Project 5: Interactive Photo Gallery

Sections of this Guide:

- **How to approach this project** includes detailed guidance to help you think about how to organize your code, project and files.
- **How to succeed at this project** lists the grading requirements for the project, with hints, links to course videos to refresh your memory and helpful resources.

How to Approach this Project

- ❏ **Download the project source files from the Interactive Photo Gallery project instructions page in your Techdegree curriculum.**

- ❏ **Create GitHub repo and push project files - Share Your Projects with GitHub**

- ❏ **Set up the rest of the directory by creating the necessary files and folders.**
    - ❏ Along with the project source files, you'll need at least a **js** folder with an **app.js** file, a **css** folder with a **styles.css** file, and an **index.html** file.

- ❏ **Create the basic markup structure and styling for your gallery.**
    - ❏ Structure your **HTML** to match the baguetteBox usage documentation.

- ❏ **Download and Hookup the lightbox plugin**:
    - ❏ Download and extract the baguetteBox JavaScript Lightbox plugin.
    - ❏ Copy the **baguetteBox.min.js** file out of the **dist** folder and paste it into your **js** folder.
    - ❏ Copy the **baguetteBox.min.css** file out of the **dist** folder and paste it into your **css** folder.
    - ❏ Add `<link rel="stylesheet" href="css/baguetteBox.min.css">` to the `<head>` of your **index.html** file.
    - ❏ Add `<script src="js/baguetteBox.min.js"></script>` above the closing `</body>` tag in your **index.html** file.
    - ❏ Add `baguetteBox.run('.gallery');` in your **app.js** file.

How to succeed at this project

Here are the things you need to do to pass this project. Make sure you complete them **before** you turn in your project.

Design

- ❑ **Make sure your design matches the layout of the mockups:**
    - ❑ Project includes the thumbnail gallery and a search box.
    - ❑ The layout doesn't have to be pixel perfect, but general spacing, position and arrangement of the elements should closely match the layout of the mockup.
    - ❑ The thumbnail gallery layout is built with CSS Grid.
        - ❑ *Related video:* CSS Grid Layout

Lightbox functionality

- ❑ **Successfully implement the baguetteBox.js lightbox plugin and ALL of the following features are functioning:**
    - ❑ When a gallery thumbnail image is clicked, larger image appears with overlay.
    - ❑ Background overlay hides the gallery and covers the entire window.
    - ❑ Images in lightbox include full captions.
    - ❑ Lightbox images can be navigated by clicking right and left arrows.
    - ❑ The lightbox can be closed.

Content Filtering

- ❑ **Implement the included searchFilter.js plugin or write your own JS for the search functionality:**
    - ❑ Gallery photos update in real time as the user types into the search box, only photos that match the caption text appear in the gallery.
    - ❑ Case insensitivity has been added so searches will ignore letter case.
    - ❑ Entire caption is searchable, not just title or keywords.
    - ❑ Add `const search = new Filter('search', 'data-caption');` in your **app.js** file if you're going to utilize the **searchFilter.js** plugin, instead of creating your own.

However, accomplishing the search functionality with your own code is an excellent bit of web dev practice and experience, so feel encouraged to give it a try. Just break the task down into small pieces.

First try to get the value of the search field whenever a user types in that field, and log that value to the screen with a console.log() statement. Checkout the **keyup event listener** for this. And be sure to make the value case insensitive by using something like the **toLowerCase()** method.

Next, start trying to target the entire caption by using the **getAttribute()** method, then **looping** and logging the all lowercase version of the captions to the console.

Next comes the fun part. Start trying to find ways to check if the current value of the search input is **included** within any of the captions, and if so, log the associated image to the console.

Lastly, use a **conditional** so that if there's a match, **display** the container of the image, and if not, **hide it**. Piece of cake!

Helpful links:

- ❑ Create a JavaScript search filter

❑ **Double check everything, validate your files, request an informal review in Slack, and then submit.**