SEAT:CODE JS Code Challenge

Technical test conducted for candidates applying for the Frontend Developer position.

Welcome!

If you are reading this document, thanks for the effort invested so far. We feel you can be a great member of our family. We want to learn a bit from your technical background to carry on with the process.

Instructions

Problem

You will be facing a Kata alike problem. It's a code challenge small enough so it can be solved without investing too much time, but complex enough to give us some hints on your thinking process, problem-solving, testing, and software engineering skills.

Timing

The code challenge should take you around 4 hours. Feel free to invest more if you want to, but we understand that your time is a precious asset. We ask you to deliver your solution maximum 48 hours after you have received the challenge. So we can speed up the process and improve your candidate experience.

How to delivery the Code Challenge

Our preferred method of delivery is using GitHub. GitHub supports personal private projects, so it won't cost you any pence. We have created a Fake Kent Beck GitHub account (@kent-beck) who you will have to invite as a collaborator. If you don't know how to do it, take a look at "Inviting collaborators to a personal repository" Github Documentation Page. If for some reason you cannot use GitHub, you can zip your code challenge and send it to candidates@code.seat. Kent Beck won't be as happy, but it will do the trick. Send us also an email in case of any issue with the delivery.

Last words

If you have any tech questions related to the challenge, take an assumption, and carry on. Please provide a small README file on how to compile and run your application. Best of luck! We hope you enjoy the code challenge!

Objective

Develop a web interface where users can search and filter different Pokemon.

API Documentation

The API to be used in this project is https://pokeapi.co, where you can find both the documentation and the structure of the endpoints.

Base API URL → `https://pokeapi.co/api/v2`

This API does not require authentication.

Images

All Pokemon images are available at the following URL:

Small

`https://assets.pokemon.com/assets/cms2/img/pokedex/detail/{pokemon-id}.png`

Big

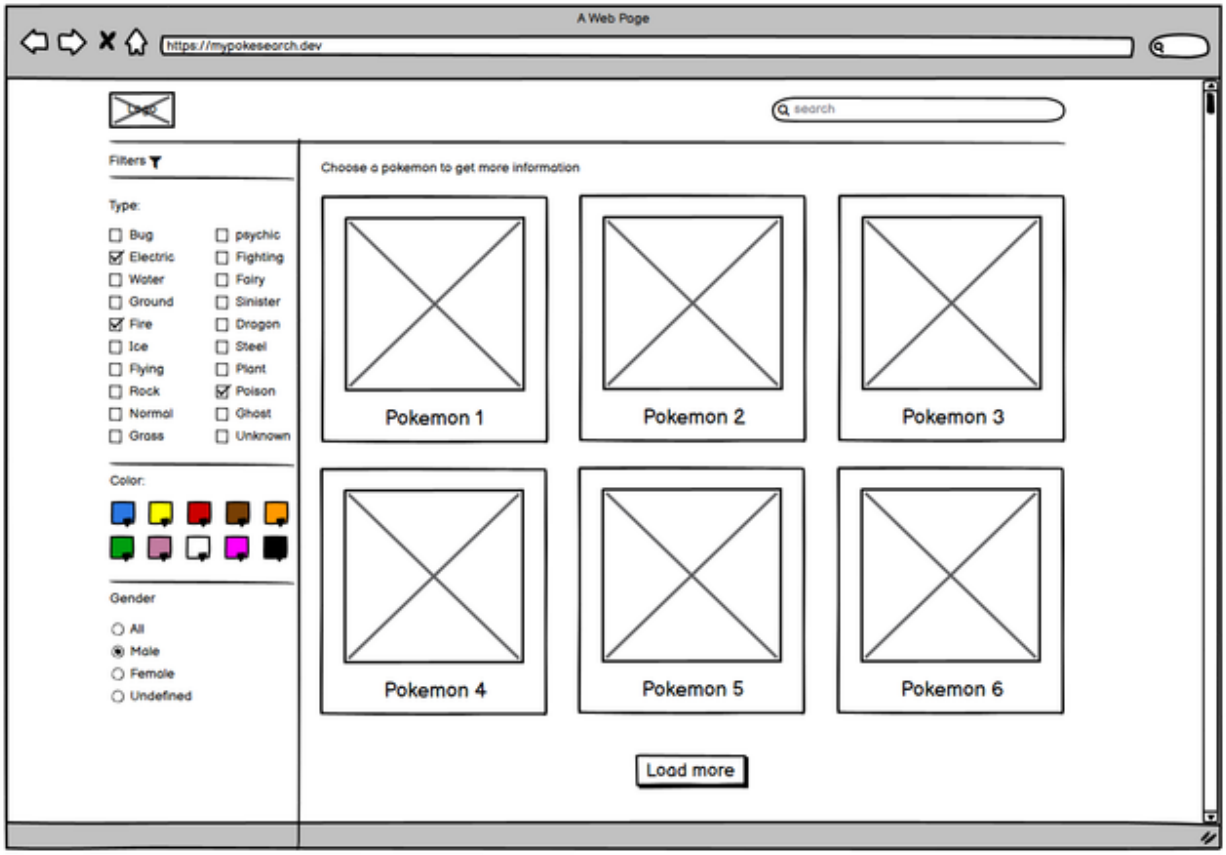
`https://assets.pokemon.com/assets/cms2/img/pokedex/full/{pokemon-id}.png`

