CO Mountain Mamas

Mamas Hike iOS Application

Client: Joy Opp

Team: Rachael Barnett, Carathryn Beutel, and Heather Paschall

06/16/2015

Introduction

Mountain Mamas Client

Colorado Mountain Mamas was founded by Joy Opp in 2003 when her daughter, Amanda, was about four months old. Joy is passionate about encouraging moms to get outside with their new babies and that passion is at the heart of Colorado Mountain Mamas. Joy's mission is to provide a safe and welcoming hiking experience for all levels of hikers no matter their fitness level or hiking experience.

Mountain Mamas is primarily run on a website. This website includes information about taking babies, toddlers, and school age children on hikes, a calendar for each of Mountain Mamas' different locations with links to upcoming hikes, and a detailed list of all hikes in all locations. Mountain Mamas also has accounts on both Facebook and Twitter that are meant to supplement the website.

Recently, Mountain Mamas has been struggling with hike attendance. Mountain Mamas moms do not check the website frequently, relying on Facebook or Twitter for hike information. If Joy forgets to promote an upcoming hike on Facebook and Twitter, hike attendance is notably reduced .

To bridge the gap between her website and social media, Joy envisioned an iOS app containing a library of Mountain Mamas' hikes that could connect and share with Facebook. Moms would have all trail information at their fingertips, enabling them to schedule their own hikes as well as attend Mountain Mamas' hikes. As a result, Joy could rely less on her website and use Facebook as the primary media for posting upcoming events, also easily maintaining all of the trail data that she has collected over the years.

Product Vision

The purpose of this project is to create an iOS application to provide moms that wish to go hiking with their babies, toddlers, and/or school aged children access to a database of hikes appropriately rated for their child's age group. Users will be able to search and filter the database by location, age group, and difficulty to easily find a list of hikes to meet their needs. Users can also save hikes to a favorites list for quick access, save hikes to a log to keep track of what hikes they have been on, view how many miles they have hiked, create a mileage goal to encourage continued hiking, and even add new hikes to the hike database.

This product will be useful for moms as it provides a solution to the difficulties of finding information about local hikes that are appropriate for children. Most similar apps on the market have hike difficulty ratings, but hikes are rated for the average fit adult, not for a mother carrying a newborn or trying to wrangle her toddler down the trail. With insufficient information about a hike's child-friendliness, the prospect of hiking with their children can seem daunting and impossible to mothers. This app will enable and encourage moms to go on hikes with their children while reasonably knowing what to expect.

Requirements

The goal of this project was to provide an iOS app to Colorado Mountain Mamas that allows a user to access and add to the database of all hikes, save hikes to the user account log and favorites, and connect with Facebook for sharing purposes. This project entailed specific functional and non-functional requirements outlined by the client:

Functional Requirements

1. Allow app users to login.
2. Allow app users to view hikes.
3. Allow app users to rate hikes using a 3 star rating system.
4. Allow app users to add a new hike.
5. Allow app users to comment on current hikes.
6. Allow app users to sort and filter hikes by name/location/difficulty/age group for easier accessibility.
7. Allow app users to share to their Facebook accounts.
8. Allow app users to save hikes to a list of favorite hikes.
9. Allow app users to save hikes to a list of completed hikes.
10. Display and keep track of users' total miles hiked using their completed hikes list.
11. Allow app users to create a mileage goal and view their progress.
12. Allow the client to easily manage the database of hikes, comments, and users.

Non-Functional Requirements

1. All code must be written in Objective-C for iOS 8.
2. The app must be highly interactive for the users.
3. The app must feel simple to use and be easy to navigate.
4. The app must adhere to Mountain Mamas' standard visual guidelines.
5. The app must be consistent in styling.

System Architecture

The Mamas Hike app system architecture is best described in two separate components: backend, which handles the database and social media connectivity, and frontend, which handles user interaction.

Backend Architecture

The general backend architecture is delineated in Figure 1, describing how Mamas Hike iOS app functions with other entities. Mamas Hike communicates with three other entities: Parse, the database, and Facebook. *Parse* is what is known as a Mobile Backend as a Service, or MBaaS. It handles storing of the databases, login functionality for users, API requests, and more. Parse manages the database entity and facilitates communication between the app and the database. Mamas Hike can post to Facebook, but cannot pull any information from Facebook.

