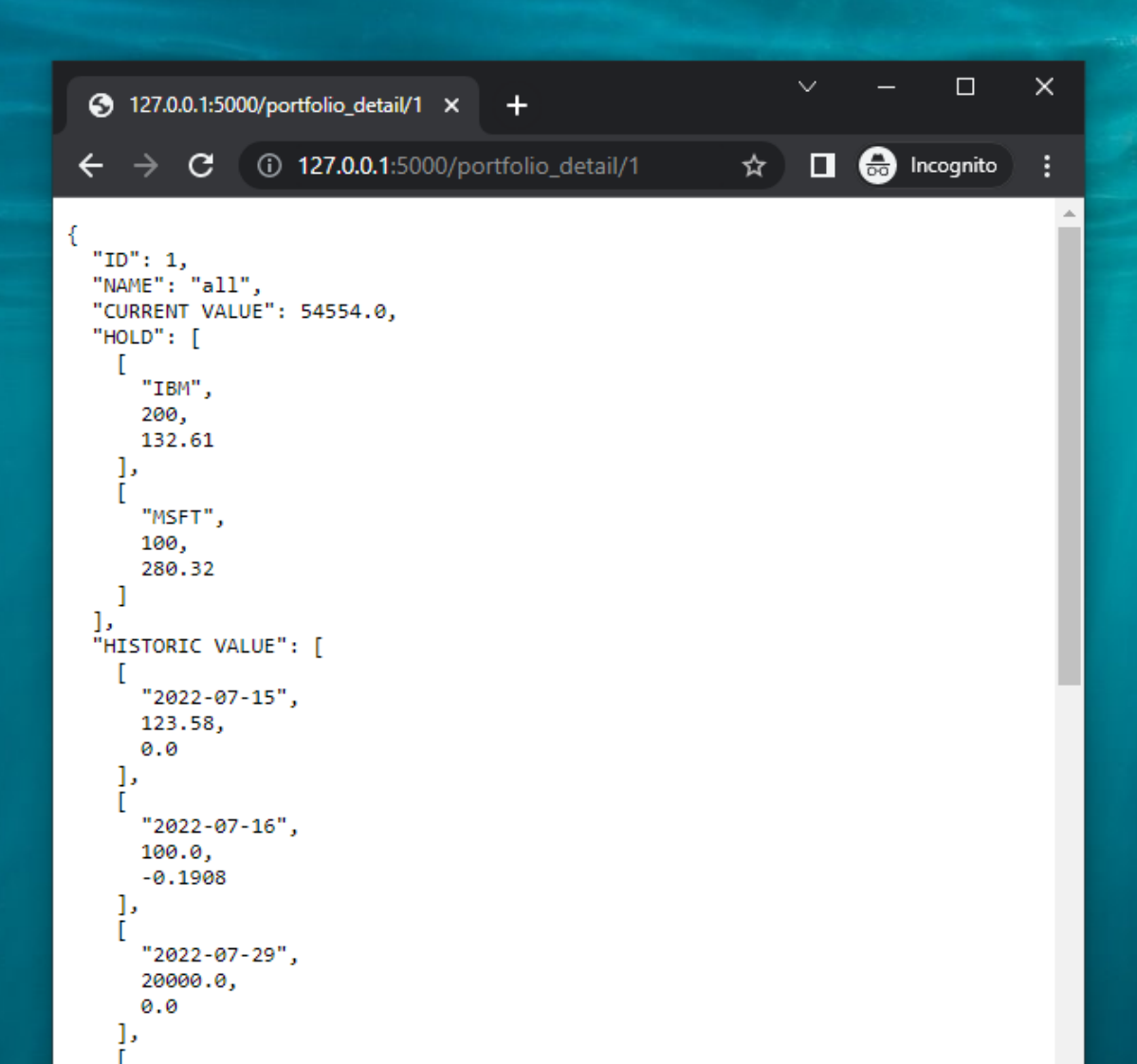


[/portfolios](#) - for list of portfolios, their names and current values



/portfolio_detail/<int:id> - for detail of portfolio

returning id, name, current value, content of portfolio and list of historic values



