Ingenico Programming Assignment – 1

1. Design and Technology Choices
    a. The solution is provided as a Spring Boot Application and was bootstrapped from http://start.spring.io as it provides easier creation of the project skeleton, basically being a simple Maven project, and faster to configure and implement. Also it can be easily containerized, for example with "Dockerizing" it.
    b. The dependencies and builds were managed by Apache Maven since I am more familiar to it than I am to Gradle, although it is more feature-rich. This choice is just personal.
    c. The Web service was implemented as a REST service, since it is easier to create and manage and also eliminates time consuming schema and WSDL creation and does not require a thick-client, rather just an HTTP client is sufficient.
    d. The persistence medium was chosen to be H2 in-memory DB, as it is easy to use with Spring Boot and requires very minimal configuration, roughly no configuration at all.
    e. There is no such separation as DTO, Entity, and Model layers. Entity classes are used to represent all three. Because, they would be nearly the same if they were separated and the code would be harder to maintain, and also would violate DRY principle.
    f. There are no DAOs or DAO Services. Rather simple JPA repositories were chosen using Spring Data, and they are injected directly into the "TransactionService", which is responsible for all possible transactions that our REST resource provides. To my opinion this applies for clean-code.
    g. There are no unnecessary interfaces for repositories and entities with respect to KISS and YAGNI principles.
    h. The whole solution is thought to be a simple REST service, and not to be overridden or extended by clients as in the case of library modules or JARs; therefore, I tried to keep it as simple as possible.
    i. Unit tests include Mockito, integration tests include Spring TestRestTemplate.
    j. Apache JMeter is used for testing application behavior on simultaneous requests
2. Limitations
    a. The Account entity is just composed of name and balance, and the currency is not stored. It is not possible to overdraw an account, they are not overdraft accounts having credit limits.
    b. The Transfer entity is composed of source and destination account IDs, the amount to be transferred, and timestamp of the transfer. It is thought that:
        i. The transfer is not authenticated within this service
        ii. This service does not know whether the request comes from the account holder. There is no such entity like account holder inside this service, for simplicity purposes.
    c. Only the asked functionality has been implemented, i.e. createAccount and makeTransfer.
3. Build and deployment
    a. The project can be built and run with Maven
        i. Open console and navigate where pom.xml resides
        ii. Run > mvn spring-boot:run
    b. The project can be imported any IDE supporting maven
        i. Import > existing maven projects > clean & build > Run as Java Application
    c. After just being built by maven, you can directly run the exploded JAR
        i. Open console and navigate where pom.xml resides
        ii. Run > mvn clean install
        iii. Run > java -jar ./target/restful-transfer-service-0.0.1-SNAPSHOT.jar