**Burn It**

Icon

Description automatically generated with medium confidence

Project Documentation

Jessica Bender

CSC 391/491 - Mobile Application Development (iOS) 2

DePaul University - Spring 2021

**Table of Contents**

[**0. Overview 3**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.d4vzku4itg9j)

[A. Team Members & Roles 3](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.z05an9kbfovn)

[B. Source Code - GitHub 3](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.rt1zuv2jqoto)

[**1. Description of all the features that have been completed 3**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.41mbwpefxi2q)

[**2. Screenshots of every screen with explanations 4**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.dt9wd37w9p1i)

[**3. Final Project Discussion - Done individually (Jess) 7**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.mxv8ajr40f12)

[**3. Final Project Discussion - Done individually (Jackson) 9**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.ajuc92snufge)

[**3. Final Project Discussion - Done individually (Jennifer) 11**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.w1jlztub64ex)

[**4. Team Contributions 12**](https://docs.google.com/document/d/1fH0jchuRpLaZxNqtXGGD3uHjB-QWR1DK5U3AoP0vvjM/edit#heading=h.kxoh66r93ey)

**0. Overview**

A. Team Members & Roles

1. Jess Bender - Front End Developer
2. Jackson Hurst - Back End Developer
3. Jennifer Nguyen -  Full-Stack Developer

B. Source Code - GitHub

<https://github.com/jacksonhurst/Stop-Spots>

**1. Description of all the features that have been completed**

* List of all CTA Lines (only red line currently functional)
* Alert notification for all the CTA Lines that are still under development
* List of Red Line Stops (only Fullerton -> Jackson currently functional)
* Pull in a list of 20 nearby spots (shops, restaurants, hotels, etc.) for each stop using the Google Places API
* Pull in ratings based on Google Reviews for each spot using the Google Places API
* Generate Google Maps with a marker on the spot for each individual spot using the Google Maps API

**2. Screenshots of every screen with explanations**

Graphical user interface, application

Description automatically generated

**Home Screen:** This is the home screen. Super simple entry page to start the user off. Simply click Pick a Line to get started

Graphical user interface, application

Description automatically generated

**Pick a Line Screen:** Another simple screen. This screen serves as a simple way to pick the line you want to pick a stop for. We chose to just implement the red line at the moment, as we believe implementing multiple lines would be overly ambitious for the scope of this project. When you tap on the button for any line besides the Red Line, you will receive a alert message saying that we are still working on other lines, and red is the only functioning line at the moment.

Graphical user interface, application

Description automatically generated

**Pick a Stop Screen:** Once again a simple screen with a limited number of the stops for the red line. Once again, due to the amount of work that goes into building the API requests and other functionality for each Line/Stop, we chose to limit the stops we had for the sake of this project. We chose to limit it to the stops between the DePaul Lincoln Park Campus and the DePaul Loop Campus.

Graphical user interface, text, application

Description automatically generated

**Pick a Spot Screen:** This was the first technically challenging screen to build out. We chose to use the detail/master view that we had learned for assignment 8. This screen requires an API call to the Google Places API to fetch all of the shops/restaurants/hotels/etc. within a given range of the stop. This is completed by getting the latitude and longitude of each of the stops, and the doing the search. For the sake of this project we store the lat/lon of each stop statically in the code.

Graphical user interface, application, timeline

Description automatically generated

**Google Maps Screen:** Finally once a spot is picked, we use our other API which is Google Maps. The request for Google Maps requires the latitude and longitude which we store in values in a “spot” struct. All of the spots for each stop are built into an array of these spot structs, and then we simply access the values once a specific one is selected. It also requires that we build a “marker” which we once again just use the same latitude and longitude.

