IS5006 Final Project How To Guide

Date : 10th April, 2022

Group 03 Algo Trading System - How-To Guide

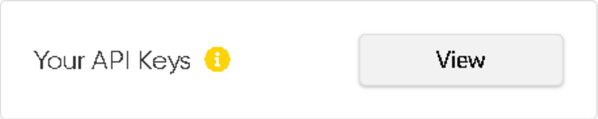
Jesper Lim Jun Hao (A0248996B) e0934611@u.nus.edu
Pitchappan Pitchappan Ramaswamy (A0236575W) e0740930@u.nus.edu
Seow Kah Yong (A0032747E) e0383424@u.nus.edu
Zhu Yexin (A0083629W) e0696677@u.nus.edu

There are 2 parts to our system, Live trading and Historical backtesting. Instructions on how to use both are given below.

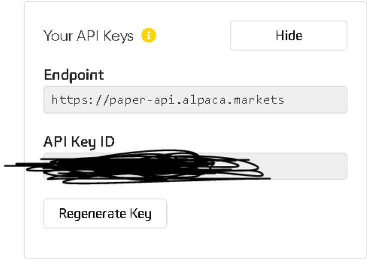
1. Live Trading

Step 1 : Create an Alpaca Account and get API Keys

1. Signup to Alpaca at https://app.alpaca.markets/signup. Create an account and verify your email
2. Login to Alpaca at https://app.alpaca.markets/login
3. Go to paper Trades dashboard by either navigating through the website or clicking this link https://app.alpaca.markets/paper/dashboard/overview
4. Click on View API keys on the right side of the screen



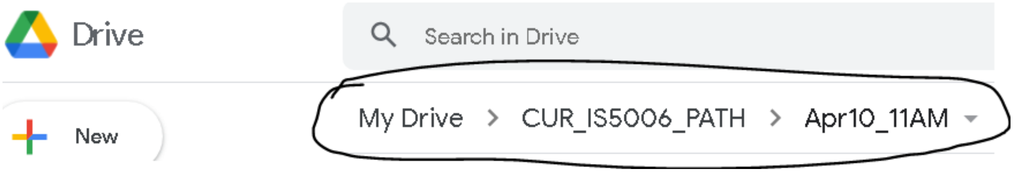
5. Copy API keys and Secret Keys and save it securely.



Endpoint
https://paper-api.alpaca.markets

API Key ID
[redacted]

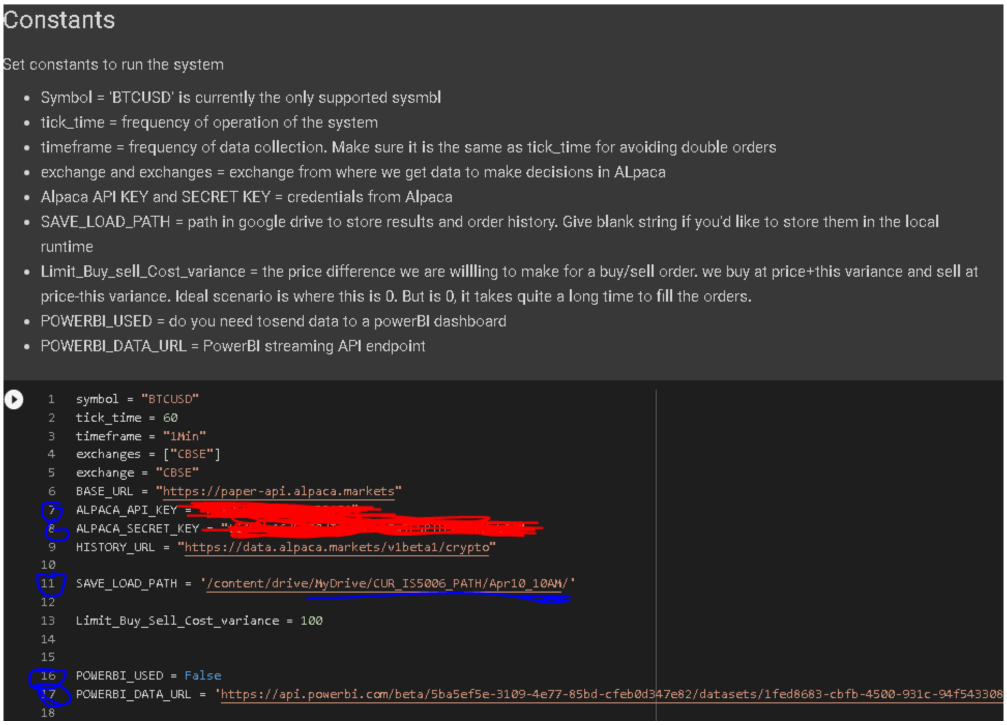
Step 2 : Create a Google Drive Folder



1. Click on the new button and click create a new folder with a name of your preference.
2. Note the directory of the Google Drive Folder. This will be used subsequently.
3. When we start a new system, the drive folder should be empty. When resuming a system, we can load CSV files to a drive to have the system continue learning and/or trading from a previous run instead of starting anew.
4. AgentWeights.csv and OrderProfitability.csv can be placed in the folder. They are the files used to make decisions and can even be loaded from completely different experiments and runs to help trade in the current run.

Step 3 : Google Colab

1. Upload the Colab File IS5006_Group3_Final_Live_Code_submission.ipynb to your Google Drive Folder and open it with Google Colab
2. Go to the Constants section



