

Desarrollo de un servidor http de Pokédex con Node.js

Introducción

En este documento, se presenta el código desarrollado para crear un servidor de Pokédex utilizando Node.js. El servidor permite buscar información sobre Pokémon por su ID o nombre en cualquier idioma.

Código Desarrollado

`fetchPokemonData`: Esta función asíncronica lee el archivo `pokedex.json` que contiene los datos de los Pokémon y devuelve una promesa que resuelve los datos en formato JSON.

```
4  const fetchPokemonData = async () => {
5    return new Promise((resolve, reject) => {
6      fs.readFile("./pokedex.json", "utf8", (err, data) => {
7        if (err) {
8          reject(err);
9        } else {
10         resolve(JSON.parse(data));
11        }
12      });
13    });
14  };
```

`handleRequest`: Esta función maneja las solicitudes HTTP entrantes al servidor. Primero, se llama a `fetchPokemonData` para obtener los datos de los Pokémon. Luego, se analiza el parámetro de la solicitud para determinar si es un ID numérico o un nombre en cualquier idioma. Dependiendo de esto, se busca el Pokémon correspondiente en los datos obtenidos.

```
16  const handleRequest = async (req, res) => {
17    const pokemonData = await fetchPokemonData();
18    const requestParam = decodeURI(req.url.substring(1));
19
20    let pokemonInfo;
21
22    // Comprobamos si el parámetro es numérico (ID)
23    const isNumeric = !isNaN(requestParam);
24
25    if (isNumeric) {
26      // Buscamos por el ID
27      pokemonInfo = pokemonData.find(
28        (pokemon) => pokemon.id === parseInt(requestParam)
29      );
30    } else {
31      // Buscamos por el nombre en cualquier idioma
32      pokemonInfo = pokemonData.find((pokemon) => {
33        const names = Object.values(pokemon.name);
34        return names.some(
35          (name) => name.toLowerCase() === requestParam.toLowerCase()
36        );
37      });
38    }
```

Finalmente, se construye una respuesta JSON con la información del Pokémon encontrado y se envía al cliente. Le he añadido emojis y un par de ajustes para entenderlo mejor y mejorar el diseño (ver código para comprobarlo).

```
response = {
  "Id del pokemon 🟠": pokemonInfo.id,
  "Tipo 🔥💧🌿": pokemonInfo.type,
  "Vida (HP) 🩹": pokemonInfo.base.HP,
  "Ataque 🖊️": pokemonInfo.base.Attack,
  "Defensa 🛡️": pokemonInfo.base.Defense,
  "Velocidad de ataque 🏃": pokemonInfo.base["Sp. Attack"],
  "Velocidad de defensa 🏃": pokemonInfo.base["Sp. Defense"],
  "Velocidad 🚗": pokemonInfo.base.Speed,
};
}
res.writeHead(200, { "Content-Type": "application/json" });
res.end(JSON.stringify(response));
```

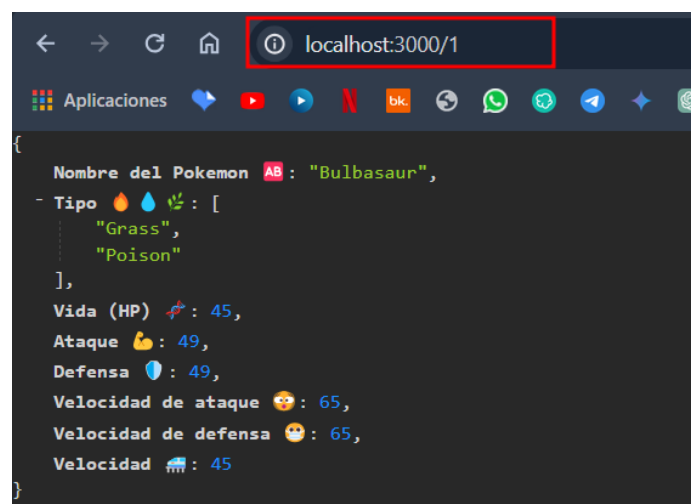
Server: Se crea un servidor HTTP utilizando el módulo http de Node.js y se asigna la función `handleRequest` para manejar las solicitudes entrantes en el puerto 3000.

```
85 const server = http.createServer(handleRequest);
86
87 server.listen(3000, () => {
88   console.log("Escuchando en el puerto 3000");
89 });
```

