Question 1:

1.1.1) The cost of a transaction on the Ethereum blockchain, is expressed by a unit called "Gas". Gas can be defined as a unit, which value is based on the amount of computational effort required to perform actions on the Ethereum blockchain. Considering its role in the Ethereum blockchain, it acts as a “fuel” for doing transactions. Every operation on the Ethereum blockchain has an associated "Gas" cost (i.e. adding numbers, getting the balance of an account or doing transactions). This cost remains constant dependant on the operation that needs to occur. Smart contracts for example, can combine many of these operations, thus causing it to cost even more Gas. To be able to determine the cost of a transaction, the Gas cost needs to be calculated, using the Gas price. Gas price is measured in "Gwei" which is essentially a smaller unit than "ETH", where one Gwei equals 1E-9 ETH. Time varying factors that influence the price of Gas, include the speed at which one wish a transaction to occur and network activity. Thus, depending on how fast one wants the transaction to occur, the gas price for that transaction will be determined. The amount of Gas needed to perform a transaction, is multiplied by the Gas price (expressed in Gwei), the result being the transaction cost. The reason for having a separate transactional unit (i.e. Gas) for doing business, is to decouple it from the price of ETH. This essentially means, given the network activity on the Ethereum blockchain remains constant, an increase in the price of ETH does not influence the cost of transactions. If the network activity on the Ethereum blockchain, however increases, the cost of a transaction will increase. The reason for this is due to the "build in auction" mechanism (i.e. time-varying factor), where users that want their transactions to be picked up by "miners", first need to "outbid" other people for space in the block being mined by miners (i.e. one user pays more Gwei for Gas than another user, driving the gas price higher). This is especially the case when users want to preform urgent transactions. Another important thing to note about Gas, is that each transaction has a Gas limit that must be equal or higher to the anticipated amount of computation needed to successfully execute a particular transaction. If there is to little Gas to perform a transaction, it is reverted (i.e. out of Gas is reached and you wil still pay for the transaction). If a surplus Gas is left after doing a transaction, the remaining Gas is converted to ETH and given back to the sender.

1.1.2) Special consideration needs to be given in cases when there is increased network activity (i.e. high volume buy and sell-offs) causing "limited block" space on the Ethereum network. These are the times when the markets are especially volatile. Each Ethereum block has a maximum size (i.e. limited to the amount of data that can be included in a block), this limit is set to the maximum block size denominated in Gas. As mentioned in 1.1, an operation on the Ethereum blockchain can have different Gas usages based on the complexity of the operation (e.g. smart contracts), meaning the maximum number of transactions included in a given block is not consistent. Since Ethereum Gas fees are paid to Ethereum miners, the miners are incentivised to give higher priority to transactions with the highest Gas price. If a transaction's Gas price is inadequate in any way, there will be no guarantee to the sender of the transaction, that the block will be mined. In this instance, a transaction might only be confirmed until a later block is mined. This is especially the case when blocks are always "full" – causing transactions to be confirmed until much later. Full blocks thus escalate the Gas price, as transaction senders are bidding for scarce block space. Thus, when block space

demand spikes suddenly or unexpectedly, sharp increases in the Gas price is caused, further affecting transaction fees.

1.1.3) In a conventional setting, traditional "market makers" provide liquidity so that investors can buy and sell an asset close to the publicly quoted price. This means the market maker will match the buyer to the seller. For any trade to happen, the buy and sell order need to match. All orders sit within an order book. Market makers thus finds buyers and sellers for assets to be bought and sold on an exchange. Traditionally, market makers in the financial industry have always been large financial organizations like banks. These organizations take on the risk associated in buying and selling these assets to investors and charge what is known as a “spread” on each asset covered, trying to mitigate the risk associated with the transaction. This means the seller is quoted a “bid” price slightly lower than the market price, and the buyer quoted an “ask” price that is slightly higher than the market price. Automated market makers (AMM) are a type of decentralized exchange protocol, based on smart contracts, that relies on a mathematical formula and liquidity pools rather than the traditional order book system. Digital assets can thus be traded, using an automated algorithm against the liquidity held in liquidity pools, meaning digital assets can be swapped at any time. The exchange price is determined algorithmically by the AMM. In the traditional sense, "market makers” provide liquidity through orders, “makers” wait for "takers" to agree to the order. With AMM protocols, there are no “makers”, thus there are no prior orders in the system only takers looking to exchange a specific cryptocurrency pair. This means you can trade cryptocurrency 24 hours a day. Also, there is no centralized entity matching buyers to sellers which removes the need for a centralized exchange.