Constants

Set constants to run the system

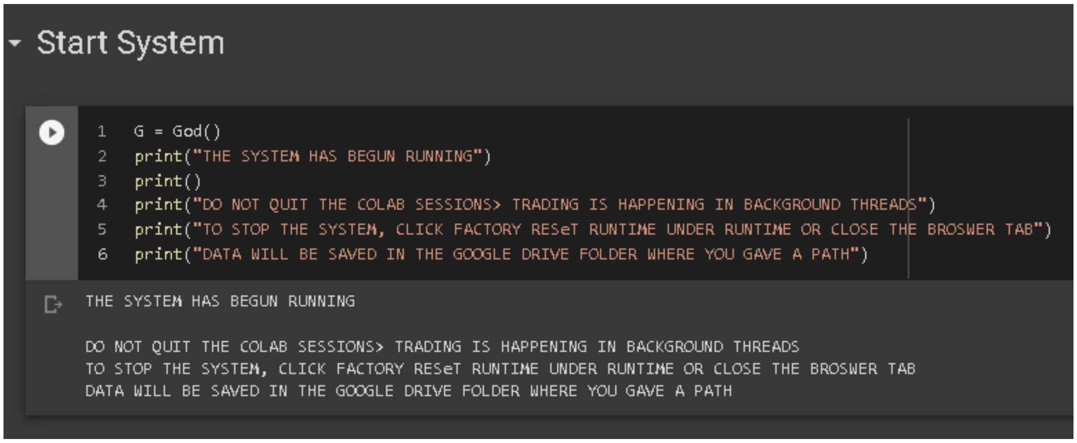
- Symbol = 'BTCUSD' is currently the only supported sysmbl
- tick_time = frequency of operation of the system
- timeframe = frequency of data collection. Make sure it is the same as tick_time for avoiding double orders
- exchange and exchanges = exchange from where we get data to make decisions in ALpaca
- Alpaca API KEY and SECRET KEY = credentials from Alpaca
- SAVE_LOAD_PATH = path in google drive to store results and order history. Give blank string if you'd like to store them in the local runtime
- Limit_Buy_sell_Cost_variance = the price difference we are willling to make for a buy/sell order. we buy at price+this variance and sell at price-this variance. Ideal scenario is where this is 0. But is 0, it takes quite a long time to fill the orders.
- POWERBI_USED = do you need tosend data to a powerBI dashboard
- POWERBI_DATA_URL = PowerBI streaming API endpoint

```
symbol = "BTCUSD"
tick_time = 60
timeframe = "1Min"
exchanges = ["CBSE"]
exchange = "CBSE"
BASE_URL = "https://paper-api.alpaca.markets"
ALPACA_API_KEY = [redacted]
ALPACA_SECRET_KEY
HISTORY_URL = "https://data.alpaca.markets/v1beta1/crypto"

SAVE_LOAD_PATH = '/content/drive/MyDrive/CUR_IS5006_PATH/Apr10_10AM/'

Limit_Buy_Sell_Cost_variance = 100


POWERBI_USED = False
POWERBI_DATA_URL = 'https://api.powerbi.com/beta/5ba5ef5e-3109-4e77-85bd-cfeb0d347e82/datasets/1fed8683-cbfb-4500-931c-94f543308

```

   a. Set **ALPACA_API_KEY** and **ALPACA_SECRET_KEY** to the credentials saved from step 1
   b. Set **SAVE_LOAD_PATH** to the Google Drive Folder directory from Step 2

    c. If you have PowerBI configured (covered in the next section), set **POWERBI_USED** to True and set the **POWERBI_DATA_URL** to the streaming API given by PowerBI
    d. You can optionally change **Limit_Buy_Sell_Variance** to 0 for an ideal model. It will help in profits but limit orders might never get filled (read code for explanation)

3. Click Run All
    a. Allow access to Google Drive when asked
    b. Scroll down to the section **Start System**
    c. It will take a few minutes to display the following



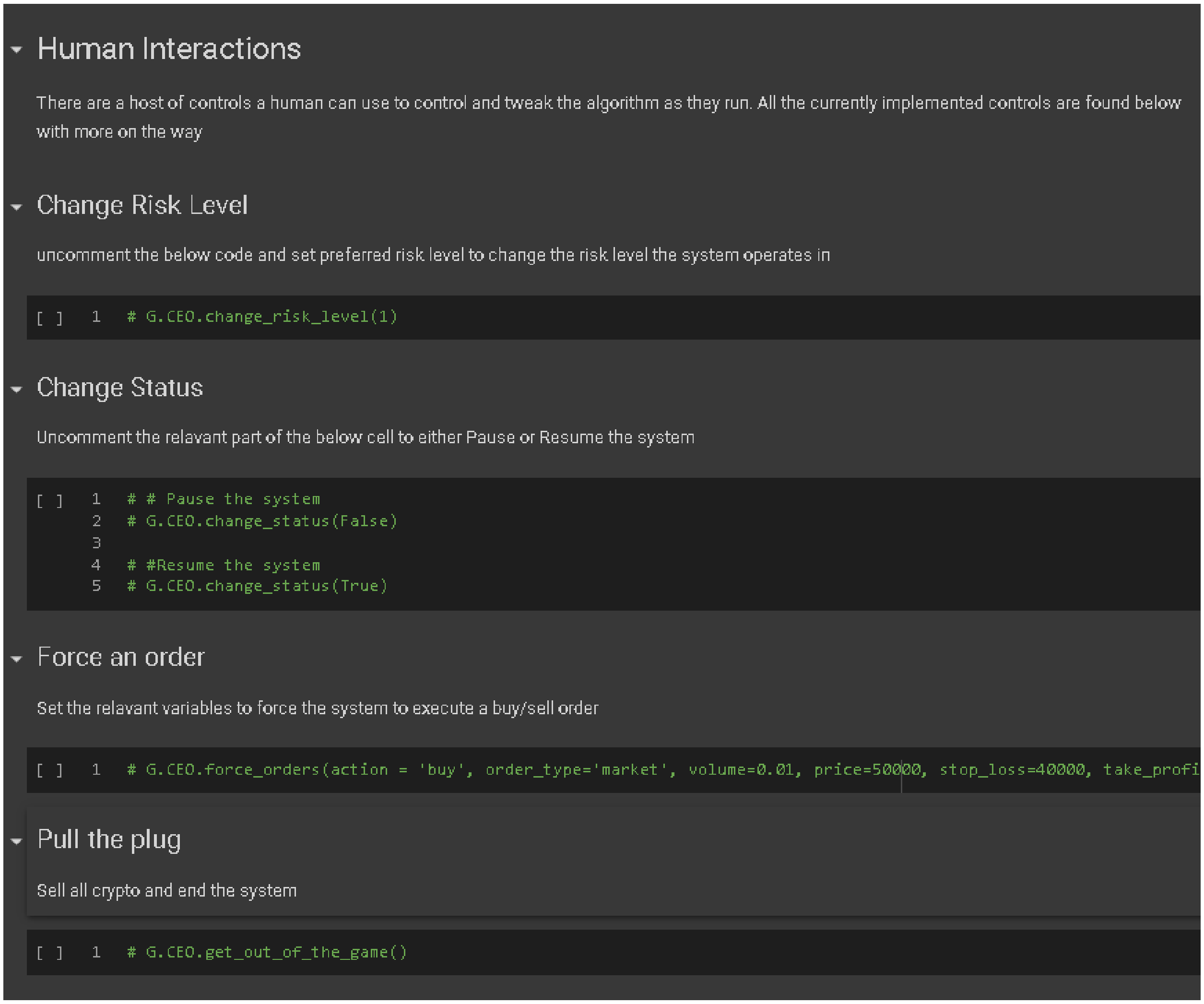
Start System

```
G = God()
print("THE SYSTEM HAS BEGUN RUNNING")
print()
print("DO NOT QUIT THE COLAB SESSIONS> TRADING IS HAPPENING IN BACKGROUND THREADS")
print("TO STOP THE SYSTEM, CLICK FACTORY RESeT RUNTIME UNDER RUNTIME OR CLOSE THE BROSWER TAB")
print("DATA WILL BE SAVED IN THE GOOGLE DRIVE FOLDER WHERE YOU GAVE A PATH")
```

```
THE SYSTEM HAS BEGUN RUNNING

DO NOT QUIT THE COLAB SESSIONS> TRADING IS HAPPENING IN BACKGROUND THREADS
TO STOP THE SYSTEM, CLICK FACTORY RESeT RUNTIME UNDER RUNTIME OR CLOSE THE BROSWER TAB
DATA WILL BE SAVED IN THE GOOGLE DRIVE FOLDER WHERE YOU GAVE A PATH
```