Mamas Hike will request data from and post data to Parse over the internet, which will then query from and post to the Mamas Hike database. Mamas Hike will also be able to share to Facebook over the internet, creating an empty post for the user to edit and post to their Facebook account, or cancel the post.

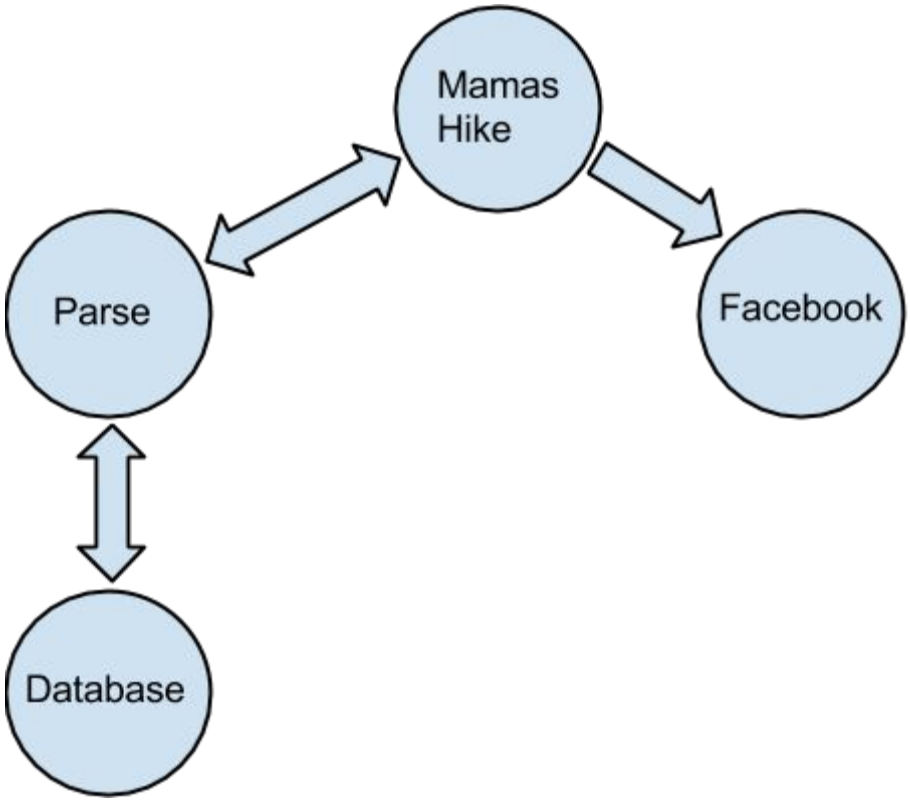

Figure 1. Basic Architecture Design

Frontend Architecture

The most basic frontend design is given by Figure 2, which shows the general user interface flow for Mamas Hike. The GUI consists of 12 functional pages: Login, Register, Home, Goals, Hike Search, Create Hike, Library, Hike Details, Comments, Favorites, Log, and About. The pages navigation is handled by a tab bar at the bottom of the screen.