1.1.4) A "flash loan" is a feature that makes it possible for one to borrow any amount of available assets from a designated smart contract pool (e.g. such as AAVE) with no collateral. Flash loans have one important caveat however, and that is the loan needs to be repaid within the same transactional block it was created in, otherwise the transaction is reverted on the Ethereum blockchain (i.e. as flash loans are "atomic" if the entire process is not completed instantly, the loan will not go through). There is usually a fixed cost associated for using flash loans, AAVE contracts for example require the borrower to return the loaned amount with a fixed fee of 0.09% of the initial loan. This is useful as one can use it for things such as "arbitrage" (i.e. buying and selling assets on different markets and making money out of the price difference) or debt refinancing (i.e. users take out a loan to pay off another loan on an exchange that might have lower interest rates – essentially switching between competitive lending platforms). The lender is guaranteed the loan is repaid as the smart contract needs to complete a set of "atomic" operations that must executed for the loan to occur otherwise the transaction is reversed.

1.2) A "stable coin" is a type of digital asset whose value is tied to an outside asset such as gold or U.S. dollar (i.e. a fiat currency) to stabilize the price. In contrast with digital assets such as cryptocurrencies, whose price can be quite volatile and unpredictable, when compared to assets such as Gold or the U.S Dollar. One of the most popular use cases of stable coins, is using it to buy and sell cryptocurrencies on exchanges where users are not able to exchange their crypto for fiat currencies due to regulation. Another advantage is that you can move funds between exchanges relatively quickly – due to crypto transactions being faster than fiat transactions (i.e. this is especially useful for arbitrage). Other benefits of using stable coins, include transparency, security, fast transactions, privacy, and low fees, all while leveraging the "stability" provided by the fiat currency. Stable coins can be constructed in different ways namely "collateralized" stable coins or "non-collateralized" stable coins. Collateralized stable coins are coins whose value is pegged by specific collateral, the three main groups been: Fiat-backed stable coins, Asset-backed stable coins and Crypto-backed stable coins. Fiat-backed stable coins, are pegged to the value of a fiat currency such as the US Dollar, an example of such a coin is Tether (USDT). Asset-backed stable coins are backed by other assets excluding fiat and cryptocurrency, example of such as might include oil, precious metals, real estate, etc. Crypto-backed stable coins are supported by cryptocurrencies, they use protocols to ensure the value does not vary with the token that it backs. DAI is an example of such a token that is a crypto-backed stable coin, supported by Ether and pegged to a US Dollar value, maintaining its price through "Maker Smart Contracts" that destroys and creates MKR tokens, according to fluctuations in the ETH price, keeping its value stable. Non-collateralized stable coins (e.g. a US Dollar-backed base coin), algorithmically use a consensus mechanism, like that of a central bank to increase or decrease the supply of tokens on a needed basis, to keep the price of the token constant. This functionality can be implemented on a smart contract, using a decentralized platform that enables it to execute automatically.

1.3) A basic futures contract, is a contract where you agree to buy or sell an asset at a specific date in the future. Using bitcoin as an example, a buyer would agree to buy bitcoin at specific date at a predetermined price, in the future and the seller is obligated to sell at the price specified in the contract. If the price of bitcoin drops the day the contract expires, the buyer is forced to buy bitcoin at a higher price, thus the seller profits. The same is true the other way around, where the price of bitcoin increases, and the seller forced to sell it at a lower price thus losing money. A perpetual future contract, is a contract that does not have a settlement date, the contract can thus be held as long the minimum requirements of the contract are met. The three central concepts behind perpetual future contracts include: the initial/maintenance margin, liquidation, and an insurance fund. The initial margin, is the margin that is committed by the buyer of the contract to open a leveraged position. The maintenance margin, is the minimum balance required for the contract buyer to keep the position open (i.e. maintenance margin is dynamic and changes with market price of the asset). If the balance of the account drops below a certain level, the buyer needs to make a margin call and add more funds to their account to act as collateral for the leveraged position. If the buyer is not able to provide collateral for whatever reason, the leveraged position is closed and the collateral in the account liquidated. The insurance fund, is used to prevent the balance of a loosing trade to drop below zero, simultaneously ensuring winning traders get their profits. It is important to note that futures contracts are traded independently from the underlying asset price. If the market expects the underlying asset to rise in value, the futures contract will be priced higher than the asset. As a normal futures contract has an expiration date, when that date gets closer, the price

difference between the futures contract and the assets current market price (i.e. spot price), will converge and the difference become close to zero. This convergence in price between the asset's spot price and the price of the future contract, is where an arbitrage opportunity lies. If one, for example, opens a short position on the perpetual market at $3000 and buy bitcoin at a spot price of $2000 the same day, one waits for the price to converge the next day where bitcoin might be trading at $4000, then close the arbitrage. The perpetual result is -$1000 and spot result $2000 netting one a profit of $1000.