    d. The code is now running

4. Human Interactions



Human Interactions

There are a host of controls a human can use to control and tweak the algorithm as they run. All the currently implemented controls are found below with more on the way

Change Risk Level

uncomment the below code and set preferred risk level to change the risk level the system operates in

```
[ ]  # G.CEO.change_risk_level(1)
```

Change Status

Uncomment the relavant part of the below cell to either Pause or Resume the system

```
[ ]  # # Pause the system
     # G.CEO.change_status(False)

     # #Resume the system
     # G.CEO.change_status(True)
```

Force an order

Set the relavant variables to force the system to execute a buy/sell order

```
[ ]  # G.CEO.force_orders(action = 'buy', order_type='market', volume=0.01, price=50000, stop_loss=40000, take_profi
```

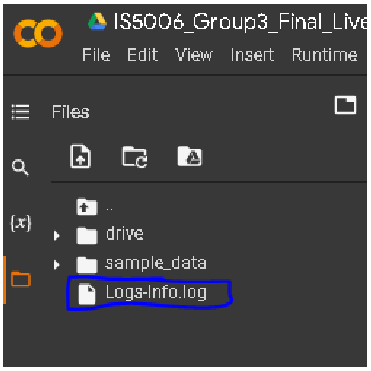
Pull the plug

Sell all crypto and end the system

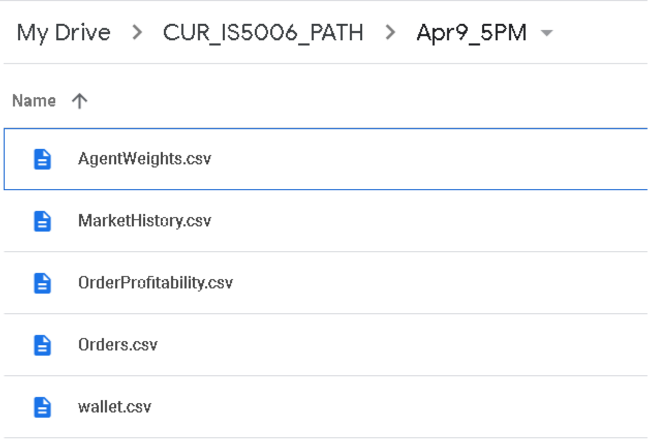
```
[ ]  # G.CEO.get_out_of_the_game()
```

You can uncomment these lines to use the 4 human interaction features to control your code

5. Log files will be present in the local runtime to debug if anything goes wrong. In case you need assistance in running the code, be sure to download this log file shown in the picture below and contact one of the authors with the file and we'll help sort it out



6. You can go back to google drive to see files being saved
    a. Overall Folder

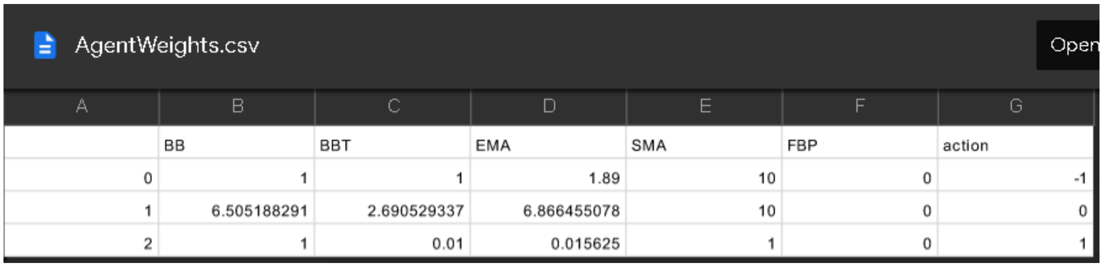


My Drive > CUR_IS5006_PATH > Apr9_5PM

Name ↑

- AgentWeights.csv
- MarketHistory.csv
- OrderProfitability.csv
- Orders.csv
- wallet.csv