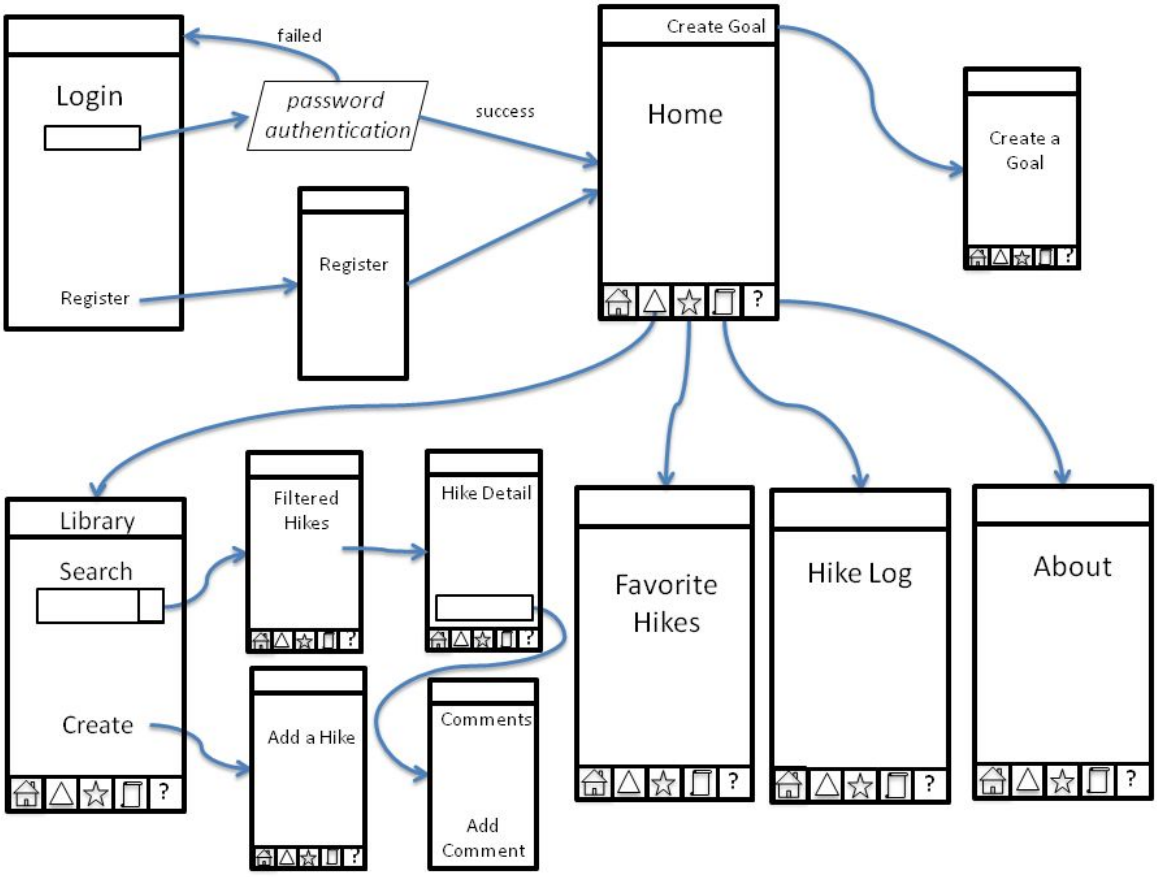

Figure 2. User Interface Flow Diagram

Login and Register

The first page is the login/register screen. The user will be able to login via credentials provided specifically for Mamas Hike. The user will also be able to register on this page in a similar manner.

Home and Goals

Once the user has logged in successfully, the Home page slides into view. The Home page displays the user's total miles hiked. If the user has created a mileage goal, Home displays the goal and a progress view bar displaying percentage of goal completion. If the user wishes to reset their total miles hiked or create a new goal, this can be accomplished by navigating to the Goal page from the Create Goal button. The Goal page easily allows a user to input a mileage

goal and save it to their profile, also displaying a blurb of text to explain the concept behind goal-making.

Navigation

The navigation between the primary pages is provided by the tab bar view controller which creates the row of tabs at the bottom of each screen. This allows the user to quickly switch between the pages of the application without needing to go back and forth in a navigational hierarchy of pages.

Hike Search, Create Hike

The second tab is the preface to the library of hikes. When the user taps this tab button, it brings up an initial Hike Search page. From here, the user can either navigate to the Create Hike page to add a hike to the Library, or search through the hike Library. To search, the user can select their general location (i.e. Denver, Salt Lake City, Fort Collins, etc.), and bring up all hikes in the database from that location, or the user can narrow the search further to filter by difficulty (casual, moderate, challenging) and age group (baby, toddler, school age).

Library, Hike Details

After pressing search, the Library page comes into view. The library page contains a list of all the hikes that matched the user's search criteria in a table view. The list displays limited hike information, including the trail name, difficulty, ideal age group, and length. To view further details, the user can tap the row containing a specific hike. This will pull up a Hike Details page that gives more information regarding the hike. From here, the user can add the hike to their Favorites, log the hike as completed, or share to Facebook. Located at the bottom of the screen is a Comments button that will take the user to the Comments page.

Comments

The Comments page displays a few extra hike details and all of the user comments on the hike. The user can add a comment at the bottom of the page. At any time, the user can return to the previous page(s) by tapping the Back button at the top of the screen.

