**Coding Problem Statement**

You need to build a cost sharing app which can solve the following pain points for the users

1.     A single person tends to pay a bill (at a restaurant, etc) when they go out in a group and they expect to settle the sharing later.

2.     When a group of people plan for a trip or similar multiple people spend on multiple items the cost of which needs to be shared later.

In such scenarios tracking of settlements becomes a challenge. What is needed is at any point in time a User should be able to know

a)    What the user owes to others or vice versa

b)    Should be able to drill down and understand why the settlement amount is X.

c)    Should be able to view expenses for an event

3. Any further enhancements/extensions which can be thought of is a Plus. For these enhancements just presenting a design artefact should be enough.

4. Users can connect from Desktop, Tab, Mobile.

Max people sharing in a group can be 1000

Group can exist for more than a year, and can have partial settlements.

5. Expect concurrent 10000 requests for Expense addition + 5000 request for settlement + 2000 group creation requests.

For testing the code, it should take at least the below inputs to populate the information.

Spender, Amount, Participants/Event

The system should allow to query for

How much Person A owes Person B

How much was the expenses for an event?

How much did an event cost a person and what is the settlement left for the event?

**Rules of the Game**

1. You have two full days to implement a solution.

2. We are interested in your data modelling and object-oriented development skills, so please solve the problem keeping this in mind.

3. Please ensure that the coding conventions, directory structure and build approach of your project follow the conventions set by popular open source projects.

4.  You have to solve the problem in Java Your solution must build+run on Linux/Windows.

5.  Please use Git for version control. We expect you to send us a standard zip or tarball of your source code when you're done that includes Git metadata (the .git folder) in the tarball so we can look at your commit logs and understand how your solution evolved. Frequent commits are a huge plus.

6.  Please do not check in class files, jars or other libraries or output from the build process. Use maven for build automation and dependency.

7.  Please write comprehensive unit tests/specs.

8. Deliverables

a) Architecture Documents, Choice of stack, frameworks.

b) Test Cases

c) Test Report

d) Working code

e) Documentation on how to deploy etc.