# Review responses

Please note that the line numbers used here are from the original review, not from the current production version of the code. Thank you for the “Awesomes”.

(nmViewModel.js:61) “Suggestion. It would be nice if this method toggle between *Pinned Only* and *All locations* mode. Currently, I have to use the *Show all categories* option from the other filter to show all locations again”

This has been corrected.

(nmViewModel.js:61) “Suggestion. It would be nice if this method toggle between *Pinned Only* and *All locations* mode. Currently, I have to use the *Show all categories* option from the other filter to show all locations again”

Done.

(nmViewModel.js:271) "Required: Google Places API request needs to be provided with an error handling method. You can use this code snippet:

function placeSearchCallback(results, status) {

if (status === google.maps.places.PlacesServiceStatus.OK) {

// success implementation

} else {

// error handling

}"

This handler was in place already, only in a slightly different format. I have matched the “if (status” line with the one specified here, and I modified my comments to make it clearer how I meet requirements. My code is different than the snippet in that:

* Between the first snippet line and the second I have to turn off a timeout that was set right after the async request. Otherwise the user sees two alerts: the error notification and the timeout notification.
* In my response handler there are 7 lines of error code and 29 lines of success code. If I slavishly follow the snippet format, the end of the success code block/beginning of the error code block would be hard to find and, by indenting the success code another tab stop, I would increase the number of line wraps and further squash the code. Instead, I invert the status test (“status !==” instead of “status ===”) and end the error block with a return. The remainder of the function is all success processing, which is the more likely code path. As nearly as I can tell, there’s no guideline against multiple returns from a function.

By the way, I’m not sure why you require an error handler here and don’t at nmModel.js:364-365 (the actual submission of the request, which bombs on a null placeId). I added one there.

Also corrects the second item in the rubric’s “Asynchronous Data Usage” section.

(nmViewModel.js:297) “Suggestion. Data manipulation of the DOM must utilize the MVVM pattern, but not by using jQuery nor Javascript DOM methods.  
For this implementation, please research on [creating a custom binding](http://knockoutjs.com/documentation/custom-bindings.html)  
Note: I'm not marking this as *required* because I think it's too much to ask you to create custom binding.”

Thank you for your consideration, and I believe it’s worse than you suggest. There seems to be a timing issue between when the data bindings happen for Knockout (data-toggle, -html, and –content for the popover) and Bootstrap (data-bind, which should bind a string to the data-content attribute). The best I could figure out (applying one framework at a time) was that Bootstrap had done its DOM processing before Knockout applied the attribute binding. I don’t think using a custom binding would make any difference in timing. That’s why I went to the lower-level libraries to work around the problem. Guess I should have documented this. It’s now noted in the README.

(nmViewModel.js:492) “Suggestion. Marker object's setMap method completely removes markers / redraws them. This is not the efficient way to show or hide markers. A more optimized way to actually show / hide markers without removing / redraw them from the map is to use Marker's' setVisible method.”

This is very helpful. I’ve implemented it, and it works fine when I test it. I wonder why Google’s sample apps don’t point this feature out.

(nmViewModel.js:623) “Awesome. Excellent job fitting map markers within map view with fitBounds! You can utilize this to make map display responsively by using window.onresize event (see [here](https://developer.mozilla.org/en-US/docs/Web/API/GlobalEventHandlers/onresize)) and call fitBounds method to make sure map markers always fit on screen as user resizes their browser window:”

I want the user to be able to manipulate the map, zooming in and out, to find new place names to search and pin. If I locked the bounds to where the pins are, it would be hard to explore new territory. It’s a feature, not a bug!

(nmViewModel.js:693) “Required. Clicking on marker or location in the list view must also show location details.”

