propsoch

# Software Engineering - Coding Round

## If you're reading this, we're excited to work with you...

Hello, hope you are geared up for some hands-on work in the coming round. So, here's an exercise we hope you'll find enjoyable as well as challenging (just how we like it).

We encourage you to ask questions wherever you feel the need for clarification, and to really show off your creativity the best way you can. Sometimes this means asking uncomfortable questions and being critical with an intent to learn and grow.

As a software engineer with Propsoch you'd need to be open to both giving and receiving honest feedback and excited about growing together. As you approach this exercise, be as honest and articulate as you can be, and don't shy away from calling us out if there's something you believe that really needs changing.

## We will build a MVP of Splitwise

If you haven't used the Splitwise app, do check it out on the play store or the app store. Splitwise is the easiest way to share expenses with friends and family and stop stressing about "who owes who". Millions of people around the world use Splitwise to organise group bills for households, trips, and more.

Here is what we're expecting you do in the time we have:

### MVP Implementation ( Take Home):

- Design database models for users, expenses, and balances.
- Write APIs for creating users, adding expenses, & viewing balances

## Specifications:

**Users:**

1. User should be able to login & create an account with an email & a password
2. User can set their default currency
3. **Users can see their profile and update their email and currency**
4. **Users can delete their account**

**Expenses:**

1. **Users should be able to add an expense.** Each expense should at the least contain Name, Value, Currency, Members, Date
2. **User should be able to view, update and delete expenses**
3. User should be able to see an activity log of all expenses they're a part of - added by them or other users, grouped by current month, last month & custom date range

**Balances:**

1. **User should be able to view their balances with all different users**
2. User should be receive a monthly report of their balances in an email

You will write and test APIs for some of these use cases as possible in the given time. Only implement the APIs that're marked in bold, use the other ones to influence your schema design. You don't have to implement an Auth Layer, simply assume that a user ID is available in the API request for now.

## Additional Complexities:

We will discuss and implement additional complexities after the above problems have been solved.
These complexities will be presented to you on the spot and you will have 45 mins to discuss and implement them.

## How will we conduct the interview?

You will have to share your screen and you can feel free to document and structure your thoughts on a Google Document / Miro Board etc. And in your preferred IDE, implement the DB, Models, Controllers, Services and Integrations if required.

Once you are done with the tasks mentioned above, please share the github link of the repo and the postman collection for testing. You can feel free to use Google / Stackoverflow / other documents or blogs or boilerplates, but **DO NOT** use Gemini / GPT / Llama or other LLM applications to write the code for you.

## Which tech stack will we use?

Database: **MySQL / SQLite or any other relational database**

Backend: **NodeJS**

Boilerplate: https://github.com/gadfaria/express-sequelize-boilerplate