# Read Me

## Assumptions

- This is a POC, therefore extensibility, scalability, authentication & authorization, detailed logging and other such considerations will be left for future implementations.
- That all instruments from both providers including dual listed are known in advance. This will allow indexing of instrument names for faster processing. Dynamic instrument processing is left for future implementation.

## Software Requirements (e.g. Tools, API's, Frameworks)

- .NET Framework 4.5.1
- Prism.MVVM (Portable) version 5
- nUnit 2.6.3
- Microsoft Visual Studio 2013 or greater
- A comfortable chair

## Functional Requirements

✓ Limit solutions to FCL of the .NET Framework

**Provider**

✓ FR2 - Use WCF is required between the components.
✓ FR3 - All unique instruments should pass through as is.
✓ FR4 - For the instruments that are available from both providers, the Aggregator should provide an average of the last incoming data from both providers.
✓ FR5 - The Aggregator must synthesize a one (1) second delay in message processing.
✓ FR6 – Processing should always use the newest data.

**UI**

✓ FR7 - The UI is required to be a WPF application.
✓ FR8 -The UI must synthesize a one (1) second delay in message processing.
✓ FR9 - The UI should employ the MVVM design pattern.
✓ FR10 - The UI should update the grid as often as possible when updates come in.
✓ FR11 - Provide the capability to launch multiple instances of the UI.
✓ FR12 - The UI should provide the capability to pause and resume the data feed.
✓ FR13 - The UI should display on a button the phrase "Pause Updates" when it is actively listening for message.

- ✓ FR14 - The UI should display on a button the phrase “Resume Updates” when it is in a paused state for message.
- ✓ FR15 - The UI shall display the most recent data from the Aggregator when the resume button is clicked.
- ✓ FR16 - Each separate UI instance running will be autonomous from any other instance running.
- ✓ FR17 - **BONUS**: The UI should provide the capability to display the Value in green if the current value for a given Name is higher than the previous value, and display the Value in red if the current value is lower than the previous value. If the current and previous Value are equal the standard color (i.e. black) shall be displayed. See Figure A below.
- ✓ FR18 – The UI’s grid should display unique list of symbol names. The data should update inline.

## Analysis & Design

- Key Abstractions
  - MarketDataProviderBase – provides the capability to perform basic client sessions management.
  - MarketDataProvider – provides the capability to control the publish or stream auto-generated symbol data process.
  - MarketDataReader – provides the capability to read the auto-generated symbol data.
  - MarketDataProcessor – provides the capability to process the auto-generated symbol data.

## Future Implementation Suggestions

- Serialize the tick information for each and every symbol.
- Provide detail logging and diagnostic information to an external data store.
- Create discovery feature to determine the Provider to connect to.
- Provide the capability for the client to select / subscribe to a discreet set of symbols.

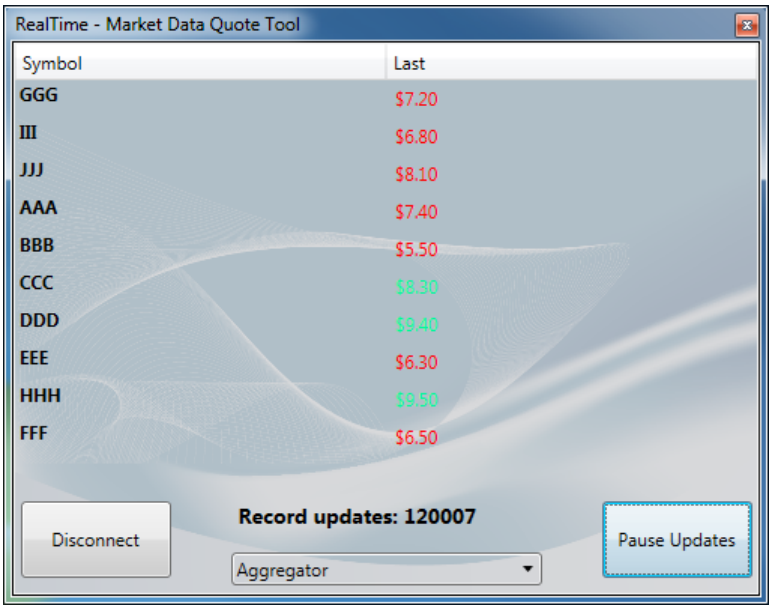


Figure A (Non-skinned version)

## How to Run Clients and Servers

1. Build the Solutions
2. Launch the MarketDataService Ecosystem.cmd batch file.
    a. The three (3) services (i.e. Aggregator, Orion, Polaris) and the client RealTime will start.
    b. NOTE: Ensure local host ports 9000, 9010, 9020 are available, respectively.
3. Bring the RealTime client to the focus, if it isn't already, and select a Provider from the drop down list box. The Aggregator provider is selected by default.
4. Click the Connect button to begin received tick information in the RealTime client window.
5. Optionally, you may press pause to temporarily stop the tick information from updating on screen. Press resume to continue updating tick information on screen.
6. Click the Disconnect button to end the connection with the selected Provider.
7. Optionally, you may connect to a different Provider without exiting the application.
8. Select the x button in the control box to close the application.
9. Services are started with a commandline i.e. MarketDataService [Aggregator | Orion | Polaris]

**Extra Bonus:** Skinned Version.