StockTrader assignment

# Objective

The objective of this assignment is to test the basic design/development skills of a Java/JEE developer. The assignment can be attempted by candidates with experience ranging from 4-10 years. Evaluation and further discussions on the provided solution will depend on individual experience level though.

This document provides details about the assignment.

# Deliverables

* Source code zip for the solution, including resources, SQL scripts if required etc.
* A simple text explaining the design, and also steps explaining how to run the solution.

# Acceptance Criteria

The candidate is expected to build a simple StockTrading application, with provision to allocate Stocks and transfer Stocks between users. The API of the service has already been designed and respective source code is provided along with the assignment.

Following is a set of acceptance criteria for the solution

1. The solution must provide implementation to the following business interfaces defined in the API:

* StockTraderServiceFactoryIF – Implementation must be named StockTraderServiceFactory and must be in the package **com.netent.platform.hiring.stockTrader.impl.** This factory class is used by the clients using the application, to instantiate Services, and also to bootstrap/cleanup the application.
* DematServiceIF – This service is used to allocate Stocks to users, and also to check the quantity of different Stocks held by users.
* TradingServiceIF – This service is used to transfer Stocks between users, and also to track the past trade transactions on a user’s trading account. Each transfer transaction can contain multiple Stock transfers. For example a transaction request could be like “remove 100 StockA from User1, add 50 StockA to User2, add 50 StockA to User3”. For each addition of a Stock to a user’s account, there must be a corresponding removal from another user’s account. Also, no incomplete transactions must be persisted.

**StockTraderIntegrationTest** can serve as a good document to understand the expected functionality.

1. All the integration tests in the StockTraderIntegrationTest class must pass, before the solution is sent for evaluation. Note that it’s not a normal Unit test, but an integration test. **Make sure to not make any changes to the test. It must pass as-is.**
2. If any external configuration is needed (host/port) to run, it must be read from a file and instructions must be provided to fill it before running application.
3. Application must persist state in a persistent store (in-memory or ondisk).
4. No classes in the com.netent.platform.hiring.stockTrader.api package must be changed/added/removed, unless you think there is a bug, which needs a change.
5. Pom.xml may be changed, but mostly to add dependencies only.

# Evaluation Criteria

The assignment will be evaluated on following criteria

* Functional correctness
* Code quality
  + Error handling
  + Testability
  + Coding standards
  + Javadocs
  + Efficient use of the Java language
* Technology choices
* Unit Tests
* Thread safety (Multiple clients may access the application simultaneously).
* Performance (Hundreds of concurrent users expected).

# Not required

* Any UI.
* Any functionality not requested in the given interfaces.

# Implementation guidelines

We expect following stack

* Maven as the build tool
* Java 8

Other than these, you are free to choose any implementation technology/library as long as it’s based on Java platform. Any 3rd party libraries that are used must be open source.

Database, if needed, must be an open source one. Prefer an embedded one (H2) if possible, but PostGres/Mysql could be used too. If using a non-embedded one, detail full setup steps in the solution document.

Application server, if needed, must be an open source one, and you must detail full setup steps in the solution document.

## KISS

**Complexity is often used to hide inefficiency… So just KISS (Keep it simple silly).**