Method SWE - Assignment Lean

👋 Hi there! Thank you for continuing in the interview process with Method. We are really excited about the prospect of working together.

Method's Interview Process

~~Intro Call (30 mins)~~

Take Home Assignment

This assignment is designed to mimic the day-to-day process of shipping products at Method. We believe traditional programing interviews (read: leet code) are an imperfect interview process, and we want to level the playing field for all candidates.

There is no specific time constraints, but feel free to take as long (or as little) as you'd like. Pixel perfect design or production level code is not required!

Background

Background

At Method, we help developers and fintechs build debt repayment into their applications. We offer a simple API that makes moving money as simple as making a POST request.

Dunkin Donuts is a Method client and they are looking to contribute to their employees' student loans. Dunkin has contracted us to build a custom dashboard where they can upload an XML roster of all their employees and make payments to their student loans.

Assignment

Create a payout dashboard for student loan disbursements. The dashboard will allow the client, Dunkin Donuts, to upload an XML file (sample attached below) that contains all the necessary information to make a one-time payout to a student loan.

Snippet of XML

dunkin.xml 35382.1KB

Notes on Dunkin's process:

- The client intends to upload the XML at the end of every pay cycle (every 2 weeks).
- The file will contain about 10,000 unique employees across 30 branches so we should optimize and limit the number of external API requests we make!
- The funds will originate from 5 unique corporate (`Payor.DunkinId`) checking accounts owned by Dunkin.
- The XML file is a list of unique payments. This means Dunkin can make multiple student loan payments to a single employee per batch.

Dashboard Features

The dashboard will be hosted in Dunkin's intranet so there's no need to build any authentication. This product will be used by Dunkin's HR team so simplicity is key! **(Simplicity > Design)** The client is looking for 3 key features: 🏦payouts and 📊reporting

🏦Payouts

The dashboard should accept the XML file via a file upload. The dashboard should process the file and interface with the Method API to create all the necessary resources (entities, accounts and payments)

Before payment processing starts the dashboard should display in a "succinct way" all the payments we are about to initiate. The user can either authorize the batch or discard it.

Once the batch is approved payments can be transmitted to the Method API.

📊Reporting

The client should be able to generate 3 CSV reports for every XML batch they have previously uploaded.

1. Total amount of funds paid out per unique source account.
2. Total amount of funds paid out per Dunkin branch.
3. The status of every payment and its relevant metadata.

Services / Architecture

You can design the UI, schema and overall flow however you'd like. Feel free to complete this in any stack you are most comfortable with. If you are agnostic, we recommend using the MERN stack. That's what we use at Method!

- Method API:
  - Create an account and get a Dev API key: https://dashboard.methodfi.com/login
  - The Method API has a 600 requests per IP per minute rate-limit. When the 601st request in that minute a 429 will be issued.
  - Important Notes:
    - When creating a corporate entity pass an empty array for `owners` key.
    - When creating an individual entity you don't need to pass the `address` key.
    - Use the Merchant `GET` endpoint to find a merchant per

Plaid Id (docs)

- `EIN` is the unique identifier for a corporation in the US.

- Useful Libraries:
  - UI Components: https://mui.com/ https://elastic.github.io/eui/

☎️ **If you have any questions, don't hesitate to ping Jose directly!**

Submitting

🎉 Once you are finished upload your code to a Github repo and share with us. Our usernames are: `josebet1` , `jmdelcarmen` and `mustafalii`

Email us with a link to your repo and we'll quickly schedule a debrief call. We are looking forward to seeing what you'll build! 🚀