**JavaScript Project 1**

**Enjoy the Outdoors: National Parks Search**

Project Overview: In this project, you'll build a simple website that allows users to search for national parks in the U.S. based on location (state or territory). You’ll be using HTML, CSS, and JavaScript to accomplish this. Don’t worry if you get stuck—this is meant to stretch your skills!

The main focus will be creating a search page where users can find national parks by state. All the data for the national parks is provided for you in a file, so your job is to connect that data to your webpage and make it interactive.

Duration: 1 day

**Project Requirements**

You will build two pages:

1. Homepage: A simple page that introduces the purpose of the site and includes a link to the National Parks search page.
2. National Parks Search Page: This page will let users search for national parks by selecting a state from a dropdown menu and/or park type from a separate dropdown menu.

**Setup**

* You will be provided with data files (in a zip file) that contain information about U.S. national parks. These files will be included in your project and used to display the parks on your search page.

**Data Files to Use**:

* locationData.js: This file contains an array of U.S. states and territories for your dropdown menu.
* nationalParkData.js: This file contains information about each national park, including the state it is located in and its park type.
* parkTypesArray.js: This file contains an array of possible park types that the user can select from.

You do not need to modify these files—just use them to help populate your page.

**Important!** These files are already written in JavaScript and include arrays of data. You’ll load them into your project just like you would load a script file. We haven’t learned about JavaScript modules yet, so we’ll work around that by including all of these files in the same HTML page as your custom script.

**Steps to Complete the Project**

Here’s how you can approach this project:

1. **Set up your project folder**

* Create an index.html for your homepage and a separate nationalParks.html for your search page.
* Create a folder for your CSS and another one for JavaScript files (you can call them css and scripts).
* Place the data files (locationData.js and nationalParkData.js) in the scripts folder. You'll also write your own JavaScript file here.

2. **Create the Homepage (index.html)**

* This page can be simple. Write a short paragraph about what the site does (it helps people find national parks).
* Add a button or link to navigate to the "National Parks Search" page.

3. **Create the National Parks Search Page (nationalParks.html)**

* Set up a basic layout for the search page. Use a dropdown menu where the user can select a state.
* Dropdown for Locations: Use the locationData.js file to populate this dropdown with all the states. You can access the list of states from the array in that file.
* Dropdown for Park Types: Use the parkTypesArray.js file to populate this dropdown with the various park types (like National Park, Recreation Area, etc.).
* Add a "Search" button.

4. **Display Search Results**

* When the user selects a state and/or a park type and clicks "Search", display a list of national parks that match the selected state and/or park type.
* Use the nationalParkData.js file to find the parks in the selected state. You can do this by checking the "State" property and “LocationName” property of each park in the array.
* Display the park names and any other details you find interesting (like their location or park type).

**Hints & Tips**

* **Populating the Dropdown**:
  + Writing out all 50 states in HTML would be a lot of work. Instead, you can use JavaScript to dynamically create the dropdown options. This is not only faster but also keeps your code clean and easy to manage.
  + Hint: Use a loop to go through the locationData array and create the <option> elements for the dropdown. innerHTML can help you add these options directly into the dropdown.
* **Styling**:
  + Use **Flexbox** or **Bootstrap** to style the search form and results to make it look presentable and responsive across different screen sizes.
* **Searching for Parks by State**:
  + Once the user selects a state, you’ll need to grab that input and compare it to the data you have about the parks. This will help you find parks that match the selected state.
  + Hint: The filter() method on arrays can be really helpful here because it allows you to return only the parks that match the selected state. Alternatively, find() could work if you are looking for a specific match.

**Stretch Goal: Mountains Information Page**

As an extra challenge, you can create a third page that allows users to explore information about different mountains. This is an optional feature designed to stretch your skills and enhance your project. If you have completed the main requirements, you can work on this feature as a bonus.

**Instructions for the Mountains Page**

**Objective**: Add a page where users can look up information about mountains.

**Steps to Complete the Stretch Goal**

1. **Create the mountains.html Page**
   * Set up a new page called mountains.html similar to the national parks search page.
   * Be sure to link to this page from the main page.
   * Add a dropdown where users can select from a list of 48 mountains (the data will come from the mountainData.js file).
2. **Populate the Mountain Dropdown**
   * Use JavaScript to load the mountain names from the mountainData.js file and display them in the dropdown.
3. **Display Mountain Information**
   * When the user selects a mountain, display detailed information about that mountain. This can include:
     + Mountain Name
     + Description
     + Elevation
     + Any other information you find interesting from the data file.
4. **(Optional) Add Advanced Features**
   * **Display Images**: If the mountain data includes an image filename, display the corresponding image alongside the mountain information.