Photo Atlas App

the app that helps photographers scout photography spots

PHOTOATLASAPP.COM

Zoraida Cabrera-Mieles

DGMD E-599 | Section 1 | Capstone Design Studio

Harvard Extension School

In Partial Fulfillment of the Requirements
for a Master of Liberal Arts, Digital Media Design Degree

Abstract **3**

Introduction **3**

The Problem 3

The Solution 4

Target Audience 4

User Profiles 5

Professional Photographer 5

Content Creator 7

Photo Hobbyist 8

Competitor Review **10**

Google Maps 11

Depalo 12

ShotHotspot 12

Competitor Comparison Matrix **13**

Plan and Design **14**

Initial Project Plan 14

Proposed Design 15

From Sketches to Mock Ups 16

Homepage 16

Location Page 18

Trips Page 19

Implementation **20**

Project Pivots 20

Time Constraints and Unconsidered Features 20

How Might We 21

Feedback 22
A Hidden API Cost 22
A New Rubric 22
Product Architecture 25
Site Map 25
Tech Stack 26
Technology 26
React 27
Flickr API 27
Leaflet JS 28
Firebase Authentication 28
Firebase Realtime Database 28
Netlify and Github 30
Testing 30
Manual Feature Test 31
Responsive Design Test 33
WAVE Accessibility Audits 36
Conclusion: Final Product and Beyond the Capstone 38
Appendix 39
Project Presentations 40
Original Project Rubric 40
References 42

Abstract

Photo Atlas App helps photographers find the right location to capture their best work. Its main goals are: 1) to ease the time consuming process of scouting for photo spots, and 2) to make it easy to find and share good photography spots with others. During my capstone project, I created this responsive web application by leveraging technologies such as React, the Flickr API, Firebase, Netlify, and Github. I successfully launched the application at photoatlasapp.com.

Introduction

The following document describes the process for designing and developing the Photo Atlas App web application during the last two semesters of my Master of Liberal Arts in Digital Media Design Degree at the Harvard extension school.

The Problem

Photographers can spend hours looking for locations both online and on-the-go. While there are a few apps out there that can ease the scouting process, one great way to find these locations is to learn where other photographers have captured pictures. Not many apps exist that allow photographers to get direct recommendations from other photographers. Those apps that do exist usually have one of the following problems:

1. Limited filtering for a photographer's need
2. Are not cross platform, only functioning in one type of device
3. Don't allow easy sharing with other photographers
4. Don't search on current location, which can slow down photographers who want to know what is immediately around them
5. Only suggest photography locations for a limited amount of places

The Solution

Photo Atlas makes it easier for photographers to find the best spots to take pictures by addressing the drawbacks of current scouting applications. To solve the main issues with other apps Photo Atlas:

1. Provides Subject Filters, which allow photographers to filter by a specific photographic subject
2. Is a responsive web app that can work on any device
3. Allows photographers to create lists of photography locations (called Trips) with 3 privacy settings:
    a. Public, which allows anyone using the app to see
    b. Private, which limits visibility only to the creator
    c. Shareable, which limits visibility to the creator and anyone with a link
4. Detects the photographer's location to search for photography spots nearby
5. Uses the Flickr API to source photography locations from around the world

Target Audience

The photo scouting process doesn't just apply to professionals. It also includes hobbyists and content creators. The interest in photography has recently grown due to better phone cameras and platforms such as Instagram. Therefore Photo Atlas has the potential to impact both SLR photographers and mobile phone photographers.

However, because scouting is most important for professionals, we can assume that the application's primary users will be professional photographers. We could also assume that the app's biggest impact of Photo Atlas will likely be within the United States, which is the app's country of origin, and where it will first be promoted. Lastly, we can place our target audience between ages 18 and 64, as this is the age group of adults more likely to use the internet, according to the Pew Research Center. While

there may be younger users of the app, adults are more likely to travel and thus more likely to adapt an app such as Photo Atlas.

With these considerations in mind, I created 3 different user profiles:

- Professional Photographer
- Content Creator
- Photo Hobbyist

User Profiles

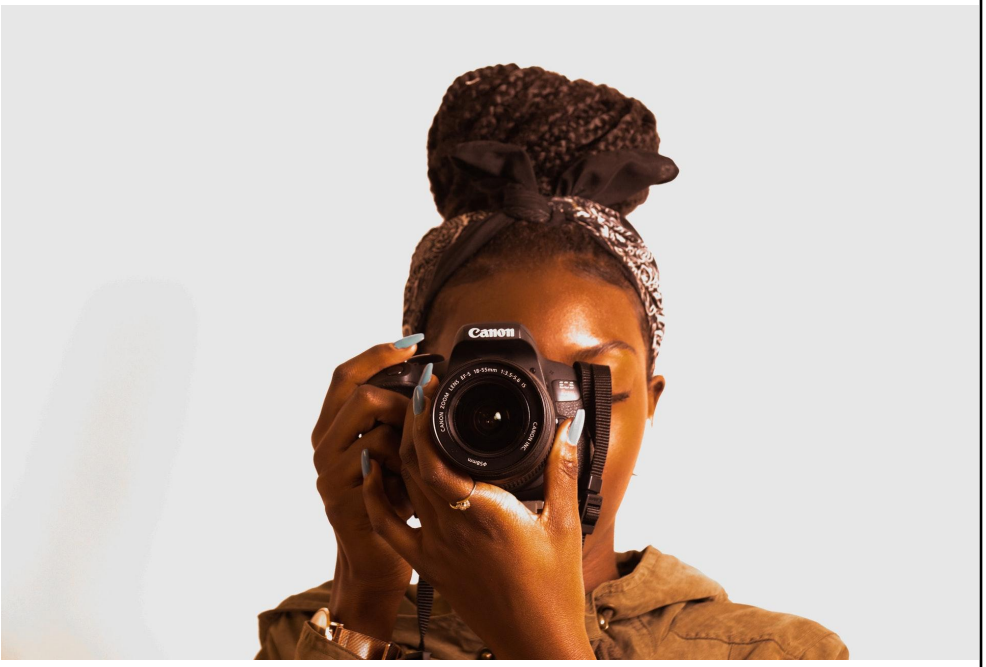
Professional Photographer

Persona Type	Professional Photographer
Name	Zachary Hernandez
Image	
Quote	"Creating the best work for my clients means knowing the best time and place to capture images."

Job	Event and Product Photographer, and owns a photography workshops business
Age	41
Goals	• Recommend the best photography resources for his students • Efficiently scout for places to take photos for his clients
Frustrations	• Working with difficult clients in general • Scheduling time for scouting during tight deadlines • Photography apps that have a high learning curve for his students
Hardware	• Full Frame Canon DSLR • Mirrorless Sony • iPhone • Macbook Pro
Software	• Lightroom for organizing photos and doing core edits • Photoshop for more advanced photo post-processing
Thinks	• Where are other photographers going to take great shots? • How do I not capture the exact same picture, but add my own style. • I should get there early so I can scout the area. • Google maps is good for scouting, but it's best to you talk to someone who knows the area • What tools and apps can I recommend to my students?
Says	• What is the best time to take pictures here? • Photography requires a lot of patience. • I need to research the area before the shoot. • A cloudy day can make for a dramatic scene. • My clients trust I know when, where, and how to capture great images.
Does	• Plans ahead of visiting a place for photography. • Uses multiple apps for scouting and bookmarking photo spots. • Wakes up early, even on family vacation, to take the best photos. • Goes on photography trips with his students or photographer friends. • Buys a lot of photography gear. • Loves learning about apps that help his work.

Feels	• Motivated • Excited • Busy • Strained for time

Content Creator

Persona Type	Content Creator
Name	Brittany Jones
Image	
Quote	"Unique images form an essential part of digital storytelling."
Job	Copywriter
Age	26
Goals	Grow her blog, Youtube, and Instagram audience by generating authentic content
Frustrations	• Finding original places to capture photographs • Creating enough original content across all her social media platforms • Staying up to date with her field

Hardware	• Switches between entry level Canon DSLR and phone camera • iPhone • Macbook Pro
Software	• UNUM app for planning her Instagram Gird • Photoshop for light editing • Snapseed for editing photos on the go
Thinks	• Is that the right filter? • I'm always taking pictures with the same background. Need to switch it up. • What does my audience think? • What is the best time to post this picture? • Is this content relevant? • Where are others taking pictures?
Says	• The right filter aides storytelling. • I don't really know a lot about photography, just enough for my work. • What is the message that we are trying to convey here? • What's your niche?
Does	• Plans her Instagram content ahead of time • Takes a lot of self-portraits as part of her blog • Tries to limit the number of apps on her phone for mental health and work-life balance reasons • Constantly researches places to travel and plans her trips ahead of time • Likes to travel light, with little camera gear • Does most of her photo editing on her phone
Feels	• Creative • Driven • Strained for time • Unsure about what content to post • In need of ideas

Photo Hobbyist

Persona Type	Photo Hobbyist

Name	Zoey Smith
Image	
Quote	"I love discovering new photo spots, it keeps it fun, and gives me ideas for places to checkout on vacation."
Job	College student
Age	19
Goals	Discover places where she can take amazing photos to print and share in social media
Frustrations	• Making time for photography in her busy class and activism schedule • Discovering photography apps that are easy to learn
Hardware	• Switches between entry level Nikon DSLR and phone camera • Google Pixel • Dell PC
Software	• Preview app for planning her Instagram Gird • Photoshop for light editing

	• Lightroom mobile for editing photos on the go
Thinks	• How are all these photographers so good? • Where should I go next for photos? • I'm always going to the same places, I need something new. • My friends and I should make a list of photo spots for all of us to visit. • It's been a while since I took out my camera, once midterms are over…
Says	• Photography gives me the creative outlet I need for my spare time. • I want to plan more photo trips with my friends. • We should try to get up early and catch the sunrise tomorrow, but it's okay if we don't. • I love golden hour! • I wish I had more time for photography.
Does	• Tries to plan her Instagram content but doesn't follow through • Captures a lot of landscape photography during camping tripsMakes lists of places she would like to travel to • Edits her photos occasionally, but likes to keep her edits light • Keeps a busy class schedule in college • Decorates her room with prints of her best photos
Feels	• Creative • Busy/overwhelmed with school • Excited about photography • Strained for time • Insecure about her photo skills

Competitor Review

Before developing Photo Atlas, I evaluated the current landscape of photography scouting apps to gain inspiration for Photo Atlas, and to see what needed to be improved for potential users. I identified 3 main competitors and made a list of pros and cons for each application.

