**Introduction**

The goal of the code challenge is to allow us to evaluate your software development skills using Java (version 8 or higher).

Please deliver a solution that meets requirements stated below.

We are looking for pragmatic, maintainable, well tested and easy to read code.

Feel free to use 3rd party libraries as you see fit to support the design of your solution.

Using frameworks like Spring, Hibernate or an in-memory database is an overkill here.

**Time & Effort**

The code challenge is not time boxed.

**The Problem**

The goal of the challenge is to implement a system that analyses financial transaction records.

A transaction record describes transferring money from one account to another account. As such, each transaction record will have the following fields:

* transactionId – The id of the transaction
* fromAccountId – The id of the account to transfer money from
* toAccountId – The id of the account to transfer money to
* createAt – the date and time the transaction was created (in the format of “DD/MM/YYYY hh:mm:ss”)
* amount – The amount that was transferred including dollars and cents
* transactionType – The type of the transaction which could be either PAYMENT
* or REVERSAL.
* relatedTransaction – In case of a REVERSAL transaction, this will contain the id of the transaction it is reversing. In case of a PAYMENT transaction this field would be empty.

The system will be initialised with an input file in CSV format containing a list of transaction records.

Once initialised it should be able to print the **relative account balance** (positive or negative) in a given time frame.

The relative account balance is the sum of funds that were transferred to / from an account in a given time frame, it does not account for funds that were in that account prior to the timeframe.

Another requirement is that, if a transaction has a reversing transaction, this transaction should be omitted from the calculation, even if the reversing transaction is outside the given time frame.

**2.1 Example Data**

The following data is an example of an input CSV file

*transactionId, fromAccountId, toAccountId, createdAt, amount, transactionType, relatedTransaction*

*TX10001, ACC334455, ACC778899, 20/10/2018 12:47:55, 25.00, PAYMENT*

*TX10002, ACC334455, ACC998877, 20/10/2018 17:33:43, 10.50, PAYMENT*

*TX10003, ACC998877, ACC778899, 20/10/2018 18:00:00, 5.00, PAYMENT*

*TX10004, ACC334455, ACC998877, 20/10/2018 19:45:00, 10.50, REVERSAL, TX10002*

*TX10005, ACC334455, ACC778899, 21/10/2018 09:30:00, 7.25, PAYMENT*

**2.1 Example Input and Output**

Given the above input CSV file, entering the following input arguments:

accountId: ACC334455

from: 20/10/2018 12:00:00

to: 20/10/2018 19:00:00

**The output should be:**

Relative balance for the period is: -$25.00

Number of transactions included is: 1

**2.2 Assumptions**

For the sake of simplicity, it is safe to assume that

* Input file and records are all in a valid format
* Transaction are recorded in order

**2.3 Deliverables**

Ensure you include tests that demonstrate that your solution is working.

Don’t forget to add a README file describing your design and how to build / run it.

You can submit your solution by sending us a GitHub link to a git repository or a zip file containing the source code.

Good Luck!