RISE INTERACTIVE – LEAD SOFTWARE ENGINEER EXERCISE

INSTRUCTIONS

The following exercise is to be completed within 3 days after being received. Once finished, please zip up your completed code, along with the design documentation for Problem 3 and return to the person who originally sent it to you.

PART 1: CODING

Refer to the Rise.Innovation.CSVTools solution that was provided with this exam for the following two questions:

1. Implement GetMergedFile in CSVFileMerge.cs

The method will take two CSV input file names, the name of a file to write the output to, and the name of a key column that can be used to join rows in the two files together. The method will create the output file and fill it with the contents of the two input files merged together. If the join column value of one file equals the join column value of the other file, then the columns from those two rows should be merged into a single row. If the join column's value exists in one file but not in the other file, the row should appear in the output file with empty values in any column from the other file.

You may assume the following are always true:

- Columns in both files are comma-delimited
- Rows in both files are CR/LF-delimited
- The first row of the file contains column names
- It is possible that one or both files are empty
- The two files may contain different column names, but they must always both contain the join column
- The values of the join key column will be unique within a single file
- If two files contain the same column name, only one file will contain a non-empty value for a given join key. The merged row in the output file should contain the non-empty value (if one exists)
- A column name will always contain data of the same type in both files and for every row in a single file

You are encouraged to test your application using Program.cs test harness that has been provided to you.

2. Add automated unit testing for CSVFileMerge

Now that you've built your file merging solution, you'll want to make sure that is functioning as expected based on the criteria explained in Problem 1.

Add as many unit tests to Rise.Innovation.CSVTools.Test project as you think are necessary to prove the correctness and robustness of your code. Feel free to add any extra frameworks (example Moq) that you deem necessary to properly test your code.

PART 2: SYSTEM DESIGN

The following question does not require any code. You will instead provide a brief technical specification for the proposed feature (like one you would give to your team of developers) to show them how they should build the feature. You may provide the specification in any format you like (text, diagrams, etc).

3. Mutual Fund Analytics Data API

You have been asked to build a data API that returns performance data for the stocks being held by mutual funds. The API accepts the following parameters:

- Mutual Fund Ticker symbol
- Start Date
- End Date
- Segmentation (either “Day” or “None”)

The ticker symbol is a unique string that indicates which mutual fund to look up. The start and end dates indicate the time range over which to fetch performance data. If segmentation is set to “Day,” then the API will return one row per day for each stock held by the mutual fund on that day. If the segmentation is set to “None” the API will return one row for each stock held by the mutual fund during the date range; the metrics shown will be aggregated over the entire date range (assume formulas for aggregation will be provided).

Note that segmenting by day can cause the result set of the API to get pretty large (especially over big date ranges).

You can assume that the performance data is stored in a SQL Server database hosted on AWS. Data will be refreshed once per day overnight. The rest of the application ecosystem is written in .NET/C#, so this application should be as well. All other technology and architectural decisions are up to you.

Provide technical specifications for your team that include the following information:

- How and where will the API be hosted?
- How will the API be architected to ensure that it scales well?
- How will access and usage controlled?
- What endpoint(s) will the API have, and what do requests and responses look like?

State any assumptions that you make explicitly in your documentation.