    b. AgentWeights.CSV



AgentWeights.csv

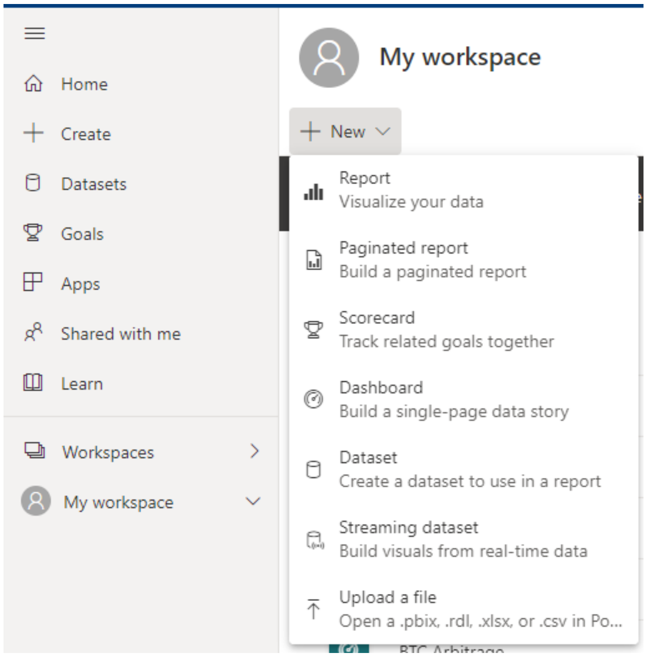
A	B	C	D	E	F	G
	BB	BBT	EMA	SMA	FBP	action
0	1	1	1.89	10	0	-1
1	6.505188291	2.690529337	6.866455078	10	0	0
2	1	0.01	0.015625	1	0	1

7. In Case of Quitting, we recommend Using the Change Status function to False and wait a minute before closing the code to ensure all progress has been saved in the google drive

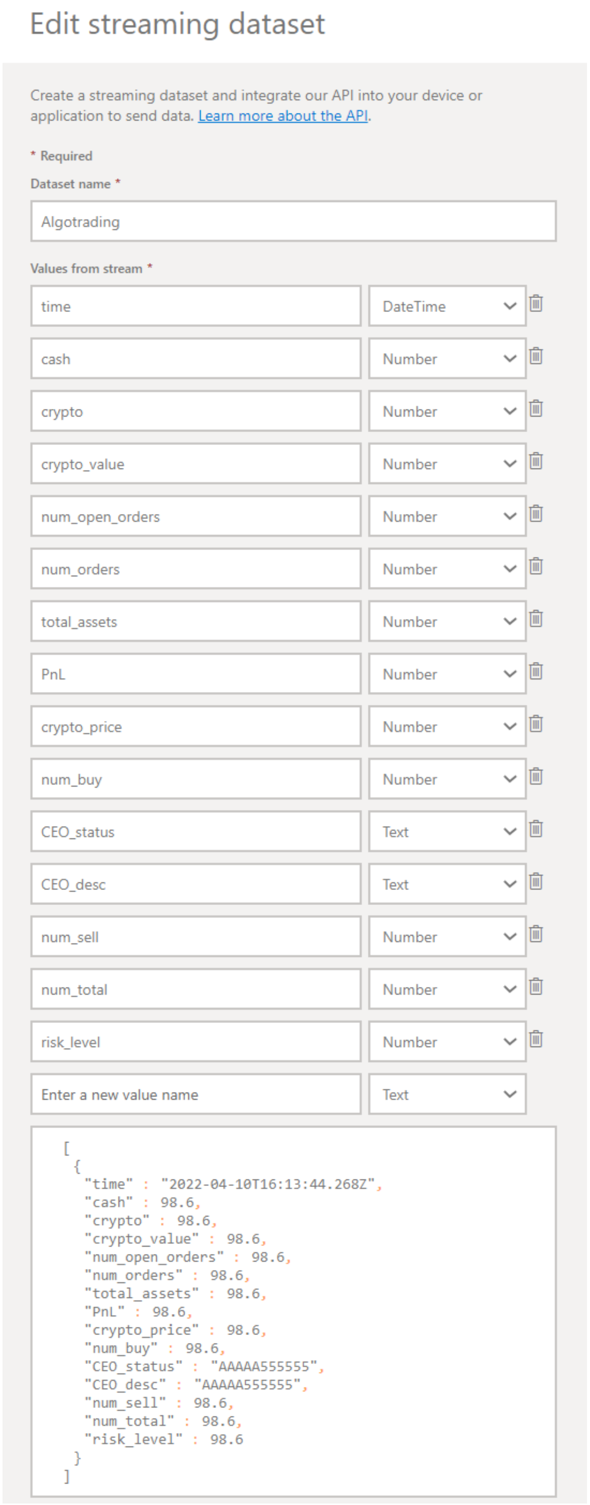
Step 4 : PowerBI

Step 4.1 : Create a new Streaming API

1. Go to My workspace, click on New button shown in the image below then click on Streaming dataset



2. Configure the values from the streaming API as shown below



Edit streaming dataset

Create a streaming dataset and integrate our API into your device or application to send data. Learn more about the API.

* Required

Dataset name *

Algotrading

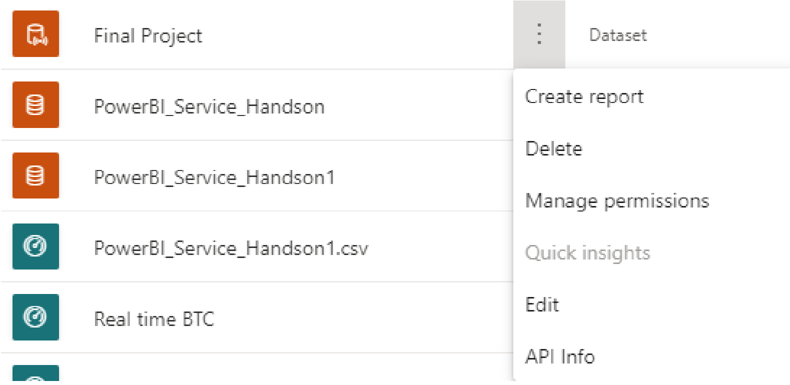
Values from stream *

time	DateTime
cash	Number
crypto	Number
crypto_value	Number
num_open_orders	Number
num_orders	Number
total_assets	Number
PnL	Number
crypto_price	Number
num_buy	Number
CEO_status	Text
CEO_desc	Text
num_sell	Number
num_total	Number
risk_level	Number
Enter a new value name	Text

