Algo-trading market emulator LLD

For terminology and other relevant information please refer to the server LLD document.

<https://goo.gl/h5MyZu>

The project id delegated to three sub-projects(layers) containing the following:

**Presentation layer:**

This layer contains the GUI Interface and it's Window Forms.

**GUI Buttons and Windows:**

**Main Menu Window:**

Buttons:

Buy Commodities – opens a new window for buying commodities.

Sell Commodities – opens a new window for selling commodities.

Cancel Request – opens a new window for cancelling a buy / sell request.

Help – opens a Help window.

History – opens the History window, shows the History when clicking "Show History" button.

Commodity Query – opens the Commodity Query window.

Buy / Sell Query – opens the Buy/Sell Query window.

User Query , All Market Query , User Requests Query – after pressing one of these buttons a message box will appear with the requested information.

AMA On / AMA Off – turns the AMA on or off

**Buy / Sell Commodities Windows:**

In this windows the user is able to send a Buy / Sell request to the server.

There are 3 numeric input fields: Commodity ID, Amount , Price.

There are 2 Buttons: Cancel – Exits the window.

Apply – Takes the numbers in the input fields and sends a Buy/ Sell Request to the server(through the BL). After that a Message Box will appear and tell the user the request id if successful, or the error otherwise.

**Cancel Request Window:**

Has one numeric input field for the request ID, sends a cancel request when clicking the Apply button. Returns a Message Box with the response.

**Buy / Sell Query , Commodity Query Windows:**

These windows have one numeric field( commodity ID for commodity, Request ID for query buy/sell). Clicking the apply button will send a query request to the server and show a Message box with the requested information ( or an error that occurred otherwise).

**Business layer:**

This class holds the History and AMA classes, and besides that this layer is used as the "Middle Man" between the GUI and the Data Access Layer. It receives input from the user for all the requests and sends the requests to the DAL. After that it receives the response from the DAL, processes it and returns an adequate response to the GUI (if there was an error with the request, it returns an error string).

**Classes**

**AssemblyInfo-** Instructs Log4Net to configure the log file "history" using App.Config.

**App.config-** Initializes the structure and shape of the History document.

**RequestAgent Class** – this class operates as the "Middle Man" between the GUI and the DAL. All the functions in this class return a string back to the GUI. In this class, the logger documents all the functions that are being used by the user.

**Functions *-*** the functions in this class are very simple; they simply take the input coming from the user about a certain request, send it to the DAL and send back a response string to the GUI. Each functions writes to the History log.

string buyCommodities(int price, int commodity, int amount)

string sellCommodities(int price, int commodity, int amount)

string cancelRequest(int id)

string QueryBuySell(int id)

string CommodityQuery(int id)

string UserQuery()

string AllMarketQuery()

string UserRequestsQuery()

The History log file shows the user all the actions he made along with the date and time they were made.

**Data Access layer:**

This class contains all the communication with the server. All the requests coming from the user through the BL are processed here, sent to the server and then this layer returns an adequate response back to the BL.

**Classes**

**AssemblyInfo-** Instructs Log4Net to configure the log file "log" using App.Config.

**App.config-** Initializes the structure and shape of the Log document.

**RequestManager Class** – this class operates as the main class of the DAL. This class contains all the functions communicating with the server with the adequate inputs as given from the Presentation Layer. This class implements the IMarketClient interface as given in the MarketClient solution.

**Functions**

Each of these functions creates an instance of SimpleHTTPClient, and use that instance in order to use the SendPostRequest function, which is used in order to translate the requests and send them to the server.In this class, the logger documents all the functions that are being used by the user.

SendCancelBuySellRequest(int id) –

SendBuyRequest(int price, int commodity, int amount) -

SendSellRequest(int price, int commodity, int amount)

SendQueryBuySellRequest(int id)

SendQueryUserRequest()

SendQueryMarketRequest(int commodity)

SendAllMarketQuery()

SendUserRequestsQuery()

**Requests Classes**

These classes include no functions, only appropriate fields (according to the input needed of the function) for creating requests instances.

BuyRequest Class

SellRequest Class

CancelRequest Class

QueryMarketRequest Class

QuerySellBuyRequest Class

AllMarketRequest Class

UserRequestsQuery Class

**Query output Classes**

These classes include one function – ToString(), and appropriate fields (according to the output received from the server) . these Classes implement the interfaces as given in DataEntries folder in the MarketClient solution.

MarketCommodityOffer Class

MarketItemQuery

MarketUserData

AllCommodityOffer

MarketUserRequests