Google Maps

Google maps is the most popular satellite map and navigation app. It allows users to search places by keywords and view satellite and user-captured images. While not a direct competitor of Photo Atlas, Google Maps is often used by photographers as a scouting tool. In his article for *Digital Photography Schoo*l, Joe Ercoli describes Google Maps and Google Earth as "the most valuable tools to add to your arsenal for finding the perfect photography location."

Pros:

- Has millions of locations around the world available
- Allows users to see the terrain and surrounding areas of the location with street view
- Provides accurate directions
- Detects user location to ease navigation
- Fast and reliable application
- Cross-platform compatible
- Allows users to share locations easily with others
- Allows users to bookmark locations and create lists of locations

Cons

- Not geared towards photographers, so its search results can be too broad. For example, searching "sunset photography in Boston" results in a list of photography businesses in Boston, instead of a list of places to take pictures of the sunset.
- Users can review places in Google Maps, but the app is not designed specifically for photographers to leave photography tips for others.

Depalo

Created by lifestyle blogger Rebecca Aneloski, Depalo is an iOS application for finding Instagram worthy photography spots. The app is mainly geared towards social media influencers or content creators, one of the main audiences of Photo Atlas. The app includes GPS coordinates of locations, photos from other Instagramers, and tips about how to get there, where to park, and how to capture the best photo.

Pros

- Provides tips about how to take the best photos in a location
- Beautiful and easy to use interface
- Allows users to save locations for trips
- Shows examples of pictures taken by others at a specific location

Cons

- Only contains locations within a handful of cities
- Only available on iOS
- Saved locations cannot be shared with others users
- The app does not provide any advanced filtering capabilities
- The user's location is not automatically detected by the app
- Due to its main audience being Instagramers, the app is widely geared towards portrait photography

ShotHotspot

Created by Darren Johnson, ShotHotspot is a search engine for finding photography spots. The app uses the Flickr geolocation API to provide "millions of potential hotspots covering many areas of the world" (Darren 2013).

Pros

- Combines both the Flickr API and crowdsourcing to provide thousands of locations across the world
- Shows examples of pictures taken by others at a specific location
- Has advanced filtering specific to photographer's scouting needs
- Allows users to create public and private lists of photography spots

Cons

- Designed for desktop, and not very responsive for smaller screens
- Does not detect the user's location
- Photography spots are shown in a map but there is no option for the user to "Get Directions" to a spot
- Out-dated UI design is intrusive to the user experience

Competitor Comparison Matrix

I created the following comparison matrix to summarize the advantages Photo Atlass has over similar applications. Currently Photo Atlas has superior functionality to most competitors, with the exception of 2 areas: users cannot add new photography spots to the application, and users cannot leave comments about a location. These are features that I plan to add in the future.

	Google Maps	Depalo	Shot Hotspot	Photo Atlas
Cross platform (website, iOS, Android)	✔			✔
Provides filters geared towards the photography scouting process			✔	✔
Shows examples of photos others have taken within a location	✔	✔	✔	✔
Provides coordinates to photo spots	✔	✔	✔	✔
Provides a way to get directions to a specific spot	✔	✔		✔

Detects user location	✔			✔
Ability to create lists of photography spots	✔	✔	✔	✔
Ability to share lists of photography spots with other users	✔		✔	✔
Ability to create private lists of photography spots	✔	✔	✔	✔
Draws from an external API to provide a larger variety of photography spots			✔	✔
Appealing visual design that contributes positively to the user experience	✔	✔		✔
Has both map view and cards view of locations	✔	✔	✔	✔
Photo spots have comments and recommendations from other photographers		✔		
Allows users to add new photography spots		✔	✔	

Plan and Design

Initial Project Plan

I originally divided my project in 4 key milestones:

1. **Design and Environment Set Up**: Finish preliminary visual and interaction design requirements and prepare the project's development environment
2. **Core Components Set Up**: Code the basic front and backend structure of the application.
3. **Core Functionality Development**: Develop the ability for a user to:
    a. Search for a photography location

b. Create an account

c. Create Trips, or lists of photography locations

d. Add new photography spots to Photo Atlas

e. Edit existing photography spots within Photo Atlas

f. Download the application and use it offline as a Progressive Web

4. **Launch**: Test and deploy the application

I developed an accompanying rubric[1] with this plan, but both the plan and rubric changed. During the implementation of my plan, I redefined what "core functionality" meant. I gained more clarity of what my users would need as a Minimum Viable Product (MVP), and what I could realistically complete during my capstone project. When I reached the Core Functionality Development phase of my project I focused on the first 3 features:

a. Search for a photography location

b. Create an account

c. Create Trips, or lists of photography locations

Proposed Design

I began designing Photo Atlas through sketches then I used Sketch App to create high-fidelity mockups. The first designs were based on the mobile version of the application, as I planned to use mobile-first responsive design to assure the app was easy to use on a phone. Later I created high-fidelity desktop designs.

Having these mock-ups helped me quickly set up my styles when it came to programming. I knew what fonts and colors to use right away, and had a clear idea of how I wanted my layout to look. However, my design did change during development. As my app came to life, I noticed areas of improvement, and tweaked

[1]In the Appendix I have included the original rubric and indicated which areas of this rubric I was able to complete.

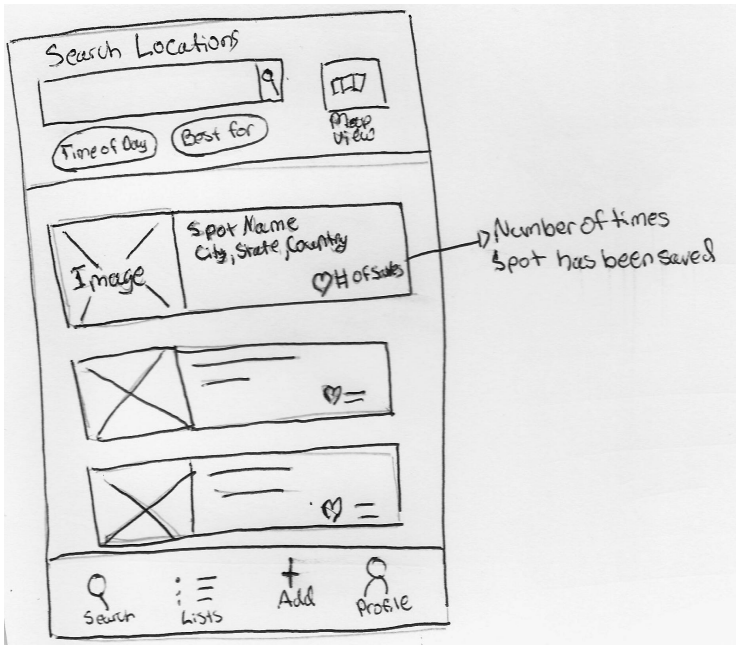
styles and labels as needed. For example, Photo Atlas currently has an Account page, which was previously called Profile. While creating this page, I realized it contained no real user profile information. Instead it was a utility page for updating the user's display name, and resetting the user's password. Therefore I renamed it to the more appropriate label of "Account."

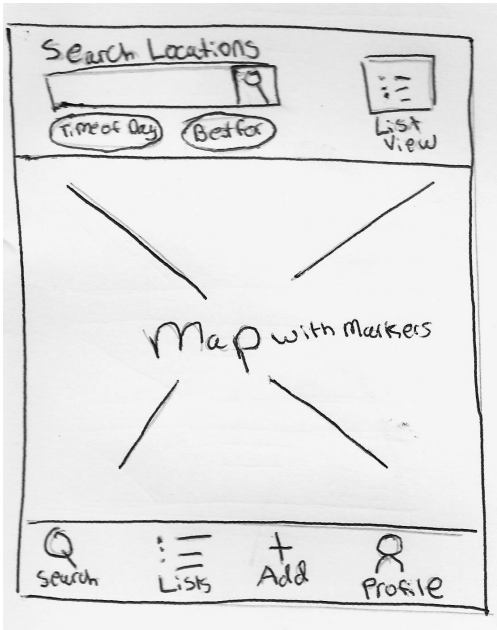
Below are descriptions of a few of my original sketches and mock-ups for the application. In [the next section of this document](), I dive deeper into how my original vision for the product changed during implementation.

From Sketches to Mock Ups

Homepage

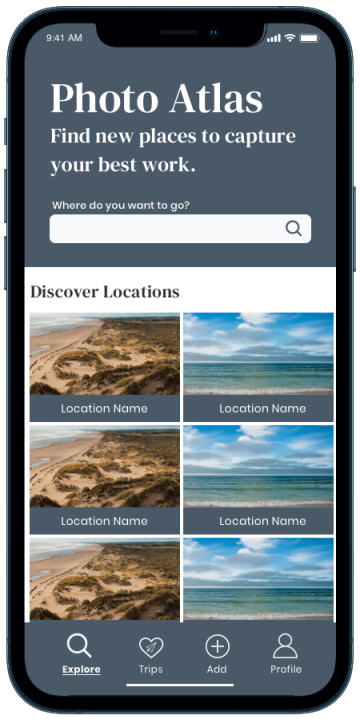
On my original drawings the Homepage consisted of a header with a search bar and some filters, and a list of locations below. Users could toggle between map view and list view of locations using a button on the top right corner of the screen.

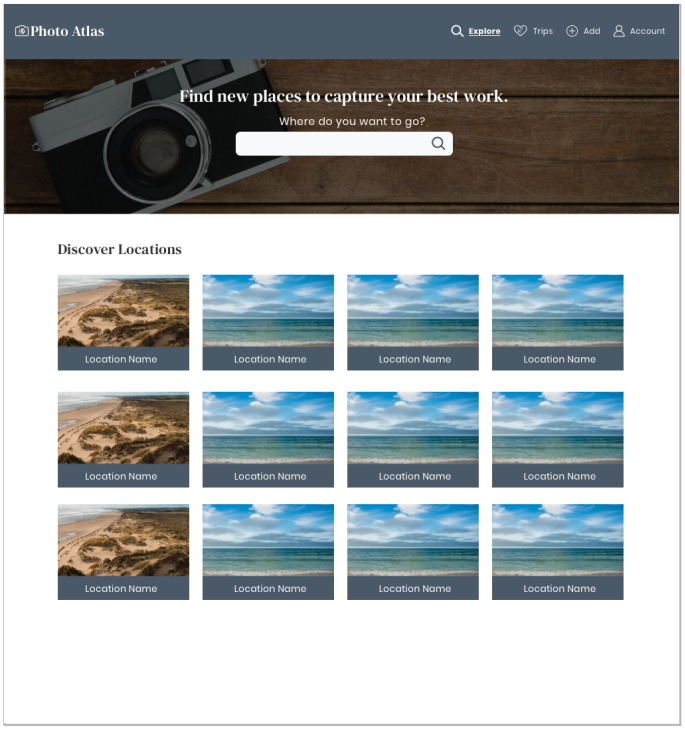


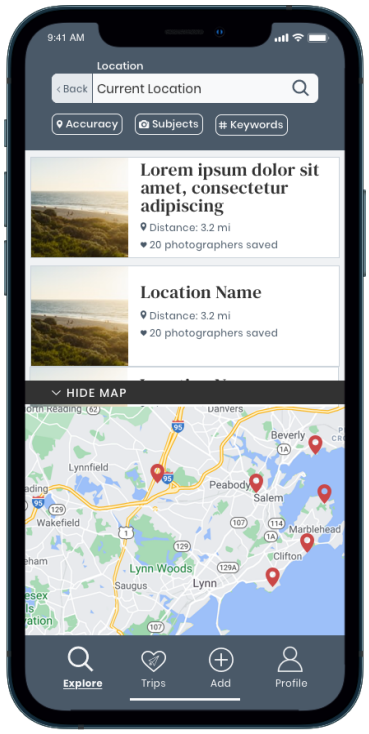


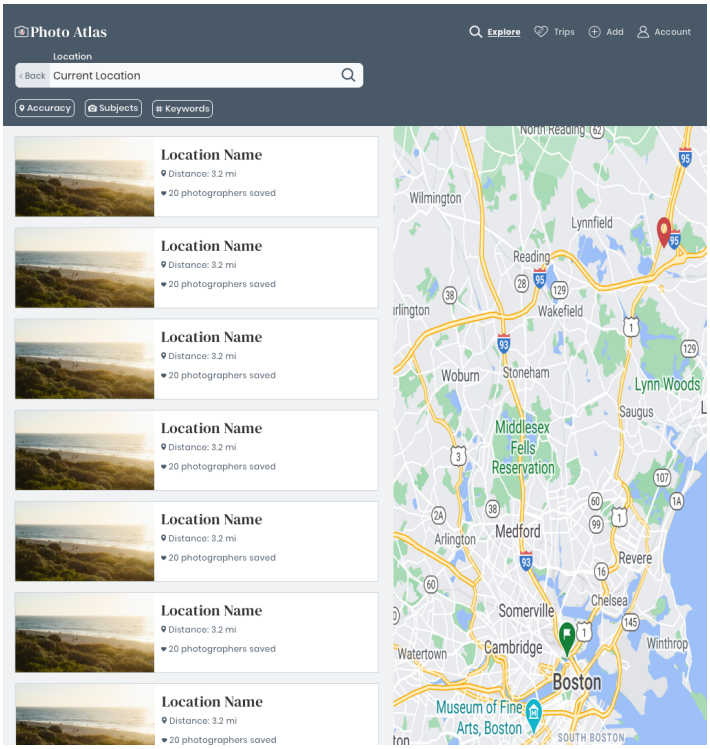
Later I designed a more visually appealing Homepage, that presented the user with the name of the application and a tagline, followed by a search bar and large cards leading to found nearby locations. Once the user searches on the Homepage they go onto an Explore screen, which, on mobile, has a similar layout to the original

hand-drawn design, except it more neatly incorporates the map and list view of locations.



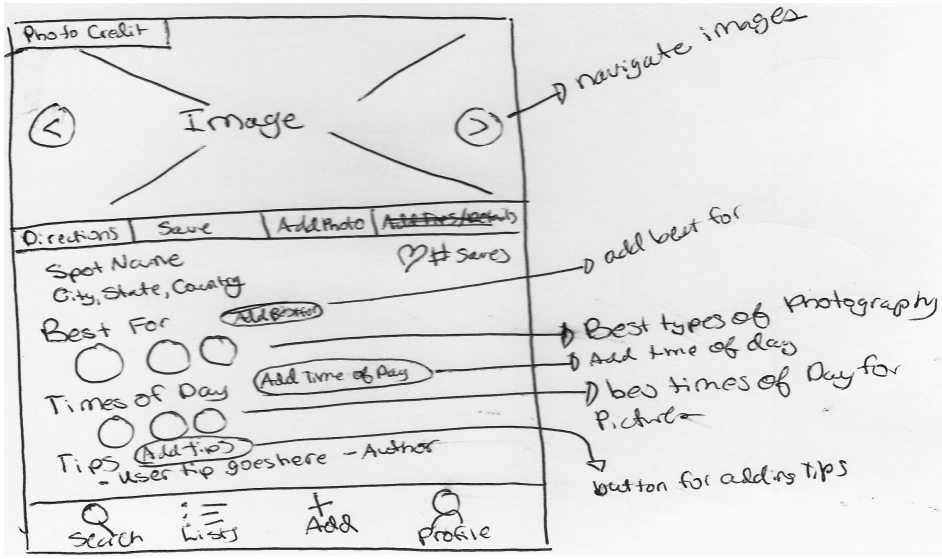


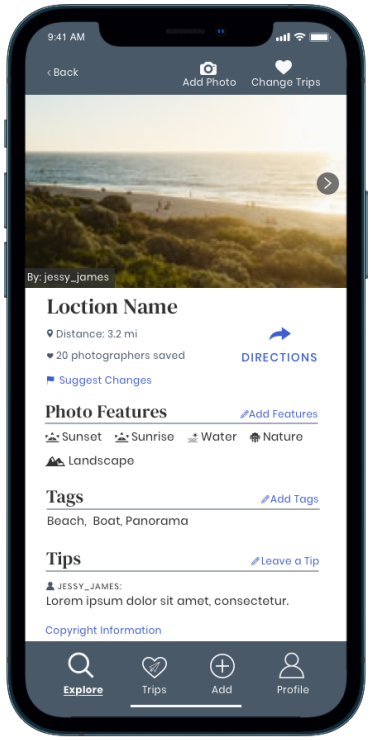


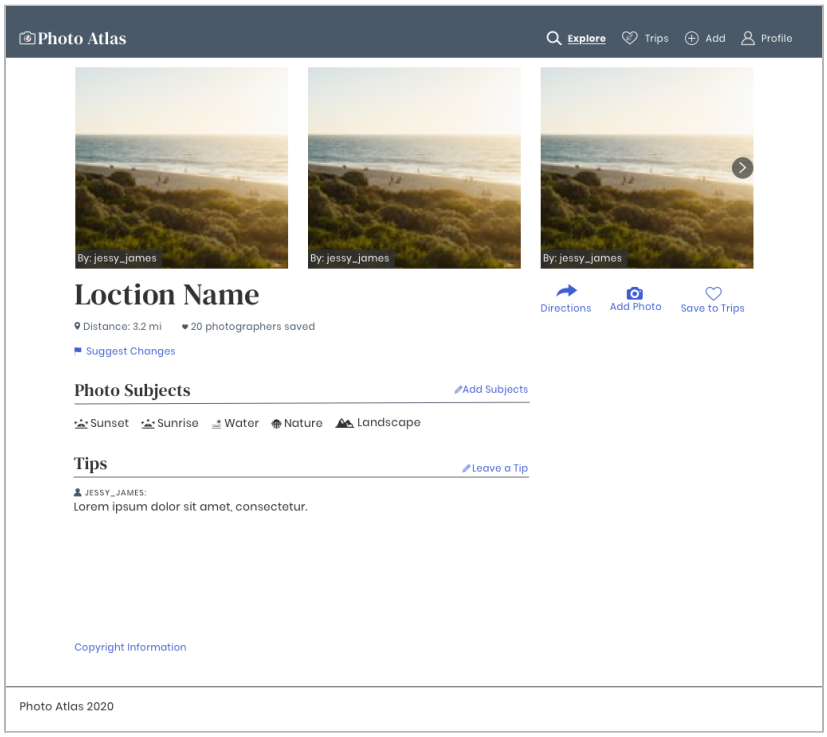


Location Page

The Location page shows the details about a photography spot. The goal here is to present images of the location, and important data that the photographer should know, such as what types of photographic subjects or features can be found in the area. These initial designs also show how I planned to allow users to add comments and details to these pages. However, this feature was out of scope for the capstone.

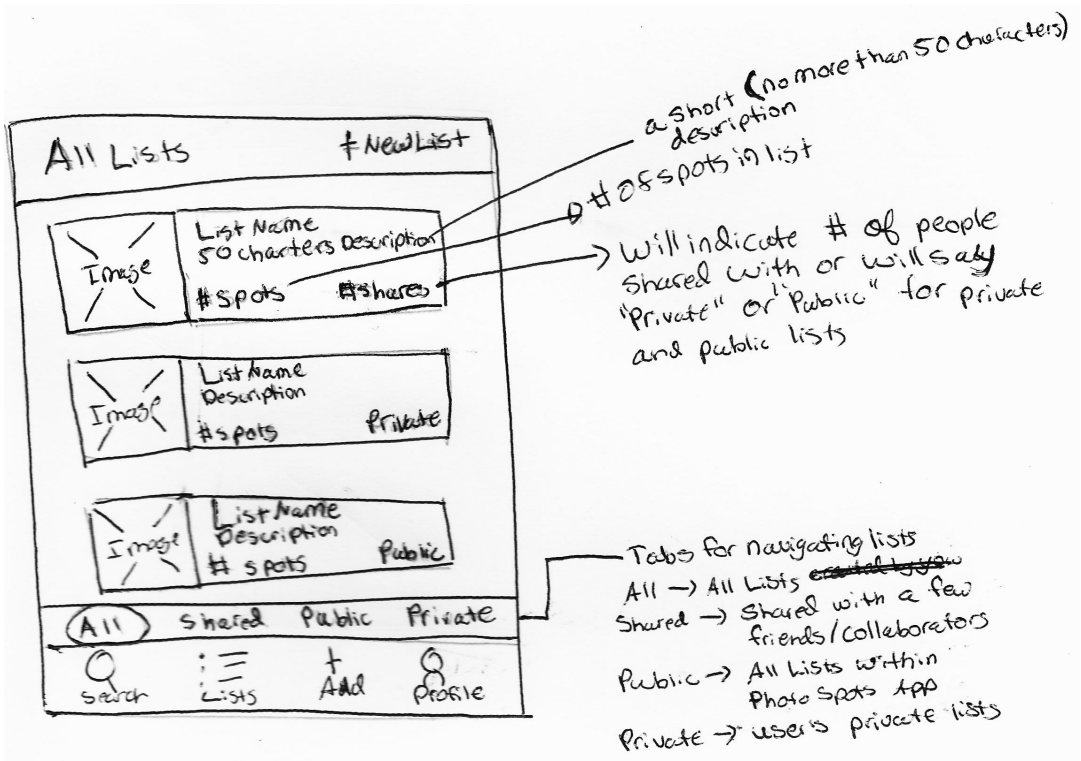


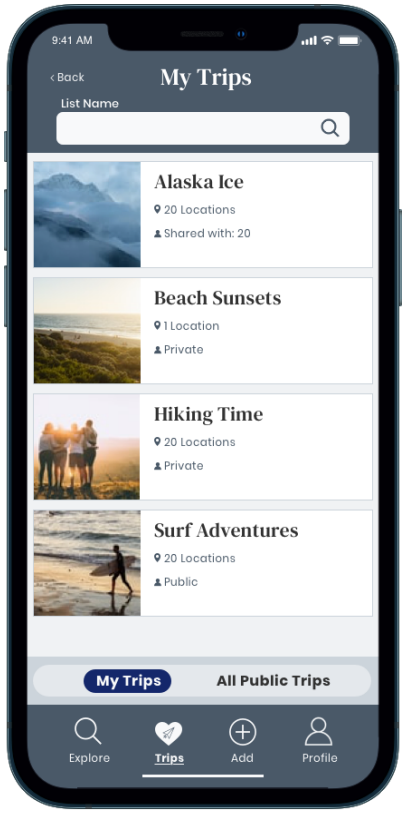


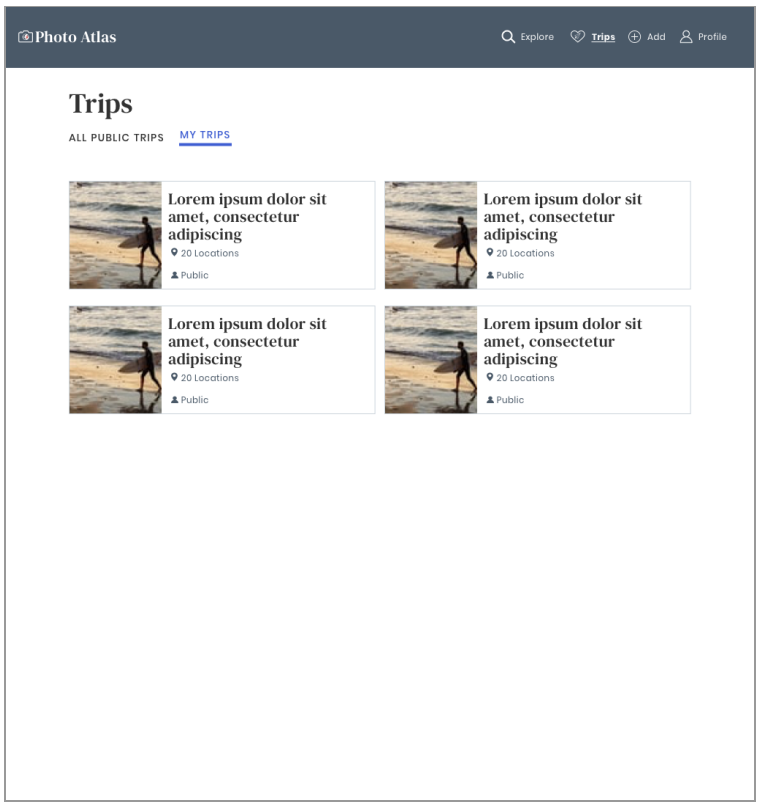


Trips Page

The Trips page was originally called Lists. It is meant for users to find curated lists of places they can visit. These lists can be created by any user with a Photo Atlas account. The layout did not change much from sketch to mock-up, but I did make some changes to the tab-structure. Originally I sketched out 4 tabs: All, Shared, Public, and Private, but while mocking up the design I condensed these to 2 tabs: My Trips and All Public Trips. During development, I made even more changes by adding a Bookmarks tab, , which I describe in the Site Map section of this document.







Implementation

Project Pivots

As mentioned earlier, the development process led to several pivots. Below I describe the biggest turning points for the Photo Atlas application.

Time Constraints and Unconsidered Features

While working on Photo Atlas, I discovered that implementing the Search Functionality was a difficult and time consuming process. I had originally allocated 5 weeks to the development of this feature, but exceeded this plan by 2 weeks. Furthermore, I began to realize that my original project rubric did not take into account the time I had spent on the following important features:

- An appealing visual design, which would give me an edge over competitors such as Shot Hotspot

- A design that meets basic accessibility standards, such as: largely semantic markup, and color contrast compliance
- A Locations page which shows the user details about a specific photography location
- The user's ability to reset their password if they forget it
- The ability to remove locations from an existing Trip
- Displaying nearby locations from the Homepage of the application, to give users and idea of what Photo Atlas can do for them before they even start their search

How Might We

An exercise that helped me better define my MVP was developing How Might We statements. I realized I needed to solve for 3 main problems:

1. How might we make it easier for photographers to find photography spots recommended by others?
2. How might we help photographers tailor the photography search to their current needs?
3. How might we create an application that works on desktop and mobile devices?

I had originally planned to give users the ability to add locations to Photo Atlas, and to edit existing Photo Atlas locations. However, these two features were beyond the scope of an MVP.

Furthermore, I had originally planned to make my application cross platform by making it into a Progressive Web Application, but I realized this too was beyond the scope of an MVP. The main issue I was trying to solve was that users could only access most scouting apps through either their mobile or desktop devices, but not both. A simpler way to solve this issue was to focus on creating a responsive web application.

Feedback

When I presented the Trips functionality to classmates, they raised the following questions, which I had not considered:

1. What if there are a lot of Trips, how would I search for one?
2. What if I like a Trip that was created by another user? How do I save or bookmark it for later?

With this feedback I realized that I needed to dedicate more time on the Trips feature of the application. I decided to add:

1. A search bar on the Trips screen to allow users to search for a specific Trip by title or tags
2. A Bookmark feature, for users to be able to bookmark Trips from other users

A Hidden API Cost

One feature shown in my original mock up was the ability for users to see the driving distance between them and a specific location. While I did manage to create this feature, I eliminated it due to hidden API costs. To create this feature I worked with the Distance Matrix API from Google Maps. Unfortunately, I discovered that the standard $200 monthly credit that Google provides for Google Maps does not cover the Distance Matrix API. I discovered this reality the hard way, when Google charged me for my usage of the API. The feature is currently too costly for the amount of value it provides. Therefore I had to make the difficult decision of removing it.

A New Rubric

During the planning phase of my project, I had created a detailed rubric, which I meant to use as a way to define success for my project. However, as my definition of MVP became clearer, I decided I needed to change my rubric in order to give myself clearer goals. The rubric below is my final rubric. Area number 3 of each section defines my MVP, whereas area 5 defines additional goals that I wanted to achieve by

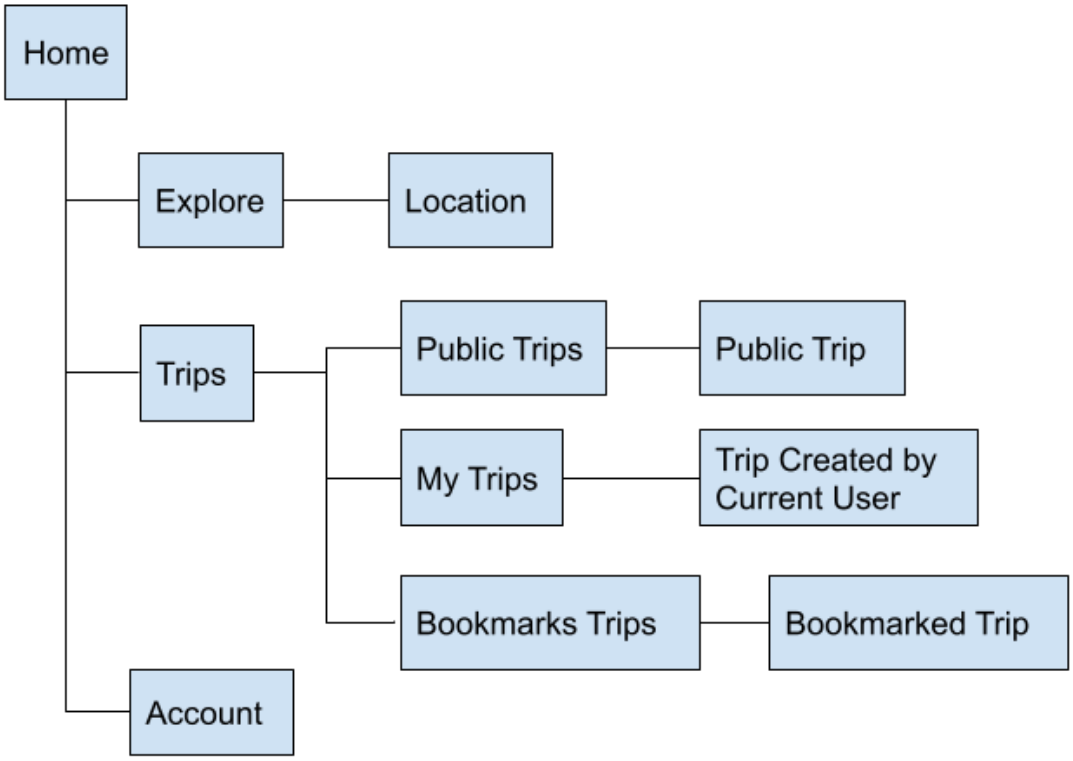
the end of the capstone project. I am happy to say I was able to meet area number 5 for every section of this rubric, as indicated by the use of the color gold.

Explore / Search Feature				
1	**2**	**3 - MVP**	**4**	**5**
When they first come to the app, users see a list of nearby locations loaded from the Flickr API.	If users click on a location on the Homepage, they can see a page with more information about that location.	Users can type an address and the app will show them a list of locations within that given area. These locations will come from the Flickr API.	When they search, users can not only see a list of photography locations, but also see those photography locations in a map.	The app has advanced searching capabilities. Aside from letting users provide a location, it also lets them filter locations.
Saving Locations / Trips Features				
1	**2**	**3 - MVP**	**4**	**5**
From a location's detail page, users can add and remove the location from different lists, called Trips.	When they create a trip, users can make locations public, for other users to see or private, so only they can see them.	Users can edit a Trip's name, and privacy settings, or delete a trip.	Users can follow another person's public trips, so they can easily access them from their Trips screen.	The Trips screen has a search bar that makes it easy for users to search for a Trip.
Account Creation and Management				
1	**2**	**3 - MVP**	**4**	**5**
Users can create an account with an email and password.	Users can set a display name to their account.	If they forget their password, users can reset it before they sign in by providing their email.	The app requires the user's email to be verified for better security and account handling.	Users can reset their password and change their display name after logging in, by going to their Account page.
Visual, Responsive, and Accessible Design				
1	**2**	**3 - MVP**	**4**	**5**
The app works	The app has a	The app has a	As an application	The app's markup

well on desktop and laptop screens.	responsive design that adjusts to the user's screen size.	largely consistent and appealing visual design.	that requires JavaScript, the app and encourages users to activate JavaScript if their browser has JavaScript off. Thus it meets base-level progressive enhancement.	and colors meet accessibility requirements. If an accessibility checker like the Wave plugin is run, no major errors are found.
Development Environment Set Up				
1	**2**	**3 - MVP**	**4**	**5**
A Create React App project has been created. A Github repository has been set up for version control.	A Firebase Firebase Auth and Realtime Database are set up in the project.	The app has been deployed with Netlify. Environment variables are set up on Netlify so that the deployed app is linked to the database.	The app has been deployed using continuous deployment with Github, so that the launched app updates upon merge to the main branch.	Not only is the app set up with continuous deployment, it also has a custom domain and an SSL certificate.

Product Architecture

Site Map



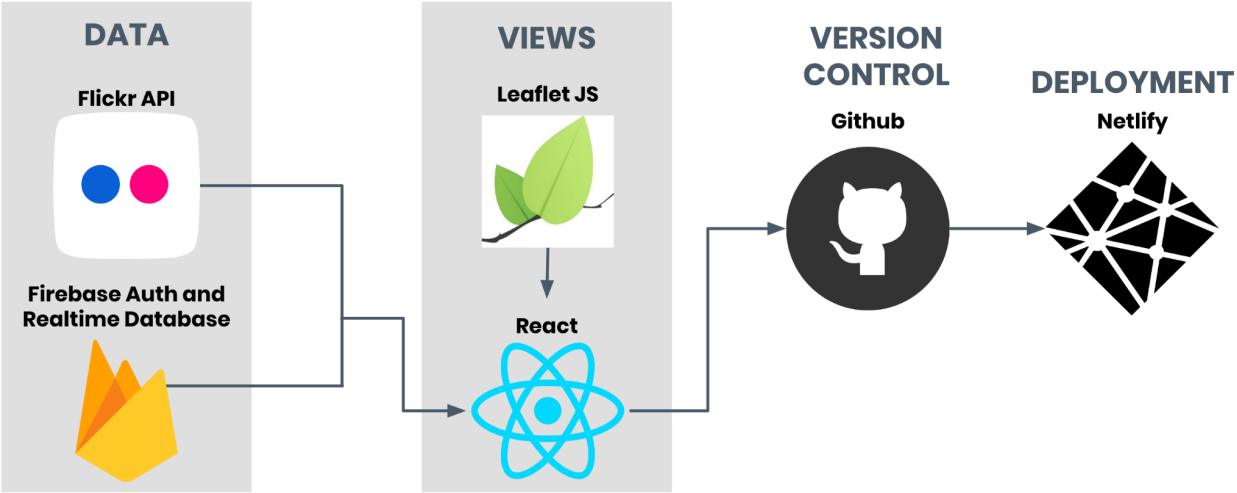
From the Homepage Photo Atlas branches into 3 main pages:

- Explore, where users can see a list and a map of searched locations, and filter those locations based on different photographic subjects.
    - When they click on a result on the Explore screen users go onto the Location page for that result, which gives them more information about that location. On this page users can also save a location to a Trip to reference later.
- Trips, where users can access lists of photography locations created by themselves or other users. These trips are divided into 3 tabs:
    - Public Trips, which contains lists viewable to all Photo Atlas users

  - My Trips, which contains the Trips the current logged-in user has created
  - Bookmarks, which contains trips that the current logged-in user has saved to reference later

- Account: where users can change their display name, reset their password, and log out of the application.

Tech Stack



Similar to a JAMstack model, Photo Atlas primarily works on the frontend, and uses APIs to load data. The application is built on React, and also uses the Leaflet JS library for mapping functionality. It queries the Flickr API to obtain photography location results for users, and users Firebase to authenticate users and to store data from Trips. I used Github for version control and Netlify to host and deploy the application.

Technology

In this section I describe with more detail how the technology behind Photo Atlas works.

React

To easily set up all the dependencies required to create a React application I used the Create React App boilerplate and customized it to my needs. To organize my React components I used the following structure on my `src` folder:

- `components/`, for storing small reusable component units
- `images/`, for storing icons and logos
- `screens/`, for storing the different pages of the application
- `styles/`, for storing all Sass files that contain the app's styling
- `utils/`, for storing reusable helper functions
- `App.js`, which contains the main routes of the application
- `index.js`, which contains the `React.DOM.render()` function that renders the application to the DOM

Flickr API

To query the Flickr API and render its results to Photo Atlas I created 4 helper functions that can be found under `src/utils/flickr/`.

- `getFlickrPhotos()`, which takes an object that contains a series of parameters that can be passed to the Flickr API's `flickr.photos.search` method. I primarily used this function to generate search results on the Homepage and Explore page.
- `createFlickrImageUrl()`, which can take a JSON result from `flickr.photos.search` and generate the url for a Flickr photo
- `getFlickrPhotoInfo()`, which takes an object that contains a series of parameters that can be passed to the Flickr API's `flickr.photos.getInfo`. I primarily used this function to properly credit photos on a Location page.

- `getFlickrPlace()`, which takes an object that contains a series of parameters that can be passed to the Flickr API's `flickr.places.getInfo` method, and returns the name of a given place, if the name is available. I primarily used this function to help name search results on the Explore Screen.

Leaflet JS

The Leaflet library renders a map of with makers to photography locations. Users can click on these markers to visit a Location page. Having a map allows users to determine how close to each other locations are. To easily connect Leaflet I used the `react-leaflet` npm package, which comes with predefined React components for Leaflet maps.

Additionally, I used the `leaflet-geosearch` library. This library allowed me to select a free data provider called Algolia to generate address suggestions as the user types into the search bar. While `leaflet-geosearch` does not directly rely on Leaflet, it easily integrates to the Leaflet library.

Firebase Authentication

I used Firebase Authentication capabilities to manage user accounts. Photo Atlas relies on the following Firebase Authentication methods:

- `createUserWithEmailAndPassword` to create user accounts
- `sendEmailVerification` to verify a user's email when a new account is created
- `signInWithEmailAndPassword` and `signOut` to handle sign-ins and sign-outs
- `onAuthStateChanged` to observe which user is logged in, if any
- `sendPasswordResetEmail` to reset user passwords

Firebase Realtime Database

I used the Firebase Realtime Database to store information about trips created by users. I organized Trips by their privacy settings, so I could more easily control the

read and write rules for them. There are 3 main branches for Trips: Private, Public, and Shareable.

```
privateTrips
publicTrips
shareableTrips
```

Under each of these branches are lists of Trip IDs, and under each ID, there is information about a Trip.

```
privateTrips
  trip-18-APR-2021-kFzBILM-13-14-2-367
    author: "Lo2ukrj8PEg0zZMmSNkyOZ7tHHs1"
    createdOn: "18-APR-2021"
    featuredImg: "https://farm66.staticflickr.com/65535/511105936..."
    lastEdited: "18-APR-2021"
    locations
      flickr-51110593690: true
    locationsCount: 1
    tags: "city, urban transportation"
    tripName: "Boston Transport"
```

To easily match a trip with a user, I created a `userTrips` branch that stores a list of user IDs, and under these are lists of trip IDs. Below each trip ID, I also stored the Trip's privacy setting so I would know which Trip branch to query.

```
userTrips
  Kqx396EXE3QvTxM9XAUqDRxlegb2
    trip-18-APR-2021-ECMQOEj-16-13-8-267
      tripPrivacy: "public"
    trip-18-APR-2021-GmqF8yG-15-8-9-336
    trip-18-APR-2021-GwHl1Ar-16-11-56-649
```

Additionally, I created a `userBookmarks` branch that stores bookmarks for a specific user, and follows a similar structure to the `userTrips` branch.

```
userBookmarks
  Kqx396EXE3QvTxM9XAUqDRxlegb2
    trip-18-APR-2021-B0yjRsL-12-44-45-318
      tripAuthor: "Lo2ukrj8PEg0zZMmSNkyOZ7tHHs1"
      tripPrivacy: "public"
  YqfdRjBXUPSgcJ8dMVoy5hsG7FG3
```

Lastly I wanted to easily query Firebase to show a user's display name on a Trip that they created. Unfortunately storing a user's display name within Firebase Authentication makes it difficult to make the user's display name public. Therefore, I choose to store user display names in the Realtime Database.

```
users + ×
  Kqx396EXE3QvTxM9XAUqDRxlegb2
    displayName: "Zory"
```

Netlify and Github

To assure easy updates of the application's code, I set up continuous deployment with Netlify and Github. This way, every time I update my code on Github by pushing code to the `main` branch, Netlify runs the `npm run build` command to deploy the updated code.

Additionally I set up environment variables on Netlify to store the application's API keys. This way I avoided having my keys public to users on my Github repo.

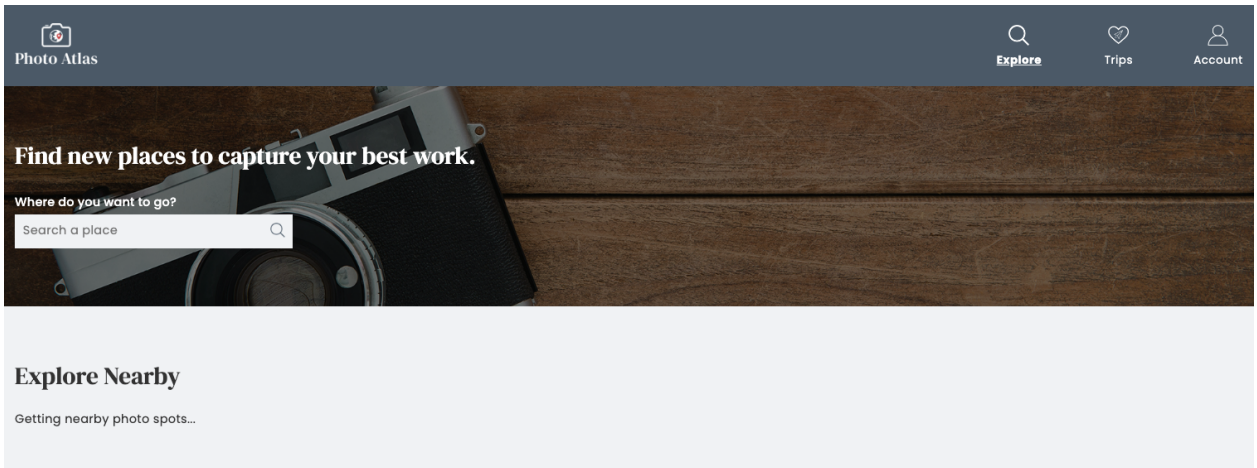
Testing

To assure Photo Atlas worked as expected, I conducted several automated and manual tests, and identified potential areas of improvement.

Manual Feature Test

I first tested the application by going through each of the main screens and assuring all features properly functioned. Using this approach I identified problems with the Homepage and the Account page.

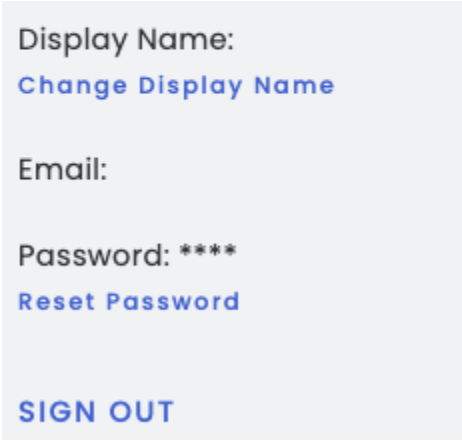
On the Homepage, I realized that no locations were loaded when there was a security setting that prevented the app from getting the user's location. Instead the app continued to say "Getting nearby photo spots..." This made the application appear unreliable.



To solve this issue, on my React code I set the coordinates of Cabo Rojo, Puerto Rico as a default location. This way if the app encounters an error or does not get the user's location within 3.5 seconds, it will show photos of Cabo Rojo on the Homepage.

```
240     componentDidMount() {
241
242         this._isMounted = true;
243
244         let defaultLocation = {
245             latitude: 18.087,
246             longitude: -67.1459
247         }
248
249         let setDefault = () => {
250             if (!this.state.currentLocation) {
251                 this.setState(
252                     { currentLocation: defaultLocation, default: true },
253                     () => this.getNearbyLocations()
254                 )
255             }
256         }
257
258         getCurrentLocation(
259             position =>  {
260                 if( this._isMounted && !this.state.default) {
261                     this.setState({ currentLocation: position.coords}, () => this.getNearbyLocations())
262                 }
263             },
264             err => this._isMounted && this.setState({ errorMessage: err.message }, ()=>setDefault())
265         )
266
267         setTimeout(function(){ setDefault() }, 3500);
268
269
270     }
```

On the Account page, the current user's display name and email did not show on the page. Logs revealed that there was a problem with database permissions.




```
2 ▸ Error: Permission denied                    Account.js:82
      at Repo.ts:368
```

I went back to my Firebase Realtime Database security rules and reviewed them. I allowed all users to read display names so that they could know who created a

specific Trip on the Trips screen, but I made display names and emails editable only by their owners.

```
"users" : {
  "$uid" : {
    ".read" : true,
    ".write" : "$uid === auth.uid"
  }
},
```

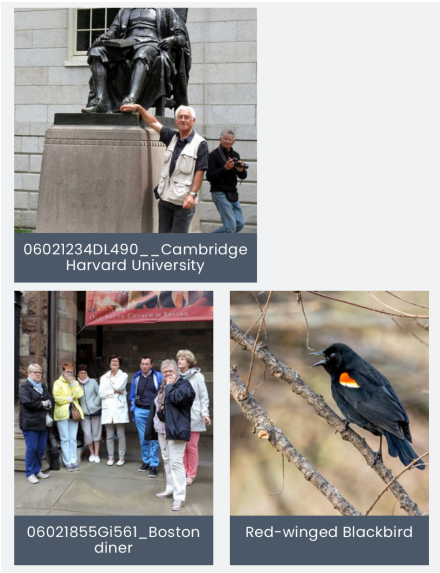
Responsive Design Test

Next I manually tested my app for responsiveness. I focused primarily on testing how each page looked in different window sizes, and different browsers. To complete this test I:

1. Used Chrome Developer tools as an initial way to view my app on different screens
2. Opened and used my application on Safari and Mozilla to assure there was consistency with Chrome
3. Opened and used my application on Safari on my iPhone
4. Used the BrowserStack extension to view my application on Android phones and Internet Explorer

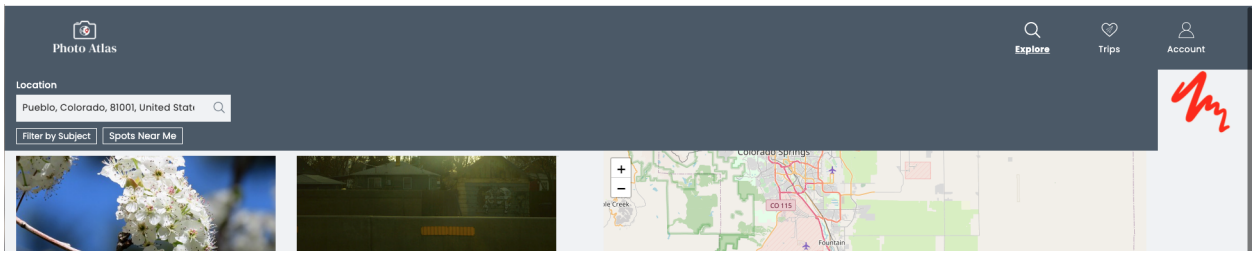
With these steps I was able to find a few issues with the app's layout.

First I noticed that on mobile devices, the grid of images on the Homepage broke, due to different image widths. To fix this issue I set a width of 320px to the cards, and hid the overflow to account for wider images.



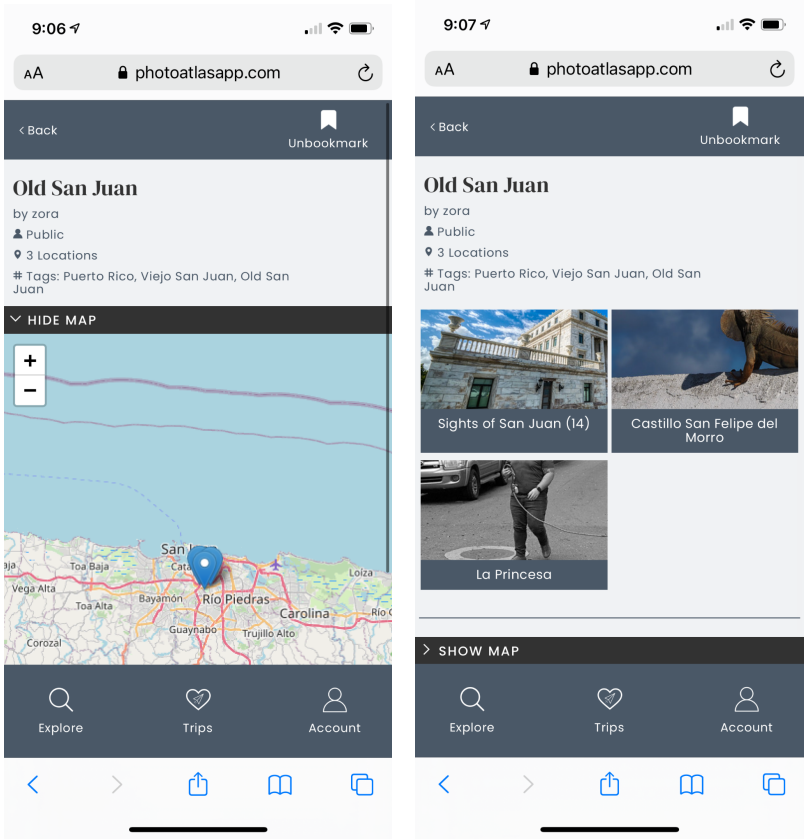

```
.title-card {
    background-color: $gray;
    color: $white_text_color;
    width: 320px;
    overflow: hidden;
```

I also discovered that on wider screens, page headers sometimes did not span the full width of the page, due to the `<main>` tag of the page having a `max-width` of 1800px. I was able to resolve this issue by eliminating this CSS rule, and using containers within `<main>` to constrain the page to 1800px when necessary.

