

**Big Data Architecture**

**Big Data Machine Learning Bootcamp**

Practica Fernando Neso Polovio

Brainstorm

* Bot de trading que opera en el mercado de acciones de Estados Unidos, según una estrategia definida. Debe comprar y vender una suma determinada en cada operación. También tendrá seteado un stop loss para proteger las posiciones.

Este robot debería poder enviar información a otro sistema donde se almacenen los datos de las operaciones.

* Algoritmo para testear estrategias de trading a través datos históricos de un activo financiero para obtener información que permita evaluar el desempeño.
* Modelo que permita predecir el comportamiento de la estrategia ante distintos escenarios de cambio en los precios, la volatilidad y el volumen de mercado.

Diseño del DAaaS

### Definición la estrategia del DAaaS

El proyecto se basa en la creación de un bot de trading que permita ejecutar operaciones de compra y venta de activos financieros en tiempo real, siguiendo una estrategia definida previamente.

El mercado principal es el de acciones americanas y ETFs (S&P 500, Nasdaq, Dow Jones, Russell 2000).

El intercambio de información para efectuar las operaciones se hace con la API Rest de TradeStation.

Este broker permite operar los distintos activos mencionados y provee información de los mercados en tiempo real.

Además, en un entorno separado y sin una conexión directa con el bot de trading (al menos por el momento), se desea crear un algoritmo que permita testear estrategias en diferentes mercados o activos utilizando datos históricos.

Este algoritmo toma datos históricos de los activos, los procesa en base a una estrategia definida en el código y devuelve información estadística útil para evaluar la eficiencia de esta en los distintos activos.

### Arquitectura DAaaS

Lista de componentes del Trading Bot.

Este algoritmo es una cloud function implementada desde cloud storage con activador de Pubsub cuyos componentes son:

* *TickerAnalyzer:* Esta función se encarga de solicitar precios de activos a la API un minuto antes del cierre de mercado. En base a estos precios decide cuáles son los tickers que deben ser operados en compra y cuales en venta. Esta información es pasada al operation bot.

El TickerAnalyzer debe recuperar desde History\_daily\_operation las operaciones de estado “open” para poder enviar la orden de cierre en caso de corresponder.

* *OperationBot:* este bot envía operaciones hacia la API de TradeStation según la información que recibe del ticker analyzer. Ejecutada la operación, envía información de la ejecución para ser almacenada en Firestor.

Aclaración importante:

Pensé este algoritmo (TradingBot) compuesto por dos bloques separados, como describí más arriba. No estoy seguro si deben ser cloud functions separadas y que de alguna manera el TickerAnalyzer sea un activador para el OperationBot o poner todo en una misma estructura de código y realizar una sola activación.

*History-daily-operation :* base de datos de firestore donde se almacena cada operación que realiza el operation bot en forma de documento. Contiene la siguiente información: estado de la operación, ticker del activo operado, tipo de operación, precio de ejecución, fecha y hora.

*TradeStation API:* Web API del broker Tradestation que permite efectuar operaciones de compra y venta de varios activos en distintos mercados financieros y provee datos en tiempo real.

Lista de componentes del Strategy tester

*Strategy tester* : Cluster de hadoop en Dataproc que analiza datos históricos de muchos activos financieros distintos para obtener, en base a combinaciones de distintas variables, el comportamiento de los precios en distintas situaciones de mercado.

*Dataset historical prices*: Datos históricos de activos financieros en formato csv.

*Strategy - results*: json con los resultado del análisis de la estrategia almacenado en Cloud Storage.

Nota: La idea a futuro es desarrollar con Strategy tester un modelo de aprendizaje automático que prediga el comportamiento de las estrategias en distintos escenarios de precios, volumen y volatilidad.

### 

### DAaaS Operating Model Design and Rollout

*Trading Bot*

1. Se activa manualmente desde consola la cloud function TradingBot con el trigger especificado en el activador.
2. Se activa la función TickerAnlayzer. Solicita precio de activos a la API de TradeStation.
3. Recupera desde Firestore las operaciones que tiene estado “open”.
4. Selecciona en base al precio que activos están en condiciones de ser operados y que tipo de operación se debe efectuar.
5. Para la operación de estado “open” verifica si debe efectuar el cierre de la misma según el precio obtenido para el activo de la orden.
6. Envía estos datos al OperationBot.
7. El OperationBot envía la orden correspondiente a la API de TradeStation y solicita confirmación de ejecución.
8. EL OperationBot envía a Firestor para su almacenamiento un documento con datos de la orden ejecutada.

