**Project Categorizing Financial Instruments**

**Context**

A financial institution manages a diverse portfolio of financial instruments, including stocks, bonds, derivatives, and other securities. Each financial instrument has a value representing its market price and a text indicating its type (e.g., stock, bond, derivative).

They implement the following interface:

interface IFinancialInstrument

{

double MarketValue { get; }

string Type { get; }

}

Currently, there are three categories based on the market value:

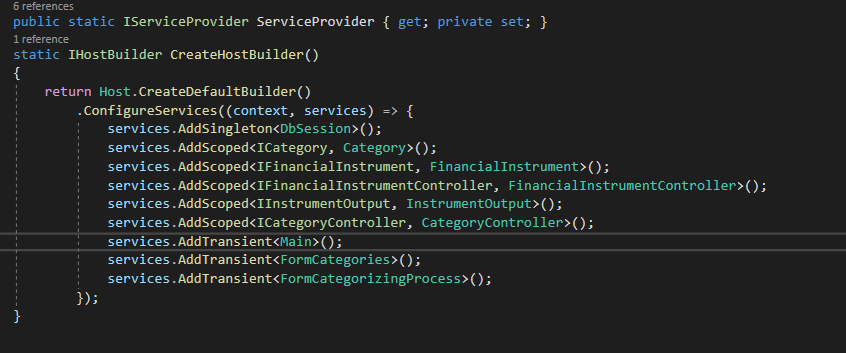
* **Low Value:** Instruments with a market value less than $1,000,000.
* **Medium Value:** Instruments with a market value between $1,000,000 and $5,000,000.
* **High Value:** Instruments with a market value greater than $5,000,000

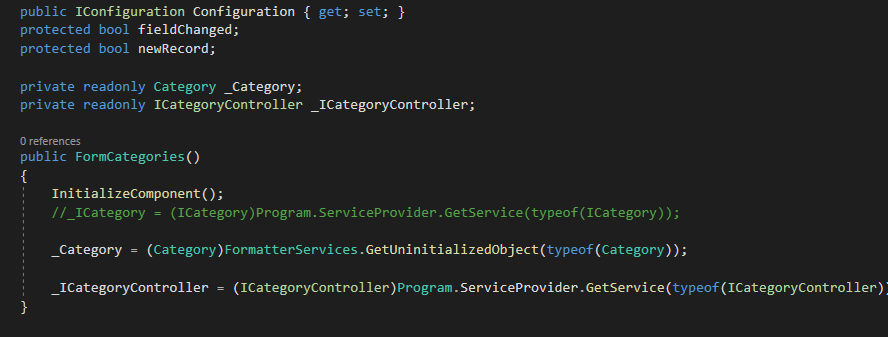
**The Software**

A Windows Forms App using .Net Core 3.1 was created under Visual Studio 2019.

I assumed that the use of Dependency Injection concepts was a necessity.

In fact, you may see it in project’s code, like for instance the piece of code at Program.cs (1st image) and at one of the form’s initialization (2nd image):



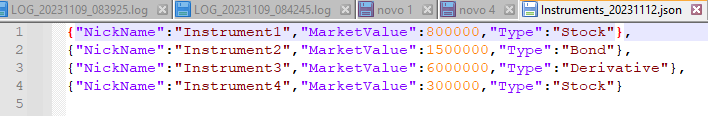


**Dapper**

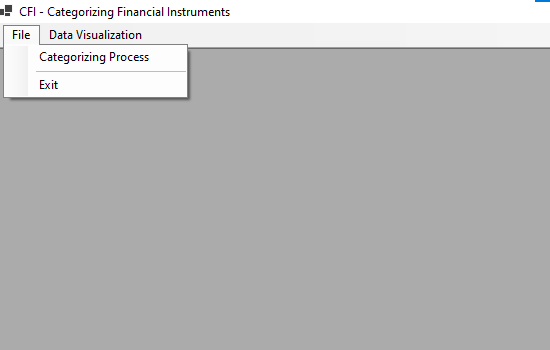
For connection and data treatment software is using Dapper as ORM.

**How it works**

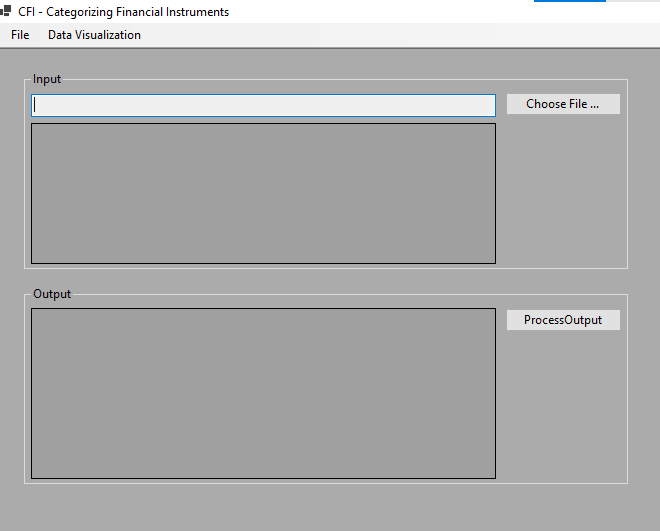
I assumed a process that received a JSon file containing the input data:



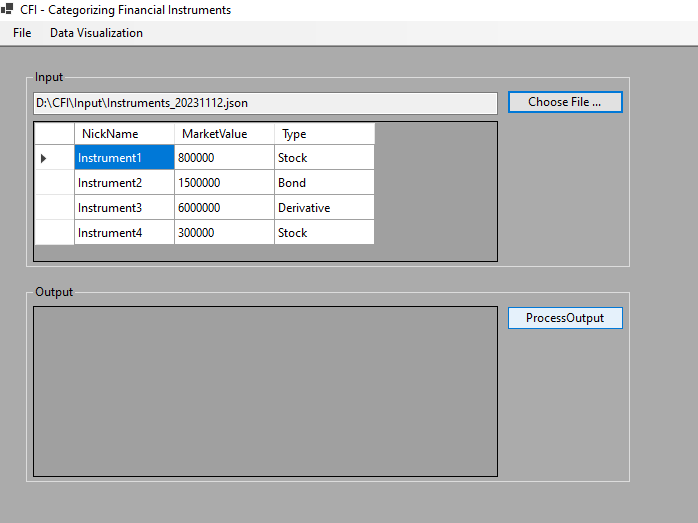
The software has a Main screen, which has a menu:



Use may import those JSon data accessing “Categorizing Process” menu, which will show the controls for it:



User may choose the JSon file, and process will automatically load its data into upper datagrid named “Input”:



By clicking at “ProcessOutput” button, the rules for Low, Medium or High value will be applied, and, at the end, it will be shown at the lower grid:

