Manticor

IntelliTrade Market Making Bot

- Students:
- Alessio Ferraro, Alessio.ferraro001@umb.edu
- Weiting Hu, Weiting.hu001@umb.edu
- Werner Ordonez, w.ordonezsalguero001@umb.edu
- Man Wong, man.wong001@umb.edu
- Philip DiSarro, philip.disarro001@umb.edu
- Client:
- Jerry Shu, jerry@manticor.io

Project Proposal v2.0 05/22/2020

1 Introduction

1.1 Purpose

The primary purpose of the IntelliTrade Market Making Bot is to reduce the spread and guarantee liquidity for the Manticor derivatives exchange market. New exchanges and newly listed coins often lack liquidity and have a big spread which discourages potential clients from using them. The IntelliTrade Market Making Bot will consistently make purchases and sales such that the spread will be minimized. In the process of improving market liquidity, the bot will generate a profit for the user based on the market spread. The target clientele of the IntelliTrade Market Making Bot will be professional market makers.

1.2 Scope

The Manticor Market Making Bot is an autonomous bot that runs locally on the host's computer. By connecting to the internet, it will be able to access the user's cryptocurrency wallet, view up-to-date cryptocurrency exchange rates from both the BitMEX and the CoinMarketCap platforms, and submit buy and sell offers to the BitMEX exchange. The user will be able to start the bot and modify some of its parameters, but they will not be able to directly control it in any way aside from terminating it. The bot will support all cryptocurrency-traditional currency pairs supported by BitMEX's platform. Using our bot, market makers will be able to majorly streamline and optimize the trading process, allowing them to make greater profits and guarantee liquidity for cryptocurrency exchanges with less effort. Traditionally, market makers make transactions based on the market spread and typically do not concern themselves with general market trends. While this can be profitable, it is also extremely risky as its possible for the market to drop so quickly that they're unable to sell their newly purchased assets. To minimize this risk, our bot will take these market trends into account and employ different strategies for downtrend side-trend and uptrend markets.

1.3 Overview

The bot will create profit for the user based on the spread of the exchange market. The larger the difference between the best-ask and best-bid price, the more profit the bot will generate. It will start with the balance of cryptocurrency in the user's wallet. It will then submit buy orders with the highest bid price and sell orders at the current ask price (the lowest sell price) from the selected platform (BitMEX or CoinMarketCap). Once one of these get a hit (another trader take's their order), the bot should go to its orders on the opposite side (as it would be likely that those prices have since changed), cancel them, and resubmit them at the latest price. By repeating this indefinitely, the spread on the exchange should continuously decrease and the bot should generate profit until the spread has minimized. If the bot detects that the spread is minimized and thus the exchange is no longer profitable, then it will cease to make transactions until the market becomes profitable.

1.3.1 Existing Work

- BitMEX Sample Market Maker: A sample bot created by BitMEX to use on their platform. This bot is free to be used and modified by anybody. We will be using this as our template for the Manticor Market Making Bot. `https://github.com/BitMEX/sample-market-maker`
- Cryptohopper: One of the most popular commercial market making bots available to the public. `https://www.cryptohopper.com/features/market-making-bot`

1.3.2 Limitations

As we are building this on BitMEX's platform, the only cryptocurrency-traditional currency pairs we will be able to use are those supplied by BitMEX. Additionally, since a market maker bot generates profit based on spread while simultaneously reducing the spread, most market maker bots target a wide variety of exchanges so that they can perform transactions of whichever exchange has the largest spread to maximize profit; however, since our bot will only function on the BitMEX platform, once the spread on that exchange is minimized, it will cease to generate profit for large periods of time until the spread increases to a profitable amount again.

1.4 Definitions

- PY: Python
- REST API: Representational State Transfer
- WebSocket: Computer Communications API
- API: Application Programming Interface
- AWS: Amazon Web Services
- PHP: PHP Hypertext Processor
- Bid Price: The price a buyer is offering in a purchase order.
- Ask Price: The price a seller is requesting in a sell order.
- Best-Bid: The highest price out of all purchase orders.
- Best-Ask: The lowest price out of all sell orders.
- Spread: The difference between the best-bid and the best-ask price.