Example:

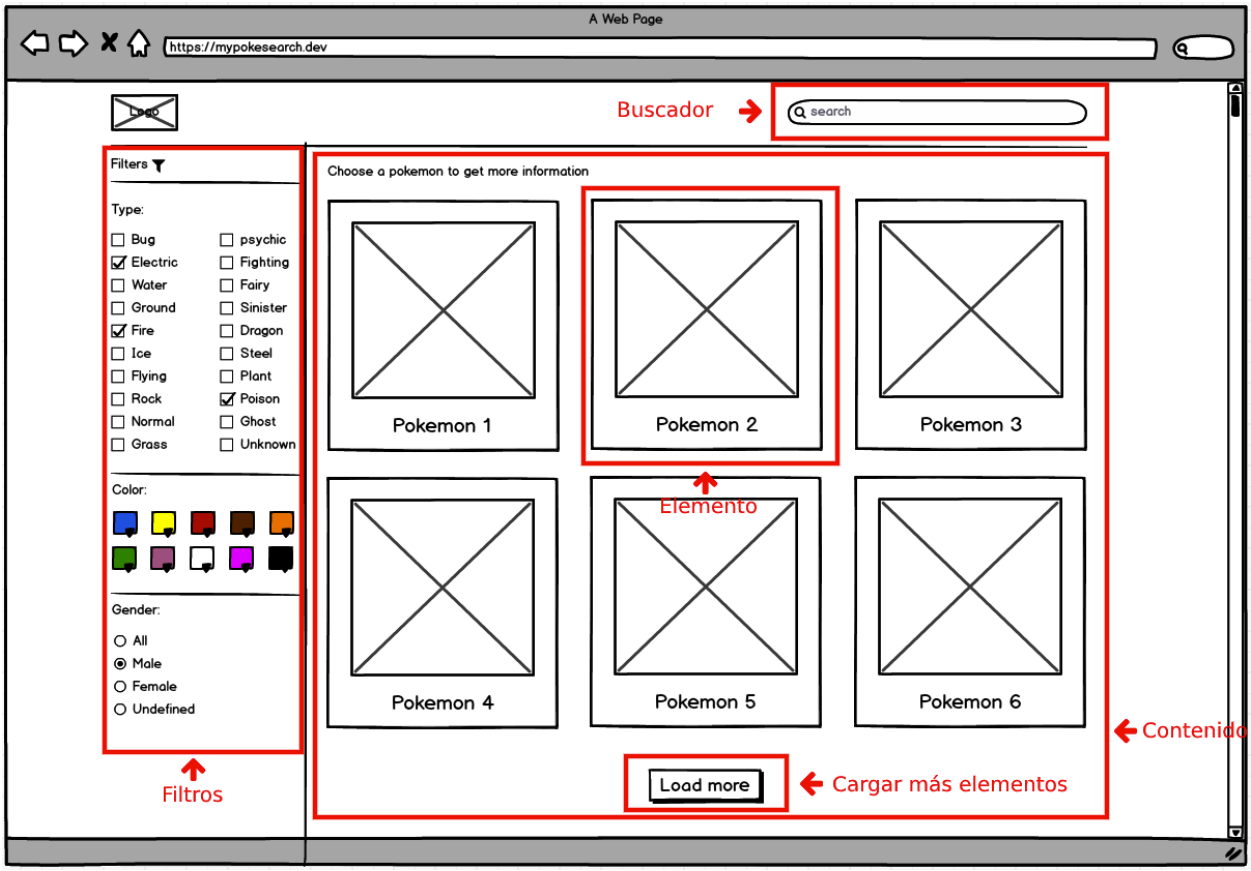
`https://assets.pokemon.com/assets/cms2/img/pokedex/detail/001.png`

Layout

The developer can design and layout before programming any functionality.