*Strategy tester*

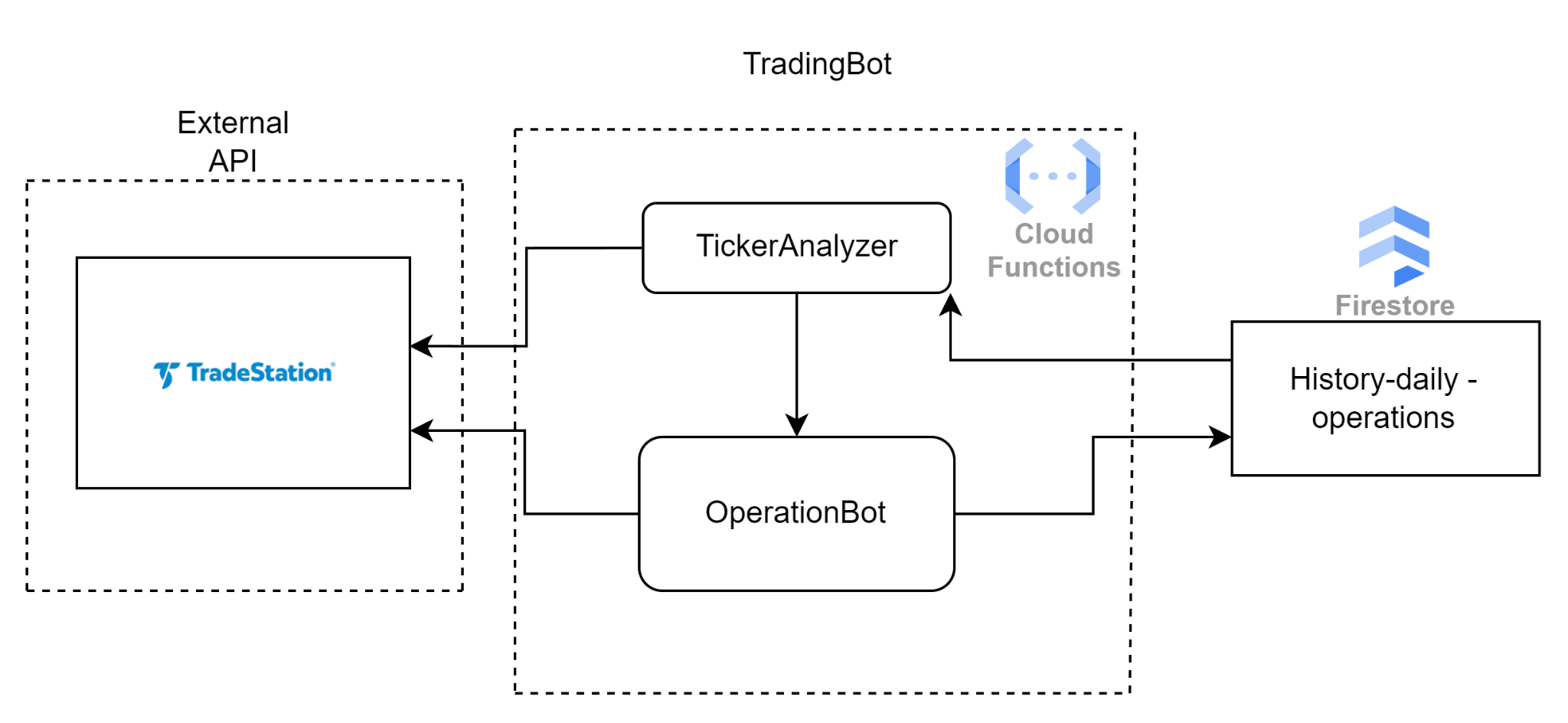
1. Carga manual de datos a Cloud storage
2. Procesamiento de los datos en el cluster.
3. Storage de los resultados para uso futuro.

### 

### 

### Desarrollo de la plataforma DAaaS.

**Trading Bot**



**Strategy Tester**

