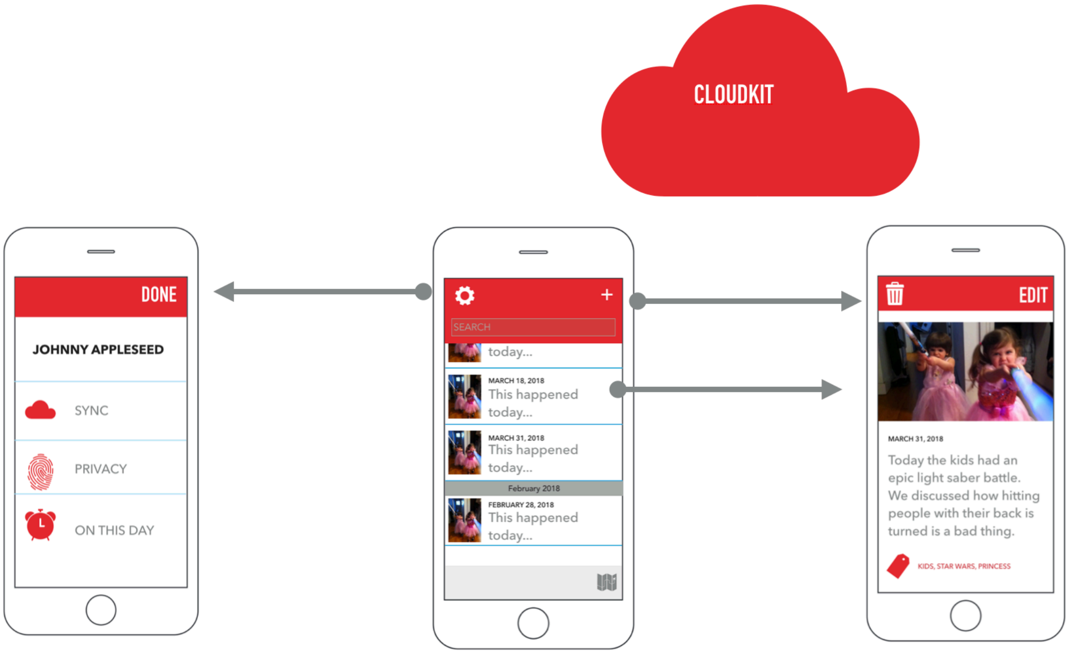
Today Was a Good Day

Journaling has been show to be beneficial for many aspects of personal health. However, the act of journaling can end up being cumbersome and a mental burden on an individual. So despite the best of intentions, maintaining a consistent journalling habit is difficult and is often stopped not long after started.

In this assignment, you will create an iPhone journaling application that will finally allow users to fulfill all of their journaling goals.



Today the kids had an epic light saber battle. We discussed how hitting people with their back is turned is a bad thing.

Figure 1. An overview of the main screens in the application.

Application Overview

This application will utilize photos as the foundation for all journal entries. This will allow metadata to be pulled from the photos and reduce the amount of information that a user needs to input for an entry. The application will also allow the users to enter some text to associate with each photo.

Will will make a concerted effort to make the app as simple as possible and only focus on core features allowing users to create, edit and delete a journal entry and to search their entries for keywords or locations (Figure 1). The application will use CloudKit to keep data synched between devices.

Home Screen

The home screes should use a table or collection view controller to display the journal entries.

- Cells should contain a thumbnail image of the entry photo, a text blurb (the entire entry does not need to be shown) and the date. You may add any additional information as you need.
- The table or collection view should be sectioned by month.

Search Bar

A search bar should be integrated into the navigation controller to allow users to filter their entries by entry text or by tags. There are many different approaches to implementing a search bar. You may consider using a UISearchBarController or a simple UISearchBar.

Bar Buttons

There should be bar buttons on the home screen that direct the user to a new view controller for Settings, creating an Entry or a Map View screen. You may arrange the buttons however you like, including putting some in a toolbar (as shown in Figure 2).

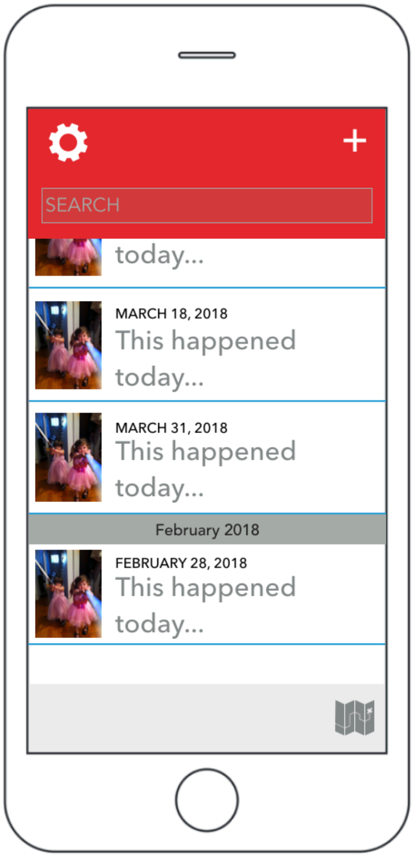


Figure 2. The home screen.

Map Screen

The map screen will allow users to identify all of their journal entries on a map. One difficulty with displaying points on a journal map will be that many of the entries will be in the same location. This would cause the map to be very dense and make it difficult to understand what will be shown when you tap on a group of points. We can use the new *clustering* feature in iOS11 to address this problem. For a good example of adding annotations to a map view please review Apple's latest WWDC video on MapKit (https://developer.apple.com/videos/play/wwdc2017/237/). They provide a sample utilizing new features, such as clustering, in this sample code project: https://developer.apple.com/sample-code/wwdc/2017/MapKit-Sample.zip.