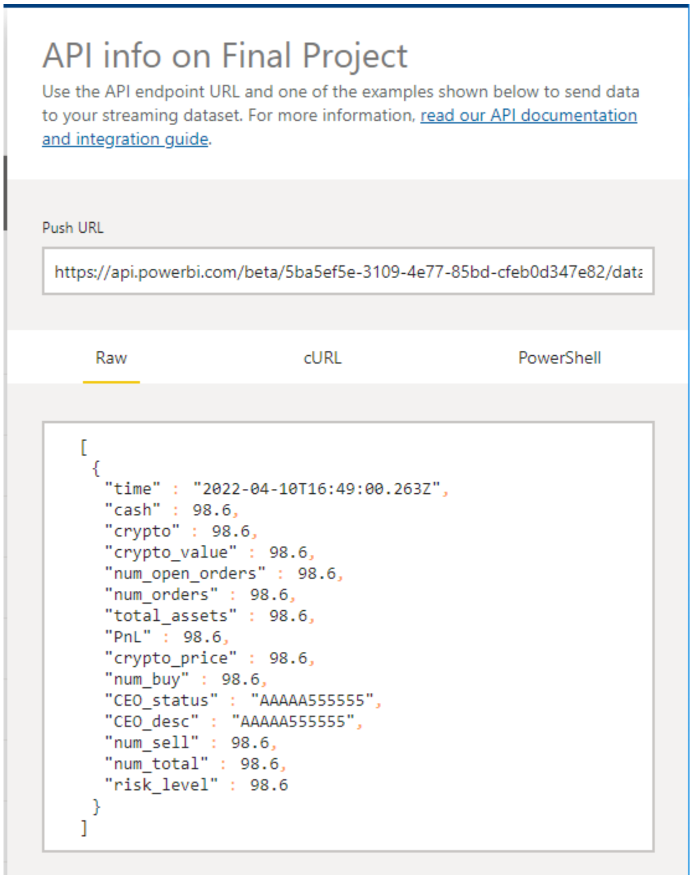
```
[
 {
  "time" : "2022-04-10T16:13:44.268Z",
  "cash" : 98.6,
  "crypto" : 98.6,
  "crypto_value" : 98.6,
  "num_open_orders" : 98.6,
  "num_orders" : 98.6,
  "total_assets" : 98.6,
  "PnL" : 98.6,
  "crypto_price" : 98.6,
  "num_buy" : 98.6,
  "CEO_status" : "AAAAA555555",
  "CEO_desc" : "AAAAA555555",
  "num_sell" : 98.6,
  "num_total" : 98.6,
  "risk_level" : 98.6
 }
]
```

Step 4.2: Obtain the API key

1. Click on the three dots beside the API streaming dataset that you have just created and click on **API Info**



2. Save the Push URL displayed. This is your API key.



API info on Final Project

Use the API endpoint URL and one of the examples shown below to send data to your streaming dataset. For more information, read our API documentation and integration guide.

Push URL

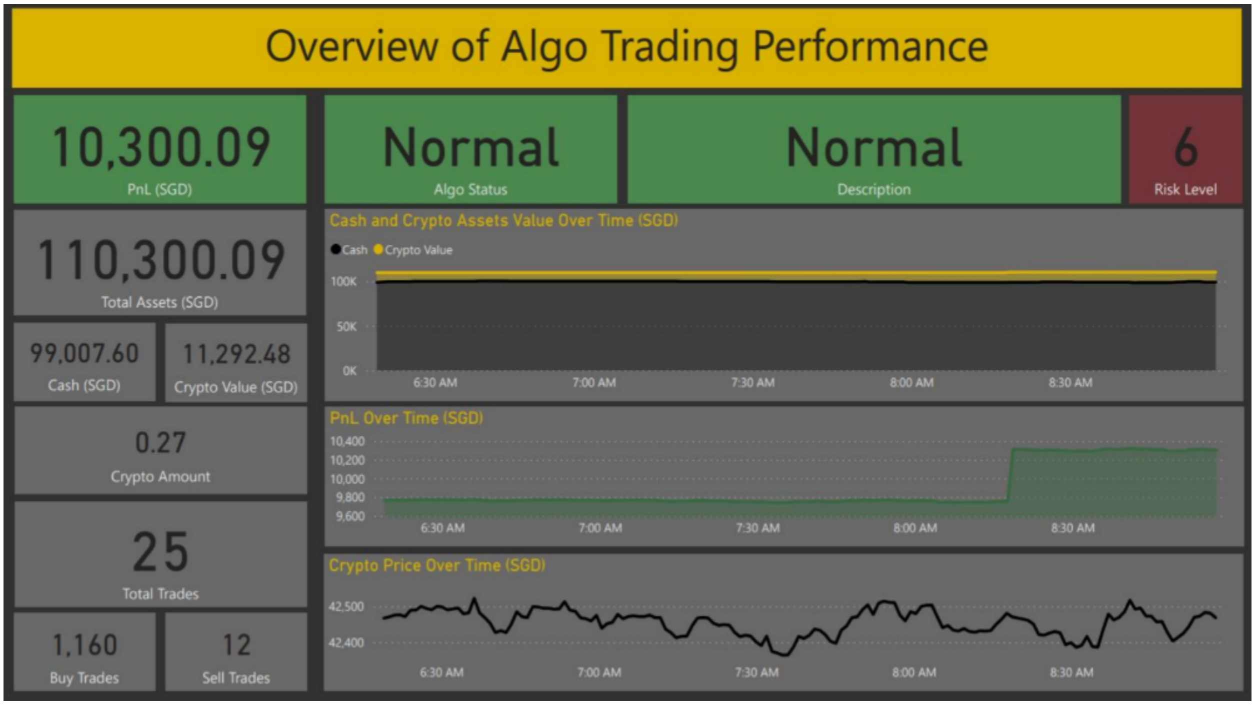
https://api.powerbi.com/beta/5ba5ef5e-3109-4e77-85bd-cfeb0d347e82/data

```
[
 {
  "time" : "2022-04-10T16:49:00.263Z",
  "cash" : 98.6,
  "crypto" : 98.6,
  "crypto_value" : 98.6,
  "num_open_orders" : 98.6,
  "num_orders" : 98.6,
  "total_assets" : 98.6,
  "PnL" : 98.6,
  "crypto_price" : 98.6,
  "num_buy" : 98.6,
  "CEO_status" : "AAAAA555555",
  "CEO_desc" : "AAAAA555555",
  "num_sell" : 98.6,
  "num_total" : 98.6,
  "risk_level" : 98.6
 }
]
```

Step 4.3: Return to Colab file

1. Set **POWERBI_USED** to TRUE in *IS5006_Group3_Final_Live_Code_submission.ipynb* and set **POWERBI_DATA_URL** in Colab to PowerBI Streaming API obtained in the previous step.

