# COMP 4513 Assignment #1: React

*Due Monday February 25th at midnight*
*Version 1.0 (Jan 29, 2019)*

## Overview

This assignment provides an opportunity for you to demonstrate your ability to generate a single-page application using React. **You can work alone or in groups of two on this assignment.** Let me know if you need a private github repo for your group.

## Beginning

Use the same techniques covered in the second React lab for this assignment.

## Sources

The image files are in a Google Cloud storage bucket:

`https://storage.googleapis.com/funwebdev-3rd-travel/`**`size-folder`**`/`

where `size-folder` is either: `large`, `medium`, `square-medium`, or `square-small`.

The data comes from the same images service used in the React 2 lab:

`https://randyconnolly.com/funwebdev/services/travel/images.php`

For the splash image on the Home Page View, please use:

`https://source.unsplash.com/`

## Submit

You won't be able to use Cloud9 for this assignment. I think the best way to submit the assignment will be to copy your local files (i.e., your my-app folder) in a folder with your login name to the Submit drive. It's a lot of files to copy however …

## Grading

This assignment is worth 15% of the course grade. The grade for this assignment will be broken down as follows:

| | |
|---|---|
| Visual Design, Styling, and Usability | 25% |
| Programming Design + Documentation | 10% |
| Functionality (follows requirements) | 65% |

## Requirements

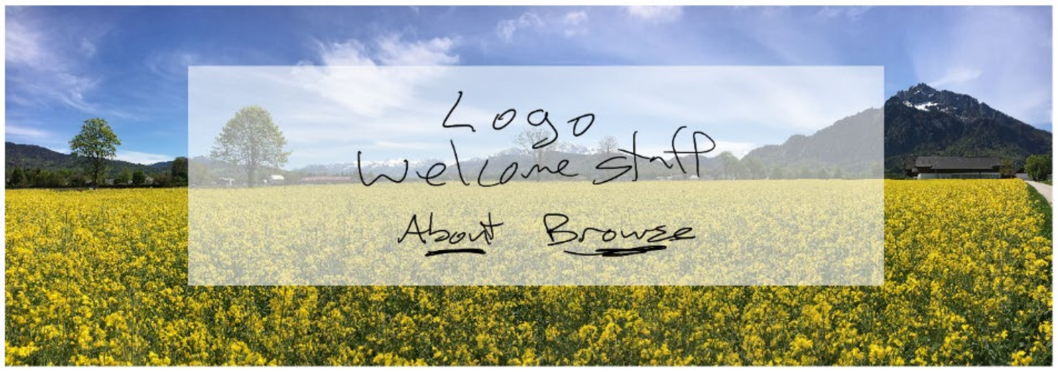
1. You must make use of the `create-react-app` starting files and structure.
2. This assignment consists of three main views (remember this is a single-page application, so it is best to think of the assignment consisting of different views). In the remainder of this assignment, I have provided a sketch of the basic layout of each view (some views have multiple subviews).

   These sketches do not show the precise styling (or even the required layout); rather they show functionality and content. I will be expecting your pages to look significantly more polished and attractive than these sketches! A lot of the 20% design mark will be based on my assessment of how much effort you made on the styling and design.