**3. Final Project Discussion - Done individually (Jess)**

For this project we used two API features. The first API feature we used was Google Places. This API was a big part of our project. Inorder to access the data we needed with this API we had to create a google cloud project. Then, we used the API key to retrieve 20 spots nearby the train stops we selected. This gave us a total of 160 spots between the 8 train stops. For each of the 160 spots we collected were given latitude and longitude. This way we were able to store the latitude and longitude and run a HTTP GET request to the Google Places API. The API then returned an Array of the 20 spots in a JSON file. Google Places is a very cool API that gives a ton of information including but not limited to the name of the place, the phone number, the latitude and longitude, and the average reviews. However the purpose of our project, we only needed the name of the place, the latitude and longitude, and the average reviews. All the results gathered within this API are in the detail/master view in our Xcode project. In the master view the user can see the spot name, for example, DePaul University, the rating out of 5 stars.

The second API feature we used was Google Maps API. For this API we had to install the Google Maps SDK package as a dependency in our project since a simple HTTP request was not available like the first API we used. Inorder to use this we had to install cocoapods and created a podfile. This allowed us to put a map on the detail view page in our app. We took the latitude and longitude of the stop and put that into the center of the app and made it the maker showing the exact location of the spot. This map also shows nearby locations and the street names. It even shows the train stop near it. The user is also able to move the map around to see anything they want to see in the area. The user can also zoom in and out of the map allowing them to read the street names and other locations better.

One of the biggest challenges I faced was getting the app we envisioned in our heads on the screen. We wanted to use a lot of things that we did not know how to use. For example the train stops page was a big challenge. We did not know how to fit all the stops onto one screen. We looked into a scroll option but did not know how to use that. Then we looked into what I believe is called a picker. This would have been a good option for us but we were not sure how this would work with our backend code. Thus we decided to limit the number of stops the user could choose from to just eight stops. This ended up working in our favor since the API took up a lot of time and we would have never been able to build out all the stops for the red line anyways.

Another issue we had was with the master and detail view page. We wanted to use this page to show the spots the user can choose from and once they picked a spot the next page would show them the location of the spot on google maps. We had a hard time connecting the master view set up to the rest of the screens we already set up. When we simply Ctrl click and drag to the master view set up for some reason it would show us the detail view page before the master view page. We had no idea how to fix this. We first tried to connect it to different pages and that all made the app crash. Then we tried playing around with the connections on the master view setup. We were able to get the app to work by deleting the bottom connection to the master view page. I have no idea why that worked but it solved our problem and were able to get the app completely functional.

Our app has a lot of limitations and definitely could have been improved with more time. Our first limitation is that we only have the redline working. If we had more time we would have liked to have all lines working. Right now our app only has the red line functioning. When the user clicks on a line other than the red line it brings up a warning box that states for example, blue line coming soon, only red is working. Thus ideally the app would eventually advance to all the lines stops.

Another limitation we have is for the one line we do have working, red, we only have 8 stops that show the local spots. Currently we only have Fullerton to Jackson working. If we had more time we would have liked to have all the redline stops available. We also had trouble finding a way to list all the stops on the screen. We did not learn about any scroll options and since there are a lot of stops for the redline it would have been nice to use a scroll to see all the stops in a reasonable size. Since we did not know how to do that we choose just to do eight stops since that would fit on the screen.

Another limitation is the pick a spot screen we only show twenty spots and we would have liked to include more. Another thing we wanted to do was have it tabbed so that you can choose how to filter the spots. For example one tab would be for restaurants and another could be for shops in the area. However, again due to time constraints we could not separate them out.

Finally, our last limitation is that the screens use segues. This just opens the screen on top of the other screen meaning if you went through the screens multiple times it would eventually start lagging. This is also not very user friendly because you can accidentally scroll down and go back a page you did not want to go back to.

Overall I like developing in iOS. It has been a challenging learning curve as for I have never used an Iphone or a mac before in my life but I found this class very interesting. I think by the end of this project I finally got a handle of coding in swift and using Xcode. There is a lot more I would like to learn with it however. I want to become an app developer in the future when I graduate, thus this class was very helpful in showing me what I might be doing after graduation. I've learned that building an app is very time consuming. There are a lot of individual parts that go into building an app. For example, you have to put all the items you want on the screen and build constraints for the app and test the constraints on different versions of the Iphone to ensure that the view is similar for each version of the Iphone. Another thing that you have to pay attention to is the app icons. For each version of the iphone the icon is a different size. Thus you have to resize the image for the icon to fit all the different versions of Iphone. Paying attention to all these small details is not something I would have expected with app development however it has proven to be very important to it.

