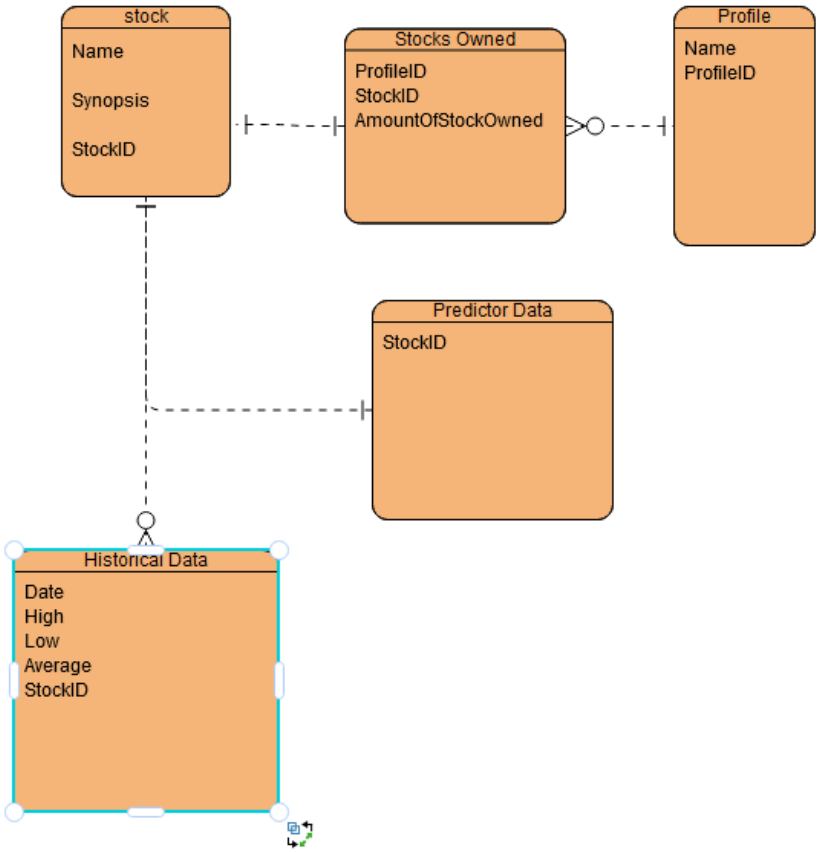
AlgoTrader Plan

We want to implement a solution that can stream live data and store historical data of stocks, and potentially crypto. The solution will run an algorithm that allows for backtesting on historical data, and runs on live data to predict whether an asset is worth buying, selling, holding, etc.

Live data could consist of sources such as stock prices, and with a little bit of exploration we could implement a natural language model to take in blog/ news article sources.

Plan for algorithmic trader implementation

https://online.visual-paradigm.com/app/diagrams/#diagram:proj=0&type=ERDiagram&width=11&height=8.5&unit=inch



Requirement directions:

Objective Definition:

- Clearly define the goals and objectives of your algorithmic trading strategy. Are you aiming for profit maximization, risk mitigation, or a combination of both?
- The goal of this project is to practice project management, planning, and development. The outcome will be a hostable project that automates market scanning, and projecting the data to predict outcomes of the market.
    - Create a plan for project steps
    - Create user stories for each requirement
    - Develop each step with weekly sprints
        - Database
        - Data retrieval
        - Programming loop, how the program will run continuously
        - Data processing
        - UI, this will be what the output will look like, not necessarily a graphical interface, but whether we use email, or display data in terminal.
        - Deployment, containerise the solution for easy to deployment.

Data Sources:

- Identify and access reliable financial data sources. This could include historical price data, real-time market data, economic indicators, news feeds, and more.
- We will need data sources for stocks, and crypto
    - https://www.interactivebrokers.com/, this is a stock platform that also has an API for making stock transactions. It also has stock data. Will require signing up and creating a profile, (requires ID and proof of address)
    - https://www.livecoinwatch.com/tools/api, potential for crypto
    - https://www.binance.com/en-AU/binance-api, binance is the main hub for trading crypto, as far as trust goes this is the best exchange. They also stream data from the API among much other functionality.
    - Blogs will take some research, but will become an important tool for an algorithm.

Data Processing:

- Develop a robust mechanism to clean, preprocess, and organize the data. Ensure that your system can handle missing data, outliers, and other anomalies effectively.
- We need to try to stick to industry standards in terms of project layout. Create unit tests when needed, and test throughput.

Feature Selection:

- Choose relevant features for your trading strategy. Common features include price trends, moving averages, volatility measures, technical indicators, and economic indicators.
- TBD, have the basic gathering of data and storage completed first.

