Emily Valenti

Professor Nicolas D’Agata

WEB 289

3 March 2021

Feasibility Assessment Document: Subscription Dock

Subscription Dock, an app to contain all of your subscription details in one place is set to be simplistic and lightweight yet powerful. The user will not be able to make any changes to the subscription from Subscription Dock, but they will be able to put the URL to the subscription’s website in a notes space in each subscription. The user will have the option to turn on notifications and get a notification a customizable time in advance of the next bill date.

Technical Feasibility:

Subscription dock will be built using Swift in XCode. This will require a comfortable knowledge of the language. Most of the interfaces will each be a separate view controller. The view controllers will be connected by Segues and have the functionality of switching back and forth between the different pages. Some of the functions will appear as a popup window or something similar, not needing its own separate view controller.

UIs:

Home interface: This interface will have several buttons, UILabels and imageViews (embedded in their own vertical stack) inside of a ScrollView. This will enable the user to see all of the “subscriptions”. When the user taps any of the buttons/icons this will trigger a specific event to go to the corresponding button. For example, when the user taps the settings icon a Segue will be performed and take the user to the settings page. When the user presses the X(done) button the segue will unwind back to the home interface. The Home interface will contain a + (add new subscription) button which when tapped will Segue to the new subscription interface. Each new subscription will be displayed as a button, imageview and UILabels instide of a vertical stack view. The home interface will contain a UIButton which will have a UILabel embedded in it to indicate the averag spending for the user. The user can tap this button to switch between average spent per week/month/year. The Home interface will have a filter button which will open up a menu that has filter options for the user to select. The home interface also has two buttons at the top, “All” and Categories”, this is another quick filtration system. As indicated the user will be able to see all of their subscriptions under the “all” tab and see the subscriptions by category under the “categories” tab. The categories will be displayed similar to the individual subscriptions (an image view and UILabels) except it will total each category and display that dollar amount.

New subscription interface: It will contain two buttons at the top of the page “Back” and “Done”. When the user taps the “Back” button, it will unwind the segue back to the Home interface without saving any of the info entered. When the user taps the “Done” button it will save the information added by the user and will return to the home interface displaying the new subscription at the top of the list. There will be an image view that will be a blank by default, the user can add an icon for this subscription in a field below and this will update the image view at the top of the interface. Underneath the icon will be a UILabel that will display the name of the subscription once the user has entered it. The placeholder text will be “Subscription Name”. There will be several UITextFields embedded in a vertical stack view for the user to enter in the subscription information. The first UITextField will be for the name of the subscription, this will reflect on the label at the top of the page once the user has entered the name. The second UITextField will be for the cost of the subsctiption. The next UI Element might be a label and button, the button when tapped will access the user’s camera roll (assuming that they have given the necessary permissions) and let the user select a custom image for the icon. If no photo is chosen, I may have a default solid color “image” for the icon. The next UITextField/UIButton will be for which cycle the subscription follows (free trial, weekly, monthly, yearly etc.). When the user taps the button, a menu will pop up allowing the user to select their cycle. The next UITextField/UIButton will be for the first bill date. When the user taps the calendar icon a UIDatePicker will appear. The user will be able to set the bill date for the subscription, this will be vital if the user chooses to receive notifications. The next UITextField/UIButton will be for the alert preference. The user will be able to select from the following: “none”, “day of 8AM”, one day before 8AM”, “2 days before 8AM”, and one week before 8AM”. The final UITextField will be for the user to enter in notes like the URL to the subscription’s website.

There are always technical problems that arise in projects. Whether it’s a bug or something you don’t quite know how to program. In the creation of Subscription Dock there are several elements and functions that I haven’t work much with in previous projects. One of them being a DatePicker element. I am still learning how to program with this element as well as ensure that the app will store the date picked correctly. I am continuing to research and learn about the best way to implement these elements and functions Another challenge will be ensuring that the subscription data gets stored correctly and not get scrambled or forgotten!

Access to the user’s camera roll is a feature that will be required if the user wants to add an icon to correspond with their subscription. The app will be clear about this access being required in the app store. The user will explicitly be warned and given the chance to decline when they try to input a new icon photo for the first time.

