CS 4720 - F18 - Final Project Proposal

Device Name: iPhone 7 Platform: iOS

Name: Hannah Thompson (solo) Computing ID: hmt5wa

App Name: UVA Accessibility

Project Description:

This App, UVA Accessibility, will serve as a platform for UVA students with disabilities. While UVA does a decent job of providing resources to SDAC students, they are very spread out and can feel unaccessible. I am proposing an app that will serve as a single point of information and aid to SDAC students. Its simplest component will be that it will serve as a consolidated point of connection to UVA’s existing accessibility services. Secondly, it will include GPS services that will map accessible routes for students to be able to get around Grounds uninhibited. Finally, it will have a form to report access barriers directly to UVA Facilities Management.

What I propose to do is create an app that will do the following:

- The system shall have a screen that contains links to all of UVA’s accessibility services, pulled down from a web-service;
- The system shall use GPS services to show users accessible routes to locations on Grounds; and,
- The system shall allow a student to fill out a form about a barrier to entry, optionally including a photo, that will then be sent as an email to UVA Facilities Management.

I plan to incorporate the following features:

- Camera - A student can take a picture of a barrier to access.
- Build and consume your own web service using a third-party platform (i.e Firebase) - Will build and pull down information on UVA’s accessibility services.
- Data storage using key/value pair storage - I will store the user’s information from the access barrier report, that is most likely to be repeated.
- Open shared activity / features - I will email the student’s barrier to access report to UVA Facilities Management.
- GPS - Will use GPS to show location along accessible pathways.

Wireframe Description:

The wireframe shows the basic envisioned layout for the app. It has a simple launch screen with the UVA Accessibility logo, and opens up to the Resources tab. The app as a whole has three tabs, as they are all “sibling” functions. The first tab is “Resources” and is denoted with the album icon and a clear label. A tableview that modally transitions to another tableview to go to the second level of resource options was considered the best way to display this information. Second, the “GPS” tab is denoted with the map icon and a clear label. In order to display the legend for the map, it was placed inside a navigation bar. It was hard to make this decision since it doesn’t quite align with design principles, but it was the best way to display the information without taking up too much screen real estate. This screen also does not immediately show the user’s location due to the fact that it is only a helpful if on Grounds. Thus, it shows the map of pathways on Grounds, and then has a button where the user can enable their location to be shown. Third, the “Report” tab is denoted with the attention icon and a clear label. This screen has a couple tab choosers as these were considered a more efficient use of space than a scrolling picker. It also has simple edit text fields to enter information, as well as intuitive buttons to choose an image to upload.

Platform Justification:

I found that iOS was the most sensical platform for this application due to my knowledge of UVA’s student population. While there are of course students who have Android devices, the majority of students are on iOS, so I figured this was logically a good place to start. However, if the app were to actually be released by UVA, I would suggest implementing on both platforms, as accessibility services should clearly be fully accessible.

Major Features/Screens:

- Resources: The resources screen holds a list that contains links to all of UVA’s accessibility resources, and upon being clicked will open the link in the user’s browser.
- GPS: The GPS screen consists of a Google Map with the UVA accessible pathways maps overlaid on top of it. This allows users to see where they are on this map by accessing their location, and thus can help guide them along accessible pathways and to accessible buildings.
- Report: The report screen is a form the user can fill out to report any barrier of access they may have encountered while on Grounds. It requires the user give a description of the issue, but all other fields are optional. When submitted this form will be sent to UVA facilities management in the form of an email (currently using my email as a default for testing purposes).

Optional Features:

- Camera -15 pts- In the report page, a user can upload a photo from their library or take one from the camera, in order to show their barrier to access. This can not be tested in the simulator, but can be tested on any device by hitting either the camera or album button and choosing your image. The image will then show up in an imageview to provide confirmation to the user.
- Build and consume your own web service using a third-party platform (i.e Firebase) -15 pts- The resources page pulls down all of the accessibility services data to fill its listview from a Firebase web service I built. This can be tested by clicking on each item to make sure that they all properly link to the expected resources.
- Data storage using key/value pair storage -10 pts- The report page stores the user's email and UVA affiliation as a key/value pair after it has been given once (and updates if a new one is input). I did this because, since most mobile devices only have one user, I wanted to minimize the amount of user input I ask for each time. This can be tested by submitting a report and noting that every subsequent time after, these fields are auto-filled.
- Open shared activity / features -5 pts- The student's barrier to access report is emailed to UVA Facilities Management. However, currently for testing purposes it is set to send it to my personal email.
- GPS -15 pts- GPS is used in the GPS tab to show a user's location along accessible pathways at UVA. This can be demoed by being on Grounds and seeing one's location on these accessibility maps.

Testing Methodologies:

I tested the app by continuously running it on my own device as I was updating the code. Each time I would test edge cases by making sure I had covered things such as if a user had left any fields blank in their forms, or what screen would be shown after sending the email or linking to the resources. To test the GPS services, I walked around Grounds and ensured that the mapped accessible pathways matched up to what I was seeing at my current location.

Usage:

The app is best run on an actual device, as a simulator will not properly do camera, GPS, or email services. Additionally, the device must have the native email service activated in order to send the barrier to access reports over email.

Lessons Learned:

In the process of building this application, I learned three major lessons: a simple and clean UI can be the hardest to implement, requesting permissions is paramount, and there are definite costs when you stray from a fully native implementation.

I will never bat an eye at a well-centered, clean form in an iOS app again. The UI that I imagined would be the easiest to implement, the barrier to access form, proved to be the most difficult. Finding button icons to try to match the native iconography was difficult, though the implementation ended up appearing so natural in the form that it seems it would have been a breeze. Trying to determine the best way to have a user choose from four options turned into an hour of research over the pros and cons of a tab chooser versus a scrolling picker. While the implementation of this UI proved difficult, it has given me a newfound appreciation for even the simplest of displays in iOS.

This process additionally helped me to truly appreciate the Info.plist. There is no greater loss than to spend an absurd amount of time wondering why your app keeps crashing only to discover that you forgot to set up user permissions to use the camera. Clearly permissions are incredibly important for getting an app to successfully run, but it was also important to remind myself that I should be thinking about the user and their privacy more while developing.

Finally, I got a lesson in why to use the native implementation of services. I opted to use a Google Map in my app as opposed to an Apple Map mostly due to my preference for it's design and features. For the most part this proved not to be too much of an issue. However, when it came to wanting to implement more robust functionalities, I found my options were fairly limited on how much I could do with a Google Map. Had I implemented fully in iOS, I may have been able to have greater ability to manipulate the maps to meet my goals.

In all, developing this app was an incredible growing experience that strengthened by understanding of design and implementation within iOS.