2 Requirements

- **R1** The user will be able to configure the bot by setting certain parameters such as: the price source, the currency pair, the amount of cryptocurrency from the user's wallet used at the start of trading, the minimum spread between the bid and ask price that the bot will consider acceptable, the bounds of the rate of change of cryptocurrency prices that the bot will consider acceptable, when to resubmit old offers at more newer prices, how aggressive the bot should be at closing the spread, and how long the bot should run before terminating.
- **R2** The bot should function mostly autonomously.
- **R3** Statistics regarding the performance and profit of the bot shall be displayed in a simple graphical user interface.
- **R4** The bot should minimize risk to the user and prevent catastrophic financial loss.

3 Specifications

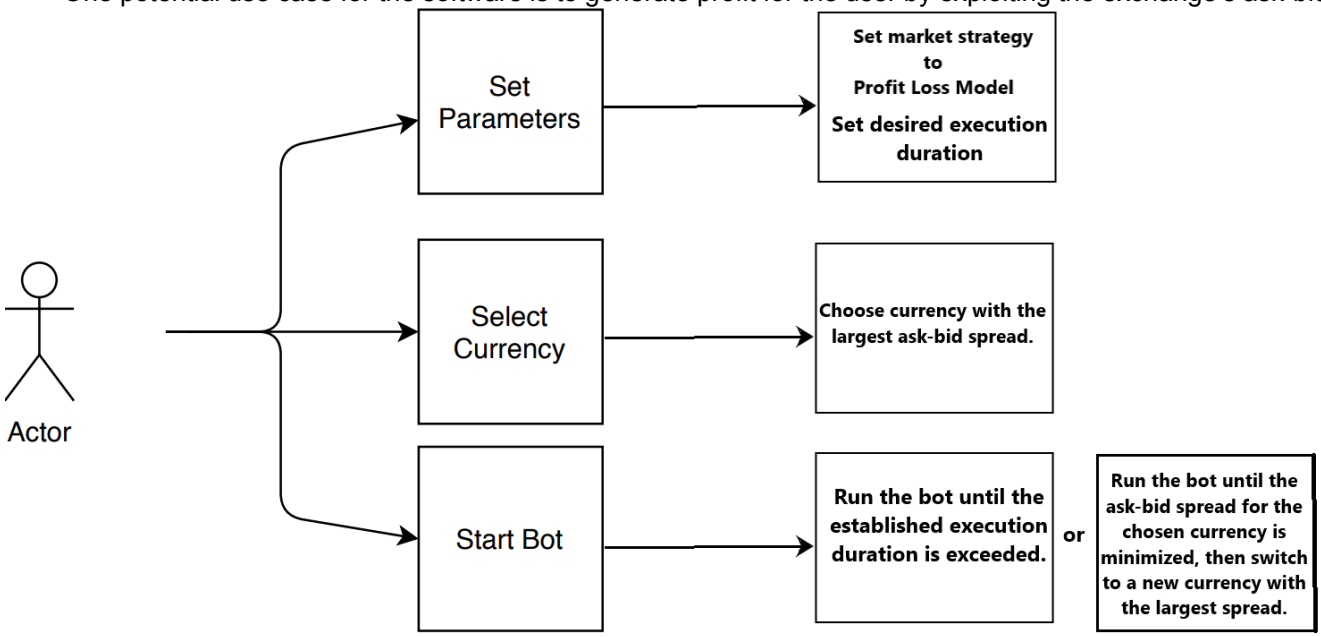
R1 Specification The software will be available as a web-based platform.
R2 Specification A user will be able to configure the bot's parameters via a graphical user interface.
R3 Specification All recorded statistics regarding the performance of the bot shall be available for view in a graphical format that is easily comprehensible for the user.
R4 Specification The users will access the service via the JavaScript/PHP based web-client. The server will run on a AWS EC2-Medium Ubuntu instance. The server client communication will use RabbitMQ or a similar Message Queuing Protocol for efficient communication.