I’ve made this change, but I’m not very happy about it. My original UI flow design was that limited place information (name and address) would be popped up in an InfoWindow on selection of either list item or marker, fulfilling this requirement. At the same time, async requests pre-loaded the Place Details dialog, so that if the user popped it up with the info button (from the list-view item) it used to appear instantly. Now, with the modal dropping down instantly on marker/list-item click, the user gets to watch the screen fill in, especially the photos which load last. It doesn’t take much network throttling to make the delay visible: emulating Regular2G in Chrome (250kb/s) it took 5-10 seconds after the modal was in place before it finished loading. But my project is now even more in compliance than before (although it’s hard to see the required marker animation with the Place Detail dialog covering it).

Also applies to nmView.js:111 and two entries under “App Functionality” on the Project Review tab.

(nmView.js:179) “Suggestion. The project rubric allows us to manipulate the layout/aesthetic components with jQuery or Javascript DOM methods. But it's recommended that we implement this feature by using *Knockout* to completely utilize the MVVM pattern”

Done, using modalDisplayFlag() observable in the Knockout viewModel. It is bound to the modal dialog with a style binding. I used a similar method to avoid some jQuery calls in the off-canvas slide-in sidebar (see below), modifying Bootstrap’s control of their component.

(Index.html:14) “Tip. We can include favicon in your <header>. It is a web site icon that usually show up in browser's address bar / tab / bookmark and will further personalize your web page.”

Excellent, I’ve often wondered about random console messages about favicons. Nice to try it out. Thanks.

(Index.html:18) “Suggestion. Highly Recommend: Since we're writing HTML5 code, we should use **HTML semantic tag** wherever possible.”

I’ve added a few, but I’ll wait to fully use semantic tags on my next project. Good suggestion, though.

(Index.html:323) “Required. Other than GMap API request, all other <script> tags shouldn't be provided with async attribute. They are vital for app to run and should be requested synchronously.”

I have done this, but I disagree with it. Some of the scripts are not executed until the user interacts with the UI to launch the Place Details page. (They are definitely “below the fold.”) There’s no need to make the user wait for them to load and run, but I can’t argue with the rubric just now.

Also corrects the first item in the rubric’s “Asynchronous Data Usage” section.

(index.html:333) “Suggestion. Code Quality. **Udacity Style Guide requires:**

* Do not use type attributes for style sheets and scripts. Specifying type attributes in these contexts is not necessary as HTML implies text/css and text/javascript as defaults. This can be safely done even for older browsers”

Removed them.

(Index.html:14) “Required. The **Asynchronous Data Usage** also requires map to be also provided with a fallback error handling method.”

I followed the example for the script tag onerror entry. However, to be sure there’s no funny business with scripts not loading or timing issues, I put the error handler (just for the async script load) in the HTML file.

(nmModel.js:372) “Tip. We can utilize shorthand conditional statement to shorten the code. Ex:

var rating;

// ( condition ) ? (if true then) : (if false then)

!!response.rating ? rating = response.rating : rating = 'No rating available';

“

Great idea. Done.

(nmModel: 491) “Suggestion. Nice job here! But we should consistently use success and error option, or done and fail method instead of mixing them.”

Makes sense. Can you tell I was working from multiple sources for my solutions? Fixed.

(README.md:5) “Required. **Documentation** rubric requires README.MD to include

* Detailed instruction on how to open the app (*locally, Github Page link doesn't count - even though it's excellent to host app on Github Pages*) (Hint: which file to open to run the app? or how to host the app locally on localhost)”

This makes no sense to me, as using a real server knocks loose bugs I didn’t see on my localhost installation. But the change is done.

(Project Review tab, “Interface Design”) “App is not yet usable on medium / small screen sizes / devices:

* The location list and filter components are not visible…”

Whew, this was a hard one. Animating DOM elements in and out on button clicks is challenging, but triggering that much complexity on viewport resizing is hard, even with Bootstrap’s support and sketchy hints about how to use it. The list view now hides if the viewport’s smaller than an iPad, to be slid in and out by a button that only appears when it’s needed (following Bootstrap’s “Off Canvas Template”). It does improve the mobile UX, though.