**3. Final Project Discussion - Done individually (Jackson)**

Discuss what are API features, such as navigation controller, quartz, or gestures, etc., you have used in your final project. Discuss exactly how they are being used. Especially, did you use features that are not covered in the lectures?

* We used two main APIs for this application.
* The first is Google Places. Google Places required us to create a Google Cloud Project and create an API key so that we could access the features. We utilized this API for a lot of our project. First, we used it to fetch 20 points of interest nearby a given latitude and longitude. By doing this, we were able to statically store the lat/lon for each of the stops that we built out functionality for, and then simply run a HTTP GET request to the Google Places API. This API returns a JSON with an array of 20 “results” which are simply the “spots” nearby. Each of these results contains a TON of information. With this you get everything from the name of the place, to the phone number, the latitude and longitude, the average reviews, and much much more. We were interest in the name, lat/lon, and average reviews. We displayed all of these results in a masterview/detailList and the label was the name (which is shortened for over 20 characters) followed by the average review out of 5 stars.
* The second API is Google Maps. This one was a unique API to use as it didn’t consist of a simple HTTP request. For this API, we needed to install the Google Maps SDK package as a dependency in our project. I spent a lot of time trying to do this dependency management manually, and then learned how to use cocoapods. Once I installed cocoapods and created a podfile, it became much easier to manage the packages we needed for this project. Once this was functioning, it was actually quite simple to build a mapview into our detailView for the list of spots. It was as simple as passing a function the lat/lon of the center point of the map and then creating a marker for the location of the spot. This map was built in the detailView, as it needed to be rebuilt for each individual spot that the Google Places API pulls in.

Discuss some of the biggest challenges you have encountered during the development of the final project. Discuss how they are resolved.

* I had three major challenges as I did the entire backend development
* The first huge challenge was creating a synchronous HTTP request. I had never done any of this sort of development for swift/iOS before, so it was an entirely new beast to me. I had to learn about dispatch groups and how to make the main thread wait on the result of a request which is typically being sent on another thread. My resolution to this in the end was to literally just wait to load the page until the request was complete. It was not the most elegant resolution, but it proved to be a functional workaround.
* The second major challenge was data modeling. Essentially, we received a JSON Data type from the HTTP request to Google Places. Now we need to take this and make it into usable, mutable data for our frontend. This was a long and strenuous process to figure out the best way to parse JSON in swift. Eventually I found my way to SwiftyJSON. I installed this external dependency and it proved to be by far the most elegant solution to parsing and handling JSON in Swift. I then created a spot struct which simply contained 4 values, a name, average rating, latitude, and longitude. These were the only 4 values we needed from each result/spot to not only display all of them in a list, but also plot them on the Google Maps Map.
* The third and final major challenge was dependency management. Once again an entirely new domain to me and it required a lot of reading and research. For the first half of the backend development I simply just downloaded and managed the packages manually. Once I got to Google Maps though (which is a quite complex package to install manually) I realized it was time to figure out what cocoabeans was. Once I properly installed and set up cocoabeans on the project though, it was a breeze handling dependencies.

Discuss any limitation of your app and how do you think the app can be improved.

* The largest and most obvious limitation is our limited number of stops that are functional at this time. If we were to build out full support for the entire CTA (all lines and stops) that would likely be many more 10s of hours of work. Right now we only serve the red line Fullerton-Jackson stops. I will admit here that the coding practices for the requests and storing the data is not the most perfect, as I am certainly new to a lot of this type of development in Swift. Should we decide to scale this project to the full CTA, it would definitely require some refactoring of the backend.

Discuss any limitation of the iOS SDK and Xcode you have encountered during the final project as well as the programming assignments.

* Swift has proven to be quite a capable language for backend development. There are a few things which I think the language does quite poorly, but overall I was able to overcome any significant issues. Xcode also seems to be a very capable tool for development. I believe in many ways it is actually better than Android Studio. Most of my issues with the IDE come from the fact that I am running the whole thing on a 2012 Mac Mini. This made emulation and some UI development quite slow and strenuous. That said, it is amazing that I can still build a fully functional modern iPhone application on a very old Mac!