4 Design

4.1 Use Cases

Use Case: Generate Profit
Pre Condition: The actor is required to have internet connection, some amount of financial assets, a BitMEX account, and a web-browser.

One potential use case for the software is to generate profit for the user by exploiting the exchange's ask-bid spread.

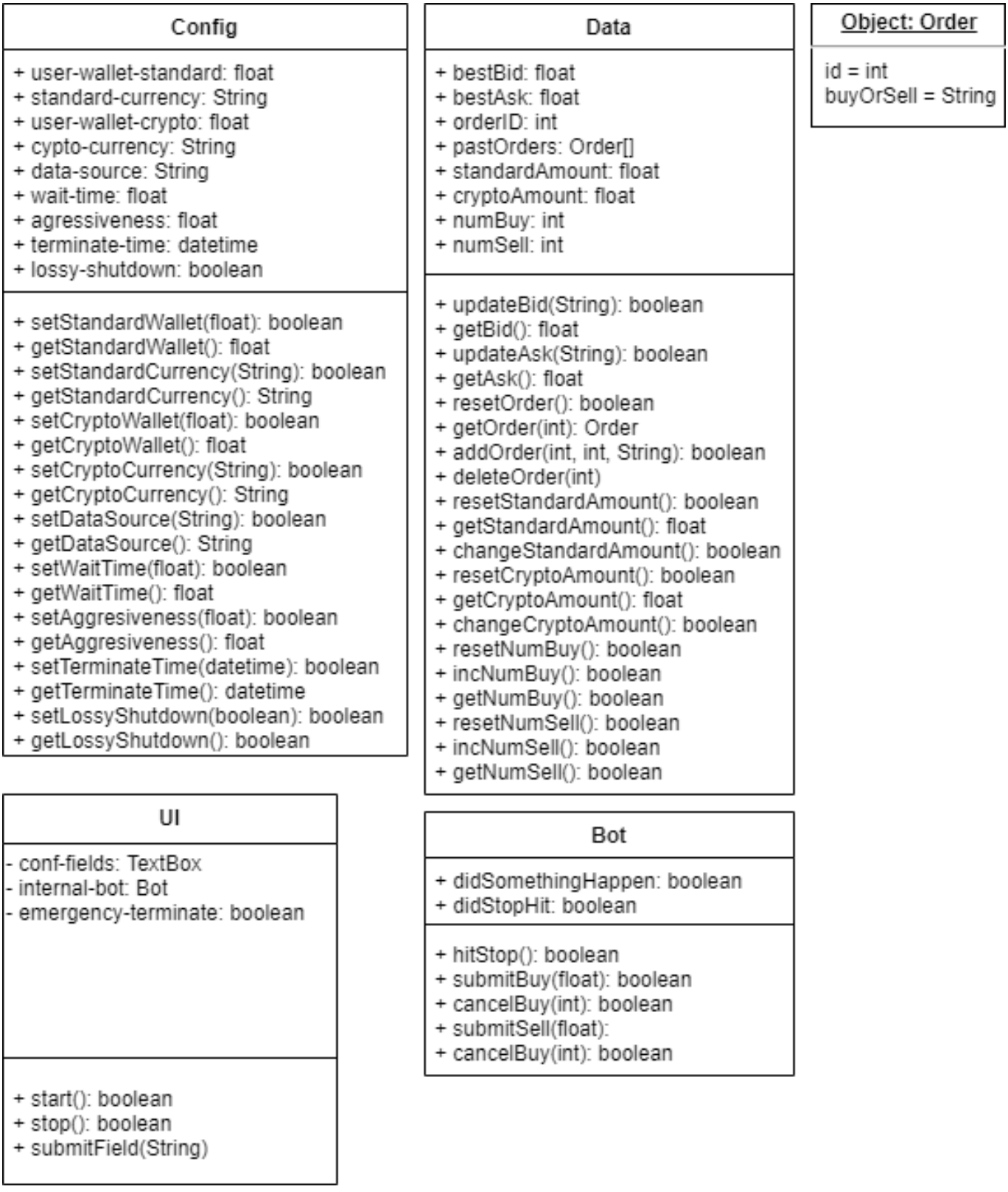


There is, however, a drawback of using the software in this way; namely, that while the profit loss model allows the user to maximize their profits by buying low and selling high, it also reduces market