Favorites

The third tab is the Favorites page, which lists the user's favorite hikes in a table view. This page enables the user to quickly find their favorite hikes. In its format, this display is very similar to the library page. The user can click any of the hikes in the list and the screen will switch to the corresponding hike detail page. The user can also click the "+" button, which will allow the user to add a favorite by first navigating to the search page, allowing the user to quickly search for a hike and add it to their favorites list. Once a hike is "favorited", the navigation jumps back to the Favorites page. On any page there is "Back" button that will navigate back a page. A hike is "favorited" by clicking the orange star on the Hike Details page. If a hike is currently on the user's Favorites, the star will be filled-in. A hike can be "unfavorited" in the same manner, and the star will be empty.

Log

The fourth tab is the Hikes Log page, containing the hikes that the user has marked as completed. This works in the same way as the Favorites page above. When the user clicks on a hike in the list, the screen will switch to the corresponding hike detail page. The user can also

add a new hike to the log with the “+” button, just as in the Favorites page. The one difference between Favorites and Log is that a user cannot delete a hike from the log, as it would not make sense to “un-complete” a hike. Once a hike is added to the log, the total miles hiked is increased by the length listed in the hike data. This will enable the user to quickly see what hikes they have completed in the past.

About

The fifth and last tab is the About page, which contains detailed explanations of how the rating system works for the different age groups. As mentioned before, the rating system is not the same across all age groups, and each group has their own system appropriate for their age. The toddler group, for example, may see a certain hike as challenging, while the school age group would view the same hike as casual. A toddler would have difficulty walking very far even if the terrain is flat; a school aged child however, can easily navigate flat terrain. The bottom of the page lists three links to external websites: a link to the Colorado Mountain Mamas site, a link to *Icons8*, the website that provided the icon images used in Mamas Hike, and a link to the app’s privacy policy (not currently an active link).

Design and Implementation Decisions

Objective-C

We chose to code the Mountain Mamas app in Objective-C over Swift since our client originally required Objective-C, and two of our three developers had experience working in it. Since Swift is a relatively new language, there is also more support and documentation on the internet for Objective-C than for Swift. As programmers with little experience in iOS application development, our team wanted the programming language with the most documentation, online resources, and printed resources as possible. Swift is also based in Objective-C, so Swift can be easily integrated if necessary in the future.

Parse - Mobile Backend as a Service

Initially, the team researched several options to serve up the database. MongoDB, CouchDB, SQLite, and MySQL were among the alternatives. Our team decided against MongoDB because it required us to either develop or find an Open Source API to allow communication between the app and the database residing in the cloud. SQLite alone was insufficient because it supports a local database only.

The best alternative we found was a design incorporating SQLite with MySQL. A combination of SQLite on the device with a local copy of the database that periodically synced with a MySQL master copy in the cloud could have worked well. Still, the SQLite/MySQL design was decided against because the team was spending too much time trying to work out how to work with the cloud and serve up the database without access to any cloud services due to expense.

CouchDB works almost identically to MongoDB, but has an API built into it already and was designed to reside in the cloud. Looking into CouchDB further revealed that the client would have needed to purchase the enterprise level of CouchDB, which was quite an expensive

option. At that point, we discovered Parse. Parse was the perfect fit because it was free to a point, had a lot of documentation, and could handle many other services which enabled the team to expand the initial design of the app.

Parse was chosen because it makes the development of an iOS app faster and allows the developer to focus more on making the app run well as opposed to attempting to serve up multiple databases from scratch and personally handle users' privacy when logging in.

Our team also wanted the app and database to be manageable for the client after the project is complete. Parse has an easy-to-use website that will allow her to add, edit, and delete any data in the database (aside from information in the user database). This will enable the client to manage the data in a straightforward manner without needing too many technical skills.

Additionally, Parse is a relatively inexpensive option; it is free to start out and as the app grows and requires scaling up, the user can choose from multiple pricing options that will meet the app's needs. For free, Parse provides the following levels of service: 30 requests/second, 20 GB of file storage, 20 GB of database storage, 2 TB of file transfer, unlimited analytics, among other features that Mamas Hike will not be utilizing. This is sufficient for many apps on the market due to their relatively low volume. If the app were to become the "next big thing," with a giant user base, Parse would not be the best option. If the number of requests and amount of storage required from the app grows exponentially, Parse can be quite costly. However, due to the somewhat small number of Mountain Mamas members, the usage of this app will most likely remain in the low volume category.