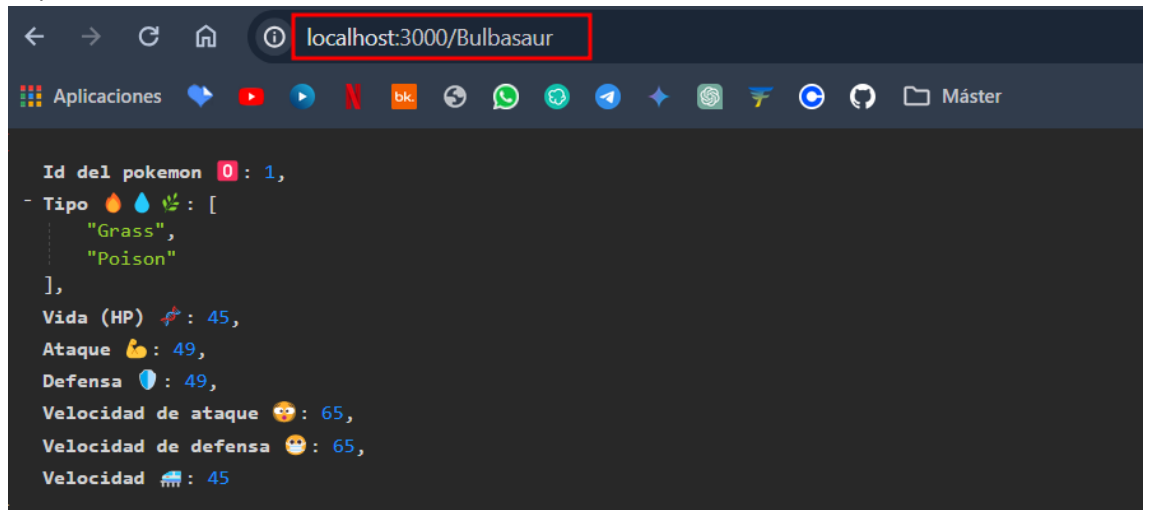
Casos de Prueba y Capturas de Pantalla

Para diferenciar ID y nombre, en el listado del json, he hecho una pequeña modificación para entender mejor el resultado ya que si se buscaba por ID, me gustaba que mostrara su nombre y si se mostraba su nombre, me gustaba que se mostrara el ID.

1. Buscar por ID: Se buscará el Pokémon con el ID 1.
 - Resultado Esperado: Información del Pokémon Bulbasaur.
 - Captura de Pantalla:

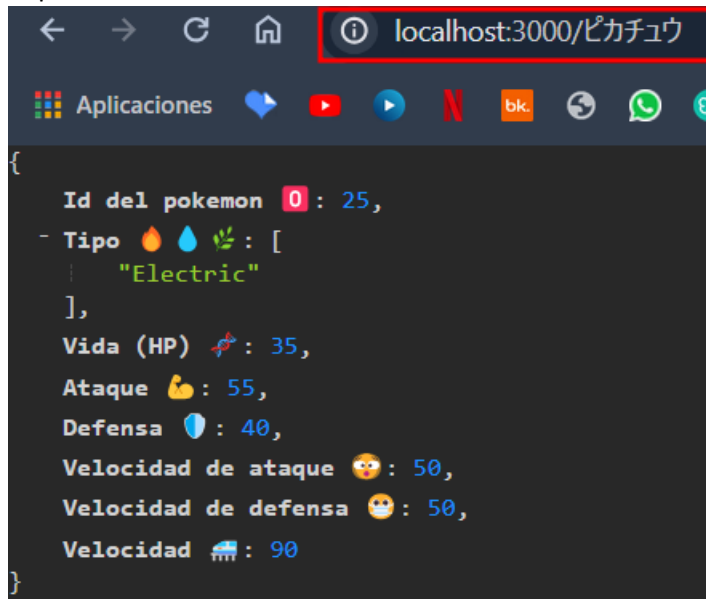


2. Buscar por Nombre en Inglés: Se buscará el Pokémon con el nombre "Charmander".
- Resultado Esperado: Información del Pokémon Charmander.
 - Captura de Pantalla:



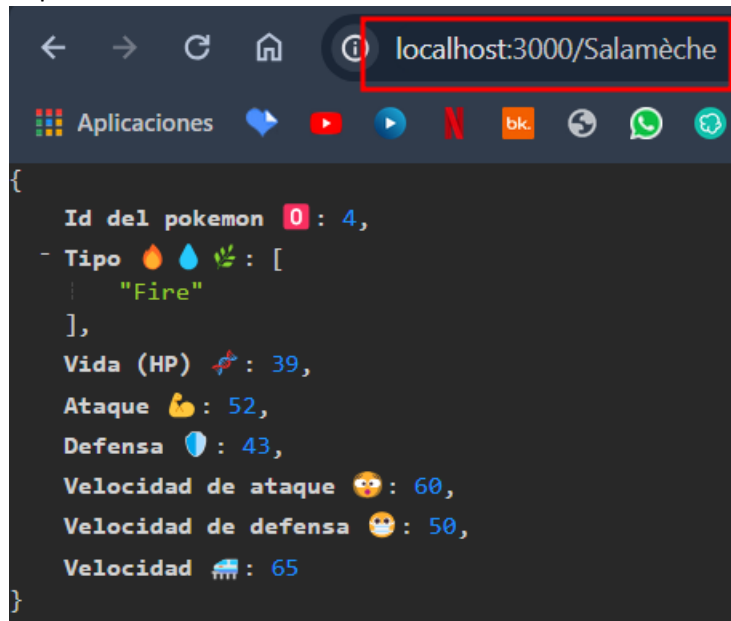
```
Id del pokemon 0: 1,  
- Tipo 🌿💧: [  
  "Grass",  
  "Poison"  
],  
Vida (HP) 🩸: 45,  
Ataque 🖐️: 49,  
Defensa 🛡️: 49,  
Velocidad de ataque 🏃: 65,  
Velocidad de defensa 🏃: 65,  
Velocidad 🏃: 45
```

