URL:
https://mirarj.github.io/CS20/Assignment8_API/

Questions:

1. What were the most challenging and most satisfying things about doing this assignment?
   The most challenging part was understanding how fetch() and promises worked. I couldn't figure out how to check for different kinds of errors using fetch(). The most satisfying part was using the form after I built it to send requests quickly and easily.

Code:

```
c<!doctype html>
<html>
<head>
    <title>Names API</title>
    <meta charset="utf-8"/>
    <style>
        li {font-weight: bold;}

    </style>
    <script src="https://code.jquery.com/jquery-3.6.4.js"
integrity="sha256-a9jBBRygX1Bh5lt8GZjXDzyOB+bWve9EiO7tROUtj/E=" crossorigin="anonymous"></script>
    <script>
    function getAge() {

        data_ajax = document.getElementById("data_ajax");
        data_fetch = document.getElementById("data_fetch");
        name = document.getElementById("name").value;
        if (name=="") {
            alert("Enter a name.")
            return false;
        }

        data_ajax.innerHTML = "Loading...";
        data_fetch.innerHTML = "Loading...";

        // AJAX
        req = new XMLHttpRequest();
        req.open("GET", "https://api.agify.io/?name="+name, true)
```

```
        req.onreadystatechange = function() {
            if (req.readyState == 4) {
                json_str = req.responseText;
                json_obj = JSON.parse(json_str);

                if (req.status == 200) {
                    data_ajax.innerHTML = disp_json(json_obj);
                }
                else {
                    error_msg = "Error " + req.status + ": " + json_obj['error'];
                    data_ajax.innerHTML = error_msg;
                }
            }
        }
        req.send();

        // FETCH
        fetch("https://api.agify.io/?name="+name)
        .then((response) => response.json())
        .then((data) => {data_fetch.innerHTML = disp_json(data);})
        .catch(data_fetch.innerHTML = "An error occurred")


        // DISPLAY DATA
        function disp_json(json_obj) {
            disp_html = "Name: " + json_obj['name'] + "<br />";
            disp_html += "Occurrences: " + json_obj['count'] + "<br />";
            disp_html += "Predicted Age: " + json_obj['age'] + "<br />";

            return disp_html;
        }

    return false;
    }
    </script>
</head>


<body>
    <h1>Agify.io</h1>
    <form method="" onsubmit="getAge(); return false;" id="form1">
```

```
        <label for="txtNameId">Enter a name to Agify</label> <br />
        <input type="text" id="name" placeholder="Bella">
        <br>
        <input type="submit" value="Get Response">
    </form>

    <h2>Getting data using AJAX</h2>
    <div id="data_ajax"></div>
    <h2>Getting data using fetch() function</h2>
    <div id="data_fetch"></div>
    <h2>About:</h2>
    <div id="about">
        <ol>
            <li> Describe the API you selected and what it does.</li>
            I chose the Agify.io API. It takes a name as input and returns the predicted age of people
with that name.
            <li>Cite the website where you found it.</li>
            I found this API on <a
href="https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/">https://mixedanalyt
ics.com/blog/list-actually-free-open-no-auth-needed-apis/</a>.
            <li>Describe the options you used for the API request.</li>
            I used the option to query a name. I also used the errors returned by the API to display
error messages. There weren't any other options available.
            <li>Give two applications where this API would be helpful.</li>
            This would be helpful in social research for finding out the average age of commenters on a
website. It could also be helpful for creative applications like making sure the names of fictional
characters match their ages.
        </ol>
    </div>
</body>
</html>
```