# Practice Challenge: Pokemon Searcher

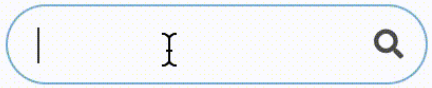
 (https://github.com/learn-co-curriculum/react-hooks-practice-pokemon-searcher)  (https://github.com/learn-co-curriculum/react-hooks-practice-pokemon-searcher/issues/new)

# Pokemon Searcher

## Add a Pokemon!

| Name | hp | Front Image URL | Back Image URL |
|---|---|---|---|
| Name | hp | url | url |

Submit



ivysaur
60 hp

venusaur
80 hp

charmander
39 hp

charmeleon
58 hp

charizard
78 hp

squirtle
44 hp

wartortle
59 hp

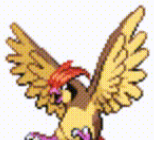
blastoise
79 hp

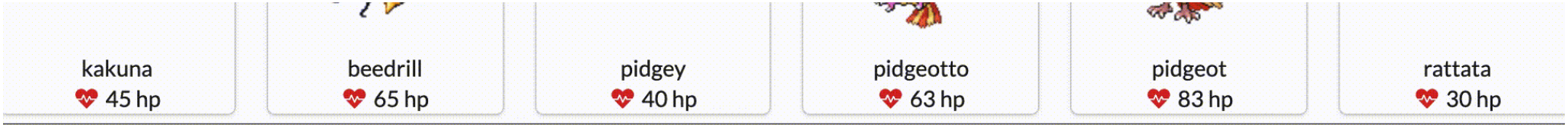
caterpie
45 hp

metapod
50 hp

butterfree
60 hp

weedle
40 hp






(Note: If the gif isn't working, view it directly by navigating **here (https://curriculum-content.s3.amazonaws.com/react/pokemon.gif)** )

# Setup

All the information about Pokemon can be found in the `db.json` file. We'll be using `json-server` to create a RESTful API for our database.

Run `npm install` to install our dependencies.

Then, run `npm run server` to start up `json-server` on `http://localhost:3001` .

In another tab, run `npm start` to start up our React app at `http://localhost:3000` .

Before you start building out the application, the first step that you should take is to draw out your component hierarchy. This will tell you how components can pass data to each other as well as where that information should be stored.

# Project Goals

- Create an index displaying Pokemon 'cards'
  - Render each Pokemon name, sprite, and hp in a card
  - *When clicked*, the card should toggle between displaying the front and back sprites
- Allow users to search a Pokemon by its name in order to narrow down the cards shown on the page
- Wire up the form to add a missing Pokemon (Bulbasaur is missing, and you can probably intuit the image links to use based on the data you have). Since there aren't any validations, you may have to manually remove additions from the `db.json` file if you make a mistake on a POST request, etc. When a new Pokemon is added, it should show on the page without having to refresh.

# Project Requirements

- components using state and props
- re-renders based on client-side events

- can both get data from, and post data to the json server/database

# Trying to Figure Out Where to Start

For the most part, the framework for this application is already made, with a simple json server, and a React app, with all the components needed. When you run `npm start` a json server will spin up and you can access the Pokemon info at `http://localhost:3001/pokemon`, and your React application will also start, on port 3000.

Given a list of deliverables, our goal is to navigate the existing code base and implement several new features. Take some time to familiarize yourself with the structure of the application, and think about where you will be making changes in order to implement the new features. While you could solve this lab without creating new components, you are encouraged to do so if it makes sense to do that at any point.

Be sure to use good programming practices, such as clear variable names and single responsibility functions. We want our react app to be clean, easy to read and understand, and of course, easy to to debug, or even add more features to later.