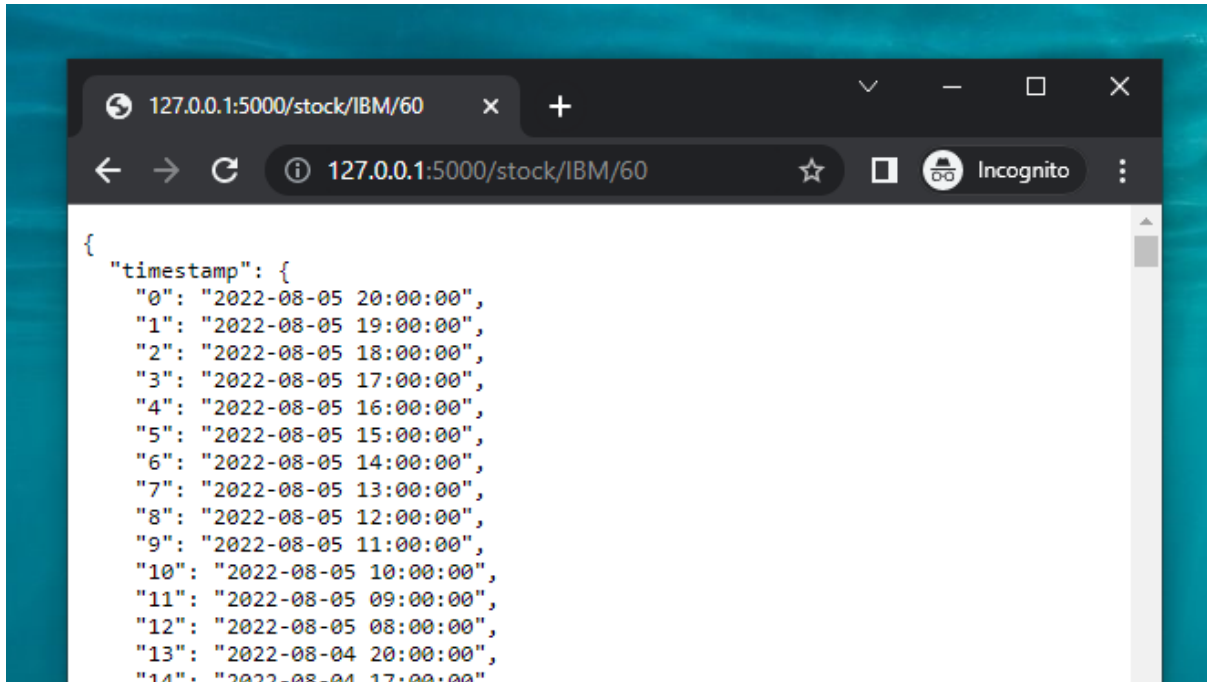
The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/portfolio_detail/1'. The browser is in Incognito mode. The main content area displays a JSON object representing a portfolio detail. The JSON includes fields for ID, Name, Current Value, Holdings (a list of stock symbols, quantities, and prices), and a list of Historic Values (dates and corresponding values).

```
{
  "ID": 1,
  "NAME": "all",
  "CURRENT VALUE": 54554.0,
  "HOLD": [
    [
      "IBM",
      200,
      132.61
    ],
    [
      "MSFT",
      100,
      280.32
    ]
  ],
  "HISTORIC VALUE": [
    [
      "2022-07-15",
      123.58,
      0.0
    ],
    [
      "2022-07-16",
      100.0,
      -0.1908
    ],
    [
      "2022-07-29",
      20000.0,
      0.0
    ],
    [

```

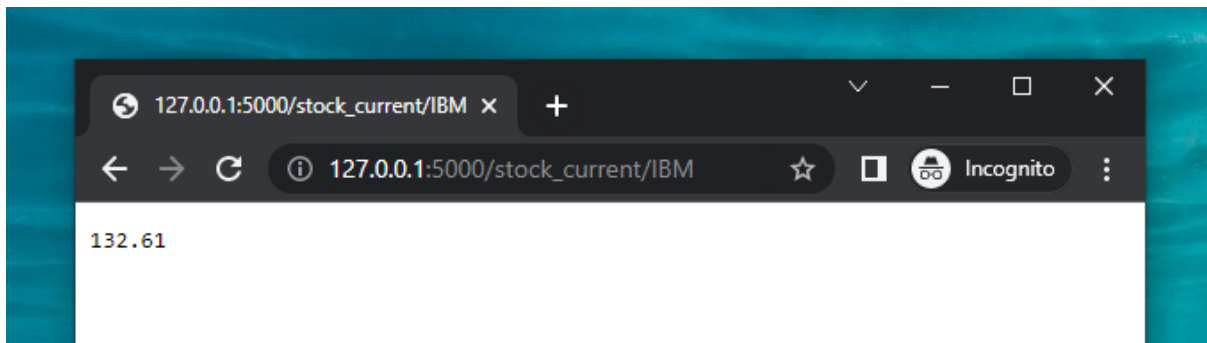
/stock/<ticker>/<int:interval> - for performance data about stock

