Privitar UI Exercise

What are we looking for?

Privitar is looking for talented developers who have an appreciation for clean code; knowledge of how to construct enterprise-grade front end applications; and a drive to keep up to date with the latest front end tools and technologies.

An appreciation for clean and well-reasoned design and UX is a bonus, but in this exercise we really want to see what the best code you can produce looks like. However, we still want to see that you can follow a specification.

You can use any secondary tools you feel comfortable with, including but not limited to state management libraries, form libraries, design system libraries and CSS-in-JS libraries. Whatever you need to get started. The only constant we expect to see is that the application is built using React.

Try to avoid setting up any build tooling; it's the application itself we will focus on. Feel free to use something like *create-react-app*.

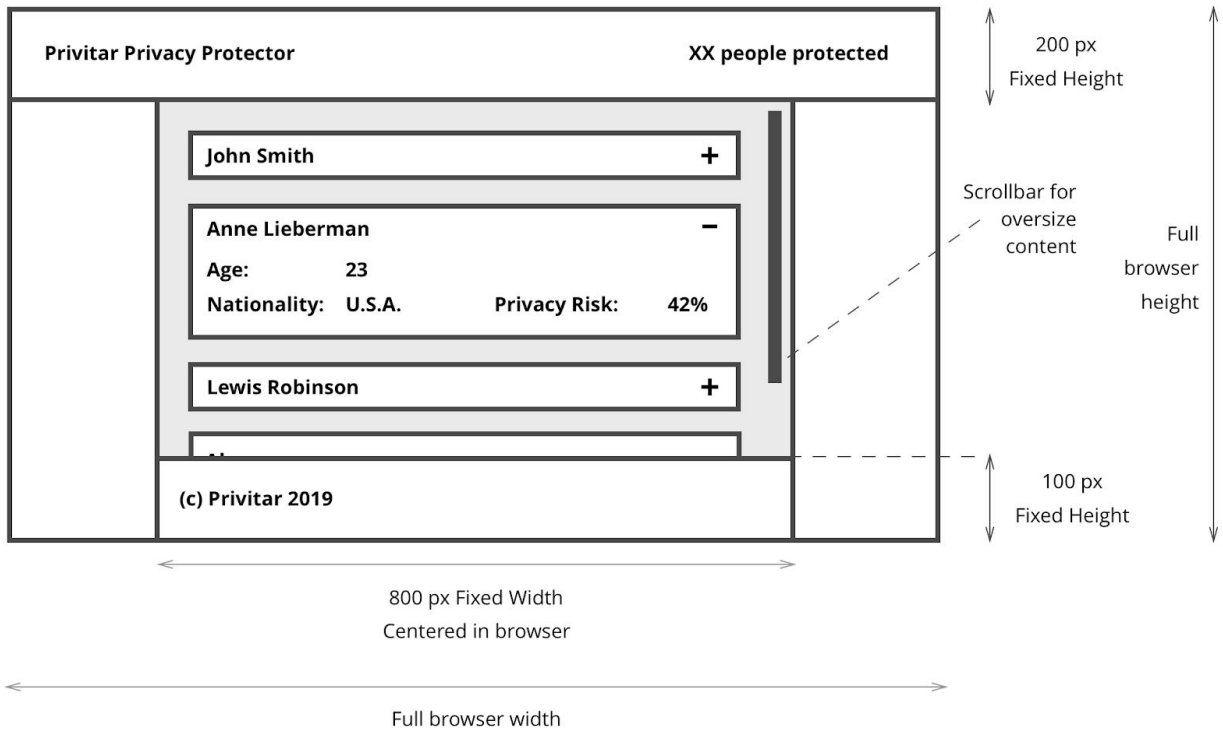
Please send the whole thing over us in a zipped folder (with a removed node_modules, of course) or a link to a private hosted git repo (github, gitlab, bitbucket).

The Exercise

Imagine privitar is building a simple application that displays the 'Privacy Risk percentage' for a list of individuals. The individuals' names and risk scores are available from an API, which has already been built. You have been asked to build the user interface.

Additionally, a form is present on the page that allows a user to add new entries. Privitar receives the data about the new persons in a predetermined bulk free-text format, which the user interface needs to parse. When the user submits the form, the user is sent back to the main listing where the new additions are reflected.

Requirements



General:

1. The page should respect the sizing constraints shown on the diagram, and look good even as the browser is resized. A fully responsive solution is *not* required; please set sensible minimum sizes.

2. The string 'XX' in the header should be replaced with the number of people in the list.

Main listing:

3. The central panel is populated with data items pulled from this URL: https://m37ov7xhd3.execute-api.eu-west-1.amazonaws.com/default/ui_example_data_endpoint

4. To call the URL, you must set an API key as an HTTP header. The API key will be provided to you by email:

x-api-key: YOUR_API_KEY

5. The central panel is a vertical accordion layout.
    a. Some examples of accordions are given here: http://ui-patterns.com/patterns/AccordionMenu
    b. The accordion should only allow one item to be expanded at a time.
6. The central panel should scroll only vertically to accommodate the content. A scrollbar is displayed to allow this. No scrollbars should be visible anywhere else on the page (unless the browser is resized too small).
7. At the bottom of the list is a form which allows the user to add new people to the list in bulk.

New person form:

8. The form that is presented includes one *textarea.*
9. The user will paste data into the *textarea* that represents multiple people to add to the data set. The format of this text is predetermined:
    a. Each line in the input is treated as a single entry.
    b. There are 4 fields per line in the order: *Name, country, age, risk percentage.*
        i. All fields are separated by 1 or more spaces.
        ii. The *name* can be any number of words, with each name separated by 1 or more spaces.
        iii. The *country* is a 2 letter country code. You can take this verbatim, you don't need to look up the country.
        iv. The *age* is either the age of the person or their date of birth in the format DD/MM/YYYY.
        v. The *risk percentage* is a decimal value from 0 to 1, where 0 is 0% and 1 is 100%. The value might have trailing 0's. Entries with a risk percent over 100% should be ignored.
    c. When dealing with invalid user input, just ignore invalid entries.
    d. Use the following string in order to test your solution:

```
Ramiro    Escobar GB   23        0.1000
Jennifer Stevens  US      34          1.02
Shreya    Khan      IN    16/02/1991    0.7
Katherine Hannah Long      FR 10/01/1978    0.1
Willem de   Leeuw Verbeek   NL 44 0.45
Joseane Silva MX  12/12/1993  1.2300
Charlie Smith IE    22  0.005e2
```

10. A button is present that when clicked parses the input and appends it to the data set.

a. The button click **does not need to make an API call**. Instead, think about how you might store the data *in memory* with the existing data.

11. When the data is successfully parsed and appended to the data set, the is it immediately reflected in the listing above the form.

Additional Considerations

It is by no means necessary, but there are some things that can really demonstrate your skills to us. You might want to think about how these could be implemented in your solution if you are invited to interview, or have a go in parts of your implementation. **However, These are not expected requirements:**

- Strongly typed. At Privitar, we use TypeScript to build enterprise-grade front end code.
- Tests. Testing is important to us, we love code that considers long term quality.
- Modern React API's. React is moving fast, and we keep up to date with the latest developments like *Context* and *Hooks.*
- Presentation. We like clean user interfaces, but we don't expect you to be a graphics designer.