Coding Task

This is the link to the TradingService.dll

https://drive.google.com/file/d/1wsk2tQ5HeZii7zIYhEBOQ1pAXR0rsfgK/view?usp=sharing

This task is to take 2.5 hours, please respond with a build-able solution within that time.

Task 1 - Power Trading Overview

The power traders require an intra-day report to give them their day ahead power position. The report should output the aggregated volume per hour to a CSV file.

Requirements

1. Must be implemented as a Windows service using .Net 4.5+ & C#.
2. All trade positions must be aggregated per hour (local/wall clock time). Note that for a given day, the actual local start time of the day is 23:00 (11 pm) on the previous day. Local time is in the London, UK time zone.
3. CSV output format must be two columns, Local Time (format 24 hour HH:MM e.g. 13:00) and Volume and the first row must be a header row.
4. CSV filename must be

   ```
   PowerPosition_YYYYMMDD_HHMM.csv
   ```

   where YYYYMMDD is year/month/day e.g. 20141220 for 20 Dec 2014 and HHMM is 24hr time hour and minutes e.g. 1837. The date and time are the local time of extract.
5. The location of the CSV file should be stored and read from the application configuration file.
6. An extract must run at a scheduled time interval; every X minutes where the actual interval X is stored in the application configuration file. This extract does not have to run exactly on the minute and can be within +/- 1 minute of the configured interval.
7. It is not acceptable to miss a scheduled extract.
8. An extract must run when the service first starts and then run at the interval specified as above.
9. It is acceptable for the service to only read the configuration when first starting and it does not have to dynamically update if the configuration file changes. It is sufficient to require a service restart when updating the configuration.
10. The service must provide adequate logging for production support to diagnose any issues.

Task 2 - Gas Trading Overview

The gas traders require an intra-day report to give them their day ahead gas position. The report should output the aggregated volume per hour to a CSV file.

Requirements

1. Must be implemented as a Windows service using .Net 4.5+ & C#.
2. All trade positions must be aggregated per hour (local/wall clock time). Note that for a given day, the actual local start time of the day is 06:00 (6 am) on the current day. Local time is in the London, UK time zone.
3. CSV output format must be two columns, Local Time (format 24 hour HH:MM e.g. 13:00) and Volume and the first row must be a header row.
4. CSV filename must be

   ```
   GasPosition_YYYYMMDD_HHMM.csv
   ```

   where YYYYMMDD is year/month/day e.g. 20141220 for 20 Dec 2014 and HHMM is 24hr time hour and minutes e.g. 1837. The date and time are the local time of extract.
5. The location of the CSV file should be stored and read from the application configuration file.
6. An extract must run at a scheduled time interval; every X minutes where the actual interval X is stored in the application configuration file. This extract does not have to run exactly on the minute and can be within +/- 1 minute of the configured interval.
7. It is not acceptable to miss a scheduled extract.
8. An extract must run when the service first starts and then run at the interval specified as above.
9. It is acceptable for the service to only read the configuration when first starting and it does not have to dynamically update if the configuration file changes. It is sufficient to require a service restart when updating the configuration.
10. The service must provide adequate logging for production support to diagnose any issues.

Additional Notes

An assembly has been provided (

```
TradingPlatform.dll
```

) that must be used to interface with the “trading system”. A single interface is provided to retrieve trades for a specified date. Two methods are provided, one is a synchronous implementation (

```
IEnumerable GetTrades(DateTime date);
```

) and the other is asynchronous (

```
&lt;Task&gt; GetTradesAsync(DateTime date);
```

). The implementation can use either of these methods. The class tradingService is the actual implementation of this service. The Trade class contains an array of TradingPeriods for the given day. The period number starts at 1, which is the first period of the day. The completed solution must include all source code and be able to be compiled from source. It may be delivered as a zip, zip of a DVCS (e.g. git) or a link to a hosted source control repository.

Example

Given the call to

```
TradingService.GetTrades
```

returns the following two trade positions:

Date	Periods	Volume		Date	Periods	Volume
01-Apr-2015	1	100		01-Apr-2015	1	50
	2	100			2	50
	3	100			3	50
	4	100			4	50
	5	100			5	50
	6	100			6	50
	7	100			7	50
	8	100			8	50
	9	100			9	50
	10	100			10	25
	11	100			11	25
	12	100			12	25
	13	100			13	25
	14	100			14	25
	15	100			15	25
	16	100			16	50
	17	100			17	50
	18	100			18	50
	19	100			19	50
	20	100			20	100
	21	100			21	100
	22	100			22	100
	23	100			23	100
	24	100			24	100

Will produce the below result set

Date	Periods	Volume
01-Apr-2015	1	150
	2	150
	3	150
	4	150
	5	150
	6	150
	7	150
	8	150
	9	150

	10	125
	11	125
	12	125
	13	125
	14	125
	15	125
	16	150
	17	150
	18	150
	19	150
	20	200
	21	200
	22	200
	23	200
	24	200