supported intervals: 1min, 5min, 15min, 30min, 60min



```
{
  "timestamp": {
    "0": "2022-08-05 20:00:00",
    "1": "2022-08-05 19:00:00",
    "2": "2022-08-05 18:00:00",
    "3": "2022-08-05 17:00:00",
    "4": "2022-08-05 16:00:00",
    "5": "2022-08-05 15:00:00",
    "6": "2022-08-05 14:00:00",
    "7": "2022-08-05 13:00:00",
    "8": "2022-08-05 12:00:00",
    "9": "2022-08-05 11:00:00",
    "10": "2022-08-05 10:00:00",
    "11": "2022-08-05 09:00:00",
    "12": "2022-08-05 08:00:00",
    "13": "2022-08-04 20:00:00",
    "14": "2022-08-04 17:00:00"
  }
}
```

/stock_current/<ticker> - for actual price of ticker



```
132.61
```

Technical specifications, used technologies & architecture

Application system requirements

- Browser (ideally Chrome)
- MySQL (any version)
- Python 3 including modules mysql, requests, io, os, csv, itertools, plotly, datetime, pandas, numpy, webbrowser, PIL, tkinter and flask (for API - optional)

Installation

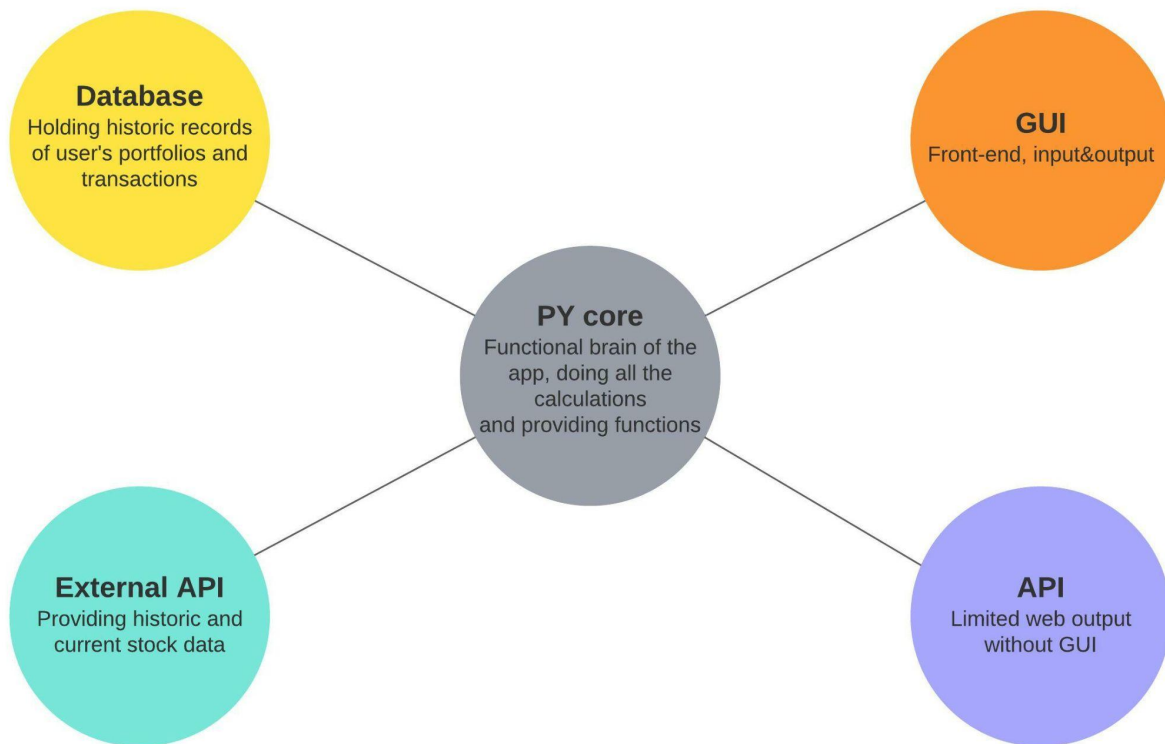
1. rise-shine-SQL-setup.sql - required
2. DEMO data.sql - optional
3. change sql connection config details in RSSQL.py
4. run tests in Test folder - optional
5. GUI.py - run the main file
6. API.py - optional for API use