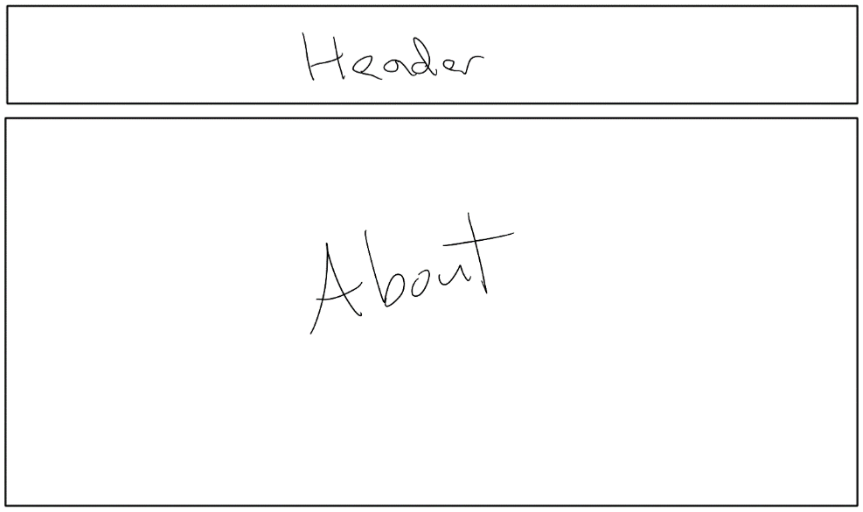
   You can change the layout if you wish. If you want your photo list to be horizontal on the bottom and the favorites to be vertical on the right, you can do so if you think that would be best … though I may not necessarily agree about the usability of your choices. I'd rather see you try something different in the layout/design than simply slap together something that is just a slightly improved version of the layout in my sketches.
3. If you've used JS or CSS that you've found online, I expect you to indicate that in your documentation and in the About view. Be sure to provide a URL of where you found it in the documentation. Failure to do so might result in a zero mark, so give credit for other people's work!
4. When your site first loads, it should display the splash page, which here is called the **Home Page View**. When the user selects a Home link/button, it should display this Home Page View.

   Your home should consist of some large image that look's like it says "Travel Photos", and then on top of the image, include your logo (or site name), some welcome text or brief description (3-8 words) of the site, plus links/buttons/icons for About and Browse.

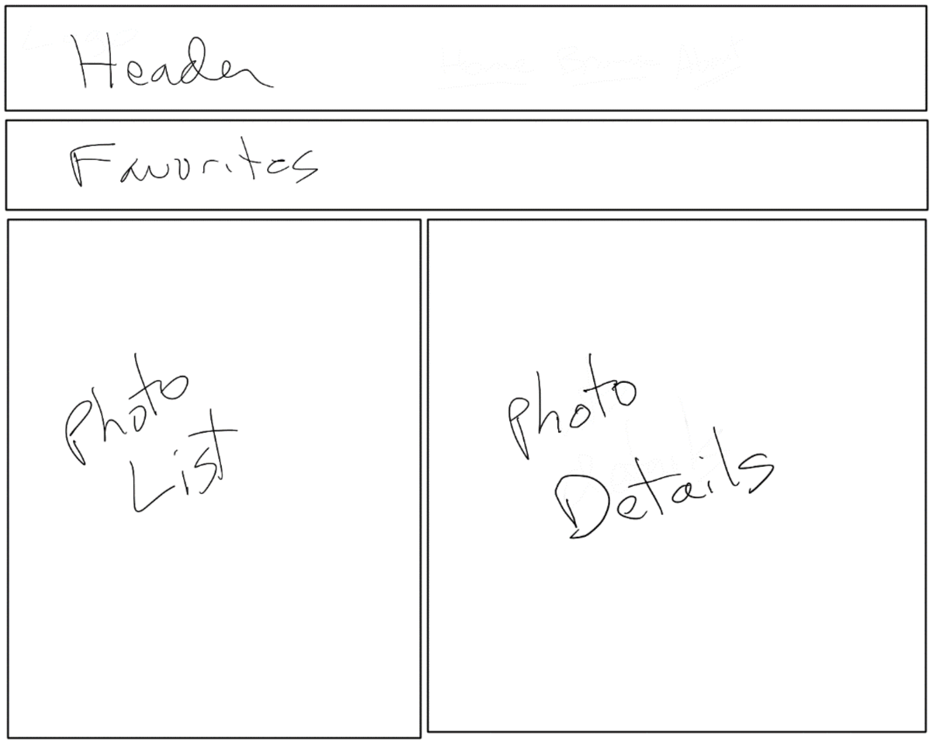
   Please use Unsplash Source for your large splash image. Be sure to credit it in About.



5. When the user selects the About link/button, it should display the **About View**. Tell us who you are, what course this is for, what technologies were used, and what tools and packages you have used. If you used other sites for help/samples, list them here with links. The header should also be visible here (described below).

