[FE] Technical Test

Summary

Some members of our team at Userzoom are crazy with Pokémon and they would like to see their favourites pokémon in a *beautiful* web application. The application should consist of two views. The first view should list the pokémons and be able to filter them. The second view should show the details of a pokémon.

The application should be a SPA so that the navigation is done always client-side, without completely refreshing the whole document.

The solution should be presented through a public repository that **must** include a README.md document where you explain how to execute the application. This test is aimed at knowing how you code a web application, the technical decisions you take along the process and cleanliness of your code. Don't worry if the website isn't complete, but it's always a bonus!

Max. duration: 4-6 hours

Guidelines

- The application will only be reviewed in the latest version of Google Chrome for desktop.
- You can create the application from an existing boilerplate, if you wish.
- Use of JS third party libraries is allowed.
- Do not spend so much time in the design, we won't evaluate that.

Restrictions

- You **must** use React.

Bonus

- Implement tests.
- Mobile-friendly.

List View

Url: /

1. This view should show the list of pokémons that are available at the endpoint http://pokeapi.salestock.net/api/v2/pokemon/.
    a. Each page should consist of 30 cards of pokémons.
    b. Optional: Infinite scroll, so that each page of pokémons is loaded as the users scrolls down.
2. Once the list is obtained for the first time from the API, it must be stored in the client so that it's only requested if more than one day has passed since the last time it was requested.
3. The user can filter the list that has already received by name and id.
    a. No extra API call is required in order to filter the list of pokémons, the filter will only take into account the already downloaded list.
    b. Filtering must have an inmediate effect, so the items on the list will change as the user types characters on the filter.
4. When a pokémon is clicked, the router must navigate to the Detail view page.

Detail View

Url: /:id

1. This view should show the details of the selected pokémon according to the endpoint http://pokeapi.salestock.net/api/v2/pokemon/{id}, where {id} is the pokémon identifier.
    a. Show only the following details for each pokémon: name, number of moves, id and types.
    b. Once the detail is obtained for the first time from the endpoint, it must be stored in the client so that it's only requested if more than one day has passed since the last time it was requested.
2. When the user clicks on the header title (Pokémunz), the website must go back to the List view.