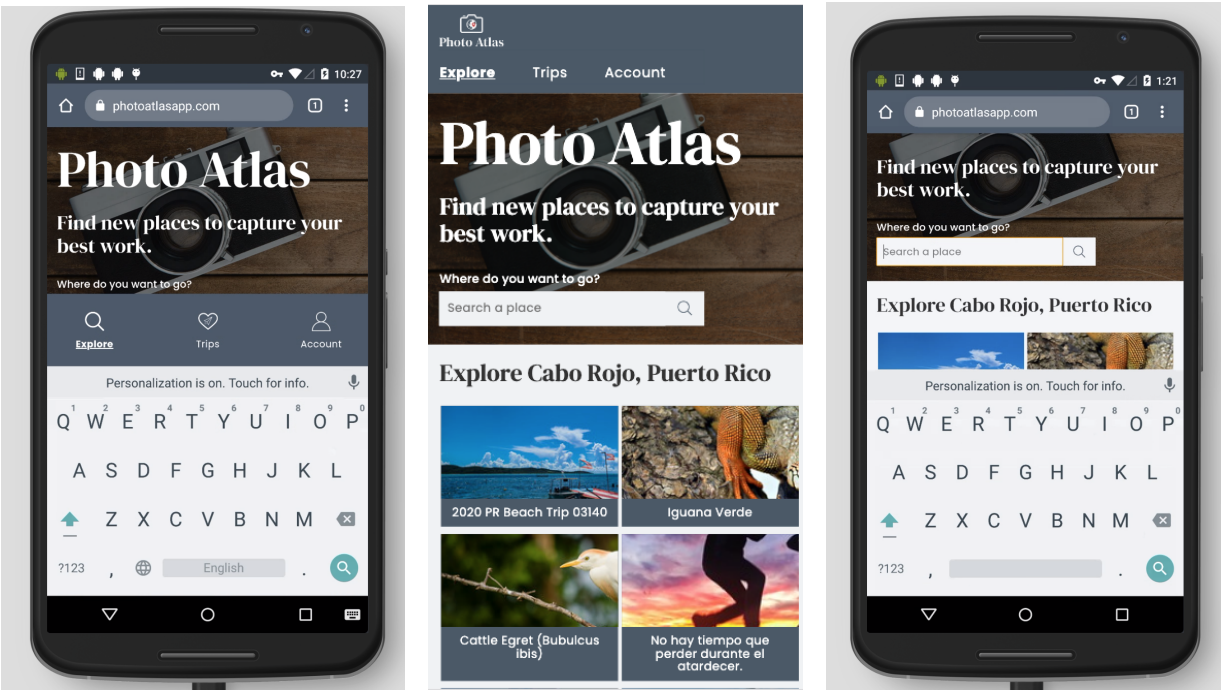



```
main {
    width: 100%;
    max-width: 1800px;
}
```

Opening the app on a phone revealed another issue. When I opened a Trip, the map on the page covered most of the screen. I resolved this issue by making the map hidden by default. If the user wants to use the map, they can tap Show Map for the map to occupy the majority of the screen.

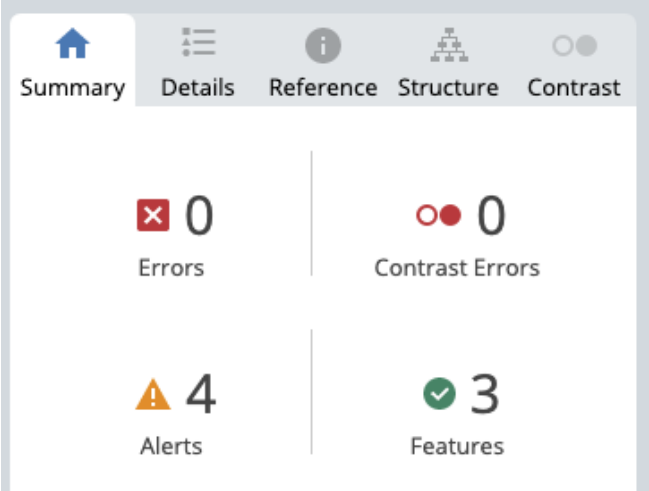


Lastly, using BrowserStack revealed a problem with Android phones. As with iPhones, whenever an Android user taps on a text bar, their keyboard pops-up automatically. Photo Atlas had no problems with this behavior on iPhones. However, on Android, the keyboard would push up the the bottom navigation bar of Photo Atlas, covering the textbox. To solve this issue, I replaced the bottom navigation bar with a more standard navigation bar at the top of the application.



WAVE Accessibility Audits

After testing for responsiveness, I used the WAVE Chrome extension to assure my application met digital accessibility standards. WAVE revealed no errors with Photo Atlas but it did show a few warnings that I thought useful to address.



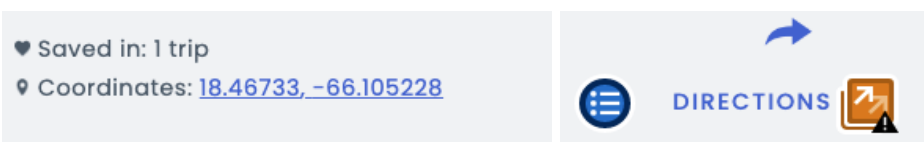
Some warnings were false calls. For example I had set several items on the page set to `font-size: 0`, to hide text from sighted users, but have it called for screen reader users. One example of this was the `<h1>` tag on the page. The words Photo Atlas were

visible on the PNG logo of the page, but images cannot be read by screen readers.

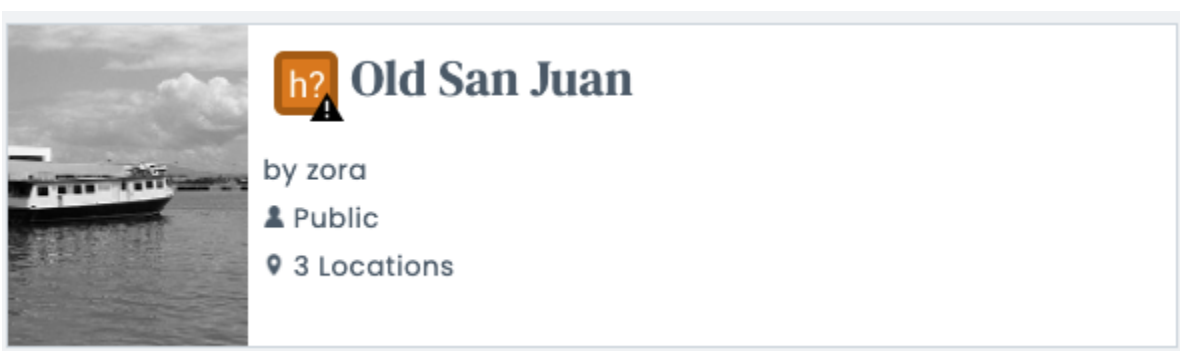
```
▼<div id="root">
  ▼<header>
    ▼<div class="container">
      ▼<div class="logo">
        ▼<a href="/">
            <h1>Photo Atlas</h1> == $0
            <img src="
            R…nUYrKsoR4FwAAAABJRU5ErkJggg=="
            alt="PhotoAtlas logo">
          </a>
```

```
element.style {
}
header h1 {
   font-size: 0;
   line-height: 0;
}
.h1-font, .h2-font,
   font-family: "DM
   font-weight: 400;
}
```

However, WAVE did call out the fact that on Location pages, the Coordinates link and the Directions link lead to the same page. This could prove inconvenient to those navigating with their keyboards, as they would hit the same link twice in a row. As a solution I eliminated the link on the coordinates.



WAVE also indicated that on my Trips screen I was using a `<p>` tag on the titles of trips, when these were headings. As a solution, I changed these tags to `<h3>`.



Conclusion: Final Product and Beyond the Capstone

I launched Photo Atlas with all the testing improvements at photoatlasapp.com. Overall the application is stable and responsive, but there is always room for growth. Beyond the capstone project, I plan to continue improving the app's functionality.

My first order of business is to run usability tests. I consider user experience a top priority, but having a full-time UX Designer/Researcher career, I wanted to take this capstone period to really focus on the front-end engineering side of product development. Thus I unfortunately did not dedicate much time for testing with real app users. But dedicating time to engineering allowed me to become more comfortable with React and with solving problems with code. Now I have a stable product for gathering feedback and programming knowledge that can help me address future product challenges.

With the feedback I gain from usability testing, I can continue improving on the current existing functionality, and later add further features. Particularly I would like to create the ability for users to add their own photo locations to application, and to leave comments and add filters to existing photography locations. I did not have time for these features during the capstone. Yet they are very important for the purpose of collaborative location scouting, which lies at the core of the application.

The purpose of Photo Atlas goes beyond helping photographers find great locations. The app also aims to help photographers see where others are capturing great photos, and to feel inspired to discover these places for themselves. Any future app improvements will be geared towards this goal.

Appendix

Project Presentations

- MVP Video Presentation
- Final Video Project Presentation
- Photo Atlas Pitch Video

Original Project Rubric

The areas colored gold indicate which sections of this rubric I was able to meet. I did not meet any of the points on the section called Photography Spots Curation Features because I realized that none of these features were essential to the MVP. Additionally, I was able to reach number 4 on Spot Search Features and Saving Features with a few caveats.

For Spot Search Features, I did not create a small set of photography spots on the Photo Atlas database, as outlined on 1. Instead of focusing on allowing users to query the Photo Atlas database, I focused on allowing them to query the Flickr API, as listed on area 5. The reason for this was that Flickr provided many more results than those which I could manually add to the database.

For Saving Features, I did not include a feature that allowed users to crowdsource their Trips, as outlined on 3. However, I did include the ability for users to change between 3 privacy settings, similar to what 5 indicates.