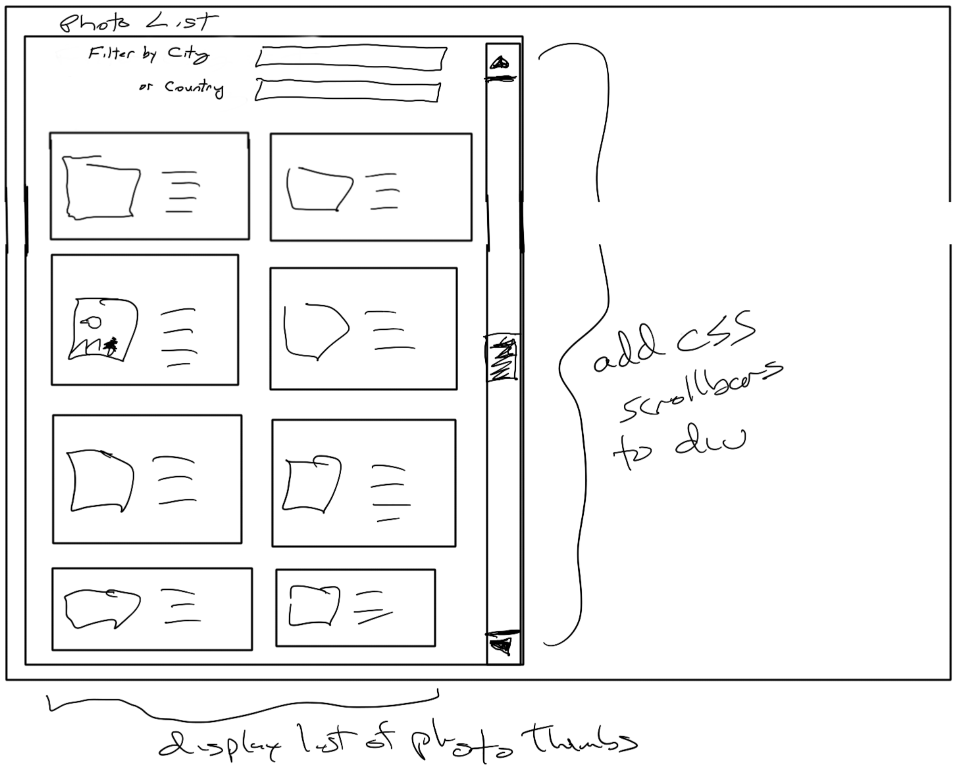


6. When the user selects the Browse link/button, it should display the **Browse View**. Most of the assignment's functionality exists within the Browse view. It initially consists of the following content areas: **Header**, **Favorites**, **Photo List**, and **Photo Details**.



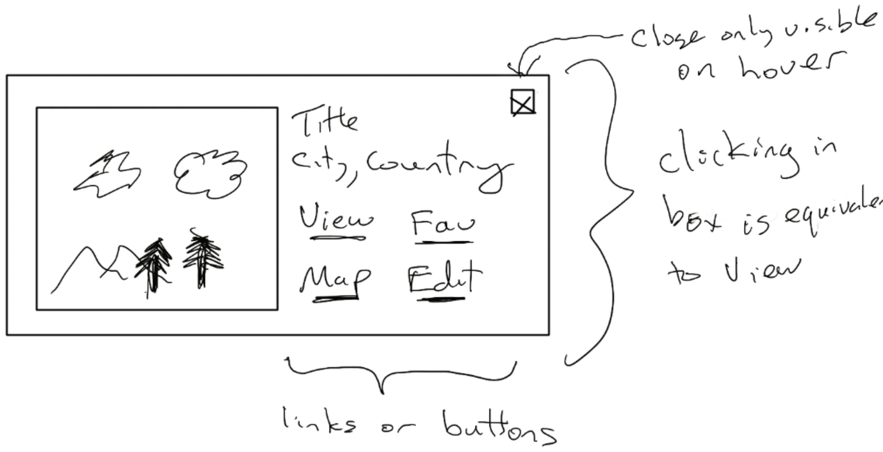
7. The Header area will contain the logo (or site name) and links/buttons/icons for Home, About, and Browse.

8. The **Photo List** contains a list of photos from the same web service used in Lab 2



The list consists of Photo Thumbs with functionality described below. There should be a way for the user to filter the list by city name **or** (but not and) country name.

