****

Test task

This document describes the test task for the position of Data Engineer at Stokex. It is a very hands-on exercise, relevant to what the person will be doing daily in the aforementioned position. Let us get started:

1. Create a free trial account here: <https://www.interactivebrokers.com/en/trading/free-trial.php> . It will nor require you to transfer funds or confirm your ID. It only create a demo account so that you could use IB products for testing which we need for the task. **HOWEVER NOTE**: it will require 1 day to be able to use your account as IB restarts their systems daily and trial demo accounts are only available to use after this reset, so create the account at least 1 day before you attempt the next steps.
2. After your demo account is set up, download and install IB TWS (locate installation files on IB’s website). You can find help on setting it up here: <https://interactivebrokers.github.io/tws-api/introduction.html>
3. Connect to TWS API via your language of choice. We recommend python and using this package as it is written in a very flexible style: <https://github.com/erdewit/ib_insync>. The goal of this is step is to make a screen shot of **ib\_insync.ib.IB.accountSummary()** method.
4. Build a script that collects and saves (to a file) some real-time data for Forex(‘EURUSD') product in ib\_insync (find the details on github/documentation of ib\_insync). Run the script to collect the data for at least 1 hour using 5-second real-time bars
5. Execute the strategy implemented in run\_strategy\_pipeline.R it will give you table of positions to enter and exit. The “position” means you have to buy 1 unit of the asset and hold it until you get to “nothing” where you should be holding 0 units of the asset.
6. Create a market order to buy some of Forex(‘EURUSD’) and after some time, sell those positions. The goal is to collect the call-back information (i.e. execution time, price, commissions paid) and write that to some file.

Bonus tasks:

1. For task 4/5, create a local Postgres db and a schema where you put some meta tables as well as the table holding real-time data and write data directly to DB rather than file. The output of this task should be some SQL files showing how you create your DB/schema/tables
2. For task 4/5, instead of manually running the scripts, write an Airflow DAG script that could be used to schedule your work.
3. For task 4/5, create docker-compose file that could spin-up airflow, IB TWS, and real-time script to automate everything. It might be really challenging to make sure IB TWS works, so you could do it theoretically, assuming it works. The other parts - airflow and the python scripts themselves - should be easy and doable.

**Deliverables**:

* 1. All code in an easy to execute fashion e.g. bash script that runs the code as intended
  2. Output files of the strategy positions
  3. Output log files of trades
  4. Any other supplementary files

## Term glossary:

IB - Interactive Brokers, an important broker we use for our systems

TWS - Trader Workstation - trading program/api by IB