


Supuesto práctico POKÉDEX


Resulta que soy muy fan del mundo Pokémon. Me he pasado todos los juegos desde la vetusta y monocroma versión de Game Boy hasta el Pokémon Go o todos los de Switch.

Como ya no sé qué hacer y además estoy aprendiendo a programar en Angular, he decidido hacerme mi propia pokédex de andar por casa. Como se los sabe todos de memoria, tiene más información que nintendo pero aún así, como no le va eso de escribirlo todo a mano, utilizará el API de Pokémon: PokeAPI.

[Home](#)[About](#)[API v2](#)[GraphQL v1beta](#)

PokeAPI proudly announces Beta support for GraphQL. Access our free console at beta.pokeapi.co/graphql/console and take a look at the [documentation](#) ✕

Sword and Shield data might be inaccurate and lacking in various aspects due to the fact that it is not taken directly from Nintendo's Pokemon ROMs. If you spot any mistake, please report it [here](#) ✕




The RESTful Pokémon API

Serving over 250,000,000 API calls each month!

All the Pokémon data you'll ever need in one place,
easily accessible through a modern RESTful API.

[Check out the docs!](#)

Try it now!

 [Submit](#)

Need a hint? Try [pokemon/ditto](#), [pokemon-species/aegislash](#), [type/3](#), [ability/battle-armor](#), or [pokemon?limit=100000&offset=0](#).

Direct link to results: <https://pokeapi.co/api/v2/pokemon/ditto>

Resource for ditto

```
▼ abilities: [] 2 items
  ▼ 0: {} 3 keys
    ▼ ability: {} 2 keys
      name: "limber"
      url: "https://pokeapi.co/api/v2/ability/7/"
      is_hidden: false
    <lot: 1
```

Para empezar el proyecto, simplemente haremos un MVP con el que podrá:

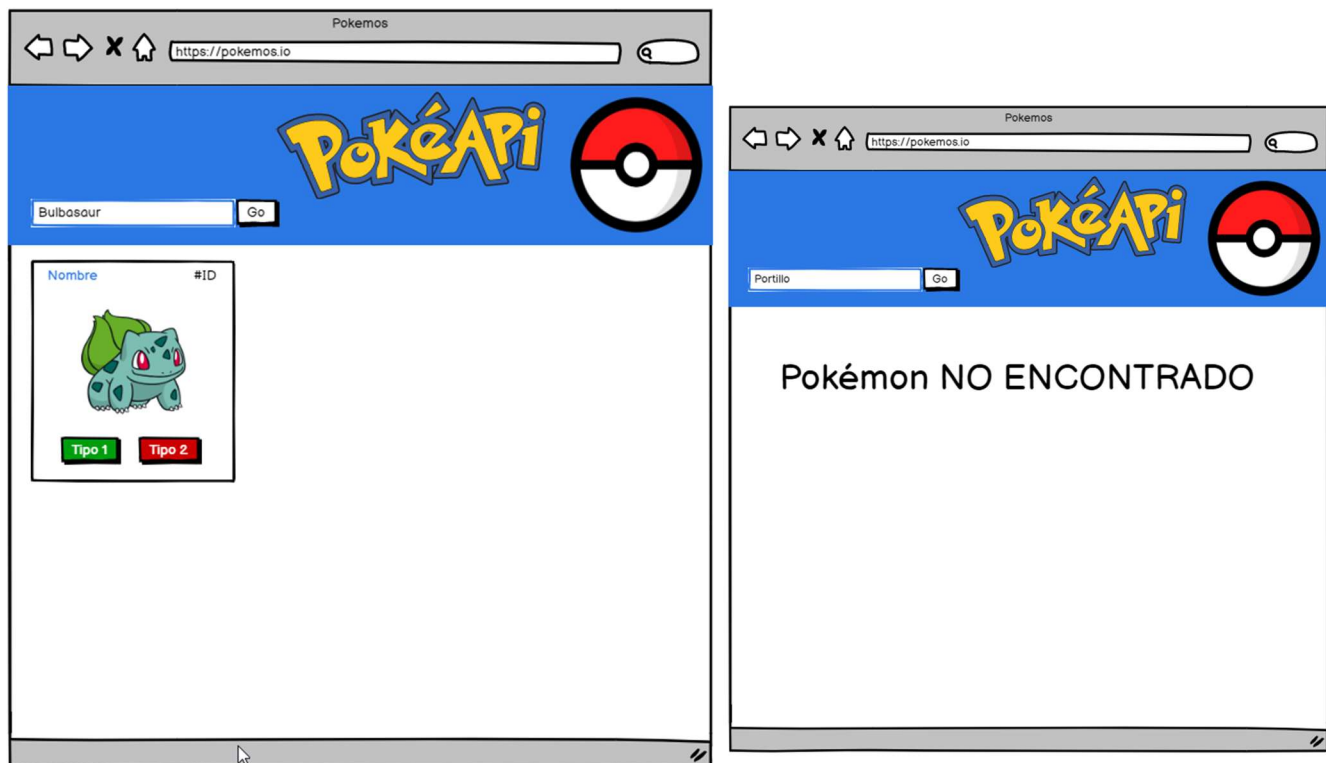
Listar unos cuantos Pokémons. Los que queramos, por ejemplo los 50 primeros (este 50 lo podremos ir cambiando). Que se vea de cada uno lo que vemos en la imagen:



Información importante:

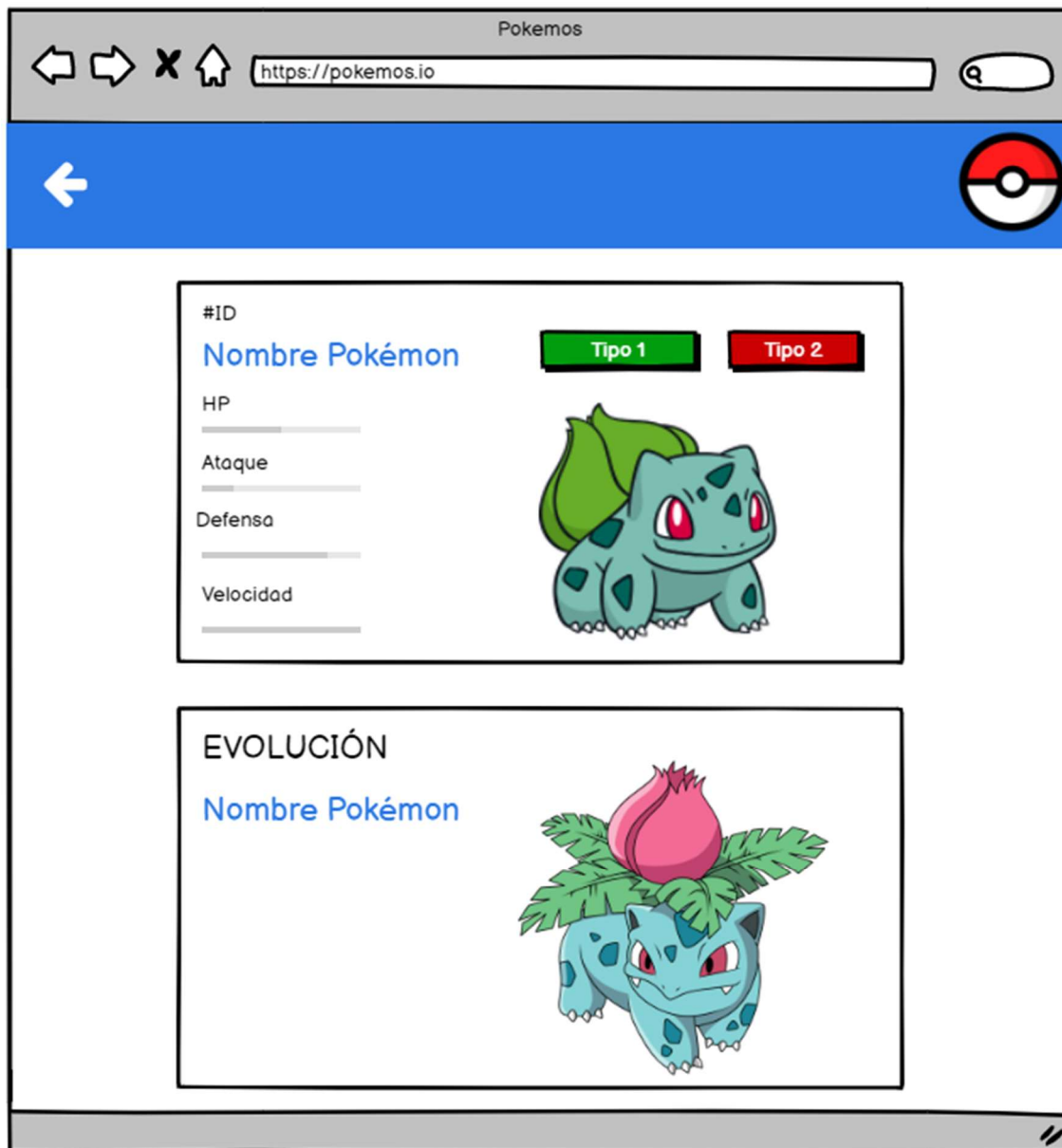
- La URL de l POkeAPI dónde podremos ver toda la información es: <https://pokeapi.co/>
- Tendremos que pensar en los componentes que tendremos que tener en el proyecto en función de lo aprendido durante el curso
- Tendremos que crear un servicio para obtener información de los Pokémon según lo que nos vaya haciendo falta, para este caso, necesitaremos un por ID para llamar a la URL correcta del API:
<https://pokeapi.co/api/v2/pokemon/:id>
- Necesitamos un modelo de Pokemon con la siguiente información:
 - o id: number
 - o name: string
 - o type: PokemonType[] -> PokemonType es un modelo con sólo un atributo ->name
- Los Pokemon serán de un tipo como mínimo y dos tipos como máximo
- Las imágenes de los Pokemon no hace falta guardarnoslas ya que siempre será una url que se montará con el ID. Por ejemplo:
<https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/other/dream-world/:id.svg>
- Si hacemos click sobre cualquiera de ellos iremos a una vista nueva dónde veremos el detalle del Pokemon

