**Task** – To monitor a directory for the incoming file ‘01-Basic.xml’ to compute statistics and output it to an output directory

**Assumptions**

* The solution being MS.net console application, many of the .net libraries used (esp. for file system monitoring) and solution design is made considering the operating environment to be a Microsoft OS.
* It is assumed that the application would have Read-Write access to the specified input and output folders.
* Since the input and output are in plain text XML, data security related aspects are not implemented/considered in the design.
* Since it’s mentioned “**ValueFactor** and **EmissionsFactor** is static data sourced from the accompanying XML file *ReferenceData.xml*. Note: it is not possible to change static data while the console application is running.” It’s assumed that the reference data file also would be in place already in the same input folder and is not modified, and thus is read at the start of the application.
* The input xml file format is considered fixed and classes for the xml are generated manually and fine-tuned.
* An email functionality is envisaged when fatal error occurs so that the right stakeholders are intimated immediately. But due to time constraints this hasn’t been implemented.

**Solution Summary** – The console application would be running continuously unless interrupted by a key press on the console window. It would be monitoring the folder (specified in the configuration file of the application) for the file ‘01-Basic.xml’ (also specified in the configuration file) being created or deleted.

On creation the appropriate processing method would be invoked and would create an output file in the folder specified in the configuration file.

**Solution details**

The console application would be initiated and start monitoring the input folder for any new file created or replaced. This is done using the ‘System.IO.FileSystemWatcher’ class. It is used as an instance in the ‘FileSystemHelper’ class object which is created when the application starts.

The reference file data is parsed at the application start by the static class method ‘ProcessRefData.ProcessAndGetRefData’.

The input file parsing and further processing are done in the static class method ‘ProcessGeneratorStats.ProcessStats’ using ‘XmlHelper’ class methods. The method is invoked in the event handler ‘OnChanged’ of the ‘FileSystemHelper’ object. The method does the processing of xml data and final calculation and output of xml file result.

**Test cases**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Test description | Expected outcome | Result |
| 1 | Basic test with the sample input file that produces an output file with matching the statistics of the sample result xml file. | The file should be created in the output folder specified in the configuration file of the application with stats matching the ones in the sample result file | **Passed** |
| 2 | Test with an additional coal generator with energy, price, emission values for a different date | The file should be created in the output folder specified in the configuration file of the application with stats matching the ones in the sample result file, with the additional generator data shown as well | **Passed** |
| 3 | Test with wrong data types for the xml tags | An error should be thrown and appropriate message shown on the console and log file., alongwith an email being sent.  Note :Email functionality hasn’t been implemented due to time constraints | **Passed** |

**External Sources**

Log4net - https://logging.apache.org/log4net/release/manual/configuration.html

https://www.kailashsblogs.com/2020/07/how-to-use-log4net-in-c-console.html

FileSystemWatcher - <https://docs.microsoft.com/en-us/dotnet/api/system.io.filesystemwatcher?redirectedfrom=MSDN&view=net-6.0>

XMLSerailizer - <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/serialization/how-to-read-object-data-from-an-xml-file>

https://json2csharp.com/xml-to-csharp (for converting xml to class)