W4 - Projets optionnels

W-WEB-405

Pokedex

Gotta Code' Em All

1.2.2

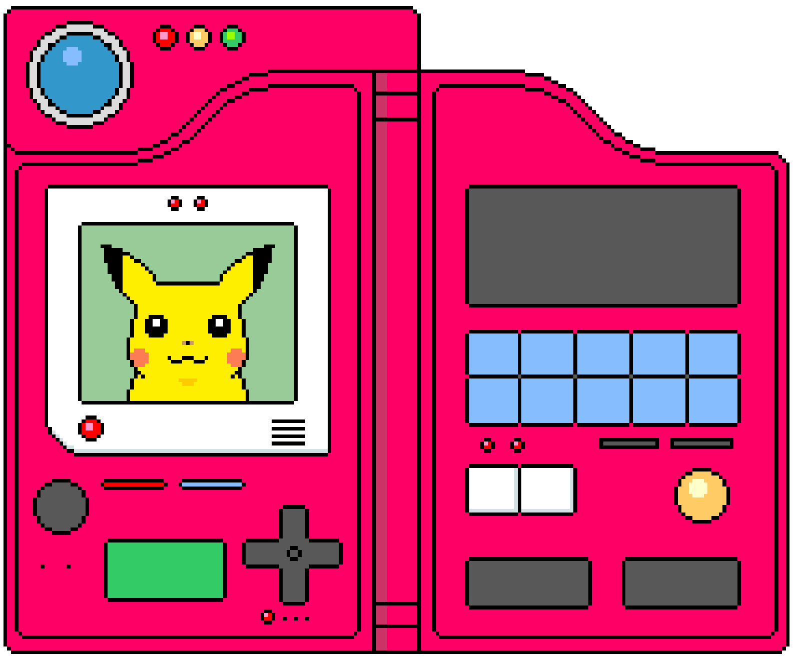
Pokedex

delivery method: pokedex on Github
language: Vue.js, Laravel

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

Refer to the general documentation of how to write a README.md in your courses to improve the quality of your project's delivery.

Let's create the mythical Pokedex from the video game Pokemon.



It is an encyclopedia listing all Pokemons' data (characteristics, types, evolutions, ...).

Do not be in a hurry to go on an adventure just yet, and carefully read the entire subject before you start.

Backend

To fetch data from the database we use an API as a communication system between the back-end and front-end.

That's why you will build your own API using Laravel, or another framework that suits you best

You are given a SQL database containing all needed data.

Authentication

An user should be able to register with:

- username
- password

Laravel's authentication may help you

Then, the user can choose a profil icon in the list:

- pokeball
- pikachu
- charmander
- bulbasaur
- squirtle
- jigglypuff

When signed-up and once a day when logging-in, the user receive a random pokemon in his team (max 6)

Routes

You must implement specific routes. You can find some details in another file.

Search

You have to implement, at least, two search mechanisms for:

- looking for a pokemon
- looking for an user

Frontend

The user can access a specific pokemon info by clicking on it from everywhere:

- Pokedex
- My Team
- Trade

The `return` arrow should accordingly returns to the previous menu.

Zeplin and Tailwindcss might interest you

Bonus

For this project, we recommend that you containerize your application using Docker so that your project can be easily tested by the pedagogical team and your entire group can easily contribute to the solution.

You need to create a Dockerfile at the root of your web application and your API allowing you to build and run at least two Docker images: one will make your application server accessible, and the other the front end on port 8080 of your localhost.

Get a close look at these examples:

```
Terminal
∼/W-WEB-405> cd ~/pokedex/pokedex-api
docker build -t morty/pokedex-api .
docker run -p 49160:4242 -d morty/pokedex-api
```

```
Terminal
∼/W-WEB-405> cd ~/pokedex/pokedex-front
docker build -t morty/pokedex-front .
docker run -it -p 8080:8080 -rm -name pokedex-front-1 morty/pokedex-front
```

To share your work, you have to describe precisely the steps for your containers (installation, update, troubleshooting, etc...) in an appropriate file. Refer to the general documentation.

You can also go further in this project with:

- a true trading system (rather than just sending)
- to handle trading evolution
- multi-lingual website, with Vue i18n
- a way to capture a new pokemon
- a battle system
- ...