Login Capabilities

Users will create new login credentials in order to use the Mamas Hike app. Users will need to provide a username and password, the email address is optional, but if it is provided the user can reset their password. These login credentials are managed entirely by Parse. Requiring the users to have a login allows the app to store information such as total miles hiked, a goal, a list of favorite hikes, and a list of logged hikes. This information enables the app to assist the user in tracking their hiking and to have quick access to the details about their favorite hikes.

Originally, the development team was going to allow users to login with either new credentials or with their Facebook account. Due to Facebook SDK being recently updated and Parse SDK being out of date, Parse and Facebook would not cooperate with each other inside of Mamas Hike. The team decided that it would be better to only allow Parse credentials, as it is important that the users have their own Parse accounts in order to utilize Parse features, while still being able to share to Facebook without being natively logged in to Facebook (this is discussed more below).

Database Design

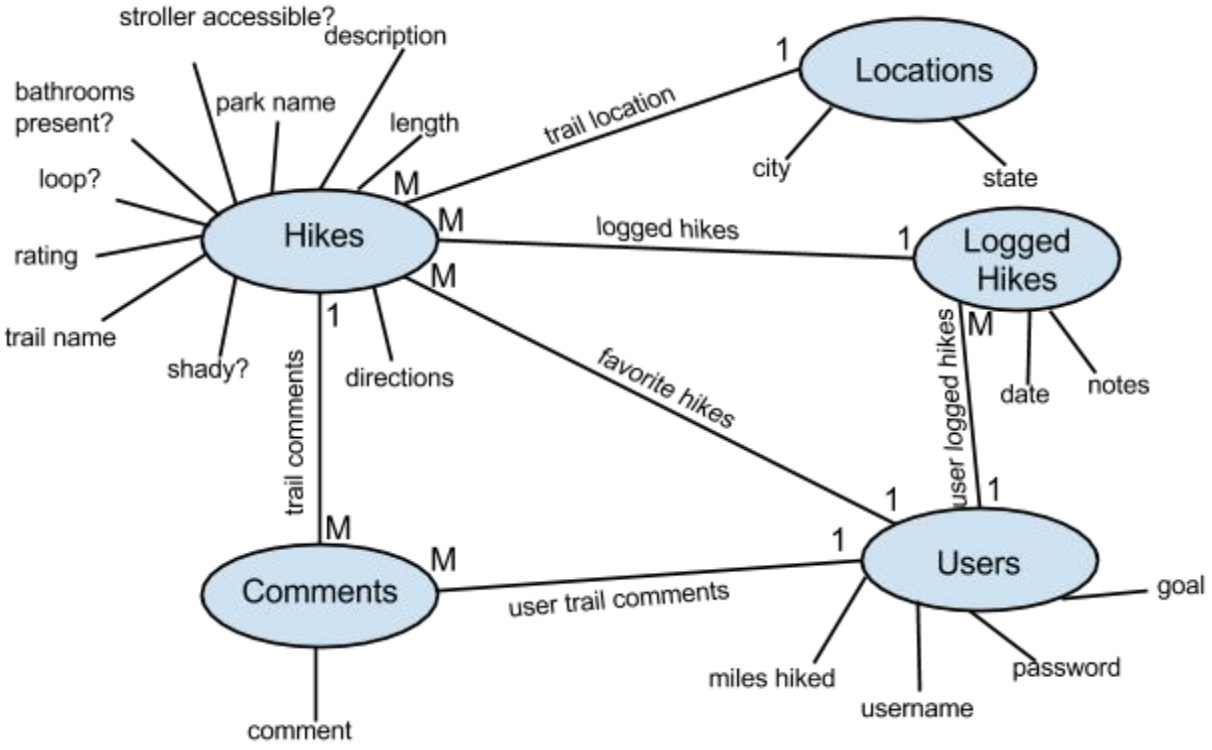

Figure 3. Database Schema

Figure 3 shows the database schema for Mamas Hike. The task of efficiently designing a database was not an easy one. The development team considered many different types of databases, flip-flopping back and forth between NoSQL and SQL databases, before deciding to go with Parse. The way Parse stores data is entirely up to the development team because the way the data is accessed is the same whether the development team decides to store document-styled data or more relational data, or any other storage method.

