The following are the changes I made in the given time frame.

AddPriceSnapshot refactoring

- Separated validation logic into its own method.
- Refactored validation logic
    - Symbols validation had faulty logic, separated it into two if blocks which checked if the symbols array was empty or less than three characters respectively.
- Made a new function for Ensuring instrument exists
    - Refactored variable cases and fixed asynchronous issues -> the use of .Wait() can lead to deadlocks and is inefficient.
- Made a new method for Fetching price
    - Did error checking for HttpRequests and parsing the double, added exception handling, ensured Http success status code.
- Added Wrapper classes and interfaces for unit testing purposes -> dependencies were exposed as static properties as I couldn't use dependency injection due to having to maintain backwards compatibility in the Snapshot util class.

Other fixes

- Fixed case issues in Instrument class.
- ExchangeRepositoryClass
    - Changed method to be async and returned Task<Exchange?>.
    - Changed SQL statement which I assumed was wrong -> was fetching Name and not MicCodes.
    - Removed .Result and replaced with await to avoid deadlocks.

Further changes I would make with more time and less constraints

- HttpClient should ideally be reused by injecting it into classes that use it, this can avoid socket exhaustion.
- Direct dependencies make for tightly coupled classes and makes unit testing difficult. Dependencies should ideally be injected into the class.
- Abstracted (defined classes) for the ExchangeRepository and InstrumentDataAccess classes to avoid direct dependencies -> this would make unit testing a lot easier.
- Additional unit testing and integration testing.
- Validation logic could be separated from core logic and put in it’s own class. (Single Responsibility Principle (SRP)).
- The database connection string is hardcoded within the method, making it difficult to change and insecure -> local connection strings should be stored in application.json and replaced when the app is being deployed with production connections.
- Would have expanded and improved error handling by defining exception classes etc.