Buscar un Pokémon



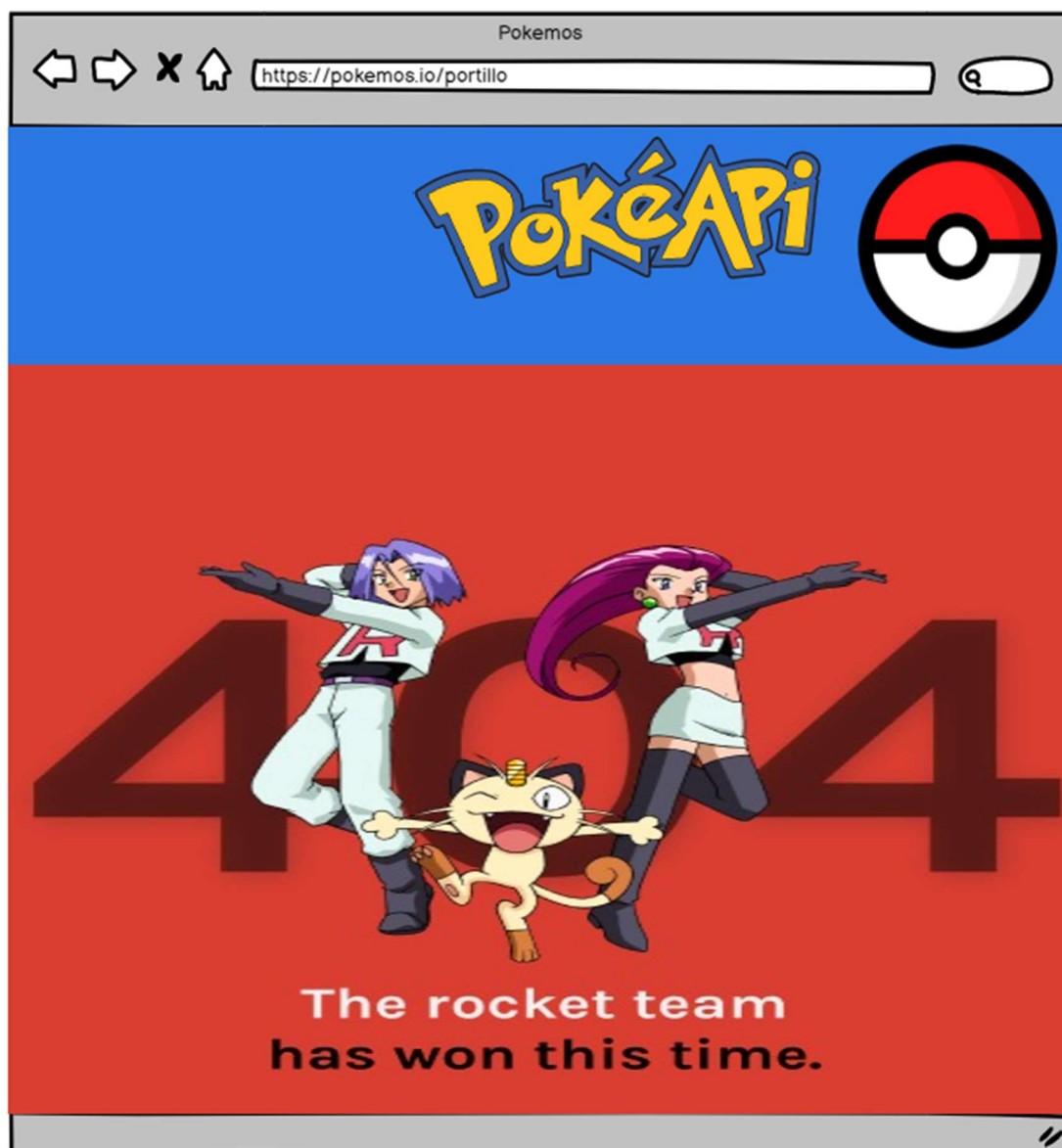
- Reutilizando lo que hemos hecho en el punto anterior, podremos buscar mediante un input, un Pokémon en concreto
- Para ello, tendremos que añadir a nuestro servicio una función más que sea igual que la anterior por ID pero que ahora nos permita buscar por NAME. La url es la misma pero se puede comprobar en PokeAPI
- Si hacemos click en la búsqueda, iremos al detalle del Pokémon
- Si borramos el input, se cargará de nuevo toda la lista

Detalle de Pokémon (<https://myapp:puerto/pokemonDetail/:id>)



- En esta ventana, principalmente recibiremos por URL el id del Pokémon y tendremos que hacer dos cargas
- Por un lado, tendremos el Pokemon recibido como parámetro mostrando los detalles que consideremos en función de la información que tenemos en el API. Podríamos ampliar el modelo de Pokémon.
- Por otro lado, tendríamos que mostrar el Pokémon al que evolucionaría, el primero de la cadena de evoluciones. Esta información en el API es compleja de obtener ya que para ello tendríamos que hacer 4 llamadas consecutivas:
 - o **Primero:** Llamada al Pokemon recibido como parámetro
 - o **Segunda:** Con los datos recibidos, obtener la url de la especie a la que pertenece el Pokemon([data.species.url](#)). Tendríamos que añadir a nuestro servicio, una función que dada una URL nos haga el get de lo que pasamos como parámetro.
 - o **Tercera:** Con los datos recibidos de la llamada anterior, tendríamos que coger la cadena de evoluciones del Pokemon([dataSpecies.evolution_chain.url](#)). Utilizando el servicio anterior para dada una url hacer una llamada.
 - o **Cuarta:** Con los datos recibidos de la llamada anterior, tendríamos que coger el nombre del Pokemon al que evoluciona ([data.chain.evolves_to\[0\].species.name](#)) y con ese nombre, hacer una llamada al servicio que tenemos para obtener el Pokemon por nombre. Este sería ya el Pokémon evolucionado, que nos guardaríamos y mostraríamos en el detalle usando el mismo componente.

Página no encontrada



Esta página la mostraríamos en caso de introducir una URL que no existiera