Stock Investment Dashboard

Group 4 - Catherine Ting, James Leslie Akerman, Hayley Lim, Vignesh Sundaramoorthi

High Level Design

Introduction

The web-interactive dashboard is designed to help a fund manager to easily visualise an interactive report/dashboard for them to review the progress of the clients' share portfolios.

Goals:

- The aim is to assist in managing client's portfolios and help them to make better decisions in a timely manner.
- Fund manager is able to review each client's portfolio and identify the short term trends in individual stocks prices.
- It also show a summary profile of the share purchased by clients.

DEMO

Our company
Receipts
Sales
Orders
100
90
80
70
60
50
40
30
20
10
0
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
Business items
Sales
Orders
100
90
80
70
60
50
40
30
20
10
0
JAN
FEB
MAR
APR
MAY
AUG
SEP
OCT
NOV
DEC

Coding Approach

Data Sources:

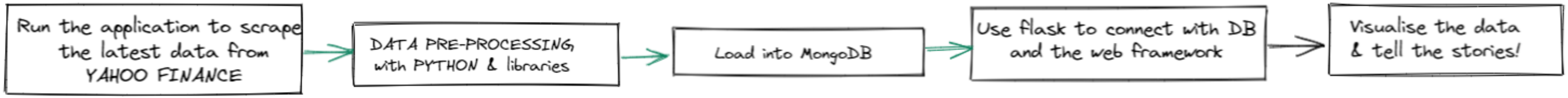
- Stock Data: https://au.finance.yahoo.com/.
- Portfolio Data: Supplied by Group 4 & Yahoo Finance.

Web Scraping: Python for data wrangling, and using BeautifulSoup to scrape data from the web.

Database Storage: MongoDB.

Flask: To create the server connection between dashboard and database.

Javascript: We primarily used D3.js to read the data from MongoDB, and used Plotly.js, Chart.js and Google charts to populate the dashboard with interactive charts.



Run the application to scrape the latest data from YAHOO FINANCE

DATA PRE-PROCESSING with PYTHON & libraries

Load into MongoDB

Use flask to connect with DB and the web framework

Visualise the data & tell the stories!

Data Wrangling PART 1 (ETL)

EXTRACT - Data preprocessing

- Data are scraped from https://au.finance.yahoo.com/ and tables are read in the HTML page. The results are converted into a dataframe using pandas.

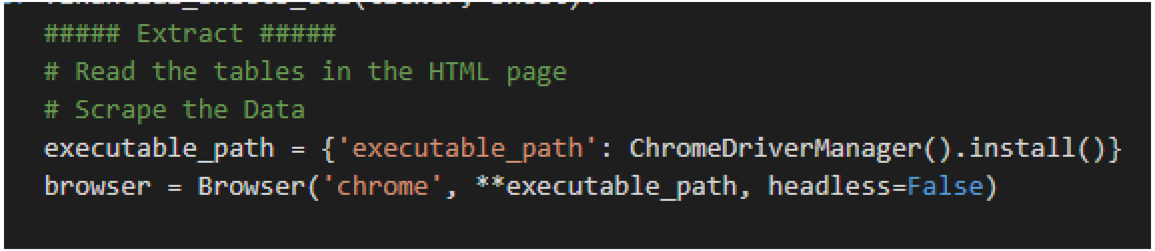
TRANSFORM

Major steps taken for data preprocessing:

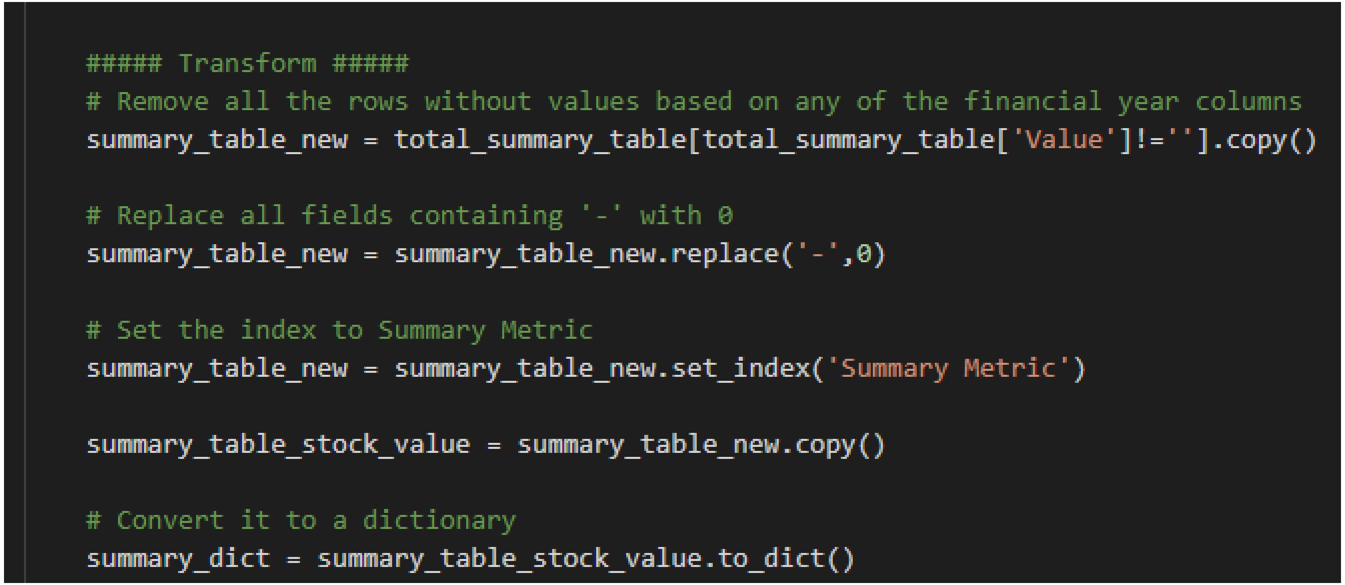
- Removing null values and replacing them with 0.
- Converting data into relevant formats (such as from string to float).
- Resetting the index before converting data into a dictionary.

