# Quote Loader Component

A junior developer was asked to create a reusable stock quote loader component, which should import quotes from a specified tab-separated file into the internal application database. Access to the database was provided by the *IQuoteRepository* interface located in the *Quotes* assembly. As a result, the junior developer produced *QuoteLoader* and *QuoteLoader.Tests* projects attached to this task.

During regular code review, a team lead pointed to a low quality of *QuoteLoader* code. One his concern was about the CSV reader/writer component, which is not generic enough to handle various possible sources of stock quotes. Another his comment was about poor unit test coverage.

## Task

Your task will be to focus on issues, which team leader considered the most critical for now, such as the CSV reader/writer component.

1. Review the existing CSV reader/writer implementation and create a complete list of problems you see in the code (i.e. for junior developer education purposes).
2. Refactor the CSV reader/writer component into a clean, performant and elegant code, without over-engineering. Make sure you follow best practices for design and implementation. Annotate your code with comments and explain any trade-offs you make.
3. Assume the CSV reader/writer already used in production, and backward compatibility must be maintained. Therefore, make sure your changes do not break any contracts. If you decide to change public interface in any way, please deprecate existing methods.
4. Make sure you write enough unit tests to achieve near 100% code coverage for CSV reader/writer component, also testing various exceptional cases. Feel free to introduce any mocking frameworks available in NuGet if needed.