**Algo-Trading client application LLD**

**Presentation layer (user interface):**

The presentation layer communicates with the user by using a console, which prints the following options on it:

1. Buying\Selling- asks the user to enter the requirements for buying\selling a commodity: commodity number (int), price (int) and amount (int).

The function sends the request with the matching parameters to the server through MarketClient by using SendBuyRequest\SendSellRequest method.

1. Canceling- asks the user to enter a transaction ID (int) which he\she wish to cancel. The function sends the request to the server through MarketClient by using SendCancelBuySellRequest method.
2. Queries-

* Buy\Sell- asks the user to enter a transaction id number for returning information about this transaction. The function sends the request to the server through MarketClient by using SendQueryBuySellRequest.
* User- sends a request for user information through MarketClient by using SendQueryUserRequest, and returns it to the user.
* Market- asks the user to enter a commodity number and sends a request for stock information through MarketClient by using SendQueryUserRequest, and returns it to the user.

1. Cancelling all requests- allows the user to cancel all his transactions (sell and buy). The function sends the server a request to get the user's info by using SendQueryUserRequest, and once it has it- the function goes through all the user's transactions and cancelling them by using the method SendCancelBuySellRequest.

Every option can be chosen by pressing the matching number:

1- Buy, 2- Sell, 3- Cancel, 4- Queries, 5-Cancel all transactions

4.1- Buy\Sell query

4.2- User query

4.3- Market query

If at any point the user wants to go back to the main menu, he\she can enter "-1".

**Logic layer functions:**

The logic layer gets the parameters from the presentation layer, and sends them to the matching function in MarketClient project, using the methods in it.

1. int SendBuyRequest(int price, int commodity, int amount)- sends the server the given parameters and returns the buying request's ID. If the request can't be executed, the function will write on the screen the reason, and will return -1.
2. int SendSellRequest(int price, int commodity, int amount)- send the server the given parameters and returns the selling request's ID. If the request can't be executed, the function will write on the screen the reason, and will return -1.
3. bool SendCancelBuySellRequest(int id)- sends the server a request to cancel transaction number "id". The function returns true if the transaction canceled successfully, and if it doesn't- writes on the screen the reason and returns false.
4. IMarketItemQuery SendQueryBuySellRequest(int id)- sends the server the given ID and returns an instance of IMarketItemQuery, which contains the data about transaction number "id" (commodity number, price and amount). If there is no such transaction, it will write on the screen a matching message and will return null.
5. IMarketUserData SendQueryUserRequest()- sends the server a request to get user's info and returns and instance of IMarketUserData, which contains the user's info (commodity number, price and amount). If it couldn't be happened, the function will write on the screen the reason, and will return null.
6. IMarketCommodityOffer SendQueryMarketRequest(int commodity)- sends the server the given commodity number and returns an instance of IMarketCommodityOffer, which contains the data about commodity number "commodity" (ask price and bid price). If there is no such commodity, it will write on the screen a matching message and will return null.

All the functions create a matching "item" to send to the server and fills it with the right parameters. The query functions create also an object, which is an instance of the interface they return.