JavaScript App (Pokedex)

Project Overview

- The Pokedex a small web app built with Javascript, HTML, CSS, jQuery, and Bootstrap framework. Users can search through a pokemon collection that is loading from an external API. Details can be displayed in a modal by clicking on a pokemon's name.
- Objective: To build a small web application with HTML, CSS, and JavaScript that loads data from an external API and enables the viewing of data points in detail
- Key features:
- ● Load data from an external source (API)
- ● View a list of items
- ● On user action (e.g., by clicking on a list item), view details for that item

User Goal:
Users should be able to view a list of data and see more details for a given data item on demand.

Technical requirements

- The app must load data from an external API; for instance, the Pokémon API.
- The app must display a list of items loaded from that API after the page is loaded.
- The app must enable the viewing of more details for a given list item (like a Pokémon) on

demand, such as when clicking on a list item.

- The app must have CSS styling.
- The JavaScript code must be formatted according to ESLint rules.
- The app must use at least one additional complex UI pattern, such as a modal, for details or

touch interactions.

- The app must not throw any errors when being used.
- The app should be deployed to a publicly accessible platform like GitHub Pages
- The app must work in Chrome, Firefox, Safari, Edge, and Internet Explorer 11.

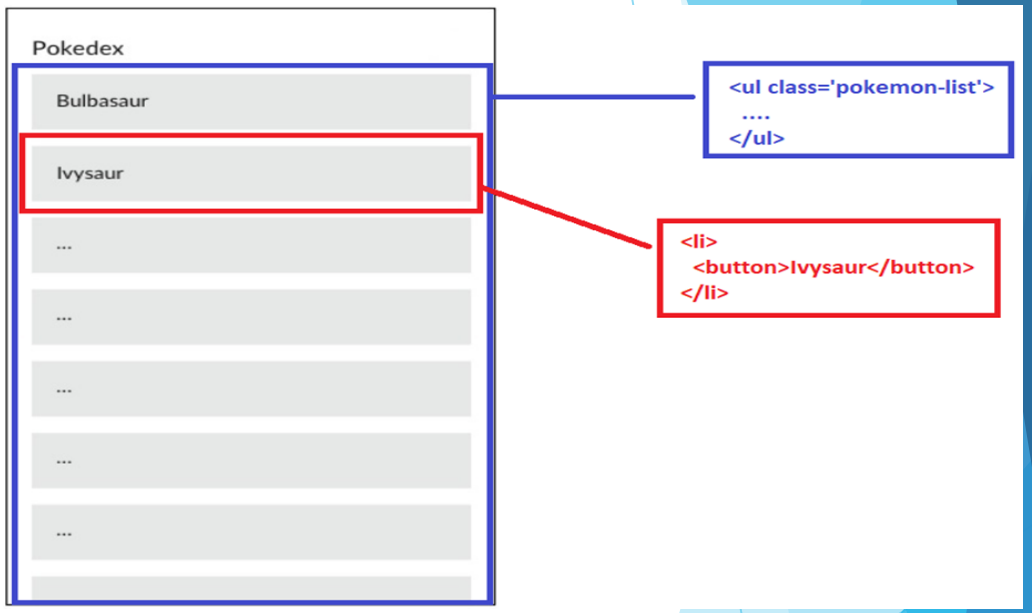
Purpose and Context/Objective

- Purpose and Context:
- The Pokedex was a personal project I built as part of my web development course at CareerFoundry to demonstrate my mastery of full-stack JavaScript development.

- Objective:
- The aim of the project was to have an ambitious full-stack project I can add to my professional portfolio. The problem I wanted to solve is to build an app that showcases pokemon and the ability to search one up and look at their stats.

DOM Interaction

- Document Object Model (DOM) is a programming interface for HTML. It represents an HTML web page as JavaScript “objects,” allowing JavaScript to interact with, and even make changes to, the HTML page.

