Hello! This coding challenge is an opportunity for you to show us your basic engineering skills and your facility with various technologies. In addition, the choices you make and document for us will allow you to show us how you think and put your engineering values on display.

How Papa Works

We'd like you to build an application that uses a few of the concepts of our company, which works like this:

- Visits are at the heart of Papa. A **visit** is a meeting **requested** by **members** of Papa (typically Senior Citizens), whose health plans allow them a certain number of visit hours per year.
- Visits are then **accepted** by our Papa **pals**, who go to the member's house to perform various tasks, generally including providing companionship and conversation.
- Pals are then compensated for fulfilling those visits.

Assignment

In order to simplify our system and give you an opportunity to show various skills, we're going to change the rules of our system a bit. Build a "Home Visit Service" application with the following core functionality:

- Users must be able to create an account. They can perform either or both of two roles: a **member** (who requests visits) and a **pal** (who fulfills visits).
- As a member, a user can **request** visits.
- When a user **fulfills** a **visit**, the minutes of visit's duration are debited from the visit requester's account and credited to the pal's account, minus a 15% overhead fee.
- If a member's account has a balance of 0 minutes, they cannot request any more visits until they fulfill visits themselves.

This application may be command-line or API-only. It does not require a graphical UI or web interface.

Data Model

You may use/extend these models or create new ones:

User	**Visit**	**Transaction**
`first_name`	`member`	`member`
`last_name`	`date`	`pal`
`email`	`minutes`	`visit`
	`tasks`	

Notes

- We recommend you use Elixir for this application, but it is not required
- Include instructions on how to run the application and tests
- Document any assumptions you made when writing the application
- Document all design decisions and technology choices
- When you're finished, push your solution to Github and share the link with us

In lieu of completing our coding challenge, you may submit a sample of your own code for review. (Please note: follow-up interviews will assume some understanding of the Papa application's concepts and challenges as explored in the coding challenge. If you are unable to complete the challenge, you will need to review and think through the details described above in preparation for those interviews.)

You may choose any code, but consider that we're looking to see how you think and put your engineering values and skills on display. As a Papa engineer, you'll be making and evaluating engineering decisions daily, and we're looking to get a sense for whether we share the same decision-making values. Ideally, we'd like to see code where you made some important design decisions relating to architecture, defining an API, managing data, handling the unexpected, and testing--and doing all of that in a clean and efficient way. Show us your best (and we'll hope that's what you'd be giving us every day at Papa).

Code samples should also follow these guidelines:

- Your code must be openly available on the public web (GitHub, GitLab, Bitbucket, etc). Do not send us any other company's proprietary code.
- You must be the primary author of the code, as determined by the commit history and git blame.
- Code should be in a language we are comfortable with, ideally Elixir, or alternatively Ruby or Javascript. We'll better be able to see the strength of your submission in a language we work in.
- The sample you select should be hundreds of lines long, but not thousands. If submitting from a larger repository, point us to a specific PR or set of files you'd like us to consider that meets the criteria.
- You must provide a clear writeup of the problem your code is solving.
- We'll be evaluating the code you send us by its accuracy, quality, understandability, and your ability to describe the problem being solved.

We may ask you to provide a second submission.

Please use the following template for your submission:

```
### Problem statement
<a few paragraphs describing the problem>

### Your solution
<a few paragraphs describing how you solved it-focus on the whys here>

### Link
<url to code or PR>
```