Lab Summary

The goal of this lab is to use JavaScript Objects Arrays, jQuery and Promises/Fetch to display in a Web Form, Country information that is stored in multiple JSON data sources. In this lab, you will use the JavaScript techniques we discussed in class to add the functionality required to bring this Web form to life! This assessment will help you practice and improve upon your skills with Objects and Arrays and help determine where you are at in your learning process. You will not be developing code in HTML code for this lab.

Coding/Resource Suggestions and Requirements

- Course Slides and Practice Exercises through week 13.
- **Use only techniques discussed in class and array prototype methods/properties wherever possible.**

Provided:

As a starting point for this lab, you are provided with an index.html file that creates the form shown in the image below.

Figure 1: Web form generated from provided index.html file

Specifications:

Part 1 – File Setup

1) Download Lab4.zip from the Lab4 link in Brightspace.
2) Extract all files/folder from the **Lab4.zip** into your VS Code Workspace. The unzip process will create the required folder structure including the top folder Lab4, and subfolders "**scripts"** and **"json"**. The **index.html** file can be found within the top folder (Lab4). The JSON file **country.json** is located in the "json" subfolder and the JAVASCRIPT file **apps.js** is located within the "scripts" subfolder (.../Lab4/scripts).
3) Review the index.html and identify Input and output web form object and their ID's to be used for reference in your JS code.
4) **Add your new code to this JS file** to the provided **apps.js**.

Part 2– Add information to Country JSON file

Add at least 3 countries and corresponding city (or cities) to the JSON file. The keys should be "country" and "cities".
Hint: Consider using an array of objects as the JSON data structure

Part 3– Setup jQuery to load JSON from file

a) Add required coding to your HTML file that allows access to the **JQUERY** functionality. Hint: use steps discuss in our last class.
b) Add appropriate JavaScript **JQUERY** code to your JS File that loads the JSON data you created in Part 2 into memory (variable) which will be used by your JS code. Hint: Remember the suggestion I made in class considering that the data is loaded within a callback function.

Part 4– Setup Promise/Fetch to load JSON from website API

Using JSON API at location https://restcountries.com/v3.1/all, create appropriate JavaScript **Promise/Fetch** code in your JS File that loads the JSON data into memory (variable) which will be used by your JS code.

Part 5 – Display Info Button - Displaying Countries and their Neighbours

a) Add JavaScript code to your promise/fetch (Part 4) that when the button is clicked the app displays the countries/city (from Part 2 data) in combination with the data loaded in Part 4. The format of the display must be as shown in figure 2 below. For displaying the information use the technique shown in class and the "div" html tag.
Hint: Recommend displaying the web data in the console, review/understand the layout of the JSON and look for a key that contains information that you can use connect the data structures. It is safe to assume that there is only one Object per Country in both data sources (no one to many relationship).
b) Modify your JavaScript coding to output the displayed countries in descending order using the **appropriate sorting parameter**.
c) Modify your JavaScript coding to display the messages "No County Data is Available to View" (Figure 4) when no country data exists and "Neighbours: None" (Figure 3) when then country has no neighbours.

Bonus – Country does not exist validation

Using array prototype methods, determine if there is not a match between an elements in both JSONs data sets. If there is not a match the display the message "Country Not Found in Web Data!" as shown in figure 5 below.

App Display Examples:

Figure 2: App displays Countries, Cities and Neighbours

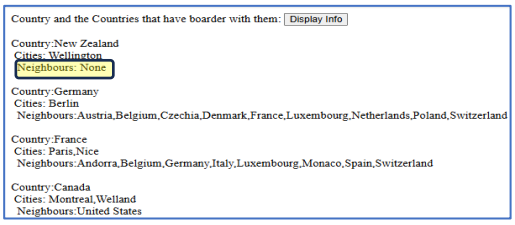

Figure 3: App handling when a Country does not have a Neighbour

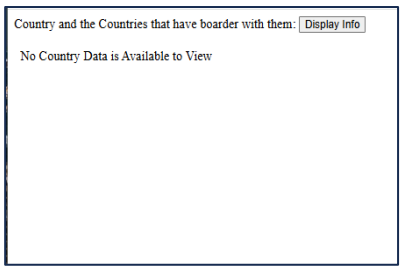

Figure 4: App Display when the JSON is empty

Figure 5: App display when matching data is not found in web data. Should use "Great Britain"

Lab Due Date and Submission:

This lab is due to be completed by the start of next class and it is worth 5%. Please submit the following to Brightspace using the Lab 4 link:

- Text File containing your JS code with file name "***YourName***_app.**txt**". Please **do not** zip this file
- Zip file containing the entire Lab4 folder compressed with filename "***YourName***_app.**zip**"

Failure to follow submission requirements will result in up to 10% grade reduction.

PLEASE NOTE: As mentioned in Week 13 class this Lab will be marked during your Week 14 class. You will participate with your Instructor in the marking of this lab. Please be prepared to demonstrate your lab and answer questions on your code and the running of your code. Your attendance is mandatory for the Week 14 and Your lab must be marked prior to the end of the Week 14 class to receive a mark. Otherwise, your lab will assigned a mark of 0.

Please refer to the following for the grading criteria:

Lab 4 Grading Rubric

Criteria	Marks
Creating the JSON data in file	2
Load JSON file data into app using JQuery	3
Load JSON web data into app using Promise/Fetch	3
Sorting of Countries	2
Display of matched data	3
Display of Messages when data is not found	4
Code is documented	3
Bonus	2
TOTAL	**20**