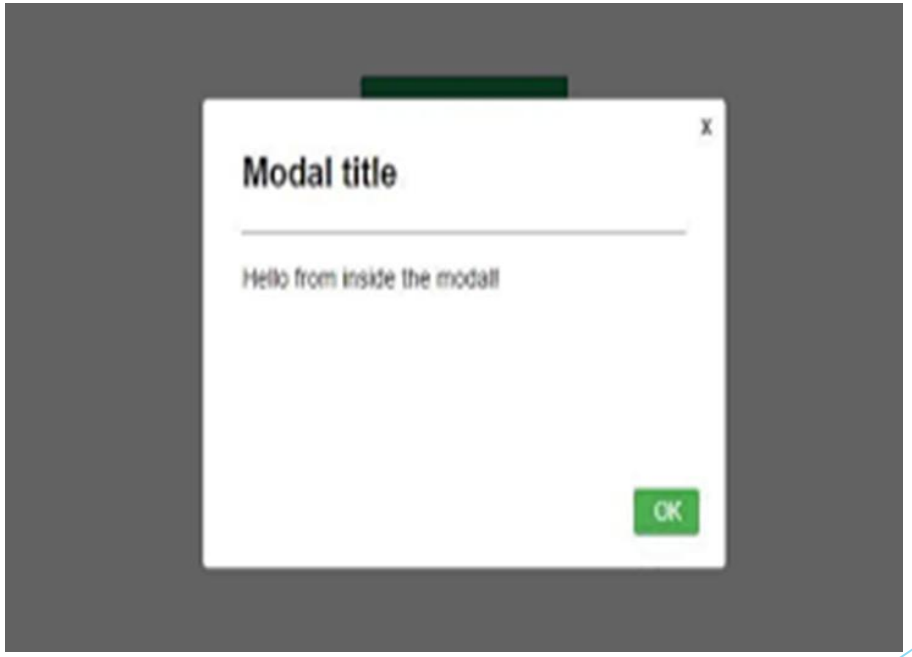


Example:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple DOM example</title>
  </head>
  <body>
    <section>
      <img src="dinosaur.png" alt="A red Tyrannosaurus Rex: A two legged dinosaur standing upright like a human, with small arms, and a large head with lots of sharp teeth.">
      <p>Here we will add a link to the <a href="https://www.mozilla.org/">Mozilla homepage</a></p>
    </section>
  </body>
</html>
```

Modals

- The Pokedex app utilizes modals when you select a pokemon in the app. A modal or dialog box is a message box that allows further interactivity of the page without navigating away from the current content in the browser. When you select a pokemon in the app, a modal pops up displaying information about said pokemon.



jQuery

- Query is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax. jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

- Example:
- function showModal(pokemon) {
- let modalTitle = $(".modal-title");
- let modalBody = $(".modal-body");
- modalTitle.empty();
- modalBody.empty();

```
console.log($) // function()
console.log(typeof $) // "function"

console.log($()) // null
console.log(typeof $()) // "object"

try {
  console.log($().jquery)
  console.log(typeof $().jquery)
} catch (e) {
  console.log(e)
} // TypeError: "$() is null"

// In jQuery, $() is an alias of jQuery():
try {
  console.log(jQuery)
  console.log(typeof jQuery)

  console.log(jQuery())
  console.log(typeof jQuery())

  console.log(jQuery().jquery)
  console.log(typeof jQuery().jquery)
} catch (e) {
  console.log(e)
} // ReferenceError: "jQuery is not defined"
```

Technologies used

- HTML
- CSS
- JavaScript
- External API

```
{layout:set name="scripts"}
    <script >
        var inputs = document.querySelectorAll( '.inputfile');
        Array.prototype.forEach.call( inputs, function(input)
        {
            var label = input.nextElementSibling, labelVal = label.innerHTML;

            input.addEventListner('change', function(e)
                {
                    var fileName = '';
                    if(this.files && this.files.length > 1)
                        fileName = (this.getAttribute('data-multiple-caption') || '').
                            replace
                    else
                        fileName = e.target.value.split('\').pop();

                    if(fileName)
                        label.querySelector('span').innerHTML = fileName;
                    else
                        label.innerHTML = labelVal;
                });
        });
    </script>