Step 4.4: Load the Streaming API and Create a Dashboard to view live results



Overview of Algo Trading Performance

10,300.09
PnL (SGD)

Normal
Algo Status

Normal
Description

6
Risk Level

110,300.09
Total Assets (SGD)

99,007.60
Cash (SGD)

11,292.48
Crypto Value (SGD)

0.27
Crypto Amount

25
Total Trades

1,160
Buy Trades

12
Sell Trades

2. Historical Trading

We already have data loaded up in github for you to test out the algorithm for the period of Feb to March 2022. If you would like to run for other time periods, create an alpaca API and uncomment code to perform experiments.

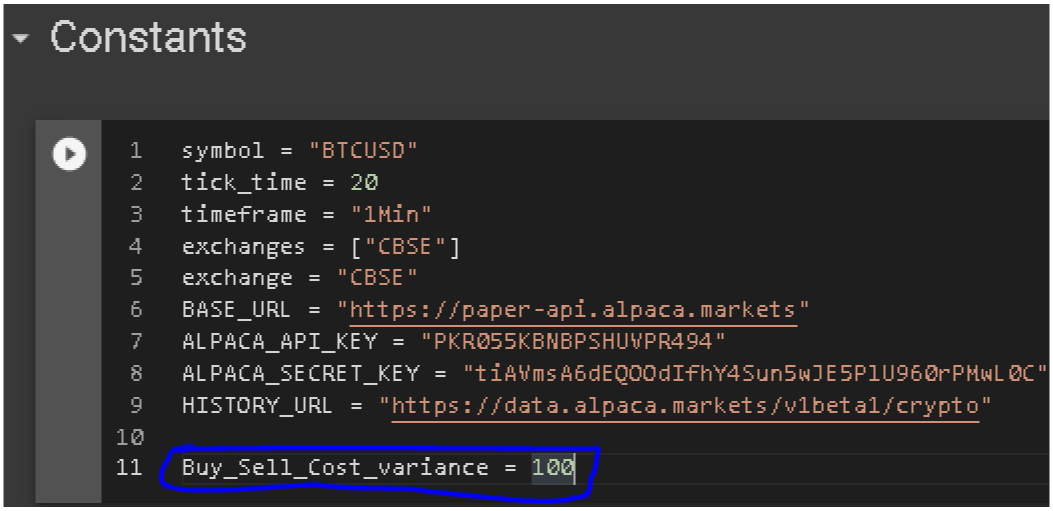
To perform experiments on a period between Feb and March 2022, follow the steps below:

Step 1 : Open Google Colab

1. Upload/Load *IS5006_Group3_Backtesting.ipynb*

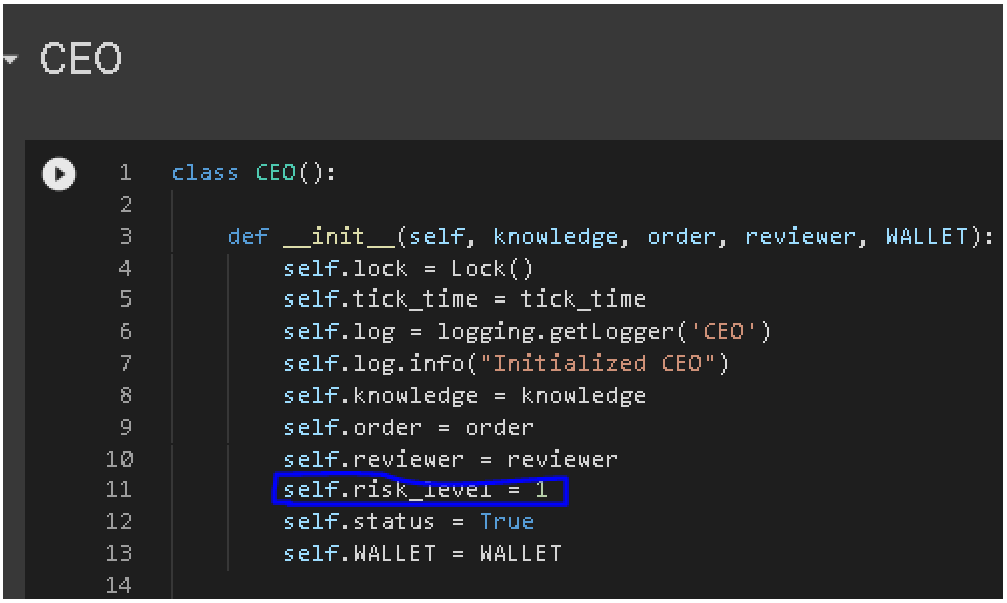
Step 2: Set **Buy_Sell_Cost_variance**

2. An ideal situation is 0, but to simulate market orders or variance in price, our system always buys at **price+Buy_Sell_Cost_variance** and sells at **price-Buy_Sell_Cost_variance**



```
symbol = "BTCUSD"
tick_time = 20
timeframe = "1Min"
exchanges = ["CBSE"]
exchange = "CBSE"
BASE_URL = "https://paper-api.alpaca.markets"
ALPACA_API_KEY = "PKR055KBNBPSHUVPR494"
ALPACA_SECRET_KEY = "tiAVmsA6dEQOOdIfhY4Sun5wJE5P1U960rPMwL0C"
HISTORY_URL = "https://data.alpaca.markets/v1beta1/crypto"

Buy_Sell_Cost_variance = 100
```