Mamas Hike's database is stored in a relational-like way, which is why the schema in Figure 3 is modeled in the way that it is. There are certain fields that make it less relational than it could be. For example, the user class contains an array of favorite hike IDs. In a truly relational schema, this should actually be a class all on it's own that stores the hike ID and the user ID. The reason the schema is not designed in this way is because it would have created an unnecessary amount of queries and/or an unnecessarily long and cumbersome queries. Our team went by the theory "de-normalize until it hurts;" this means that the developer should separate the data until it starts to hurt your project by creating too many queries or other similar overhead. In general, the data that is contained in a class is data that will be displayed on a single page. This means that only one query needs to be completed per page, which is ideal in the app environment.

Not shown are few bookkeeping fields for each database entry including *createdAt*, which stores the date and time of the object creation, *updatedAt*, which stores the date and time of the most

recent object update, and *ACL*, which is the Access Control List, storing the information regarding who can access/edit the object.

The database schema has been split up into five separate classes. The locations class stores the nearest large city so that app users can easily narrow down the list of hikes in the library by location. The hikes class stores all hikes and their hike data (name, park, description, etc.) including a location ID which points to the nearest large city. The users class stores authentication details, an array of hike IDs the user has favorited, an array of logged hike IDs, total miles hiked, and an optional goal. The logged hikes class stores the hike IDs, the user ID who created the log, any notes the user made about the log, and the date the log was created. The comments class contains the actual comment, a hike ID of the hike that the comment goes with, and the user ID who composed the comment.

Most fields are strings because it makes the field more flexible and not too restrictive. For example, city name, hike description, directions to the hike, and more are all strings. For filtering purposes, there are some boolean fields in the hike object, such as is it stroller accessible or is it a loop. Hike length is a number so that users cannot spell out a distance (i.e. one mile), which helps ensure the data is more uniform. Total mileage and goal are both numbers as well so that simple calculations can be performed on the data without casting.

Hard Coding vs. Database Querying

Certain app fields we elected to hard code, such as the rating system descriptions, while the list of locations (Denver, Colorado Springs, etc.), hike details, and comments are pulled from the database. These hard coded fields are not expected to change; however, the fields that are pulled from the database are expected to change by being added to or deleted from. Changing fields are often lists to which any user is able to add and the client is able to edit or delete.

The downside of harding coding fields is that if the client ever wishes to edit, she will have to do so in code and then update the app in the App Store. If the client decides in the future that she wants to change these fields frequently, then it would be easy to add a new database to Parse and would simply pull from the new database. Still, these fields should not need editing very often, if ever, so our team decided that it would be better to hard code than to add unnecessary API requests to Parse for data that will stay the same.

Sharing to Facebook

Even though the user will not be logged into Facebook natively in Mamas Hike, they will have the ability to share to Facebook. When the user taps the Facebook Share button, the screen will switch to either the user's Facebook app or open Facebook in the browser with a post already open sharing a link to the Colorado Mountain Mamas website. The user can then edit this post to their liking and tap Post which will then switch the screen back to Mamas Hike.

Facebook Event Use

Originally, the Mamas Hike app design included event handling, such as creating a new hike event, adding hike events to the calendar, inviting friends from Facebook to the hike event, and syncing with Facebook events. Due to limitations in syncing with Facebook, our app would

have to handle events independently from Facebook and users would manually sync them. Since our client wanted to integrate our app with Facebook while still using Facebook as a social platform, it made more sense to let Facebook continue to handle any event functionality, but allow users to post to Facebook from the app.

Size Classes

In order for an app to be accepted into the App Store, one of the criteria the app must meet is that it is capable of running on any iOS device. This includes iPhone 4S, iPad 2, iPhone 6 Plus, and more. These devices are all very differently shaped and sized, which creates difficulty when trying to design a layout that will look good on all devices. The method provided by Xcode to handle multiple screen sizes is called *size classes*. Each screen size fits into at least one of several size classes, but most importantly, each screen fits into the class *Any by Any*. By setting constraints for each object in a view in Any by Any, our team was able to design a single view that would format correctly for any screen size. Due to size classes being a relatively new feature of Xcode, there is little documentation and tutorial help, requiring much trial-and-error to understand size class implementation. Currently, Mamas Hike is only tested for formatting with iPhones (not iPads) in the portrait orientation.