Interface structure

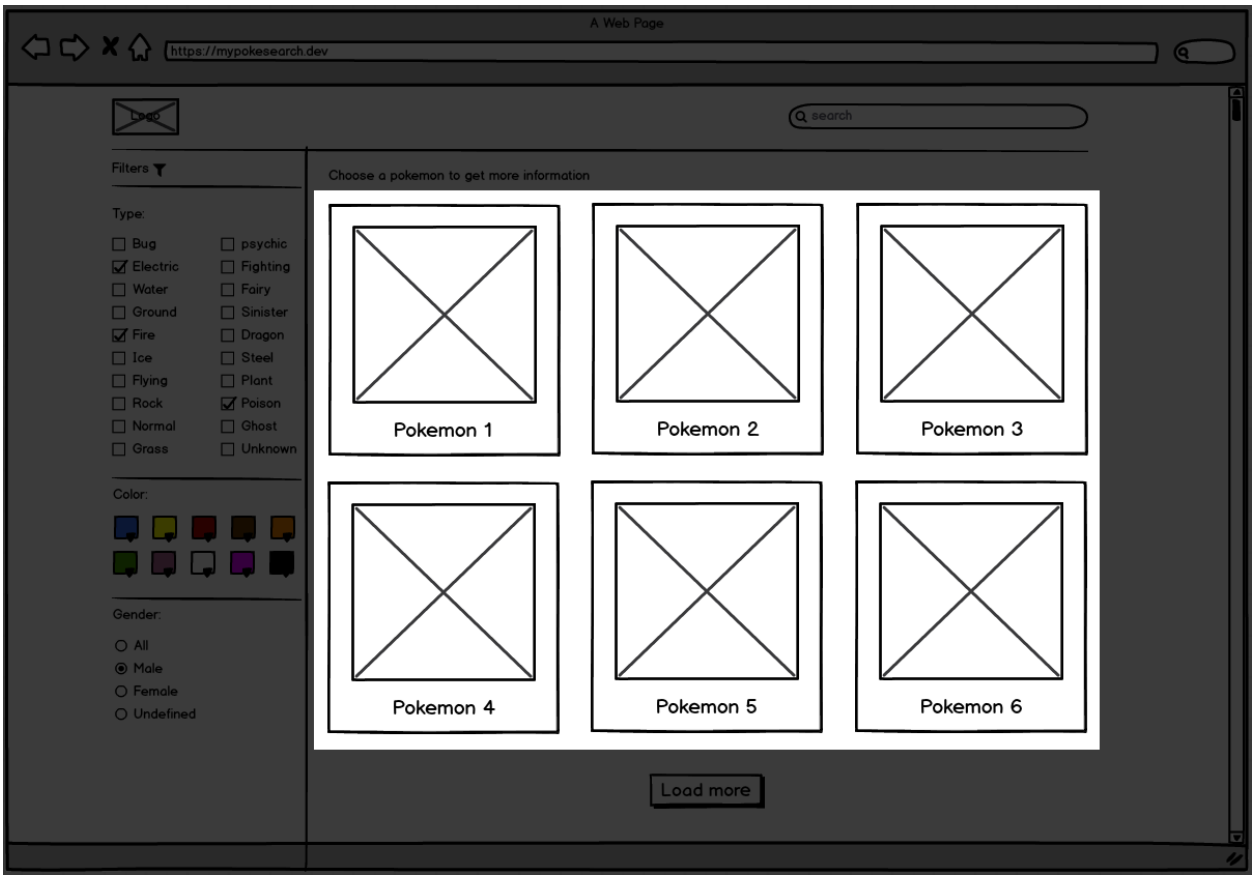


Functionality

Initial state

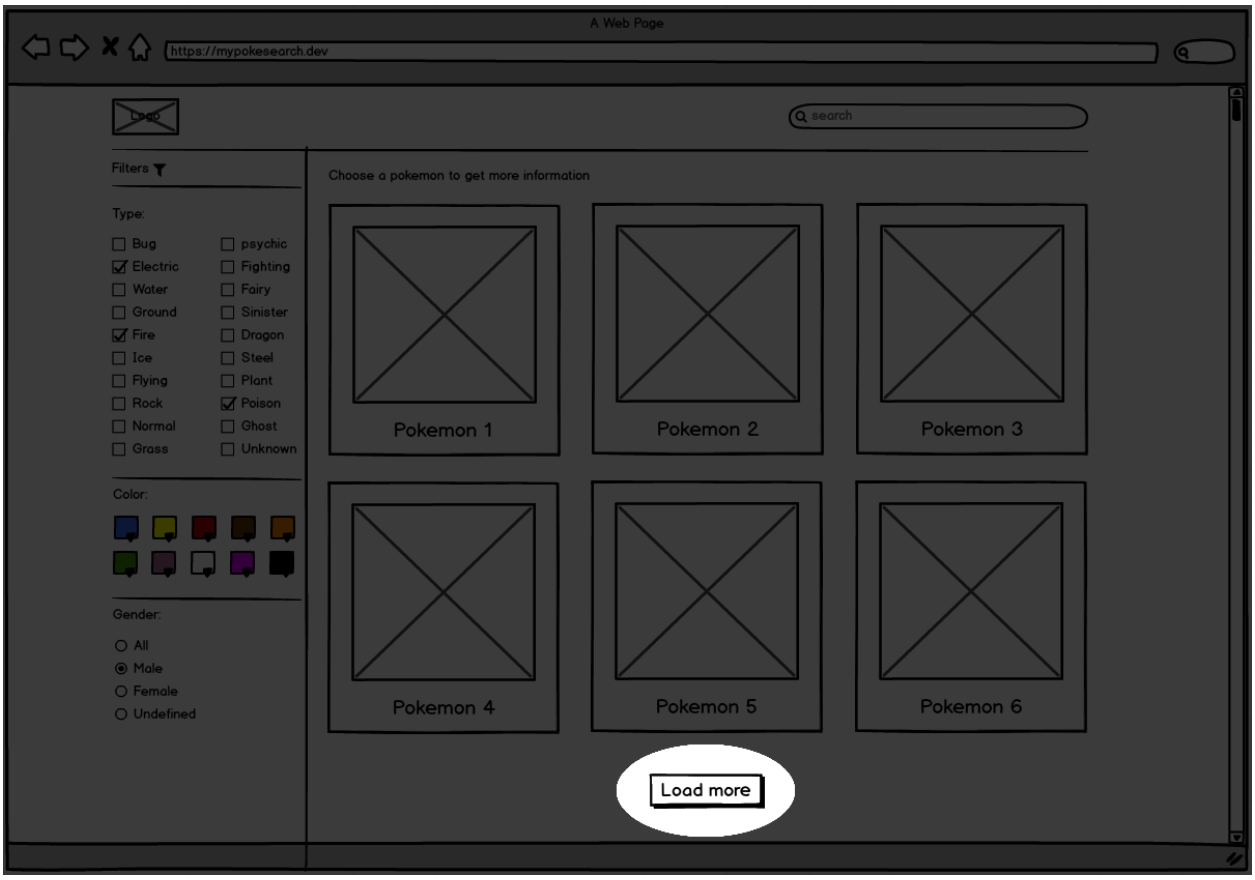
Endpoint: `GET /pokedex/national`

Although this endpoint lists more than 800 Pokemon, the initial state of the interface should display the first 20 Pokemon (with images) sorted by ID. The rest of the information should be stored in memory to list the remaining Pokemon (Load More) or for the search.



