# Crypto-trade-bot requirements

By: Blake Hachen

# Description

- **Problem:** The current state of the cryptocurrency market makes trading the traditional way somewhat dangerous. There have been many times where nightly sales have caused me great losses.
- **Solution:** A self-made secure bot that let's me trade any crypto asset automatically across multiple exchanges

# Requirements

- Trading bot
  - models for: order, price, exchange, dataset, currency
  - API keys for various exchanges: binance, coinbase, coingecko, etc.
- Features
  - Import backtestable datasets
  - Config file (.env) that will let the user change attributes of the bot
  - Testing mode and live trading mode, testing mode will work off the imported datasets
  - SQL database (PostgreSQL) this will let us POST our api requests to a database maintaining one Rest API for all exchange APIs

# Tools Used

- Python
  - Most exchanges have site-packages that can request information from the api
  - I will use the python requests module to create the Rest API
- PostgreSQL
  - I want to use PostgreSQL to upload external requests to an internal database of various datasets
- Node.JS or DockerFile?
  - These are also options being considered for internal API requests

# Data source possibilites

- CoinBase
- Coingecko
- Binance
- All exchanges should bring me similar information regarding data of the crypto asset

# Config File

- The config file will act as a starting point for the user
- It will have variables: API KEY, API SECRET, START, END, CANDLE INTERVAL, API URL, API URI, and any other editable attributes
- The purpose of the config file is to provide versatility to the trading bot

# Trading

- The bot will trade normally on any dataset whether it is a testing dataset or live trading session. There needs to be a variable within the config file to determine whether the user wants to live trade or trade over test data.
- There will be multiple strategies that the user can specify in the config file. These strategies will be used by the bot. One strategy that I may implement is the arbitrage strategy.

# Database

- The goal of the database is to build on it. Everytime a dataset is imported it will be moved into the api/datasets portion of the database. Api will be the central database. This will make it easy to do internal api calls. Inside the dataset will be information regarding the currency and asset being traded and attributes of the asset that will determine the trade.

# Timeline

- June
  - Choose Project x
  - Requirements x
  - Create Skeleton of bot (models) x
  - Finish bot
- July
  - Import datasets functionality
  - Backtest datasets functionality
  - Test, fix bugs
  - Final Presentation

# Thank You!