turnover (volume of assets traded) and thus does not provide maximum liquidity for the market. This use case would be preferable to clients who do not have any interest in the success of the exchange platform and instead are interested in personal profit.

Update in final version: Selecting the currencies is done at the same time as setting the rest of the parameters. Also there is no longer an option for manual switching of market strategies.

4.2 Architecture

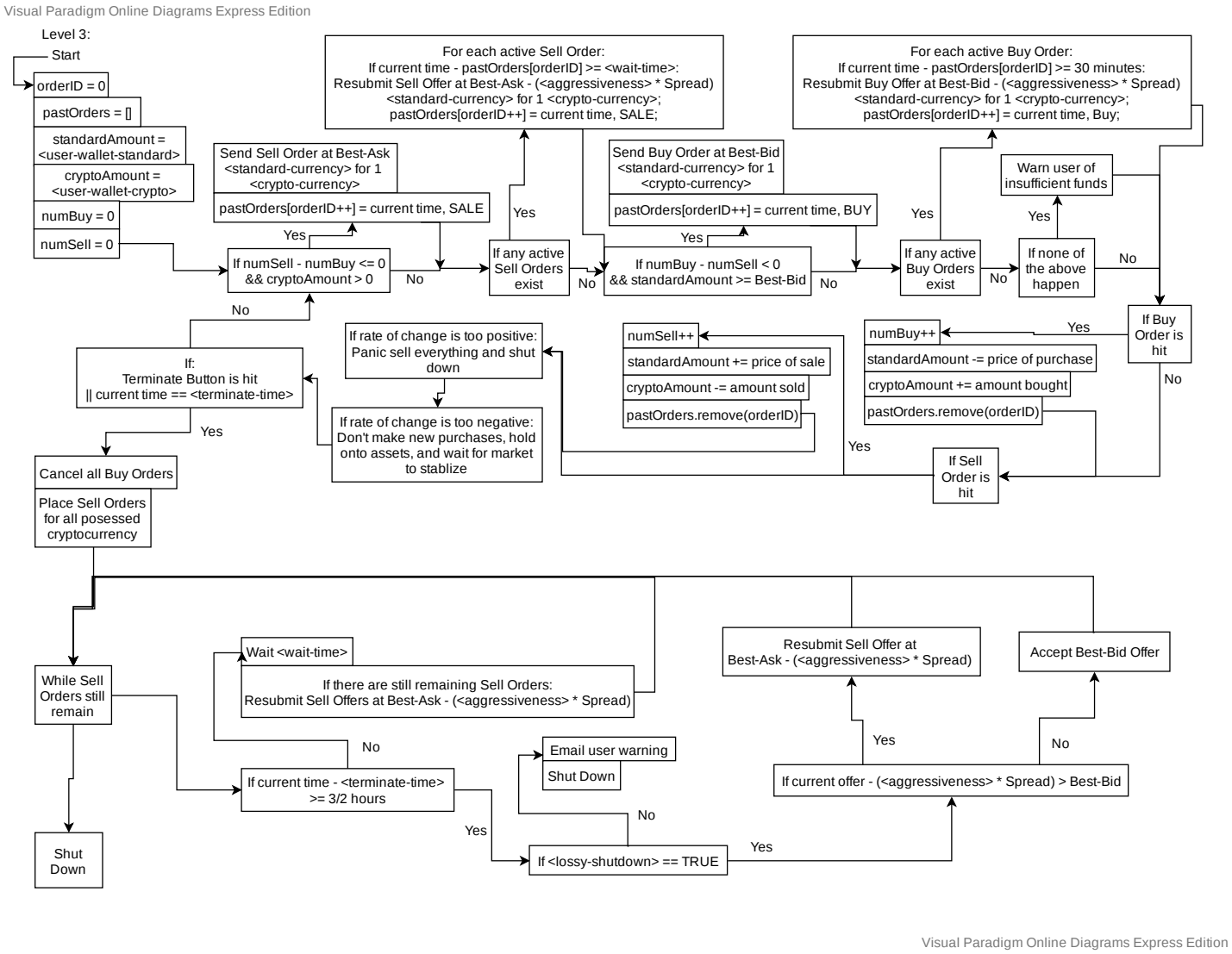
Below is the class diagram for the trading bot.