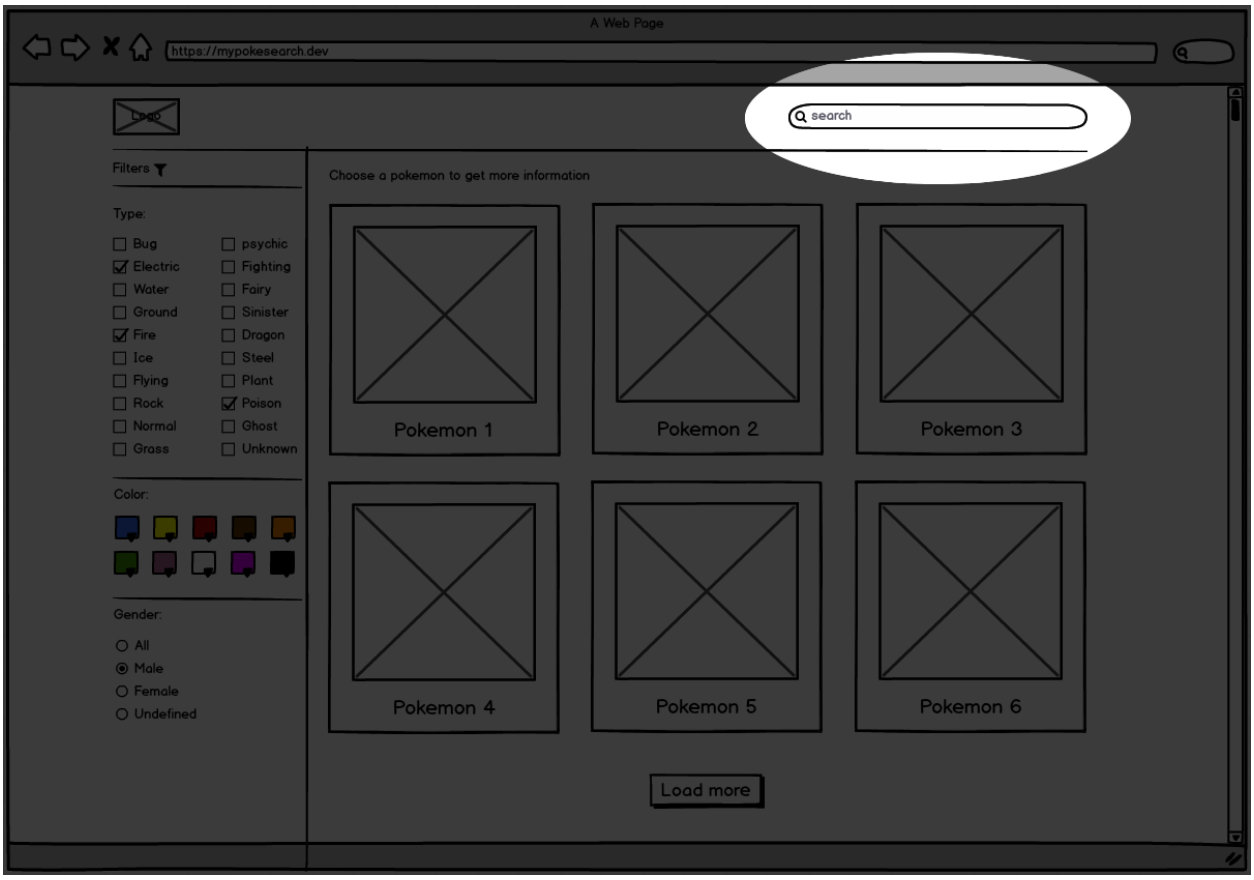
Load More

By pressing the "Load More" button, we should load the next 20 Pokemon into the interface, and so on. Once all Pokemon have been loaded, the button should disappear.