Other used technologies during project

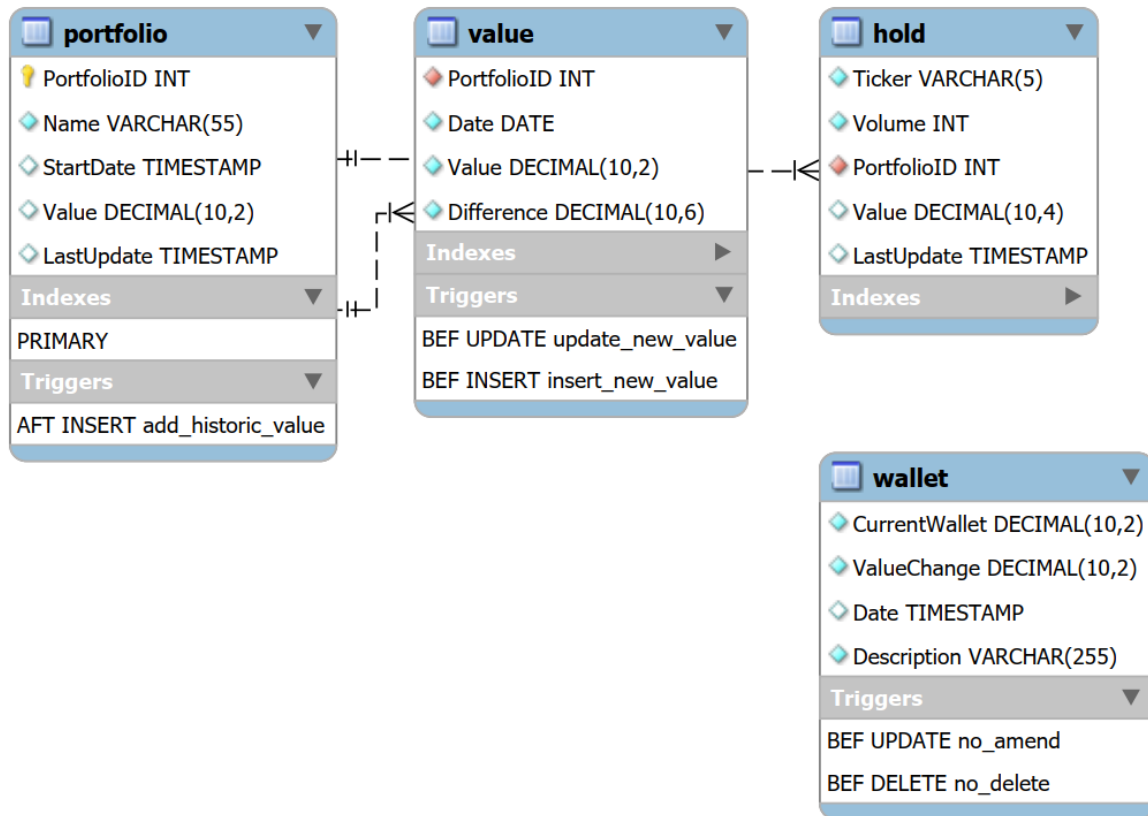
Application was developed in following environment:

- Asus G752VS, Intel i7--6820HK 2.7GHz, 32GB RAM
- Windows 10 Pro 64-bit
- MySQL Community Server - GPL 8.0.27 64-bit
- PyCharm 2022.1.1 (Community Edition) 64-bit
- Python 3.10.4 64-bit
- Google Chrome 104.0 64-bit
- ALPHA VANTAGE API - <https://www.alphavantage.co>

