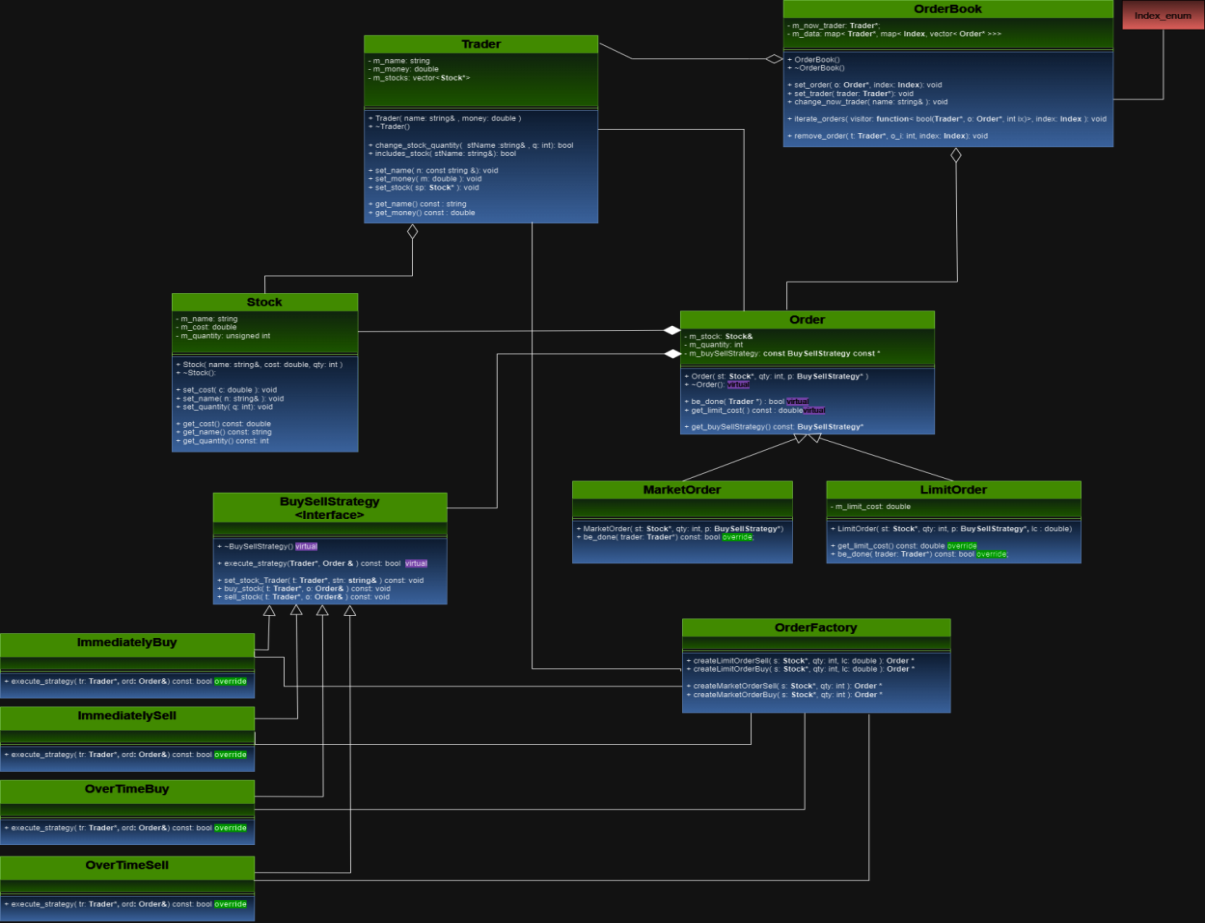
Stock Trading System

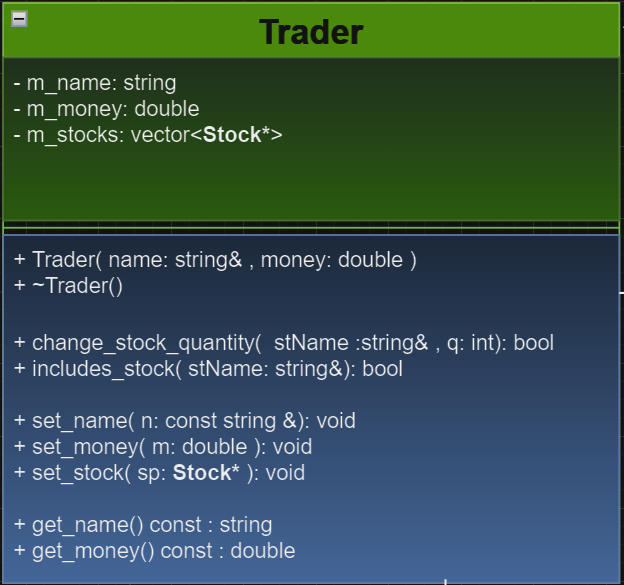
* **Introduction**

The task was to create a system in which many traders can create orders to buy or sell stocks. All orders are stored in the Order Book. Buy and sale orders must be executed in a multithreading way.