Update for final version: The class structure used changed greatly as we wrote the code, barely resembling this original plan. The UI and Config classes and the Order object have been removed entirely (UI is handled by the website instead of the bot, Config is now a json file held in Data, and Order is handled via BitMEX's own order tracking system). Along with that, the contents of Data and Bot have been massively altered, and new files have been added such as coinmarketcap and static instances.

4.3 Trading Logistics and Algorithm Overview

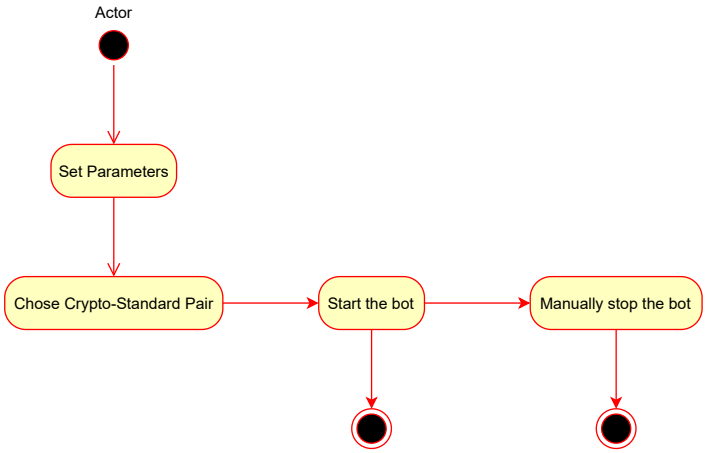
The following figure describes the trading algorithm which our bot will employ.



Update for final version: While coding the bot to fit with BitMEX's API, we ended up completely discarding this algorithm and rebuilding it from the ground up, making this chart basically useless.

4.4 Workflow

The workflow activity diagram is depicted below. The user will launch the program, configure the bot's settings via the GUI (this step is presented as the "Set Parameters" node in the activity diagram) and then start the bot by clicking the "Start" Button. Then the bot will begin placing purchase and sell orders in a manner determined by the market strategy the user selected; or, in the absence of a user selected market-strategy, the bot will autonomously determine the appropriate strategy based on the current market trend.



Update for final version: The plan for the user to manually stop the bot were thrown out, so that branch of the diagram is no longer relevant.

4.5 Technologies and Implementation Details

For this project, we chose the following technologies:

- **For the back end**: The back-end will be developed using Python, the BitMEX market_maker library and the BitMEX REST API, all being run on an AWS cloud server.
- **For the front end**: We will use JavaScript and PHP for front-end web-development.
- **Issue tracking and Project Management Software**: We will use Jira for bug tracking, task distribution, and general project management.