Notice that there are four links/buttons/icons: to view, to add to favorites, to see image location on map, and to edit the info for the photo. View, Map, and Edit will be displayed in the **Photo Details** area.


Close only visible on hover
Title
city, country
View
Fav
Map
Edit
clicking in box is equivalent to View
links or buttons

Clicking on the box or the View/Map/Edit buttons should change the visual display of clicked Photo Thumb component in some way (e.g., change its color, add a shadow, or a border, etc). Initially, the first photo should be selected.

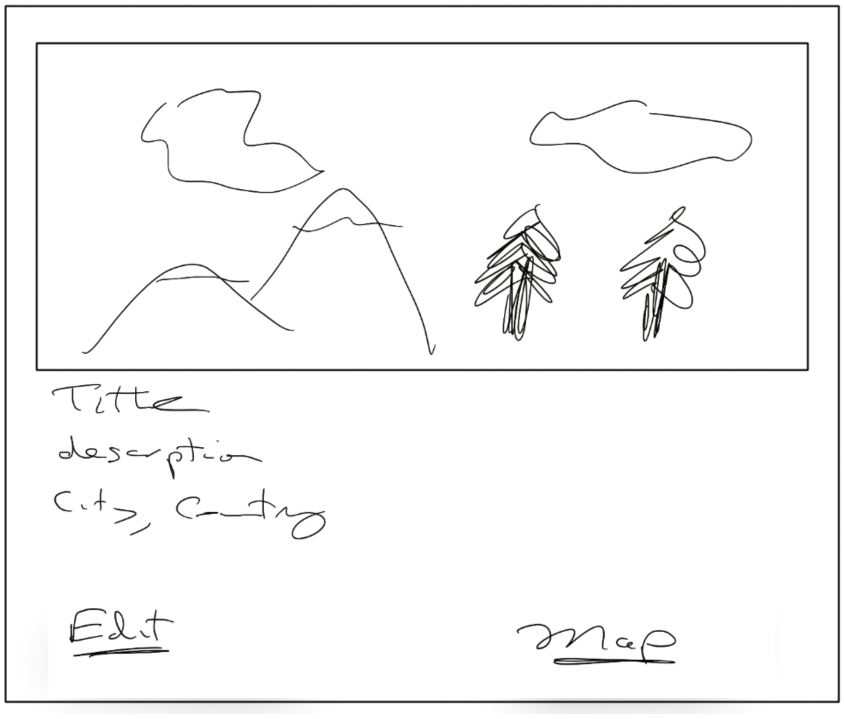
When the user clicks on the Close button, remove the photo from the list (and from state). If the photo is in the favorites, remove it from there as well.

The best way to implement this is to implement the close button as an image (or use just CSS transforms) and position in upper right. Use CSS :hover selector to change its opacity or visibility. If you want to always display the close button, that's fine but there should be some type of state change on hover (e.g., change button opacity, or color, or size).

Clicking on the close button must remove the photo from state. Ideally, there will be some type of animation/transition on the Photo Thumb box to provide visual feedback that the photo is being deleted.