3. Buscar por Nombre en Japonés: Se buscará el Pokémon con el nombre "ピカチュウ" (Pikachu).
- Resultado Esperado: Información del Pokémon Pikachu.
 - Captura de Pantalla:



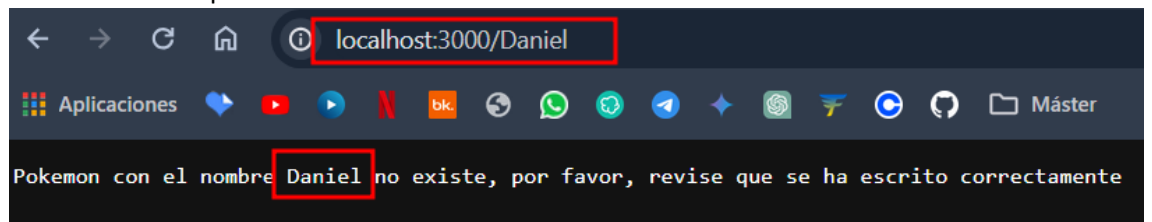
```
{  
  Id del pokemon 0: 25,  
  - Tipo 🌿💧: [  
    "Electric"  
  ],  
  Vida (HP) 🩸: 35,  
  Ataque 🖐️: 55,  
  Defensa 🛡️: 40,  
  Velocidad de ataque 🏃: 50,  
  Velocidad de defensa 🏃: 50,  
  Velocidad 🏃: 90  
}
```

4. Buscar por Nombre en Francés: Se buscará el Pokémon con el nombre "Salamèche" (Charmander en francés).
- Resultado Esperado: Información del Pokémon Charmander.
 - Captura de Pantalla:

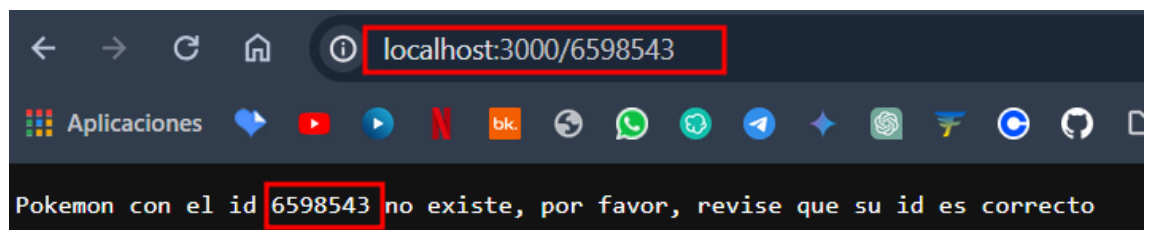


5. Buscar Pokémon no Existentes: Se buscarán Pokémon con ID o nombres que no existen en la Pokédex.
- Resultado Esperado: Mensaje de error indicando que el Pokémon no existe.
 - Captura de Pantalla:

Con un nombre que no existe:



Con un ID que no existe:



Conclusiones y Observaciones:

Gracias a el servidor de Pokédex desarrollado, puedo buscar Pokémon por su ID o nombre en cualquier idioma, proporcionando información detallada sobre cada Pokémon encontrado.

El código está bien estructurado y modularizado, lo que facilita su comprensión y mantenimiento, además, con comentarios, indicando en todo momento la explicación del procedimiento del código.

Se manejan adecuadamente los errores, como la búsqueda de Pokémon no encontrados, mostrando su nombre o id no correspondientes con ninguno de la Pokédex.

Finalmente, se ha hecho una mejora de la resolución del ejercicio implementando con ello emojis y una diferente esquematización (a modo diseño y visual) gracias a una extensión que utilizo para Google Chrome.

URL DEL CÓDIGO:

<https://github.com/danisentamans/Entregables-BACKEND.git>