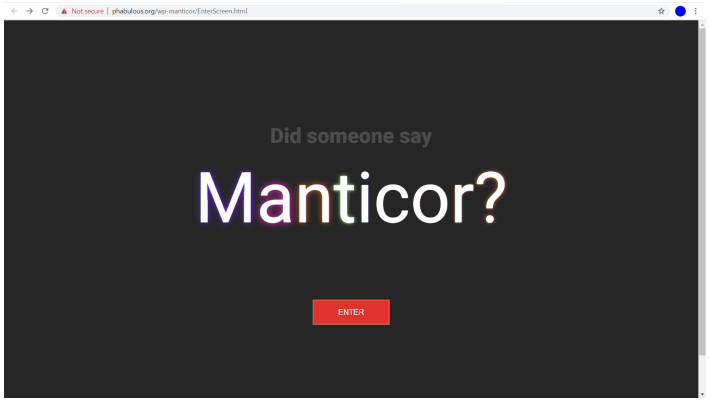
5 Timeframe and Milestones

- April, 4: Have the bot be able to connect to the BitMex platform, receive data, and submit buy and sell offers.
- April, 10 : Implement the logic described in section 4.3 with the BitMEX Python library.
- April, 18 : Integrate the data-aggregation from the CoinMarketCap website. At this point, we will simply have the bot acquire the best-ask and best-bid data from CoinMarketCap, but as a stretch goal the bot will analyze the market trend from CoinMarketCap and invest accordingly.

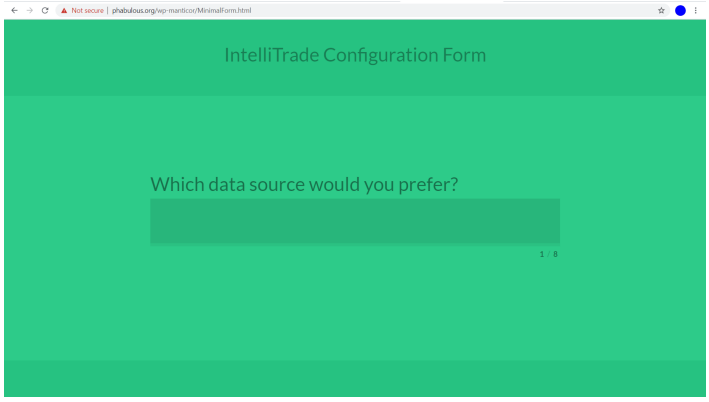
6 Supplemental Documents and Notes

- ”What is the Best Crypto Trading Bot in 2020?”
- `https://www.hodlbot.io/blog/ultimate-guide-on-crypto-trading-bots`
- ”Battle of the Bots: How Market Makers Fight It Out on Crypto Exchanges” :
- `https://medium.com/swlh/battle-of-the-bots-how-market-makers-fight-it-out-on-crypto-exchanges-2482eb9371`

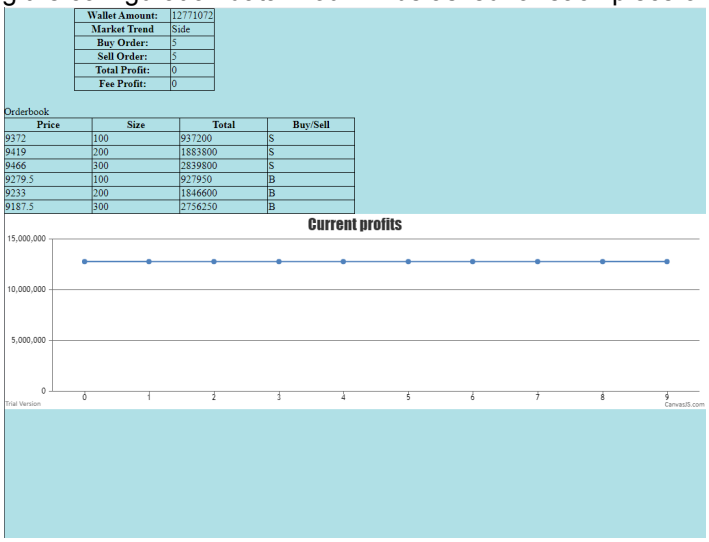
7 Product Screenshots



This is the front page our site you will see after clicking the url.



This is the form for entering the configuration data. You will be asked for each piece one at a time.



This is the UI for when trades are occurring.

8 User Documentation

NOTE: Before using our software, you must have an account at `https://testnet.bitmex.com/`, a wallet with a usable amount of cryptocurrency, and an API Key and API Secret (which you can get at `https://testnet.bitmex.com/app/apiKeys`).