Question 2:

This has been heavily commented in code (i.e. following comments will give intuition to how the algorithm works). Also check the requirement.txt files and “how to run” “.txt” file on github repo.

Question 3:

3.1) The Sharpe ratio can be defined as the average return of an investment when compared to a risk-free investment per unit of volatility (i.e. a measure of price fluctuations of an underlying asset).

From data extracted between 24 May 2020 and 24 May 2021 (i.e. extracted from ftx in 5 minute intervals) the following Sharpe ratios where calculated:

Annualized Sharpe ratio BTC/USD: 23.616

Annualized Sharpe ratio ETH/USD: 18.118

Generally, a Sharpe ratio above 1 is considered good because the investment is showing higher annual return relative to its volatility. Both BTC/USD and ETH/USD show Sharpe ratios well above 1 indicating that these digital assets have very good returns relative to their volatility. However, BTC/USD shows an even higher Sharpe ratio compared to ETH/USD, indicating BTC/USD to be the better investment choice -- according to this metric. The problem with the metric, however, is that it assumes that the returns are normally distributed, which is not the case in financial markets as there might be large sudden drops or spikes in price. Another problem with this ratio is one can cherry pick data if you make the time interval large enough (i.e. meaning a larger standard deviation in the denominator of the Sharpe ratio formula). Measuring the Sharpe ratio over a long enough time interval, meaning the standard deviation over a year in smaller time intervals (i.e. minutes, hours and days) is generally larger than the standard deviation (i.e. weeks and months) over the same period in longer time intervals, does alter your results as day to day volatility is typically more than day to month of day to week. Essentially, increasing or lowering the ratio, makes it show better results.

The Sortino ratio is an adaptation of the Sharpe ratio, differentiating itself as it only considers downside volatility instead of overall volatility. It thus measures the standard deviation of an asset’s negative returns rather than the overall portfolio returns; this is relevant since investors usually only worry about downside risk. Sortino ratio thus only considers downside risk. This is useful since it gives a better view of the portfolio or asset being tracked risk-adjusted performance, making it a useful measure of bad risk.

From data extracted between 24 May 2020 and 24 May 2021 (i.e. extracted from ftx in 5 minute intervals) the following Sortino ratios where calculated:

Annualized Sortino ratio BTC/USD: 0.00028

Annualized Sortino ratio ETH/USD: 0.00049

Like the Sharpe ratio a higher Sortino ratio indicates the investment is earning more per unit bad risk taken. Both BTC/USD and ETH/USD have extremely low Sortino ratios (i.e. basically zero) -- ETH/USD showing slightly better performance -- indicating that the investor is not really being awarded for the additional risk taken. This suggest there is still a large downside risk that needs to be considered, which makes sense given the volatility of crypto markets.

3.2) The information ratio is a measure of an investment or portfolio's returns beyond that of a specific benchmark. We compare BTC/USD returns and compare it to our benchmark returns ETH/USD.

From data extracted between 24 May 2020 and 24 May 2021 (i.e. extracted from ftx in 5 minute intervals) the following information ratio was calculated:

Annualized information ratio between BTC/USD and ETH/USD: 23.765

A higher information ratio indicates that BTC/USD in this case produces better consistent returns considering the risk-adjustment than the benchmark ETH/USD. This means less risk is taken and more consistency in performance relative the benchmark.

3.3) The tracking error can be used as a measure of financial performance that determines the difference between the return fluctuation of an investment portfolio and the return fluctuation of a chosen benchmark (i.e. how well an investment portfolio "tracks" a specific benchmark). This is measured using the standard deviations between the difference between the portfolio and the chosen benchmark. Lower tracking errors means the performance of the portfolio is close to that of the benchmark.

From data extracted between 24 May 2020 and 24 May 2021 (i.e. extracted from ftx in 5 minute intervals) the following tracking error was calculated:

Tracking error for Bitcoin vs Bitcoin perpetual futures: 0.0118

Tracking error for Ethereum vs Ethereum perpetual futures: 0.0158

For the 7-day rolling window:

7 day rolling window tracking error Bitcoin vs Bitcoin perpetual futures: 0.00121.