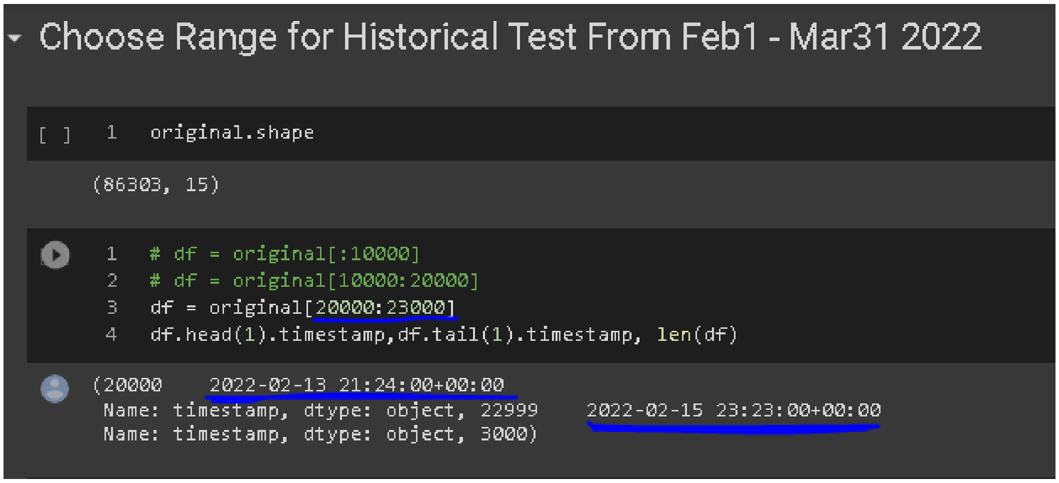
Step 3: Go to CEO and set preferred risk level for experiment



CEO

```
class CEO():

    def __init__(self, knowledge, order, reviewer, WALLET):
        self.lock = Lock()
        self.tick_time = tick_time
        self.log = logging.getLogger('CEO')
        self.log.info("Initialized CEO")
        self.knowledge = knowledge
        self.order = order
        self.reviewer = reviewer
        self.risk_level = 1
        self.status = True
        self.WALLET = WALLET

```

Step 4 : Run Till Choose range. Then tweak indexes and rerun till you are happy with the date range



Choose Range for Historical Test From Feb1 - Mar31 2022

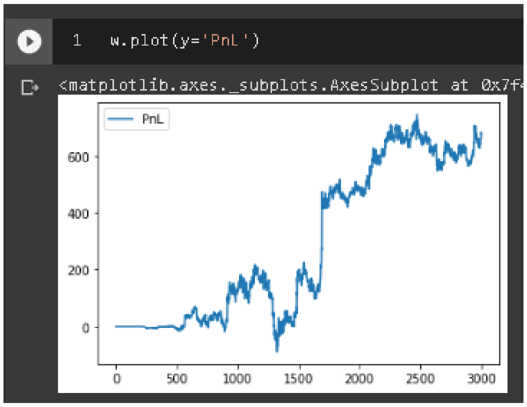
```
original.shape
```

```
(86303, 15)
```

```
# df = original[:10000]
# df = original[10000:20000]
df = original[20000:23000]
df.head(1).timestamp,df.tail(1).timestamp, len(df)
```

```
(20000    2022-02-13 21:24:00+00:00
 Name: timestamp, dtype: object, 22999    2022-02-15 23:23:00+00:00
 Name: timestamp, dtype: object, 3000)
```

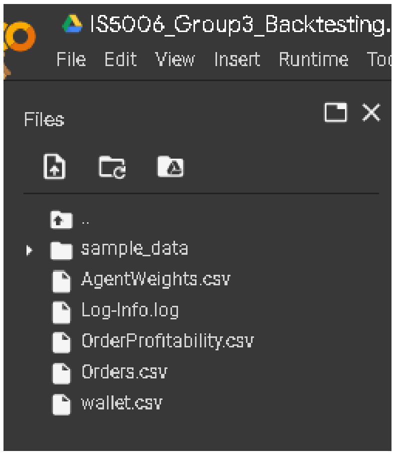
Step 5: Run the rest of the code to generate reports



```
w.plot(y='PnL')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4
```