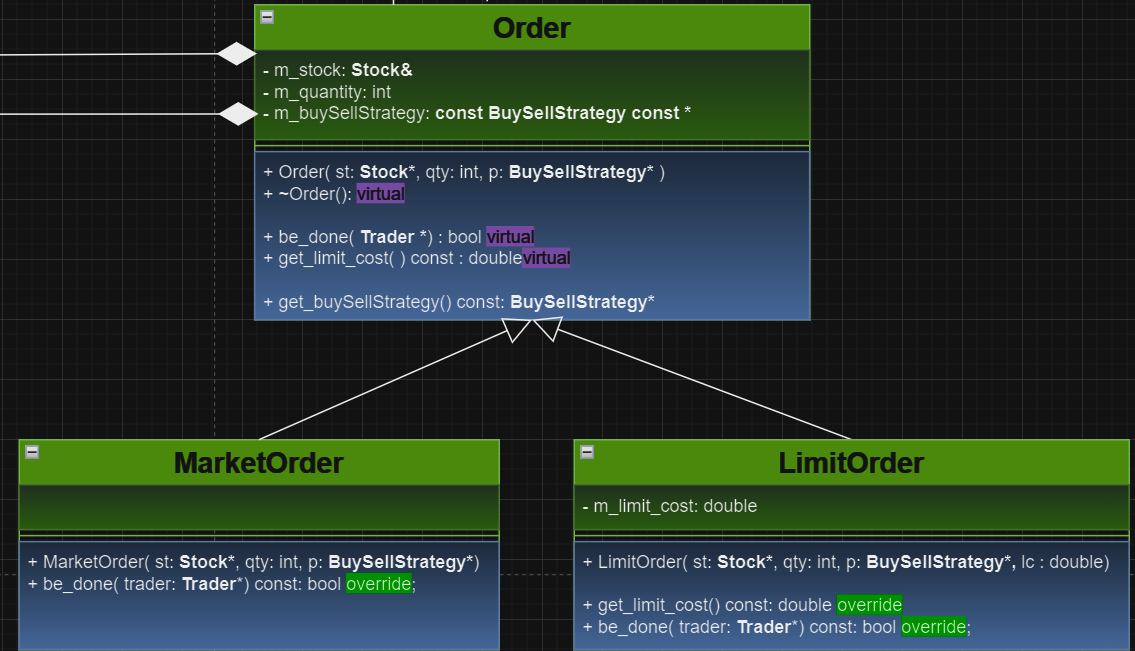
After receiving the assignment, I compiled a UML project. I have identified for each object the characteristics that are important for my task. [URL](https://drive.google.com/file/d/1LLLy_XP7RcJd2PpRpPIRQ41PEWYBWjn8/view?usp=sharing)



