Project 2 requirement checklist:

* Two api’s (google maps and yelp in our case)
  + Yelp api has yet to be implemented
* ~~Cloud Storage API (firebase in our case)~~
* ~~Local Storage~~
* ~~Required Controls~~
  + At least 3 inputs
    - Search field
    - Latitude input filed
    - Longitude input field
    - Radius select
    - Category select
    - Search button (doesn’t count)
    - Get List button?
* No Javascript errors or exceptions
* Pleasing graphic design
* Widget are well laveled and follow interface conventions
  + Example: radio buttons are for mutually exclusive options, checkboxes are for when you want to let the user choose multiple options --> <https://delib.zendesk.com/hc/en-us/articles/203430309-Radio-button-vs-checkbox-what-s-the-difference->
* Users should be able to figure out how to use the app with minimal instruction
* User errors must be handled gracefully
  + for example, if the user forgets to type in a search term before clicking the Search button, the app should tell the user something like "Please enter a search term first"
* Users must know what s*tate* the app is in at all times
  + for example, when they click the search button, there should some indication that a search is happening:
    - text that says "Searching for 'Tacos' near you" and so on
    - a "spinner" or other "indeterminate progress" animation --> [Google search "indeterminate progress"](https://www.google.com/search?q=indeterminate+progress&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj-sNCal4neAhVr34MKHWKqA98Q_AUIDigB&biw=1036&bih=583)
    - here are some "spinner" images you could use (show them when the search starts, and hide them when the search ends): <http://ajaxloaders.net/2012/10/spinner-loading-animations-set-1/>
* While the app doesn’t need to be fully responsive, it should look good on a range of displays
* Valid html
* Valid css
* Images are properly optimized
* Allowed and encouraged to use CSS frameworks on the UI
* Must use data binded and an MVC / MVVM
* Use ajax (fetch in this case)
* Use at least 1 ES6 custom class, written by you:
  + for example, if your web service was downloading and displaying state parks in a list, you could create a class called StatePark to model the data
* Conventions and structure:
  + All code is an external JavaScript files
  + ES6 module pattern preferred
  + let and const must be used to declare variables
  + D.R.Y. - Don't Repeat Yourself. Repeated blocks of nearly identical code must be factored out and placed in a separate function
  + Variable and function names must begin with a lowercase letter
  + Well-commented code. Each and every function gets a comment indicating what it does
* It is expected and required that the code in the assignment (other than from approved libraries) is written by you. If you do end up using a small amount of code you found on the web, you must document where you got it from. Give credit and a link for all code (fragments or otherwise) that are not written you. Failing to give credit opens you to charges of academic dishonesty:
  + examples of acceptable use for this project:
    - copying a GUID generating function "whole cloth" from StackOverflow - <https://stackoverflow.com/questions/105034/create-guid-uuid-in-javascript>
    - copying and lightly modifying code for a "hamburger" menu - <https://www.google.com/search?q=vanilla+javascript+hamburger+menu>
  + Cite the code source both in the source code itself as a comment, and in your final documentation
  + Be sure to make borrowed code "your own" as much as possible for example by simplifying or improving the clarity of the code, using let or const instead of var, getting rid of inline event handlers (which are prohibited in this project) and so on
  + You do not need to cite code that you received from our in-class exercises, demos or HW
  + If you have any doubt about what is acceptable to "borrow", ask the professor in advance of using it
* NOT allowed:
  + DOM queries and manipulation via document.querySelector(), document.querySelectorAll(), element.innerHTML, element.appendChild() and so on. Instead, use data-binding and your MVVM framework to update the DOM
  + jQuery DOM manipulation - for example $.html() - is NOT allowed
  + var
  + inline event handlers in your HTML
  + console.log() calls (delete or comment them out)