Step 5 : Download Files



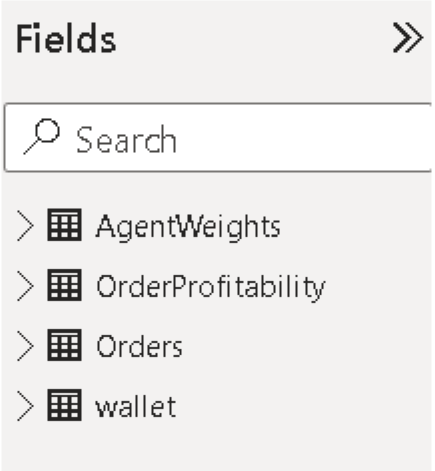
Download all these CSVs

Step 6: PowerBI

Step 6.1 - Open the *IS5006_Group03_Backtesting.pbix* File in PowerBI desktop

**Note: you may need to download PowerBI desktop or use PowerBI service to open the file*

Step 6.2 - Load the downloaded CSV



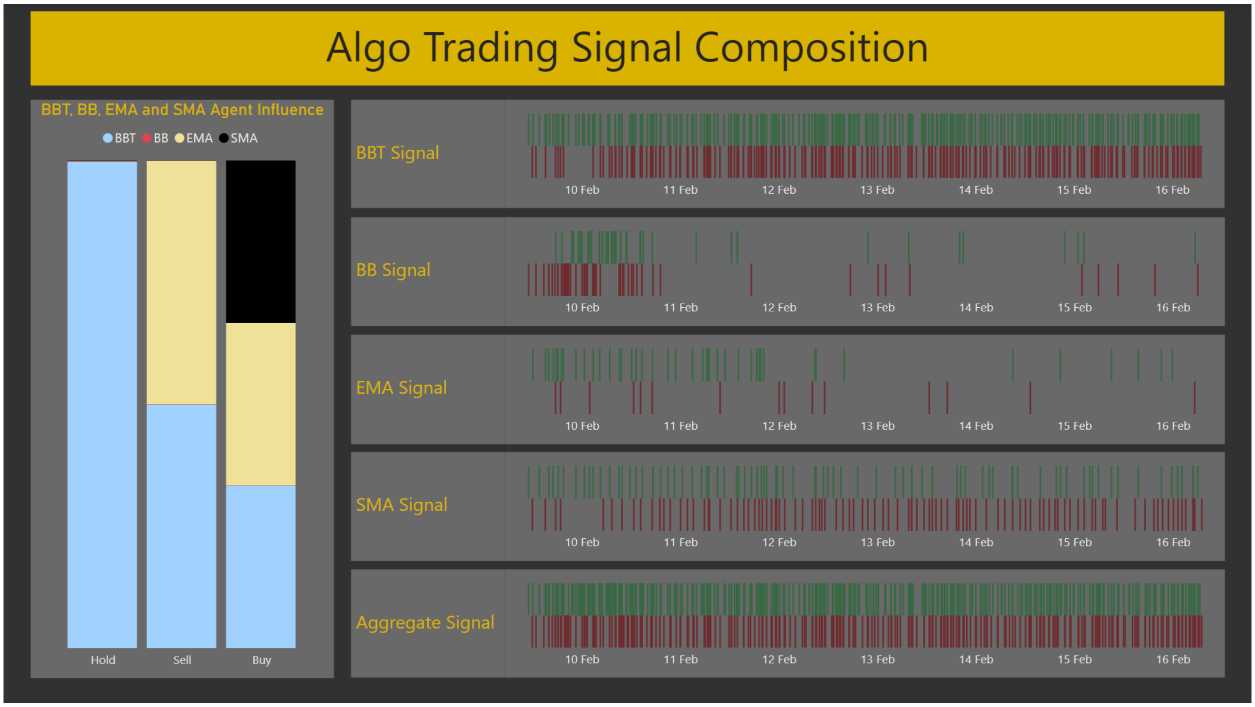
Fields

Search

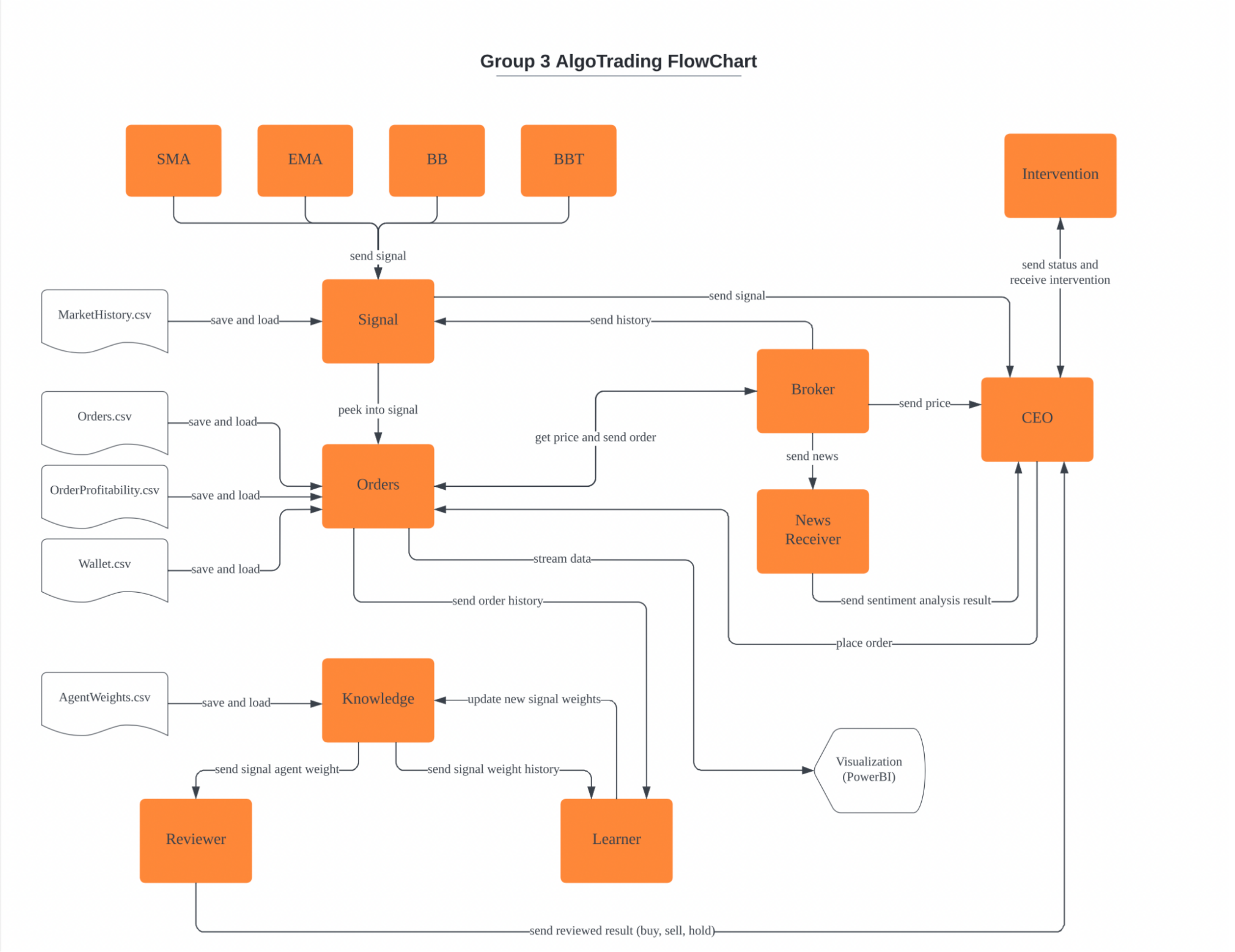
- AgentWeights
- OrderProfitability
- Orders
- wallet

Step 6.3 - Visualize

Create your own visualizations or use ours as a template



Flow-chart of our model



Data that is used

There are various sources of data that we obtained from the internet:

1) Alpaca paper-testing historical data[1]
2) Alpaca Live Data
3) News Data from BenZinga
4) Coinbase web scraped news data

[1] https://paper-api.alpaca.markets