I think the best way to approach the technical challenges that I can clearly see right now in the planning phase is to fully flesh out the details of these challenges and take it one step at a time. I may have to strike some of the features from the first release if I realize that there is some hidden baggage that comes with a feature. My main goal currently is to plan and design an app that focuses more on the stable functionality of the app rather than all of the bells and whistles that don’t work as well. For example, I am not sure if I am going to be using any APIs at this point in my app. I may integrate Google Sign-In and Sign in with Apple in a future update. This will require the use of Apple REST API as well as A Google API.

Updates and bug fixes are definitely included in the future of Subscription Dock post deployment. Who has ever gotten their first app totally perfect on their first release? Any bug fixes, performance enhancements, and additional features that are released will be accessible to the users in the app store, all the users will have to do is update on their device. I will also have some of my friends and family test out the app and give me feedback. This will give me more eyes to spot the glitches or bugs that may occur.

Operation Assessment:

While this app will not contain any credit card or payment the user may want to protect the data that they have entered. I am still working on what kind of protections I will offer in this app. Even a passcode that the user will put in when they enter the app will offer a security to the user and their data.

The data being stored is the list of the user’s subscriptions and their corresponding data. This will be stored locally at launch. One of the future updates may include linking the account to a cloud-based service so the user can access the list of all of their current subscriptions over many different devices. Another future update may include Google Sign-In as well as Sign in with Apple.

The app will be sure to inform the user of all of the data that is going to be use and what it is intended to be used for. Keeping the integrity of the app will be a top priority for me. I don’t want any user to feel uncomfortable using my app and will do my due diligence to communicate this to the user.

I will be sure to keep my app in the guidelines of the App store including appropriate content, data policies, displaying in-app experience screenshots in the App Store, keeping the app updated by removing any depreciated features frameworks or technologies etc. (Apple)

Schedule Feasibility:

I would like to have Subscription Dock totally done and ready to submit to the App Store at least 10 days before it is due, May 6, 2021. The following is how I have broken my timeline out:

Design phase – 2/18/21 – 3/11/21:

* I want to continue to work on all of the UI details as well as solidify a plan for all of the functional details.
* Decide if I am going to use an API for user authentication.

Development: 3/12/21 - 4/14/21:

First UI developments and some backend development:

* Creating the basics framework for my interfaces
* Coding the UI interfaces to some of the basic functions, i.e., setting up the Segues, buttons etc. and ensuring that the main navigation works well

Completing the UI developments:

* Ensure that all of the visual elements are present
* Ensure that all of the elements are in the correct layout
* Ensure that all of the visual elements are fluid and can transfer smoothy to different screen sizes.

Complete the backend coding:

* Ensure that all of the features and functions are working correctly
* Ensure that the user’s data is being protected
* Ensure that all of the “new subscriptions” are being stored correctly and displayed correctly

Hoping to have 4/7/21 - 4/14/21 as an extra week for any unexpected things that may crop up

Testing and refinement 4/15/21:

* Send app out for review to friends, family, fellow classmates.
* Do complete testing of the app to ensure is fully functional and meeting all of the requirements. Ensure that it is bug free and ready for the approval process. Adjust and refine the necessary details/features.

App Approval 4/26/21

* Send the app and other necessary documents in for the approval process
* If necessary, adjust features to meet the App Store approval

Full App Release! 5/5/21

* Launch the app to the App store! Yay!
* Keep on working on updates and more features

|  |  |  |  |
| --- | --- | --- | --- |
| Feasibility Matrix | | | |
|  | Weight | Website | Mobile App |
| Technical Feasibility | 50% | **80** | **100** |
| Does not solve the problem of a convenient central location for a list of your current subscriptions very well. Mobile apps are more popular for users rather than typing in the website URL especially for this category of app. | A mobile app will be able to contain all of the data for each subscription easily and in a convenient format. |
| Operational Feasibility | 25% | **40** | **100** |
| Does not support scheduled notifications to the user. It is harder to send out scheduled notifications based on the subscription bill date through a website. | A mobile will be able to send a notification to the user’s mobile device easily. The user will be more likely to actually see the notification if it comes across their mobile device screen even when they are not actually in the app. |
| Schedule Feasibility | 25% | **85** | **70** |
| 8 Weeks | 11 Weeks |
| Total | 100% | **68%** | **90%** |

Works Cited

Inc., Apple. “App Store Review Guidelines.” *App Store Review Guidelines - Apple Developer*, developer.apple.com/app-store/review/guidelines/.