LOAD

- Setting up the MONGODB connection when the data scraped from the website is ready to be loaded into the database.
- Different functions will load the respective info different databases. (ex: balance sheet / stock history data).



```
    ##### Extract #####
    # Read the tables in the HTML page
    # Scrape the Data
    executable_path = {'executable_path': ChromeDriverManager().install()}
    browser = Browser('chrome', **executable_path, headless=False)
```

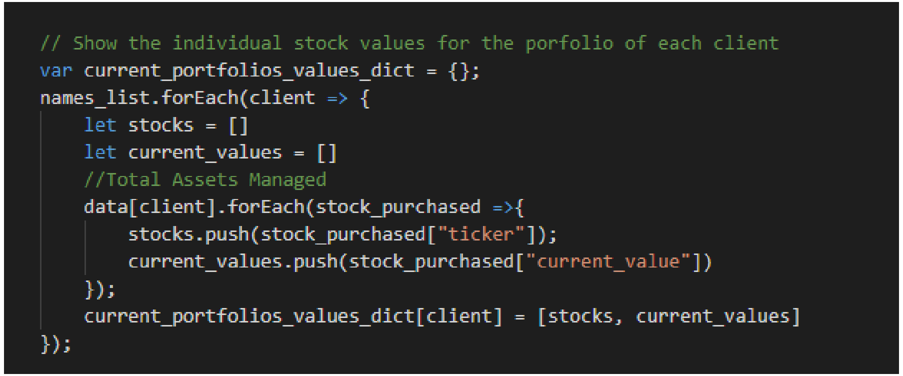


```
    ##### Transform #####
    # Remove all the rows without values based on any of the financial year columns
    summary_table_new = total_summary_table[total_summary_table['Value']!=''].copy()

    # Replace all fields containing '-' with 0
    summary_table_new = summary_table_new.replace('-',0)

    # Set the index to Summary Metric
    summary_table_new = summary_table_new.set_index('Summary Metric')

    summary_table_stock_value = summary_table_new.copy()

    # Convert it to a dictionary
    summary_dict = summary_table_stock_value.to_dict()
```

Data Wrangling PART 2 (Methodology)

D3.js

1. Loading data from mongoDB in a json format.
2. Using JS to calculate some information to be displayed in the graph, such as Total Current Portfolio Value of each client.
3. Converting data to arrays so that they could be visualised using JS libraries later.



```
// Show the individual stock values for the porfolio of each client
var current_portfolios_values_dict = {};
names_list.forEach(client => {
    let stocks = []
    let current_values = []
    //Total Assets Managed
    data[client].forEach(stock_purchased =>{
        stocks.push(stock_purchased["ticker"]);
        current_values.push(stock_purchased["current_value"])
    });
    current_portfolios_values_dict[client] = [stocks, current_values]
});
```

Visualisation Tools we used:

Plotly/Chart.js/Google Chart (charts)

1. Line chart - to show the overall performance of the stocks purchased by the clients.
2. Bar graph - to display total purchase and current value of all the clients.
3. Pie chart - summarise the current distribution portfolio value of the holding stocks.
4. Implement D3.js event listeners to allow the interactive navigation of the dashboard. User is able to selectively looking at a specific portfolio.

Bootstrap / HTML / CSS / Jinja (dashboard structure)

1. Make the main website’s structure by using HTML, enhance the styling using CSS & Bootstrap.

Final Thoughts

What we did well

- Built a comprehensive firm wide portfolio dashboard.
- Included filters to show information individual portfolio composition.
- Created informative and engaging interactive visualisations.

What didn’t so well

- Underestimated complexity of some of the graphs proposed initially, resulting in us opting for simpler graphs.

Proposed future ideas for the dashboard

- Include more complex graphs such as a line graph showing changes in the overall value of managed assets over different periods of time, such as the last month, year and so on.
- Include more granualised graphs of client portfolio compositions, such as changes in the value of their portfolios over the last day, week, two months and so on.

THANK
YOU