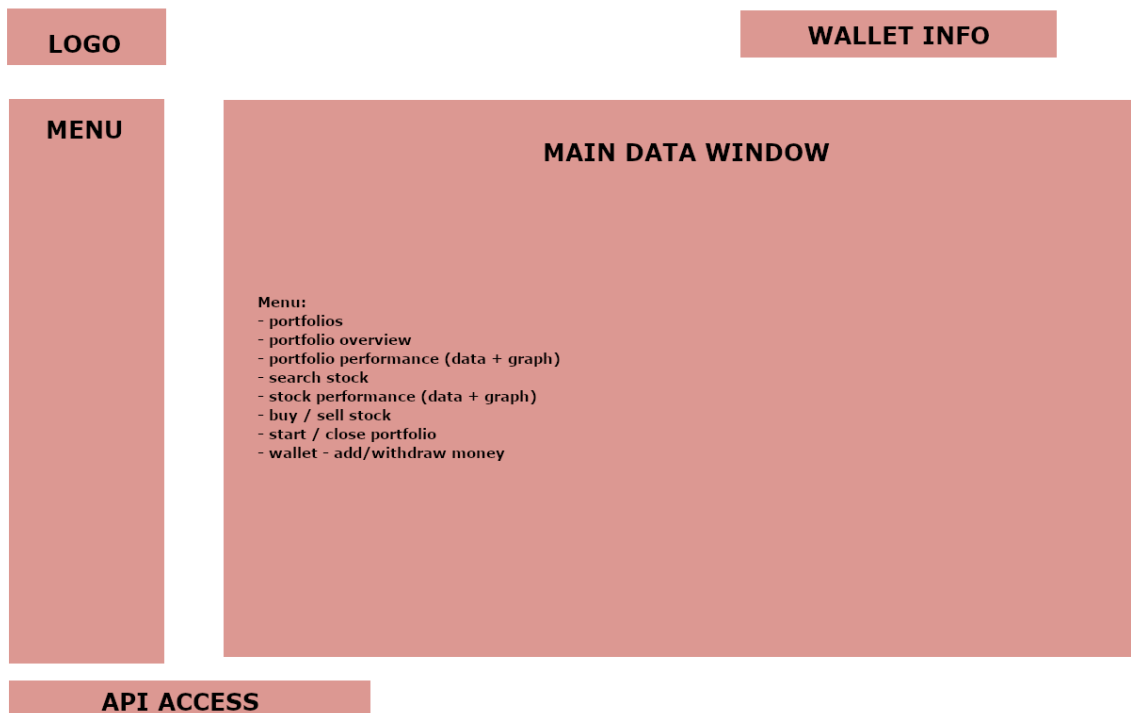
Application architecture



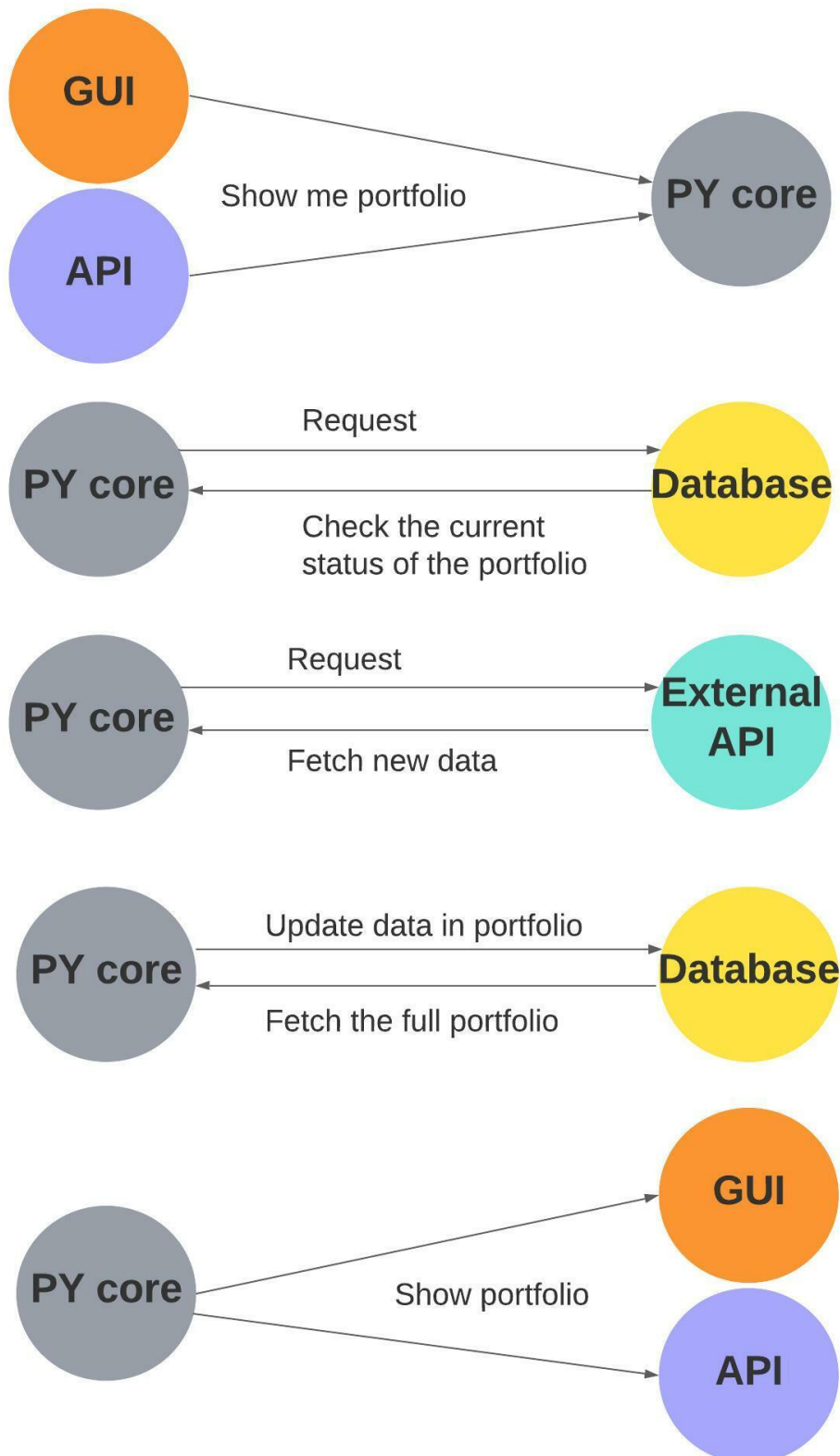
SQL architecture



GUI design



