**Rich Media Project 3**

How We Met the Requirements

1. Functional Requirements
   1. Must use two distinctive APIs
      1. We utilized both Google Maps API and the OpenWeatherMap API
      2. The Google Maps API is used to show the location the user searches for on a map, and the OpenWeatherMap API is used to show the user what the weather is at that location, including a description of what the weather is like, the temperature, and the wind speed.
   2. You must utilize cloud storage in some way
      1. We utilized Google’s Firebase API to store each search, keeping track of the location, the weather, the temperature (and units), and the wind speed (and units).
      2. All of these searches are shown in our proj-3-admin.html page
   3. You will save the last search using local storage
      1. When a user leaves our page and returns, the last term they searched for will still be in the search bar.
   4. There will be a minimum of 3 controls
      1. Our site utilizes a search bar to search for a location, and two dropdowns. One dropdown is for temperature units (Kelvin, Celcius, and Fahrenheit) and the other is for wind speed units (m/s, km/hr, mi/hr).
   5. There will be no Javascript errors or exceptions
      1. None of the code we have written causes JavaScript errors
      2. There is an error being caused by the Firebase API, but Professor Jefferson said this is okay.
      3. There is a quirk with Google Chrome where a 404 error will be shown when a user inputs a location that does not exist. This is handled by our code, so that there are no errors caused by a user inputting an incorrect location. The 404 error is simply a quirk of Google Chrome, and the error does not show when the site is loaded on Firefox.
2. Design & Interaction
   1. Pleasing graphic design
      1. In order to differentiate our project from the GIPHY assignment, we put the UI we programmed into a hamburger menu on the side of the screen.
   2. Widgets are well-labeled
      1. All of our widgets have labels on them. The search button has “Search” on it, and is right next to the search bar. The two dropdowns have labels next to them that display what the drop down does.
   3. User should be able to figure out the app with minimal instruction
      1. Because all of our controls are clearly labeled, it’s very easy for a new user to figure out how to use our app.
   4. User errors must be handled gracefully
      1. When the user inputs a location that does not exist, our app will change the search bar and location in the display table to say “location not valid” to inform the user that their search was invalid.
   5. Users must know what state the app is in
      1. Our search takes less than a second when the user hits the search button, so it’s very clear what state the app is in.
   6. The app should look good on a range of displays
      1. We’ve tested our app on our big desktop screens, the lab machines, and on my small laptop, and the app looks good on all displays.
3. HTML/CSS & Media
   1. Valid HTML5
      1. Our HTML validates, aside from Vue errors.
   2. Valid CSS
      1. Our CSS validates, aside from Vue errors.
   3. Images are properly optimized for Web delivery
      1. We don’t use any images in our app.
4. Code Requirements
   1. You MUST use data binding and an MVC or MVVM framework
      1. We use Vue.js in our app.
   2. Ajax
      1. We use Fetch in our app.
   3. Use at least 1 ES6 custom class
      1. We have an ES6 custom class that stores the data the user searches for in order to upload to our cloud storage
   4. Conventions and structure
      1. All of our code is in external .js files
      2. We use let and const to declare our variables
      3. We put any code that would be repeated into external functions
      4. All of our functions and variable names begin with a lowercase letter and use camel case
      5. All of our code is commented, with each function we wrote being commented to show what it does
   5. Not allowed
      1. We don’t use DOM queries and manipulation
      2. We don’t use jQuery DOM manipulation
      3. We don’t use var unless absolutely necessary for an API
      4. We don’t use inline event handlers in HTML
      5. All of our console.log() lines have been commented out
5. Impact
   1. Does the app work as intended and do something useful?
      1. The app both works as a map using the google API and a weather service using the OpenWeather API. When searching a location it will zoom to said location and give the current weather conditions.
   2. Does the app functionality and programming go beyond what we did it class?
      1. As a group we had to figure out how to zoom into a specific location based off of a search bar. This combined both the OpenWeather and Google Maps APIs into a fluid construct.
   3. Is this project “portfolio quality” that you would not hesitate to show a potential employer?
      1. I think it’s a very well done project that could be presented on a portfolio to an employer. Since neither myself nor Allie are artists or looking for a job in design, the code that runs the app would stand above how well designed it looks.

Sources

Google Maps API: <https://developers.google.com/maps/documentation/javascript/tutorial>

OpenWeatherMap API: <https://openweathermap.org/api>

Firebase API: <https://firebase.google.com/docs/database/>

Hamburger Menu Tutorial: <https://codepen.io/erikterwan/pen/EVzeRP>

What Each Member Did

For a majority of this project, we tried to get together in person when we did work. Because of this, both of us worked on essentially every element of this project. While one of us wrote code, the other would be there offering ideas. Both of us worked to get the APIs implemented and working. Both of us worked on designing the app to make it look as aesthetically pleasing as possible.

In terms of specifics, Allie implemented the functionality for the admin.html page displaying the items put up to Firebase. She also implemented the search functionality with Google Maps, the drop down menus, and the handling of errors

In terms of specifics, Conor implemented the Firebase API functionality, the hamburger UI menu, the fetching of weather data from the Weather API, the conversion rates for different units, and the map moving to the location searched by the user.

Our Grade

We believe that our app is deserving of a 100 grade. We feel that we met all given requirements in a satisfactory manner. Our app utilizes the two APIs we implemented well and efficiently, displaying to the user exactly what we intended from the start of this project. With the hamburger menu, the UI is clean and the feedback is quick and easy for the user to understand. Everything is clearly labeled so that any user could sit down and understand our app within seconds of using it for the first time. We feel that the work both of us put into this project and the end result are deserving of a 100 grade.