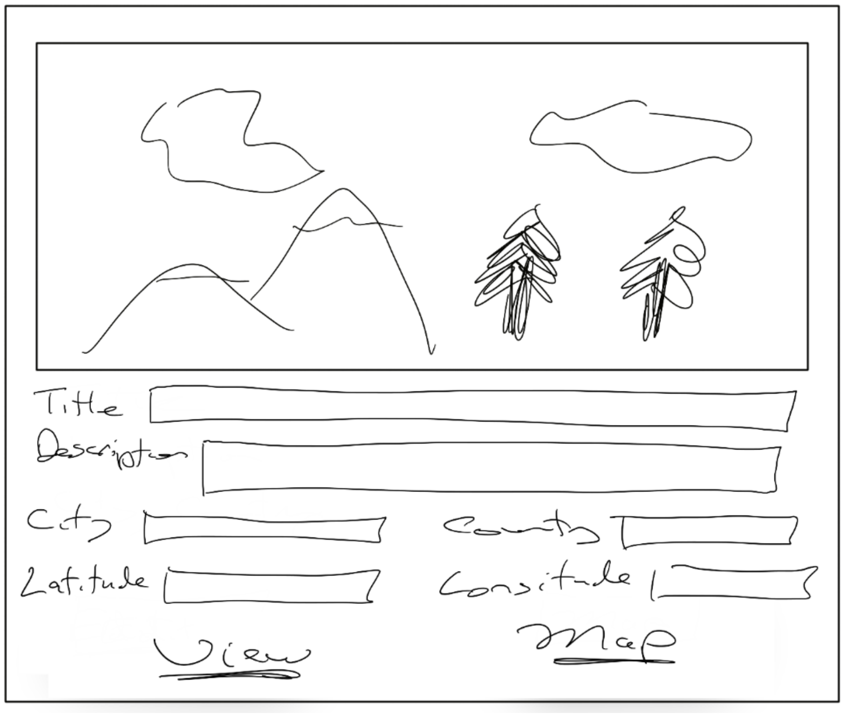
9. The **View Single Photo** displays a larger version of the image, along with the title, description, city, and country data. This should be the default view in the details area whenever a new photo is selected (or when the **Browse View** is first displayed).

   It should have two links/buttons/icons: Edit will replace this view with the **Edit Photo Form**; Map will replace this view with the **Photo Map**.



10. The **Edit Photo Form** allows the user to edit the data for the form. Changes to the form data should change data in the state and elsewhere on the page (just like in the React 2 lab). Latitude and Longitude must be numbers.

    The form should have two links/buttons/icons: View will replace this view with the **View Single Photo**; Map will replace this view with the **Photo Map**.

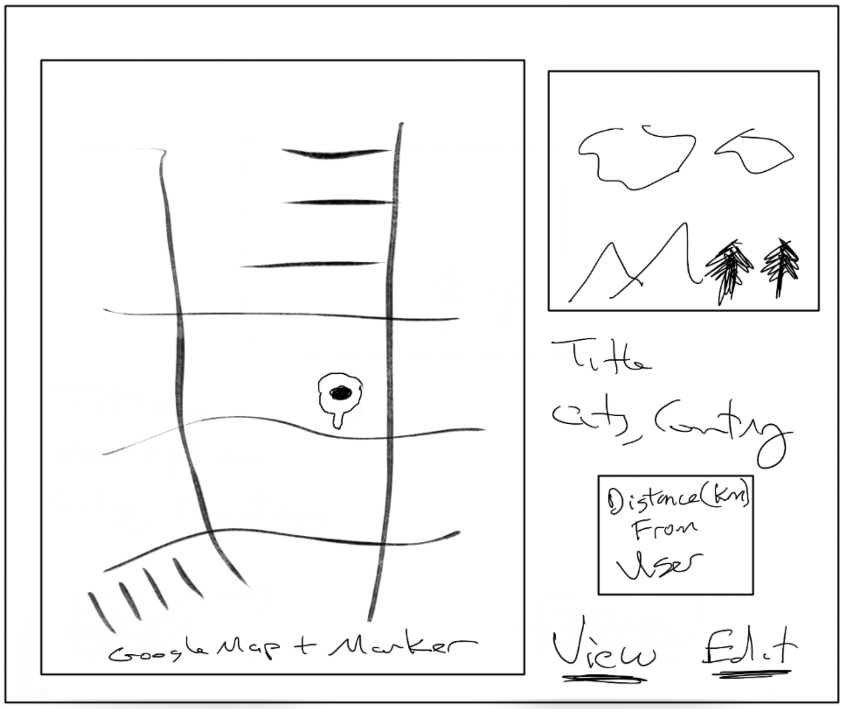


11. **The Photo Map** should display a Google Map of the photo's latitude and longitude. I would recommend using the `google-maps-react` package. Be sure to put down a marker for the photo's location. Choose a sensible zoom (city block level).

    Your view also needs to display the distance in km from the photo's lat/long and the user's current location. You can retrieve the user's current lat/long location using the geolocation api in the browser (the browser will ask user for permission to "Know Your Location"). If no permission is obtained, your page must handle this gracefully.