Example of function Show portfolio:



Development process

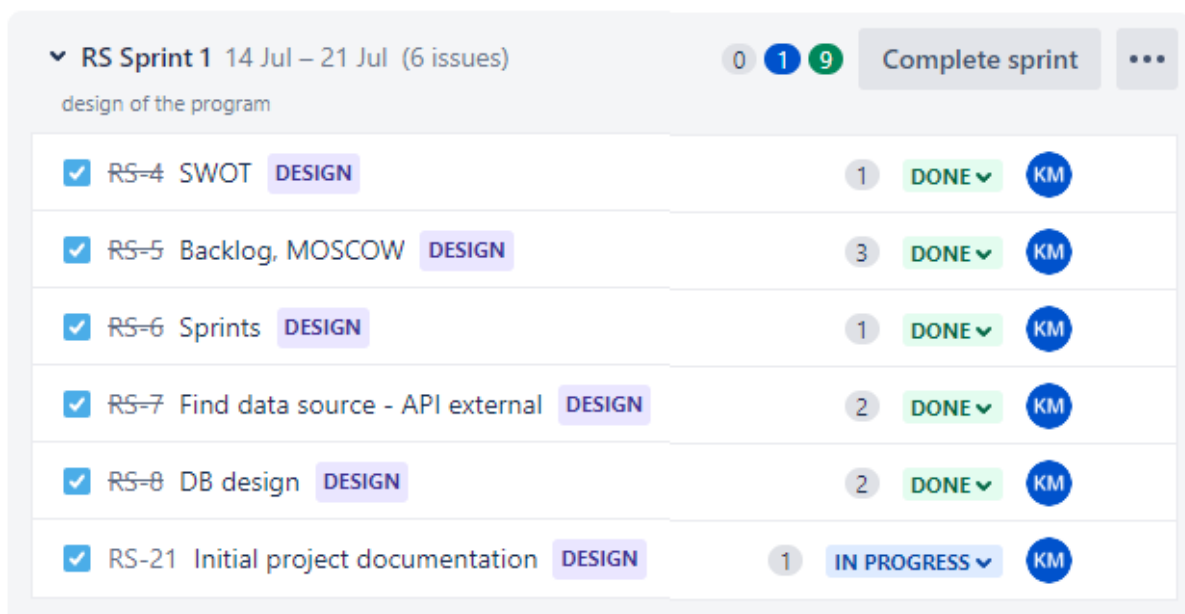
During project many other applications were used:

GitHub, Jira, Confluence, Gimp, Canva, Lucidspark, Google Docs, Slack, Zoom

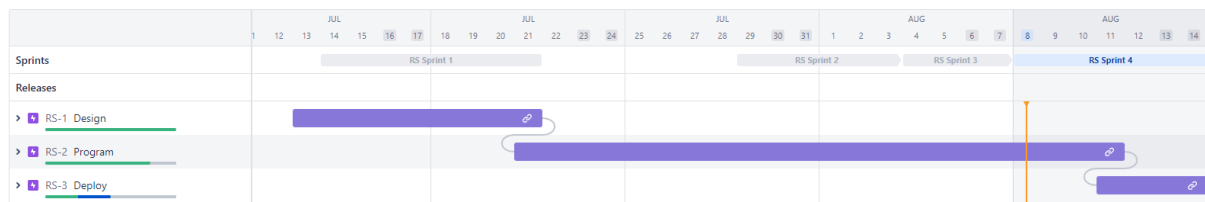
Planning of development was made in Jira.

Sprint 1

Planning and designing the application without any programming using agile methodology. SWOT analysis completed as first, then stories were added to backlog, evaluated by MOSCOW and assigned story points estimate. Further sprints were created, database architecture designed (not programmed) and initial documentation created.



Roadmap created:



Sprint 2

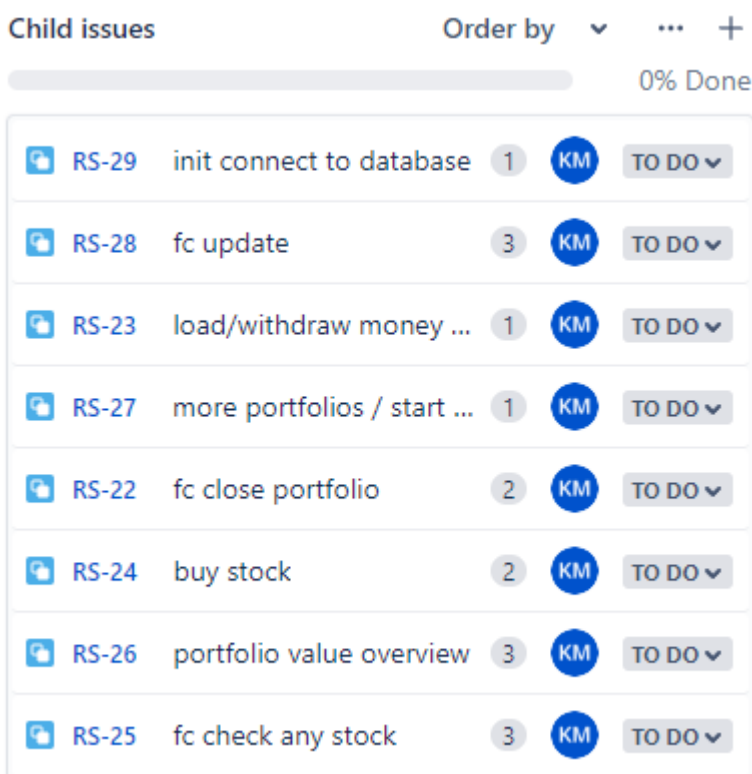
Main programming part without GUI. First creating a database including triggers and functions. Then move to Python - creating functions for external API (exAPI.py) and classes including methods for wallet (wallet.py) and portfolio (portfolio.py). File RSSQL.py holds connection functions to database and configuration details. File Main.py administers a dictionary of portfolios and starts of new portfolios.

The most complicated function was the hidden one - update. It is updating historic values of the portfolio based on data for each held ticker from external API, so merging SQL and external API data through calculations.



▼ RS Sprint 2 21 Jul – 8 Aug (3 issues)		3	0	0	Start sprint	...
<input checked="" type="checkbox"/>	RS-9 DB PROGRAM		TO DO ▼	KM		
<input checked="" type="checkbox"/>	RS-10 Core PROGRAM		TO DO ▼	KM		
<input checked="" type="checkbox"/>	RS-11 API PROGRAM	3	TO DO ▼	KM		

Children issues of Core:



Child issues		Order by ▼	...	+
0% Done				
<input checked="" type="checkbox"/>	RS-29 init connect to database	1	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-28 fc update	3	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-23 load/withdraw money ...	1	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-27 more portfolios / start ...	1	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-22 fc close portfolio	2	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-24 buy stock	2	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-26 portfolio value overview	3	KM	TO DO ▼
<input checked="" type="checkbox"/>	RS-25 fc check any stock	3	KM	TO DO ▼