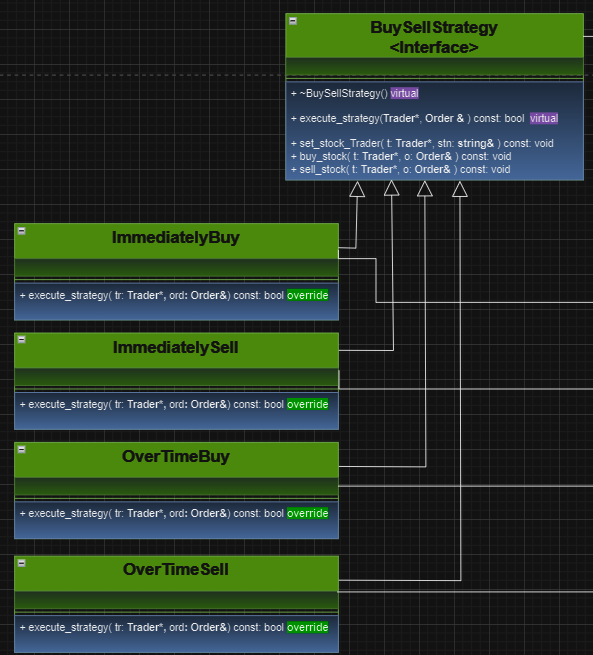
Trader class has a Stocks Vector, where its stocks are stored, and provides functions for increasing or decreasing stocks.



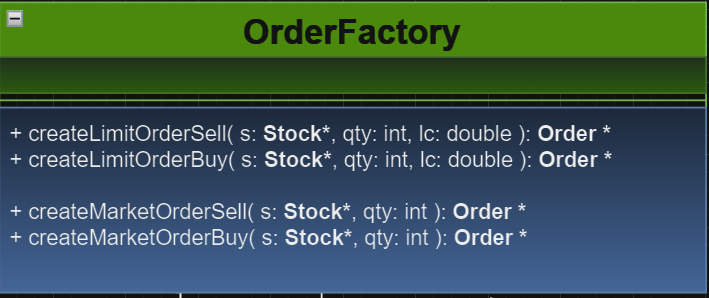
Order is an abstract concept. While designing, Order is an abstract class to which MarketOrder and LimitOrder extend. This is very flexible, since in the future there may be other Orders that can extend the abstract class Order.



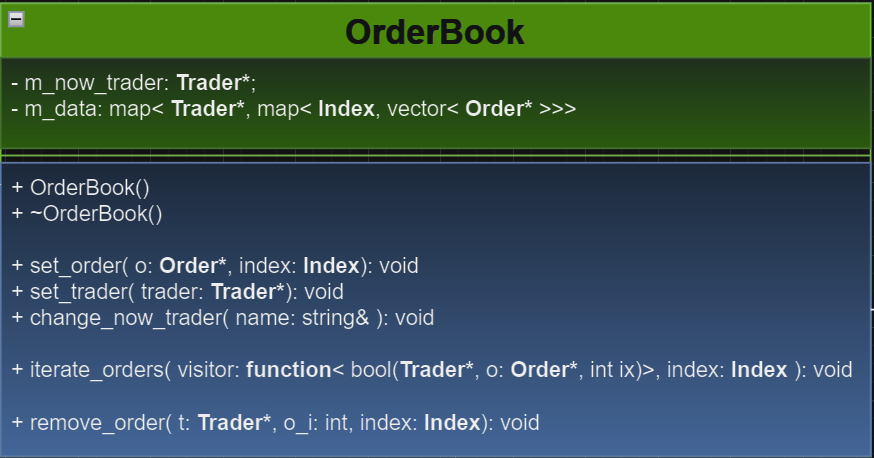
Each order should have an appropriate strategy of how it should be sold or bought. I used the strategy design pattern to solve this problem.



I also used the Factory Method pattern, which creates orders by giving them the appropriate strategy.



All Orders are stored in OrderBook. Since buy orders or sell orders must be served through different threads, I have given the following solution: for each trader there is a map in which the vector of buy orders and the vector of sell orders are stored. I store them in the map, because in the future there may be such a Vector of Orders that we want a separate thread to work and we can do it very easily.



OrderBook provides a function (the name of the function is iterate\_orders) that passes through the order vectors that clientcode wants. We can use this function in different threads to solve our problem. To store orders, I use Vector because each member is addressed in a constant time, which is quite effective from the point of view of my problem. To remove an Order from the Orders vector, I use the following method: move the deleted Order along with the last one and delete the last one.

* **Conclusion**

This task taught me how to develop a flexible and extensible system, as well as how to use design pattern.