7 day rolling window tracking error Ethereum vs Ethereum perpetual futures: 0.0016.

Considering the tracking error for BTC and BTC perpetual futures on a 5 minute yearly basis. The tracking error is low indicating that their prices are very closely related to each other. The same is true for Ethereum showing similar trends to BTC. However, considering an average 7 day rolling window of BTC, ETH and their perpetual future, their price converges making the tracking error even smaller (i.e. almost 10 times smaller in this case).

Question 4:

A simple algorithm was used where the average low, high and spot prices was used and compared with each other every 12 minutes. Streaming live BTC data via websockets using the coin base pro API. If the average price is higher than it was 12minute ago we sell, if it is lower, we buy, based on this input data is written to mysql database using SQLAlchemy and the transactions recorded and made available using the following endpoints. Also, the flask server and websocket streaming are run on different threads.

Endpoints: where created on google cloud to access MySQL database:

http://34.69.210.197:5000/current_positions :

Show account ID and the BTC and USD values currently available in these accounts (Note “acc_id” is the account ID is different for USD and BTC and timestamp used to keep track of transactions):


```
[
    {
        "acc_id": "4363e394-d40d-4968-a929-f1403d95402e",
        "available": "28.93447683",
        "balance": "28.9344768300000000",
        "currency": "BTC",
        "hold": "0.0000000000000000",
        "id": 1,
        "time": "1622188140.3653705"
    },
    {
        "acc_id": "8be78d37-2f52-426b-a80d-c34595e81a35",
        "available": "816826.40991778578034",
        "balance": "816826.4099177857803400",
        "currency": "USD",
        "hold": "0.0000000000000000",
        "id": 2,
        "time": "1622188140.3750813"
    }
]
```


http://34.69.210.197:5000/balances :

Shows the rolling balance of the account as bitcoin is bought and sold:


```
{
    "acc_id": "4363e394-d40d-4968-a929-f1403d95402e",
    "available": "30.89579914",
    "balance": "30.8957991400000000",
    "currency": "BTC",
```

```
    "hold": "0.0000000000000000",
    "id": 1,
    "time": "1622149765.3966725"
},
{
    "acc_id": "8be78d37-2f52-426b-a80d-c34595e81a35",
    "available": "740903.82964309575624",
    "balance": "740903.8296430957562400",
    "currency": "USD",
    "hold": "0.0000000000000000",
    "id": 2,
    "time": "1622149765.4043117"
},
{
    "acc_id": "4363e394-d40d-4968-a929-f1403d95402e",
    "available": "30.89318722",
    "balance": "30.8931872200000000",
    "currency": "BTC",
    "hold": "0.0000000000000000",
    "id": 3,
    "time": "1622149862.835894"
},
{
    "acc_id": "8be78d37-2f52-426b-a80d-c34595e81a35",
    "available": "741003.64950798259848",
    "balance": "741003.6495079825984800",
    "currency": "USD",
    "hold": "0.0000000000000000",
    "id": 4,
    "time": "1622149862.8409488"
},
{
    "acc_id": "4363e394-d40d-4968-a929-f1403d95402e",
    "available": "30.8905816",
    "balance": "30.8905816000000000",
    "currency": "BTC",
    "hold": "0.0000000000000000",
    "id": 5,
    "time": "1622150040.4574816"
}
```


http://34.69.210.197:5000/balance/<id> :

This is just the specific balance of account given the ID (i.e. in this case it is “2”):


```
{
    "acc_id": "8be78d37-2f52-426b-a80d-c34595e81a35",
    "available": "740903.82964309575624",
    "balance": "740903.8296430957562400",
```

```
    "currency": "USD",
    "hold": "0.0000000000000000",
    "id": 2,
    "time": "1622149765.4043117"
}
```


http://34.69.210.197:5000/orders :

These are the “buy” and “sell” orders placed using our simple algorithm (e.g. the smallest id value in this case “1“ was our first buy recorded):


```
{
"amount": "100",
"id": 1,
"product_id": "BTC-USD",
"side": "sell"
},
{
"amount": "100",
"id": 2,
"product_id": "BTC-USD",
"side": "sell"
},
{
"amount": "99.82032341",
"id": 3,
"product_id": "BTC-USD",
"side": "buy"
},
{
"amount": "100",
"id": 4,
"product_id": "BTC-USD",
"side": "sell"
},
{
"amount": "100",
"id": 5,
"product_id": "BTC-USD",
"side": "sell"
}
```