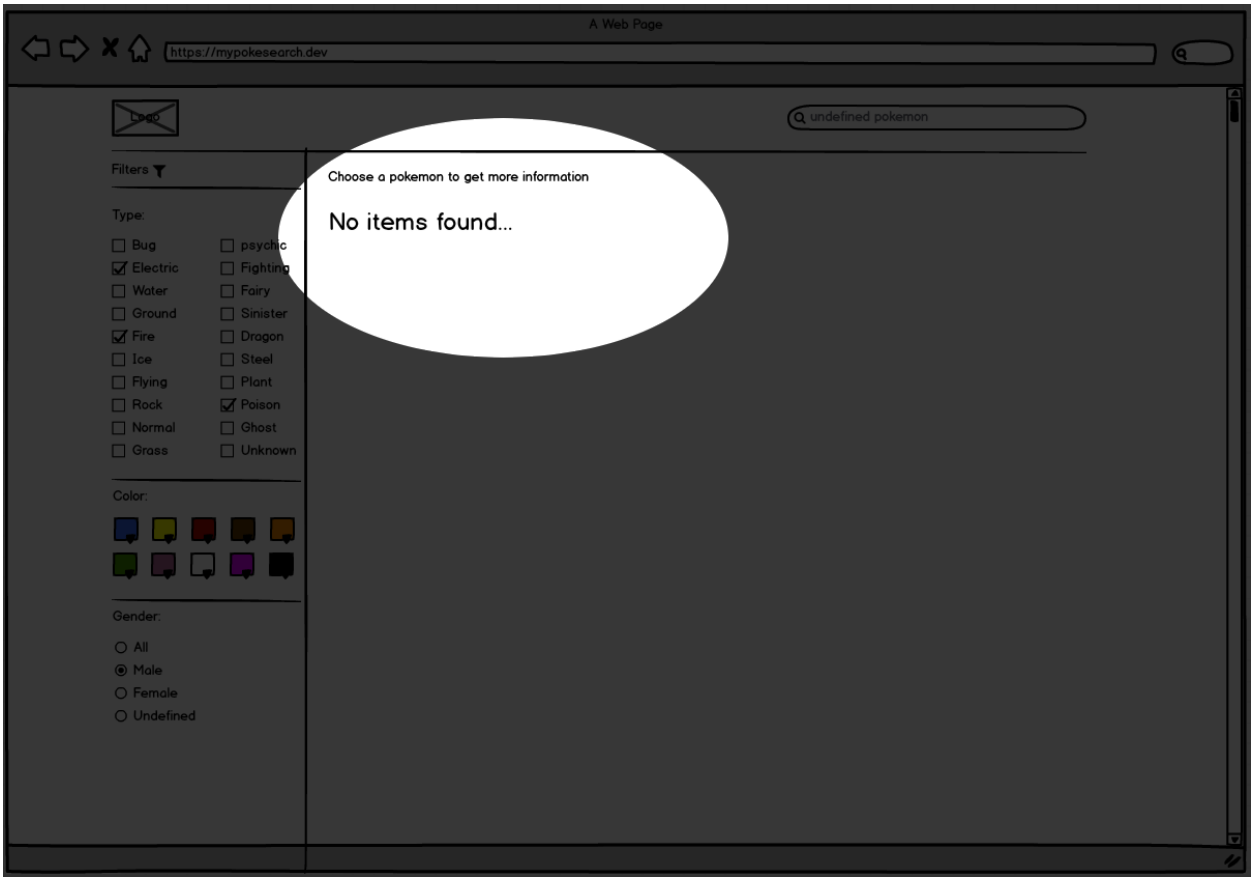


Search

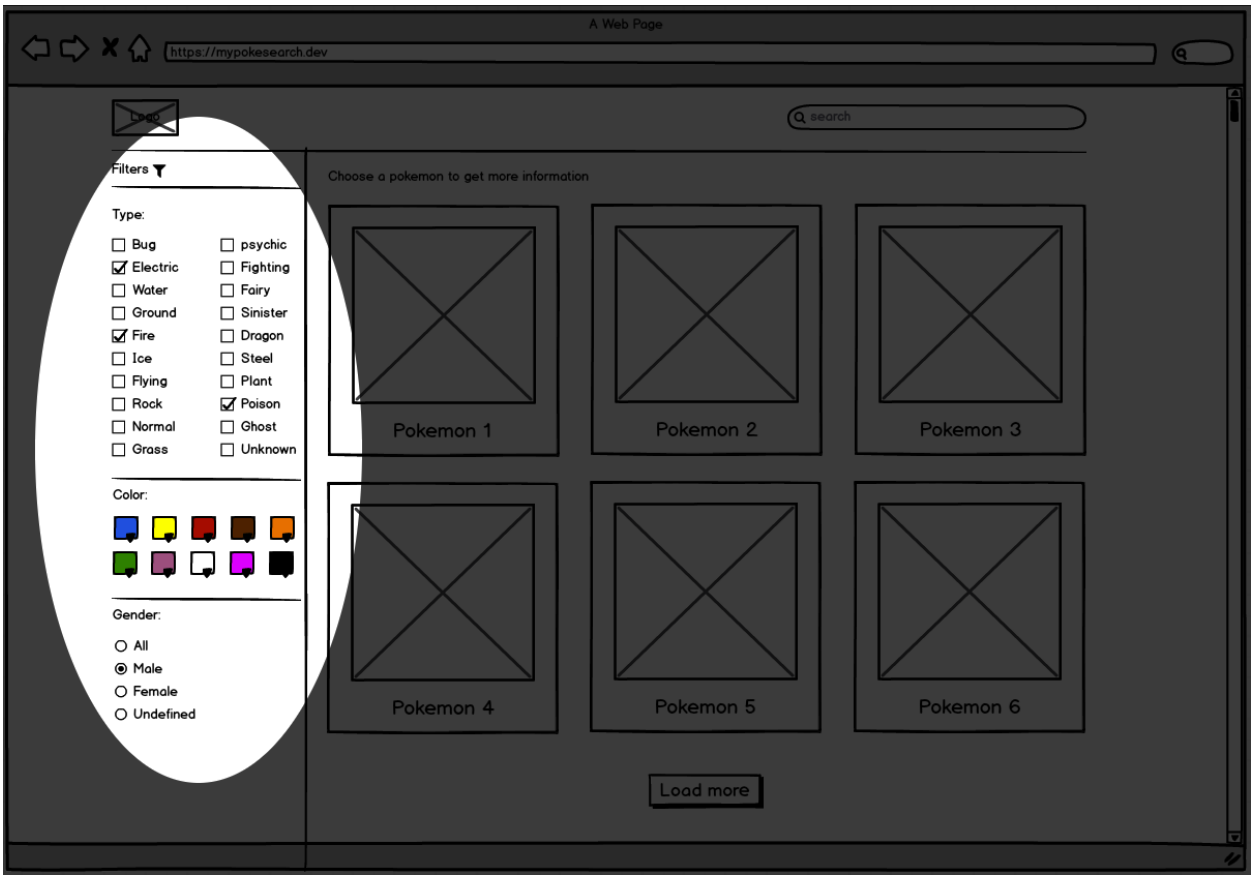
Upon entering a letter in the search bar, the search should immediately begin by filtering all items that match the written term. In this case, the content area should be cleared so that only the search elements appear. When the search box is empty, the application should return to the initial state. The search for elements should be done within the initial payload `GET /pokedex/national`. The search should also work when inserting the ID (number) of a Pokemon. Example: if I enter the number 2, list all Pokemon with the number 2 in their ID.



