Welcome to the Thriva front-end coding challenge

Intro

Thriva is on a mission to put better health in people's hands with tools they can trust, advice they can action, and the support to get them there. We're doing that by working on a bunch of exciting features like predictive results, the ability to upload past blood test results, and a personalised health plan to help our customers achieve their health goals. As developers in the product team our aim is to create delightful experiences for our customers, be the voice of the user within the company, and produce world-class code.

A user's journey with us begins with our onboarding survey — we collect the minimum amount of information we need to be able to make relevant recommendations of what someone should be testing.

This challenge is actually a simplified version of our older onboarding survey so it gives you a chance to see how we work and structure our code. If you're interested in finding out more about the tools we use please read our blog post. As already mentioned, we expect you to spend an evening on this and ask you to send it back to us within a week. Feel free to work at your own pace and make any additions that you think will make the user experience even better!

Preparation checklist

1. Fork the https://github.com/thrivadev/frontend-coding-challenge repo and follow installation instructions to set up the project. We prefer receiving a github link for the challenge but if you use a different versioning tool feel free to share a zip with us.

2. You will find one more PDF with the design guidelines attached to the email.

3. There is a NOTES.md file; please leave anything you'd like to let us know in there. This could be reasoning as to why you have done x this way, your thought process or a comment on one of the tasks. We will use these as discussion points when running through this tasks in person.

Instructions

Part 1: Completing the survey

What we're evaluating: Javascript/Vue knowledge and communication with an API

We use Vue and Vuex but we don't expect you to have worked with this framework before. However, remember you can look around in the code for examples of how we do things at Thriva. That combined with Vue's excellent documentation should be enough to get you started and should you get to the next stage we expect you to be able to support your implementation choices.

Tasks

1. **Name**
    - Store the user's first name inputted in the Name stage.
    - Display it in the Goals stage's question.

2. **Goals & Diet**
    - Extend the functionality of the CheckButton component so it can handle the selection of the user's goals and diet.
    - Store the user's goals and diet.
    - The users can select up to 4 goals but only 1 diet.
    - The users should not be able to progress to the next stage if the above requirements are not met.

3. **Date of birth**
    - Store the user's date of birth.

ⓘ The value will be stored in this format `2020-09-23T17:05:04.606Z` by default.

4. **Survey progress**
    - Supply the correct data to the SurveyProgress component so that it will display how far through the survey stages the user currently is.

ⓘ The component is built, no need to worry about the styling, everything is there for you.

5. **API call**
    - Call the API using the Vuex *sendDataToApi* action when the user submits their date of birth.
    - The call should be a POST request to the */users* path.
    - The API requires the request body to be in the JSON format. The format for the JSON is as follows:

```
// Request body format

{
  "user": {
    "Name": "String"
    "goals": ["String"],
    "diet": "String",
    "dob": "String"
  }
}
```

```
// Errors from the server will look as follows

{
  "error": "Name must not be an empty string"
}
```

    - On a successful API call the user should be redirected to the *Success.vue* page.

ⓘ If the API accepts your request you should get a 201 response, otherwise you'll receive a 400 with details of the error in the response body.

Part 2: Adding a Success screen

Tasks

1. **Success screen**
    - Extend the */connectors/Success.vue* file based on the design provided.

ⓘ Some styles should work already because they're part of the Thriva default styles but we have provided them in the design in case you need them.

Part 3: Writing tests

We use Jest and Vue Test Utils but we don't expect you to have worked with these testing libraries before. To facilitate your work, the two testing files you will have to work on have already been created and, at least partially, set up to ensure that it performs the task that it is meant to perform.

Tasks

- **CheckButton tests**
  - Implement the already defined unit tests in the CheckButton.test.js file to make sure the CheckButton component behaves as expected.
  - Define and implement additional tests in the CheckButton.test.js file to reflect any change or addition to the CheckButton component logic.
- **sendToApi tests**
  - Define and implement any useful tests for the sendToApi action in the actions.test.js file, based on both its initial implementation and additional logic.

What we're evaluating

Your application will be reviewed by 2-3 of our engineers. The aspects of your code we'll evaluate include:

- Correctness: Does the Survey do what was asked? If there's anything missing, does the NOTES.md explain why it's missing?
- Code Quality: Is the code DRY, easy to understand, and maintainable?
- Testing: How thorough are the tests? Are they testing useful things?
- UI/UX: Does the survey work end to end? Has the design been followed accurately?
- Technical Choices: Do they seem appropriate for the challenge?