Morningstar Software Engineering Code Test

As part of our technical assessment we would like you to complete a refactoring exercise.

The exercise is based on a Visual Studio solution which currently has no unit tests and a number of code smells and maintenance issues.

If you do not already have Visual Studio installed you will be able to install a free version of the IDE following this link https://visualstudio.microsoft.com/vs/community/

Exercise:

Existing consumers of the `InstrumentService` class are reporting that the service often slows down their application and may even cause it to crash.

We would like you to begin refactoring the `AddPriceSnapshot` method in the `InstrumentService` class in order to make the class easier to maintain and more reliable. You can change anything in the `ServiceLibrary` project (method signatures, constructors, etc.), apart from making the `InstrumentDataAccess` class and its methods non-static.

You should assume that this service forms part of a larger system. You must **not** change anything in the `SnapshotUtil` project, which is an existing integration and backwards compatibility must be maintained. You do not need to update the `SnapshotUtil` project in any way, but it must remain part of the solution and it must compile after you have completed the refactor.

During the refactoring process, you should consider the SOLID principles, the readability of the code and where tests might be appropriate.

By the end of the refactor we expect **at least one test** that covers the successful addition of a price snapshot. You may use the test and/or mocking frameworks of your choice.

Notes from us:

You should aim to spend no longer than 2 hours improving the solution to make it more maintainable and reliable, applying basic engineering principles such as SOLID, DRY, YAGNI and KISS. We're not expecting the work to be 'finished', we want to see how you break down the problem and spot potential errors. We're looking for simple, clean readable code to demonstrate this. Please try avoid over-engineering the solution.

Whilst some comments are fine we do not want lots of explanatory comments in the code. If you do run out of time and would still like to explain how you would continue to refactor something then feel free to add a comment in the relevant place.

Please note all connection strings and API URLs supplied are fake, but we expect you treat them as if they were valid.

MORNINGSTAR

Make sure any unit tests you implement pass, a submission with failing tests will not pass the review stage.

Once done please reply to the email you received containing this test, attaching a nice zipped solution that compiles.

What will happen next?

A panel of Morningstar engineers will review your submission, if we like your solution and would like to discuss it further, we'll get in touch to make arrangements for the next stage of the interview process, where we will expect you to explain the changes you have made so far and what your next steps would be.