Spot Search Features				
1	**2**	**3**	**4**	**5**
The app's database has a small set of photography spots, which users can see in a map on the front end of the app.	Users can type an address and the app will search within its database for any spots in that area.	Users can authorize the app to detect their location and use this location to search for spots within its database.	The app has advanced searching capabilities. Aside from letting users provide a location, it also lets them filter locations.	The Flickr API supplements the app's database, giving users a larger set of spots. The app searches within a typed location or the user's current location, and

				allows users to add filters.
Saving Features				
1	**2**	**3**	**4**	**5**
Users can save photography spots to their profile to see later.	Users can organize their saved spots into different trips (lists of photography spots).	Users can share their photography trips with all users of the app to crowdsource them.	Users can turn their private trips into public trips and vice versa.	Users have more advanced privacy settings for their lists. They can choose to share their trips with a select group only.
Photography Spots Curation Features				
1	**2**	**3**	**4**	**5**
Users can leave comments on photography spots.	Users can add comments and their own photos on photography spots.	Users cannot only add details to a spot, they can also report issues with the current information of a spot through a contact form.	In addition to comments and photos, users can add tags and filters to photography spots, which the app will use to help refine searches. Users can also report issues with a spot.	Aside from being able to add information or report issues, users can create their own photography spots that will be added to the app's database.
Cross Platform Design				
1	**2**	**3**	**4**	**5**
As a mobile first platform, the product looks good as a mobile website, but does not adjust well to larger screens. It also does not have any markup	The app has markup and CSS enhancements that help optimize images for mobile data loading.	The app is not only fully functional in mobile, it also adjusts well to desktop/laptop screens.	Aside from being responsive, the app follows principles of progressive enhancements, by partly loading when JavaScript is off on the	Product has cross platform layout and graphics optimizations, plus the caching and offline capabilities of a Progressive Web App.

and CSS enhancements needed for optimizing images on mobile.			browser, and encouraging users to activate JavaScript.	
Development Environment Set Up				
1	**2**	**3**	**4**	**5**
A Create React App project has been created, along with a Github repository.	A Firebase reference has been added to the project.	The Firebase Realtime Database for the app has a small set of photos. Firebase Auth is set up in the project so that users can create accounts.	The app has been deployed with Netlify and has continuous deployment with Github. Environment variables are set up on Netlify so that the deployed app is linked to the database.	Not only is the app set up with continuous deployment, it also has a custom domain and an SSL certificate.

References

Achard, C. (2020, March 18). Tips for Organizing React Projects. Retrieved October 12, 2020, from https://dev.to/chrisachard/tips-for-organizing-react-projects-191

Ayodeji, B. (2019, April 01). Building & Deploying your first Progressive Web App. Retrieved October 05, 2020, from https://medium.com/backticks-tildes/build-your-first-progressive-web-app-a-weight-converter-app-and-deploy-to-netlify-b55ed4a86413

Batschinski, G. (2020, May 04). Firebase: Top 5 open-source alternatives. Retrieved October 11, 2020, from https://blog.back4app.com/firebase-open-source-alternatives/

Build a PWA app with React.js. (2020, July 10). Retrieved October 25, 2020, from https://koala42.com/build-a-pwa-app-with-react-js/

Caching Files with Service Worker | Web | Google Developers. (2019, May 1). Retrieved September 26, 2020, from https://developers.google.com/web/ilt/pwa/caching-files-with-service-worker

Clark, B. (2017, November 30). Firebase Alternative: 3 Open-Source ways to follow. Retrieved October 11, 2020, from https://medium.com/@brenda.clark/firebase-alternative-3-open-source-ways-to-follow-e45d9347bc8c

Clark, M. (2020, September 21). Netlify vs Heroku: What are the differences? Retrieved October 11, 2020, from https://blog.back4app.com/netlify-vs-heroku/

Create a New React App. (n.d.). Retrieved September 05, 2020, from https://reactjs.org/docs/create-a-new-react-app.html

Darren, J (June 9, 2013). The Photography Locations Search Engine. Retrieved July 28, 2020, from www.shothotspot.com/about/press/release/shothotspot-helps-you-find-the-perfect-photography-location/

Data USA: Photographers. (n.d.). Retrieved September 07, 2020, from https://datausa.io/profile/soc/photographers

Davis, S. (2017, April 04). The Most Distinctive Baby Names for Each of the Past 7 Generations. Retrieved September 20, 2020, from https://www.mentalfloss.com/article/93923/most-distinctive-baby-names-each-past-7-generations

Demographics of Internet and Home Broadband Usage in the United States. (2020, June 05). Retrieved September 07, 2020, from https://www.pewresearch.org/internet/fact-sheet/internet-broadband/

Domes, S. (2017, June 27). Build your first progressive web app with React. Retrieved September 26, 2020, from

https://engineering.musefind.com/build-your-first-progressive-web-app-with-react-8e1449c575cd

Enabling Offline Capabilities in JavaScript. (n.d.). Retrieved October 24, 2020, from https://firebase.google.com/docs/database/web/offline-capabilities

Ercoli, J (2014, May 25). Find the Perfect Photography Location Using Google Maps. Retrieved July 28, 2020 from https://digital-photography-school.com/find-perfect-photography-location-using-google-maps/.

Erikson, M., & Shoemaker, S. (2017, January 25). Redux and Why it's Good For You. Retrieved September 05, 2020, from https://www.newline.co/fullstack-react/articles/redux-with-mark-erikson/

Exifreader. (2020, October 10). Retrieved October 11, 2020, from https://www.npmjs.com/package/exifreader

Farrugia, K. (2016, August 11). A Beginner's Guide To Progressive Web Apps. Retrieved September 26, 2020, from https://www.smashingmagazine.com/2016/08/a-beginners-guide-to-progressive-web-apps/

The Flickr Developer Guide. (n.d.). Retrieved September 05, 2020, from http://www.flickr.com/services/developer/

Gaunt, M. (2013, March 20). Introduction to fetch() | Web | Google Developers. Retrieved November 08, 2020, from https://developers.google.com/web/updates/2015/03/introduction-to-fetch

Get started with build configuration. (2020). Retrieved October 25, 2020, from https://docs.netlify.com/configure-builds/get-started/

Get Started with Firebase Authentication on Websites. (n.d.). Retrieved October 12, 2020, from https://firebase.google.com/docs/auth/web/start

GitHub Pages vs. Netlify: A Comparative Breakdown. (n.d.). Retrieved October 05, 2020, from https://www.netlify.com/github-pages-vs-netlify/

Google Maps Platform: Overview. (2020, August 28). Retrieved September 05, 2020, from https://developers.google.com/maps/documentation/javascript/overview

Johnson, T. (n.d.). Retrieve a gallery using the Flickr API. Retrieved September 05, 2020, from https://idratherbewriting.com/learnapidoc/docapis_flickr_example.html

Kakodker, R. (2019, September 28). 10 rules for user sign-up journeys. Retrieved October 12, 2020, from https://uxdesign.cc/10-rules-for-user-sign-up-journeys-7f9b19079a72

Manage Users in Firebase. (n.d.). Retrieved October 24, 2020, from https://firebase.google.com/docs/auth/web/manage-users#send_a_user_a_verification_email

MDN contributors. (2020, June 11). Progressive web apps (PWAs). Retrieved September 05, 2020, from https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

O'Donoghue, R. (2018, May 29). PWA Minimus: A minimal PWA checklist. Retrieved September 26, 2020, from https://mobiforge.com/design-development/pwa-minimus-a-minimal-pwa-checklist

Palchuk, M. (n.d.). Mapbox vs Google Maps: What maps API is best for your app? Retrieved October 11, 2020, from https://uptech.team/blog/mapbox-vs-google-maps-vs-openstreetmap

Perez, S. (2017, December 07). IPhone models dominate Flickr's list of most popular cameras in 2017 with 54% of top 100 devices. Retrieved September 07, 2020, from https://techcrunch.com/2017/12/07/iphone-models-dominate-flickrs-list-of-most-popular-cameras-in-2017-with-54-of-top-100-devices/

Pope, T. (2019, March 27). Transform a React App into a Progressive Web App (PWA). Retrieved October 25, 2020, from https://medium.com/@toricpope/transform-a-react-app-into-a-progressive-web-app-pwa-dea336bd96e6

Richard, S., & LePage, P. (2020, February 24). What makes a good Progressive Web App? Retrieved September 26, 2020, from https://web.dev/pwa-checklist/

Roberto, B. (2013, June 12). Get Flickr images depending on a search result. Retrieved October 24, 2020, from https://stackoverflow.com/questions/17071187/get-flickr-images-depending-on-a-search-result

Smith, C. (2020, July 11). 20 Interesting flickr Stats and Facts: By the Numbers. Retrieved September 07, 2020, from https://expandedramblings.com/index.php/flickr-stats/

Stepnov, E. (2020, September 06). Top JavaScript Maps API and Libraries. Retrieved October 11, 2020, from https://flatlogic.com/blog/top-javascript-maps-api-and-libraries/

Stevenson, D. (2018, October 25). What is Firebase? The complete story, abridged. Retrieved September 05, 2020, from https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0

Structure Your Database | Firebase Realtime Database. (n.d.). Retrieved October 24, 2020, from https://firebase.google.com/docs/database/web/structure-data

Temprano, V. (n.d.). Google Maps API or Leaflet: What's Best for your Project? Retrieved October 11, 2020, from https://www.codementor.io/@victorgerardtemprano/google-maps-api-or-leaflet--what-s-best-for-your-project-faaev60vm

VanToll, T. (2015, September 13). Handling Failed HTTP Responses With fetch(). Retrieved November 08, 2020, from https://www.tjvantoll.com/2015/09/13/fetch-and-errors/

Vu, L. (2020, September 01). How to Make Your PWA Work Offline - SimiCart. Retrieved September 26, 2020, from https://www.simicart.com/blog/pwa-offline/

Wik, A. (2018, July 15). Extract GPS Location And Other EXIF Data From Photos Using JavaScript. Retrieved October 11, 2020, from https://awik.io/extract-gps-location-exif-data-photos-using-javascript/

Yurieff, K. (2019, June 22). Some locations are Instagram gold. This app helps you find them. Retrieved July 28, 2020, from https://www.cnn.com/2019/06/22/tech/instagrammable-places-depalo-app/index.html