We'd like to ask you to complete this coding challenge, so you can showcase your creative problem-solving and technical ability.

We hope you'll enjoy it. It should take around a week, give or take.

We're on the lookout for best practices and evidence that you understand the technology.

We strongly suggest you use the libraries suggested on the bottom of this description and to follow functional programming practices to complete this challenge, but you're free to use any other libraries or tools on top.

**The background**

As a Hospital manager I would like to have a web app that shows a list of patients assigned to a practitioner and when selecting each patient, it shows a graph using the glucose measures uploaded.

This app doesn’t need a login screen, but it has to be able to return results by using a practitioner ID.

The app should have 3 sections, home, patient’s list and graph.

Home – home.png

This should contain a component that will allow the practitioner to input his/her user id and if the user id is valid it should navigate to the list section and show the results, otherwise it should show a message saying that the user id is not valid.

List – list.png

This section should show the results of all the patients assigned to that practitioner, showing a scrollable list (if it is needed) and the name, DoB, the diabetes type and a button to navigate to the graph section

Graph – graph.png

Using the data for each patient a graph showing the evolution of a single day should be created as in the screenshot provided, using the ‘glucoseMesures’ (GM) values from the json file, the graphic should be created dynamically using the ‘ranges’ value for each patient (have a look at graph.png for correlations).

Each glucose measure should be represented by a cross and on roll over should show the data for that GM and it should use a different color depending on its level (have look at tooltips images).

**Your task**

- Build the screen so it follows the guidelines from the supplied design screenshot.

- Implement the logic as described above.

- Provide unit tests.

**Things we are looking for**

* Maintainable and well-written code using good functional programming practices.
* We like to see good vanilla JS, but for this test we are looking for a react application.
* Browser support: IE11+, latest 2 versions of Chrome, Firefox, Safari.

**Things that we are NOT looking for**

* Database or any data source connection or any kind of ORM. It’s OK to read the provided JSON file all at once.
* The app does not need to looks exactly like in the screenshots, they should be used as a guidance.

**Suggested Libraries**

* [react](https://github.com/facebook/react): A declarative, efficient, and flexible JavaScript library for building user interfaces.
* [redux](https://github.com/reactjs/redux): Keep the entire state of your app in a single place.
* [redux-observable](https://github.com/redux-observable/redux-observable): middleware for action side effects in Redux using "Epics".
* [Reselect](https://github.com/reduxjs/reselect): Simple “selector” library for Redux.
* [Recompose](https://github.com/acdlite/recompose): Recompose is a React utility belt for function components and higher-order components.
* [Ramda](https://github.com/ramda/ramda): Practical functional Javascript.
* [Jest](https://github.com/facebook/jest): A comprehensive JavaScript testing solution.
* [styled-components](https://github.com/styled-components/styled-components): Use tagged template literals to style your components.
* [Typescript](https://github.com/Microsoft/TypeScript):  a superset of JavaScript that compiles to clean JavaScript output

**How to submit your web application**

When you're happy with the result, send us a link to your private repository (if you have one) or a zip file with the source code and remember to include a README.md file so we can install, build and run it locally. Please don't publish it to a publicly accessible repository.