Strategy Development:

- Implement and backtest your trading strategy. This involves coding the logic that will decide when to buy, sell, or hold assets based on the selected features and indicators.
- TBD, this will be an ongoing process as there is no correct answer for what a good algorithm is.

Risk Management:

- Integrate risk management techniques into your algorithm. This includes setting position sizes, stop-loss orders, and other risk control measures to protect against significant losses.

Simulation and Backtesting:

- Develop a simulation environment to backtest your strategy using historical data. Ensure that your algorithm performs well across different market conditions and is not overfitting to specific datasets.

Transaction Costs and Slippage:

- Consider transaction costs and slippage in your backtesting. These factors can significantly impact the performance of your algorithm in a real trading environment.
- Definitely important to consider fees and tax when doing transactions in a mock environment. For now we will only progress with a mock environment.

Live Trading Interface:

- Develop a live trading interface that can connect to brokerage APIs or other trading platforms. Ensure that your algorithm can execute trades in real-time.
- Mikael loves doing front-end, not my problem ;)

Monitoring and Logging:

- Implement robust monitoring and logging mechanisms. Track the performance of your algorithm in real-time, log important events, and set up alerts for potential issues.
- This will be important if it ever went live, whether notifications come through an app, email, or text. I expect an app that interfaces with a DB will be the most effective way for us to send notifications, this gives us space to develop a UI as well. Problem is with deploying apps to apple and android, its quite an involved process and also requires Mac OS for development on apple devices.

Compliance and Regulation:

- Be aware of and adhere to any legal and regulatory requirements related to algorithmic trading in your jurisdiction.
- I believe we would simply be taxable as day traders in this regard, otherwise it should be fine.

Security:

- Implement security measures to protect your algorithm, sensitive data, and trading accounts. Use secure authentication and encryption protocols.
- The platform if it ever went live should definitely use an authenticator. As it would have access to your capital.

Documentation:

- Create comprehensive documentation for your algorithm, including the strategy logic, data sources, and technical details. This will be crucial for future maintenance and collaboration.
- Documentation is as important as the project itself in terms of learning for us. When working for clients, or companies, they will want this. Try to maintain documentation as you go.

Testing and Validation:

- Conduct thorough testing and validation of your algorithm in different market scenarios. Consider stress testing to evaluate its robustness.

Continuous Improvement:

- Plan for continuous improvement by regularly reviewing and updating your algorithm. Stay informed about market trends and adjust your strategy accordingly.
- It would be cool if we could implement some CI/CD. (Continuous integration/Continuous Deployment) this means when you make a code change and push it to your repository (using git), it will automatically be tested, then integrated and deployed, this shows any errors immediately.

Backup and Recovery:

- Establish backup and recovery procedures to ensure the integrity of your algorithm and data in case of system failures or other emergencies.
- Super important, how will our database be resilient? We can have a database to store transactions, we need to have a regular backup, this will have to be hot as we want to do this probably daily maybe more, losing data can be brutal especially if it involves money invested.

Performance Metrics:

- Define and monitor key performance metrics to evaluate the success of your algorithm. This could include measures such as Sharpe ratio, maximum drawdown, and annualized returns.
- The obvious consideration would be are we making money, but apparently there are better ways to determine if a strategy is good. More research required.

Ethical Considerations:

- Be mindful of ethical considerations related to algorithmic trading. Avoid market manipulation, insider trading, and other unethical practices.
- This is probably worth considering on a personal level, as if you work for a publicly listed company, or are affiliated with them, you must be careful around them.

List of requirements:

DB AND DATA REQUIREMENTS

- I would like to stream stock data into my project
- I would like to have a historical store of data in my project
- I would like a db with historical stores of data on stocks
- I would like to stream crypto data into my project
- I would like to have a historical store of crypto data in my project
- I would like a db with historical stores of data on crypto
- I would like to store data on currently held stocks/crypto

Processing and general runtime requirements

- I would like to be able to get specific stock data between specific date ranges
- I would like to generate day start and day end high/low
-

Cloud/Infrastructure Requirements

- I would like to deploy to AWS
-

Containerization

Formatted Requirements for development:

Kanban board: https://bybalgotrader.atlassian.net/jira/software/projects/KAN/boards/1
Github: https://github.com/Doggydo123/BYBAlgoTrader

Need to decide on services:

- Python for the general computation
- Vue JS for frontend UI, flutter for a mobile frontend (universal for apple and android)
- We could implement a .Net backend if we wanted, unsure what the use case of this is though.
- SQL server or something similar which will be hosted on AWS
- We will probably host our program on an ubuntu VM, unless we can strip it down.

Planning Poker:

Maintaining Best Practice:

This will be a first direction for a python project:

https://dagster.io/blog/python-project-best-practices

I have never deployed a python project, so I am still to learn what goes into running the python code autonomously on a server and interacting with the required services, e.g connecting with azure or AWS.