If no Pokemon matches the search term, feedback should be given to the user.


A Web Page
https://mypokesearch.dev
Logo
undefined pokemon
Filters
Type:
Bug
Electric
Water
Ground
Fire
Ice
Flying
Rock
Normal
Grass
psychic
Fighting
Fairy
Sinister
Dragon
Steel
Plant
Poison
Ghost
Unknown
Color:
Gender:
All
Male
Female
Undefined
Choose a pokemon to get more information
No items found...

Filters



By Type: When selecting one or more "types," the application should only retrieve Pokemon belonging to those "types." The complete list of types can be obtained from the following endpoint `GET /type`, and Pokemon belonging to a certain type are found in `GET /type/{type-id}`.

By Color: When selecting one or more "colors," the application should only retrieve Pokemon belonging to those "colors." The complete list of colors can be obtained from the following endpoint `GET /pokemon-color`, and Pokemon belonging to a certain color are found in `GET /pokemon-color/{color-id}`.

By Gender: When choosing a "gender," the application should only retrieve Pokemon belonging to that gender. The complete list of genders can be obtained from the following endpoint `GET /gender`, and Pokemon belonging to a certain gender are found in `GET /gender/{gender-id}`.

Filters must be additive, meaning I can search for Pokemon by type + color + gender at the same time.

Delivery

1. GitHub URL with the repository. Must be included a Readme with a short presentation and all that you consider (like the reason why you added a library, for example).
2. GitHub Pages URL from which the test can be viewed in a web browser.

Points to evaluate

Mandatory

1. Use **Less** to create styles, compile **Less** files to locally generate CSS, and add them to the project.
2. Less/CSS code must be writen using BEM metodologhy.
3. JavaScript must be written in Vanilla (No jQuery, React ...).
4. Use HTML5 and CSS3.
5. Third-party libraries are permited, Use CDNJS for include it.
6. Responsive: The given prototype is designed for mobile, but it should be responsive **Mobile First**. For example, on desktop screens, the product grid may change to 4 columns.

Optional

1. SEO, Accesibility & Performance
2. Documentation
3. Animations
4. Caching the payload
5. Loading during queries
6. Datalist in search field
7. Problems resolution

Bonus

Completing one or all of the following points will result in extra points when evaluating the test results.

1. Include an option (design and location at the developer's discretion) with which the application can be reset to its initial state, clearing filters, search, and loading initial elements.