Results

This project has been thoroughly tested and is working really well using the iOS simulator on Macs. Aside from the occasional slow network connections with Parse, the team can find no issues with the app in its current version. The team has found that creating an app from scratch and fitting all of the various moving pieces together can be quite challenging and spent a significant amount of time researching and creating test apps to help get Mamas Hike in working order. The app will most likely work just as well on actual iOS devices, but may require some optimizations or more formatting to run quickly and smoothly and look perfect across all the various device sizes. The team has found difficulty in considering this project finished because, particularly in app design, developers can always do more or add one more feature to make the app just a little bit better. This is where the functional requirements came in handy because once our team met the requirements, we needed to keep reminding ourselves that it would be ok if that feature was not there.

All the functional requirements have been met and the team feels that we have met all the non-functional requirements as well, though the non-functional requirements are a matter of personal opinion. Therefore, it would be extremely helpful to test the app with some sample users to get a better idea of how easy-to-use and interactive the app actually is. The project could not be submitted to the App Store for review due to time constraints, though we have provided the client with a compressed copy of the project on a USB flash drive and created a GitHub account for the client with the project in her own repository. The client has also been provided with a Parse account containing the database and instructions on how to use Parse to track app usage and manage the database. The client will be able to either submit the app for review or give the project to another team for testing and/or further development.

Future Work

Test App on Devices

Due to time constraints, it was not reasonable for our team to request the client to purchase developer licenses for our entire team. For initial development, using the iOS simulator on the computer is sufficient, but before submitting the app for review it should really be tested on real devices. This enables developers to test how the app behaves in a “real environment”, for example if the app causes the battery to drain rapidly or produce excessive heat. Our team was able to test most features on the simulator, but to ensure a quality product, the app should also be tested on a physical device.

Submit to App Store

Without developer licenses and the time necessary to test the product on real devices, the team cannot submit the app to the App Store at this time. The submission application is rather extensive and will take a significant amount of time by itself. Some of the tasks needed for submission include the drafting of an app description, creation of app icons and screenshots, category and genre selections, assignment appropriate ratings, and assignment of appropriate keywords for the app.

There are also requirements for the submission that the developers cannot meet; for example, a Privacy Policy needs to be drafted and posted on a website. All URLs in the application must be in working order and link to valid documents for the App Store to accept the app.

The App Store has extremely strict guidelines that developers must follow in order for an app to be accepted. A list containing the guidelines that pertain to this particular project are located in Appendix A.

Geolocation Features

One of the difficulties in going on hikes is finding the actual trailhead to the hike. When searching on a map, trailheads are usually not marked. One of the ways in which this app can be improved is to add a geolocation feature to the hikes database. This geolocation feature would be a pin at the trailhead. An app user could tap the geolocation link to pull up the location on the device’s maps app. The maps app would then give exact directions to the trailhead.

The team decided not to pursue this functionality because it would have cost a lot of time to compile the data required to initialize the database. There are many websites that have geolocation pins of trailheads, but they would need to be compiled and added to Colorado Mountain Mamas’ current data set. This could be a very time consuming task and the data may still be incomplete after the task has been completed.

Another concern about geolocation functionality is that when users create a new hike they may not be at the trailhead. The easiest way to obtain a geolocation pin is by determining the current location of the device; if the user was not at the trailhead though, this would not be

accurate data. Therefore the development team would need to somehow take this scenario into account.

Expand the Login Capabilities

The development team had to abandon the option to login via Facebook because Parse SDK has not been updated recently enough to account for Facebook SDK's recent updates. Once Parse SDK has been updated, it should be a rather simple task to add a Facebook login to Mamas Hike.

The option to login via Facebook would be an invaluable feature for Mamas Hike to possess. Many users prefer not to create new login credentials, as having additional login credentials can be inconvenient. The option to login via Facebook would streamline the registering and logging in process, which would encourage new users to continue using the app, thereby increasing Mamas Hike's user retention rates.

