Thanks for your interest in joining the MRHB Network engineering team! Before we proceed with more formal interviews, we ask that all candidates submit a coding challenge. The coding challenge is a foundational piece of our process and it's referenced later in our process during the technical interviews.

The coding challenge revolves around building a task list. Tasks belong to groups and can have dependencies on one another (i.e. if task X depends on task Y, task X cannot be completed until task Y is completed). The challenge includes 3 components:

- Build React-based UI
- Complete a backend interface
- Design database schema
- Implement Web3 Wallet connection preferably MetaMask (or WalletConnect).

Build React-based UI

The UI consists of 2 screens:

- **Overview**: Displays a list of all the groups along with their completion status. Clicking on a group should render the detail screen.

- **Detail**: Displays a list of all the tasks in the selected group and allows the user to toggle the completion status of unlocked tasks.

The data you should use to implement your solution is available at a GQL Query called tasks, you can find it on *./client/src/graphql/getTasks.graphql*. SVG assets for the icons used in the design live in *./client/public/*.

Some things to keep in mind:

- When clicking the box to complete a task, you must use the toggleTask mutation.

- Your implementation should resemble the above design

- We value well-structured code that follows best practices

Complete a backend interface

As mentioned above, you will need to use the toggleTask mutation to complete/un-complete a task. To accomplish this, you will need to implement the backend logic for toggling the task.

Design Database Schema

Design a schema in SQL to store the task list data. You can add the SQL code needed to create the schema to *schema.sql*. The schema should define all tables, columns, and constraints needed to store the task list data.

Implement Web3 Wallet

A user should be able to connect their MetaMask wallet to the app and app should load tasks for that specific user. You can use Metamask Wallet Address to identify the user, as this is unique public key that only that specific user has access to.

Getting Started

- To get started, clone this repo to your local machine.
- Next you'll want to install the dependencies and run the client and server (we recommend using node: v14.15.4):
- cd server
- yarn install

yarn dev

open other terminal on the root of this project

cd client

yarn install

yarn start

At this point, the client and server should be running in development mode and any local modifications you make will be automatically detected and result in reloads.

You should only need to add/modify code in the *./client/src/* and *./server/src/* directory.

Submission

To submit your coding challenge, zip up your solution return it to: denizdalkilic@marhabadefi.com

Thanks for taking the time to do this coding challenge and here's hoping we talk soon!