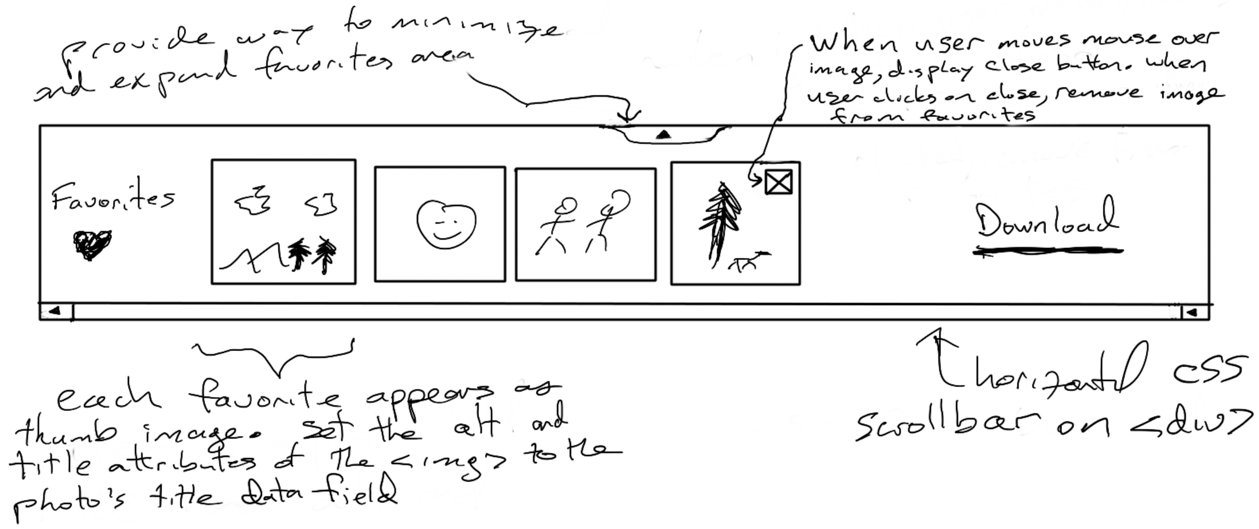
    If user does give permission, you can then calculate the distance between the two locations using the haversine formula (see for instance, [https://www.movable-type.co.uk/scripts/latlong.html](https://www.movable-type.co.uk/scripts/latlong.html) for copy+paste JS code to perform this distance calculation). If you have room, you may want to display a second smaller map next to the distance showing the user's location on this smaller map.

    The form should have two links/buttons: View will replace this view with the **View Single Photo**; Edit will replace this view with the **Edit Photo Form**



**12.** Clicking the Favorites link/button/icon will add the photo to the current Favorites state. This will also display the square photo thumb in the favorites area/component.

Each time the favorites state is changed be sure to also save the changed favorites in local storage as well. This means when the app loads, you will have to check if the favorites exists in local storage. If so, then initialize the favorites list in state to be the same as that in local storage.


provide way to minimize and expand favorites area
When user moves mouse over image, display close button. when user clicks on close, remove image from favorites
Favorites
Download
each favorite appears as thumb image. set the alt and title attributes of the <img> to the photo's title data field
horizontal CSS scrollbar on <div>

Notice that the user needs a way to remove favorited items. The best way to implement this is to implement the close button as an image (or use just CSS transforms) and position in upper right. Use CSS :hover selector to change its opacity or visibility. If you want to always display the close button, that's fine but there should be some type of state change on hover (e.g., change opacity, or color, or size). Clicking on the close button must remove the photo from favorite state (and update local storage). Just as with removing photos from the Photo List, ideally add some type of CSS transition or animation to provide visual feedback that an image is being removed.

The user needs a way to hide or unhide the favorites area. Ideally, there should be some type of CSS transition or animation to provide visual feedback when the favorites area is hidden.

Finally, the favorites area needs a Download link/button/icon. When the user clicks this, large versions of all the favorited images will be packaged into a single zip file and then downloaded. I would recommend using the simple JSZip library (https://stuk.github.io/jszip). See also https://davidwalsh.name/javascript-zip.