Expand Facebook Functionality

Once Mamas Hike is on the app store, it would be useful to add a link to Mamas Hike in the app store or open the app if the user has it installed on their device instead of linking to Mountain Mamas' website.

It would also be an added benefit to make a custom open graph story to share hike details such as trail name, park name, hike length, and the ideal ages. This expansion of the sharing feature would make it simpler for users to invite their friends to hike with them.

More Facebook functionality could be added such as liking the Mountain Mamas Facebook page, inviting friends to use the app, and sharing hiking achievements such as reaching a goal.

Appendices

Appendix A. Relevant App Store Review Guidelines

This list of the App Store Review Guidelines that directly pertains to the Mountain Mamas iOS app project. It contains a numerous list of reasons why the app could be rejected from the App Store in the review process. The development team has worked to minimize these risks, but the relevant risks have been provided so that it is understood how strict the App Store can be about accepting apps into the market. This list has been compiled from https://developer.apple.com/app-store/review/guidelines/, to view the full list please see the link provided.

2. Functionality

- 2.1 Apps that crash will be rejected
- 2.2 Apps that exhibit bugs will be rejected
- 2.10 iPhone Apps must also run on iPad without modification, at iPhone resolution, and at 2X iPhone 3GS resolution

2.15 Apps larger than 100MB in size will not download over cellular networks (this is automatically prohibited by the App Store)

2.23 Apps must follow the iOS Data Storage Guidelines or they will be rejected

3. Metadata (name, descriptions, ratings, rankings, etc.)

3.3 Apps with names, descriptions, screenshots, or previews not relevant to the content and functionality of the App with rejected

3.4 App names in iTunes Connect and as displayed on a device should be similar, so as not to cause confusion

3.5 Small and large App icons should be similar, so as not to cause confusion

3.6 Apps with App icons, screenshots, and previews that do not adhere to the 4+ age rating will be rejected

3.7 Apps with Category and Genre selections that are not appropriate for the App content will be rejected

3.8 Developers are responsible for assigning appropriate ratings to their Apps. Inappropriate ratings may be changed/deleted by Apple

3.9 Developers are responsible for assigning appropriate keywords for their Apps. Inappropriate keywords may be changed/deleted by Apple

3.12 Apps should have all included URLs fully functional when you submit it for review, such as support and privacy policy URLs

10. User Interface

10.1 Apps must comply with all terms and conditions explained in the Apple iOS Human Interface Guidelines

10.3 Apps that do not use system provided items, such as buttons and icons, correctly and as described in the Apple iOS Human Interface Guidelines may be rejected

10.6 Apple and our customers place a high value on simple, refined, creative, well thought through interfaces. They take more work but are worth it. Apple sets a high bar. If your user interface is complex or less than very good, it may be rejected

13. Damage or Injury

13.2 Apps that rapidly drain the device's battery or generate excessive heat will be rejected

14. Personal Attacks

14.3 Apps that display user generated content must include a method for filtering objectionable material, a mechanism for users to flag offensive content, and the ability to block abusive users from the service

17. Privacy

17.1 Apps cannot transmit data about a user without obtaining the user's prior permission and providing the user with access to information about how and where the data will be used

17.2 Apps that require users to share personal information, such as email address and date of birth, in order to function will be rejected

17.4 Apps that collect, transmit, or have the capability to share personal information (e.g. name, address, email, location, photos, videos, drawings, the ability to chat, other personal data, or persistent identifiers used in combination with any of the above) from a minor must comply with applicable children's privacy statutes, and must include a privacy policy

17.5 Apps that include account registration or access a user's existing account must include a privacy policy or they will be rejected

Appendix B. Useful Links

- Mamas Hike Project
  - Project repository on the client's GitHub account
  - https://github.com/mamasHike/MamasHike
- Parse iOS Developers Guide
  - Documentation detailing how to use Parse in Objective-C and Swift
  - https://parse.com/docs/ios/guide
- Submitting Your App to the Store
  - Step-by-step instructions on submitting your app to the App Store
  - https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/SubmittingYourApp/SubmittingYourApp.html
- Facebook Developers
  - Home page to all links to developing an application with Facebook functionality
  - https://developers.facebook.com