The side effect of using this clustering technique will be that we will not be able to tap on a specific point to view the entry. Instead, we will use the location of the point as a query to the users data and show all entries close to the point. For example, if there was an annotation cluster on a zoomed out map of Illinois, then tapping on the cluster near Chicago would show all entries near Chicago. The exact distance filtering you use is up to you to define base on what you think is reasonable.

I would recommend presenting the Map view controller as a popover. When the user select a point on the map, dismiss the popover and pass the location back to the presenting controller, the Home screen.

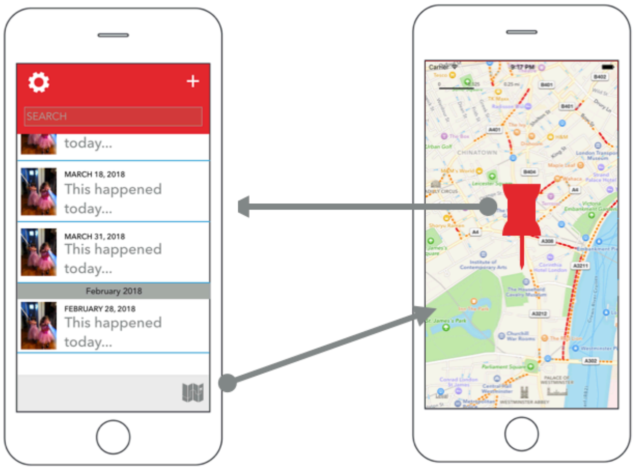


Figure 3. The Map screen.

Settings Screen

Create an additional view controller that allows the users to access some app specific settings. The screen should show the users name from the iCloud user account. In addition, there should be the following buttons (Figure 4):

- Sync - This button should delete the locally stored data and trigger a download of all user data from iCloud. The exact strategy is up to you, but you should consider NOT downloading images until you need them.

- Privacy (Bonus Point) - Implement an additional layer of security to prevent prying eyes from viewing the journal on an unlocked device. You can use Local Authentication framework to implement touch Id or write your own. The application should require a password when opening from a suspended or terminated state. *Think about how you implemented a splash screen for the same effect.*

- On This Day (Bonus Point) - Figure out a way to send on an On This Day notification that fires daily and directs the user to entries taken on the current date.

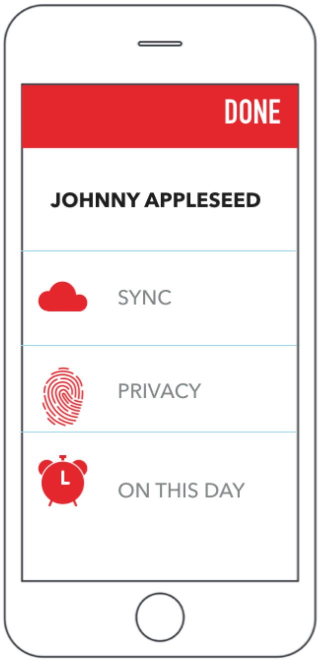


Figure 4. The settings screen.

Journal Entry Screen

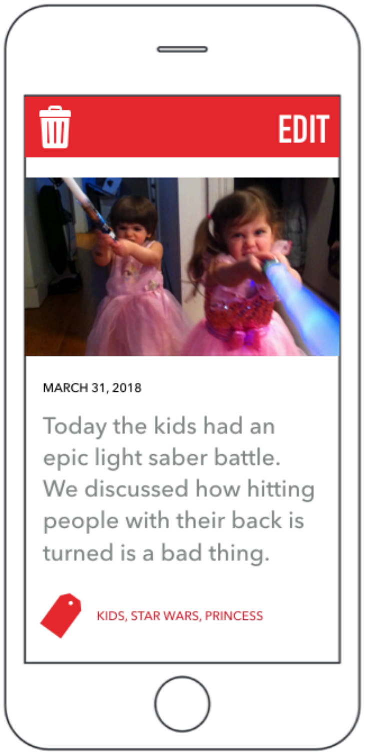
The user should create an entry that requires a photo. The photo should provide important metadata that should be stored with the entry, as follows:

- Location
- Date created

The date created should be used to query the Dark Sky weather API (https://darksky.net/dev) to get the weather for that day. The weather should be displayed in the user interface.

Use the Vision framework to automatically create tags based on the objects detected in the photos. These tags should be displayed as part of the user interface.

The entry screen should allow for editing of photo, date, text and tags. During editing the users should be allowed to remove tags that were automatically assigned or add their own.



MARCH 31, 2018

Today the kids had an epic light saber battle. We discussed how hitting people with their back is turned is a bad thing.

KIDS, STAR WARS, PRINCESS

Figure 5. The journal entry screen.

Data and Synching

This app should follow an offline first approach and ensure that the local data persistence is always the primary goal, with synching with iCloud done opportunistically through CloudKit. Use a custom class or struct that conform to the coddle protocol to store the journal entries as JSON to disk. Images should be cached locally.

Use CloudKit subscriptions to manage synching between devices. Create subscriptions that inform changes to the database and then handle them all all devices that receive the notification.

Extensions

Add a photo share extension and a today widget extension to the application. Users should be allowed to add an entry from the photos app. You can use the standard UI or create a custom UI for creating the entry. Your implementation may require the use of app groups and/or embedded frameworks to make the best practices in code reuse. Create a today widget to show the 5 latest entries made to the journal. You can include whatever information you want, but you should at least include the photo

Grading

The application should compile with no errors or warnings and perform all described behaviors. If the app does not compile you will receive a 0. Feel free to explore additional enhancements to improve the appearance or functionality of the application.

Due Date

Assignment 2 is due May 5, 2020 at 5:29pm.