{/layout:set}
```

- JavaScript:
- JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. JavaScript gives web pages interactive elements that engage a user. Common examples of JavaScript that you might use every day include the search box on Amazon, a news recap video embedded on The New York Times, or refreshing your Twitter feed.

- Example:
- ```
  function showDetails(pokemon) {
  ```
- ```
    loadDetails(pokemon).then(function () {
  ```
- ```
      console.log(pokemon);
  ```
- ```
      showModal(pokemon);
  ```
- ```
    });
  ```
- ```
  }
  ```

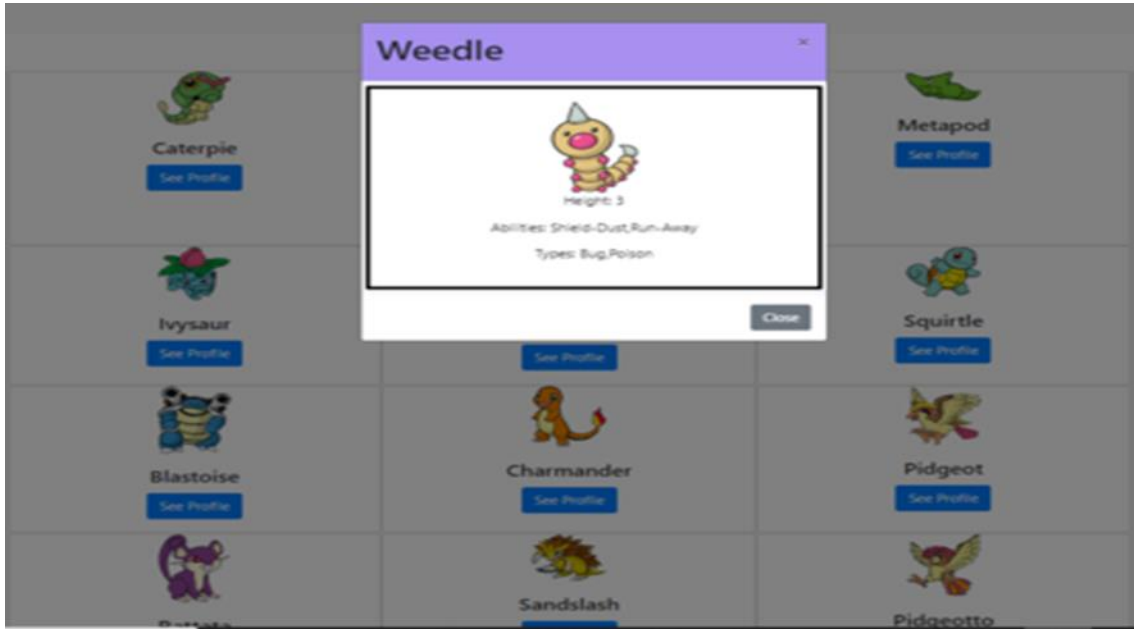
- External API:
- An external or open API is an API designed for access by a larger population as well as web developers. An application programming interface is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software.

- Example:
- {
- "name": "bulbasaur",
- "url": "https://pokeapi.co/api/v2/pokemon/1/"
- },
- {
- "name": "ivysaur",
- "url": "https://pokeapi.co/api/v2/pokemon/2/"
- },

My Application

- Here is my link to my GitHub repository:

- https://github.com/1998-creator/simple-js-app



Challenges

- Besides my portfolio, this was my first project that I built so a lot of the material involved was new to me and took time to learn.
- Understanding how to implement the API into the project
- Taking my Javascript I wrote previously and translating it into jQuery
- Updating the forEach loops to work within the IIFE
- Making sure I was learning the material in the project while also keeping a good schedule

Retrospective

- What went well:
- After the completion of the project, no errors were found. The app displays a list of items loaded from the API after the page is loaded. The app properly loads the specific colors for each pokemon type that is selected by the user.

- What didn’t go well:
- It took some getting use to when it came to understanding some of the functions and loops for the app.

- What can be improved:
- Some more CSS stylings could be added to make it look more like a legit app. Some more in-depth information of each pokemon could be added for the user to see more details about each pokemon.