- Step 1: Go to the url: `https://phabulous.org/wp-manticor/EnterScreen.html`
- Step 2: Complete the 2 CAPTCHAs (if you fail the first step, refresh the page)

- Step 3: Fill in the configurations when asked
- 3.a: Which data source would you prefer?: Unless you type in exactly ”CoinMarketCap it will pull from BitMex
- 3.b What currency pair would you like to trade with?: The available pairs are XBTUSD (Bitcoin), ADAM20 (Carnado), BCHM20 (Bitcoin Cash), EOSM20 (EOS), ETHXBT (Ethereum), LTCM20 (Litecoin), TRXM20 (Tron), and XRPUSD (Ripple).
- 3.c: How much money would you like to trade with?: Keep in mind typing in ”1” here means 1/10000000 currency.
- 3.d: What would you like to set minimum spread as?: This is how close the buy price and sell price of the currencies can be before the bot halts trading. Our recommended value would be around ”0.0002” (Note that this value is a percent so 1 = 100
- 3.e: What would you like to set the market low threshold at?: This determines how drastically the price of currency can shift before the bot will react. This value is a percent and should be negative. Our recommended value would be -0.5.
- 3.f: What would you like the relist threshold at?: This is how far off from the current price old orders can be before the bot resubmits them at a newer price. This value is a percent. Our recommended value is 0.01.
- 3.g: What would you like to set the aggressiveness at?: This is how quickly the bot will diminish the spread between the buy and sell prices for currency. This value is a percent. Our recommended value is 0.0005.
- 3.h: How long would you like the bot to run for?: The number you enter is the number of seconds.
- Step 3.5: If this is you first time using the bot, please allow ”Insecure Content” from the site by clicking on the lock symbol by the page url and going to ”Site Settings”.
- Step 4: If you see a pop-up, and that pop-up does not read an error, then the bot is now running and you can allow it to run automatically until the time you entered runs out.

9 Testing

Much of our testing was done manually. For the main bot, we did a lot of testing in PyCharm, where we wrote most of our code. The BitMEX API we used came with a logger function that would display given information on the terminal, so throughout our code we filled it with numerous logger outputs and saw what was output. When the bot was functional we let it run for a while, saw what kind of profits were gained, and made any adjustments as needed. When we had the bot connected to the site, we similarly just allowed the bot to run and saw what kind of profits were produced. Testing a program like this can be very tricky and annoying, as the bot can only perform using what the market gives it, and the markets are very fickle. Sometimes the market’s are just not fertile enough for trading, resulting in large spans of time that we had the bot running and we just had to sit and wait for even just a single of our orders to get a hit. This made attempting to test for specific scenarios very difficult. Overall though from what we have seen we are quite confident in the performance of our bot.

10 Conclusion

At the beginning of this project many of those on our team knew little about cryptocurrency and or the industry of trading and market making, but in the end we were able to pull together and make a quality product. Early in development we were shown BitMEX’s own market making bot and we were told to find a way to improve it. Despite our teams admitted lack of knowledge in the field of economics, we were able to find a new way to improve the pre-made bot. In our attempts to get the pre-made bot working to get a base-line, we found it very difficult to work with, the mechanics of which were a tangled mess that took us hours to just get running let alone understand to a point we could recreate. With our newer bot, we have not only made the inner works far more organized and understandable, but we have also connected it to a convenient web-based interface, unlike the old one which required the use of a python terminal. Unlike the previous bot, ours will be usable by those not at all versed in programming, and those who are versed should have a far easier time expanding and iterating upon, while still maintaining the previous bot’s level of performance. As happy with our work as we are, we do admit there is some features that had to be cut due to time constraints, such as: the ability to stop whenever upon a user’s command, the ability to cross reference our local and BitMex’s data to ensure bot sanity, and a customizable shutdown procedure that would determine how the user’s leftover trades would be handled upon shutdown.