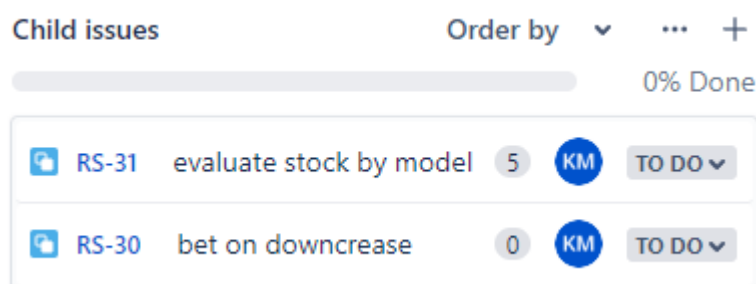
Sprint 3

Graphic user interface is made using tkinter. With almost 400 lines of code it proved to be a much bigger piece of work than originally planned. Apart from one option in the menu all other options require interaction with the user and therefore following functions called by buttons. Great focus was given to safeguard the program against any user error.

The delay led to reevaluation of sprint and moving extensions, which had MOSCOW value Could and Won't back to backlog. Hopefully in future new version of Rise & Shine will appear, where the functionality of stock being assessed by machine learning model will be available.



Children issues of Extension:

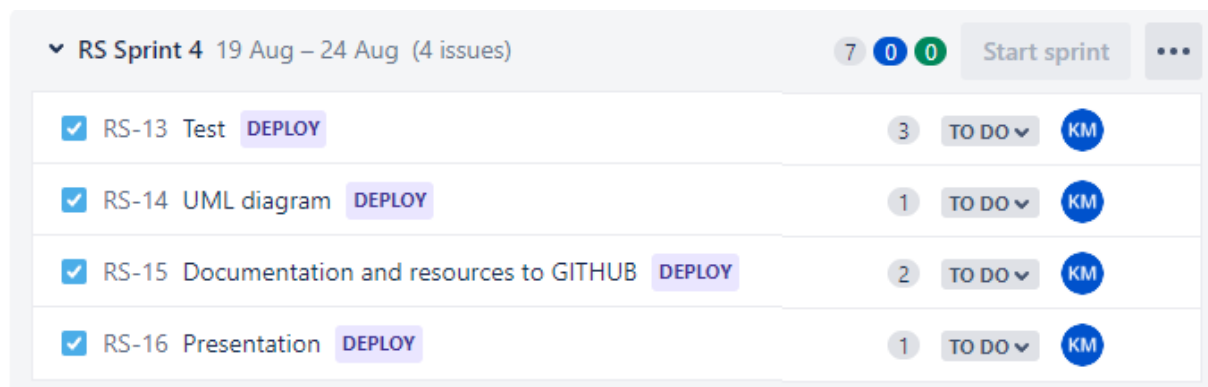


Sprint 4

Testing was done alongside the development, but after all parts finished tests were run again. Some issues occurred as small changes were made throughout the older code in terms of shaping it to the needs of GUI.

New SQL demo data was created as well.

SQL diagram was created and two UML diagrams from initial documentation were used in this document. As well, this document will be used at the final presentation of the project, together with the live presentation of the application.



▼ RS Sprint 4 19 Aug – 24 Aug (4 issues)			
<input checked="" type="checkbox"/>	RS-13 Test	DEPLOY	3 TO DO KM
<input checked="" type="checkbox"/>	RS-14 UML diagram	DEPLOY	1 TO DO KM
<input checked="" type="checkbox"/>	RS-15 Documentation and resources to GITHUB	DEPLOY	2 TO DO KM
<input checked="" type="checkbox"/>	RS-16 Presentation	DEPLOY	1 TO DO KM

Final words

This project took around 100 hours of screen time, where at least half of the time was used for learning and researching the solution. Test and fail was the most used method, there was no time for posting questions to online forums and waiting for any answer.

Where I started with basic knowledge of Python, minimum knowledge of optional modules and zero knowledge of tkinter, I am very proud of the result. This project took me completely from procedural programming (I was very comfortable with :-)) to object oriented programming. I thank Code First Girls for the challenge, it was a tough one, therefore great!

Plan for the future is to come with a new release which will include data science tools for evaluating stocks. Then it can have that nice dodgy label "We don't hold any responsibility upon results you receive" :-)