Discuss your overall experience as an iOS developer so far.

* I will be entirely honest here. In the majority of this course I did not enjoy swift or building out user interfaces for iPhone. This only change on this final project, when I realized some of the really exciting and fascinating stuff that you can build with Swift and Xcode. I might still prefer Android development by a bit but this is only because of the native support for Java (my strongest programming language by far)

**3. Final Project Discussion - Done individually (Jennifer)**

Discuss what are API features, such as navigation controller, quartz, or gestures, etc., you have used in your final project. Discuss exactly how they are being used. Especially, did you use features that are not covered in the lectures?

* For our project we used two APIs, the Google Maps and Google Places API. The Google Places API was used to create a Google Cloud Project that allowed for us to use the API key to fetch addresses and information of 20 restaurant and shop locations near a red line stop that the user chooses. We decided to pull destinations for eight red line stops, giving a total of 160 spots recommendations. Each spot had it’s on latitude and longitude which we stored and ran a HTTP GET request to the Google Places API. As a result, the API returns an array of the 20 results of the nearby destinations in a JSON file. Not only did the JSON file consist of latitude and longitude information, but it included other information like name of the location. Phone number, all customer reviews, and address. For our project, we used additional information like the name of the place as well at the average rating of reviews for the users.
* For the Google Maps API, we used the Google Maps SDK package as the dependency for the project. To get this working, we installed cocoapods and created a podfile to manage packages needed for our project’s functionalities. Cocoapods was used to build a mapview on the detailView screen to list out nearby spots near a specific train stop. The display was a pinpoint on a map which used the latitude and longitude pinpoints to give the precise location.

Discuss some of the biggest challenges you have encountered during the development of the final project. Discuss how they are resolved.

* One of the biggest challenges the team encountered during the development of the final project was finding ways for each of the team members to work on different parts of the project on our own. We found that it worked out doing the front-end side with the screen design, but when we had to start implementing the back-end, it was best to work together in-person.

Discuss any limitation of your app and how do you think the app can be improved.

* Limitations of our app include features only available for popular destinations on the Red Line stop from Fullerton to Jackson, aimed towards providing restaurant and shop recommendations for DePaul students who commute from the Lincoln Park campus to the Loop campus. The app can be improved with adding a feature that allows for the user to get restaurant and shop recommendations from all the stops on the redline as well as of the other CTA train lines. Since there are so many train lines and train stops, we limited ourselves to just popular stops on the red line to really get to.

* Another feature that could be added to our app to improve it is the front end where we can have more designs and aesthetics to improve the visual side since we were really focused on the back-end side of things and getting the screens to interact with each other and with the Google Maps and Google Places API.
* We could add a table and key with information such as which train stations had elevator access for the handicapped or which station had the possibilities for transfers. Another feature that we could have added is the directions component, where the user could be interested in going to a destination close to their train station which gave them directions on how to get there instead of showing the pinpoint of the location.
* Another feature that could be added to the app to improve it, is the capability for the user to save destinations that want to potentially visit. Instead of them having to go back to pick the train line and stop to search for the destination, it would be saved into a different tab that they can easily access.

Discuss any limitation of the iOS SDK and Xcode you have encountered during the final project as well as the programming assignments.

* A limitation of the iOS SDK and Xcode the team encountered during the final project was a problem where a button kept redirecting us to a page that we didn’t want to be directed to. We spent quite some time trying to figure out the issue even though the arrow of the page was not pointed to it. Some other issues I ran into programming assignments throughout the quarter are just some constraint and formatting issues that kept giving me errors even though it was addressed. In the beginning of the quarter during the XCode and tools installation process, I ran into an issue with getting XCode 11. Although I had it downloaded and updated, it just would open and run on my computer and this was because of the Mac Version I had. I ended up updating to macOS High Sierra where I was still experiencing the same problem. I then updated to macOS Mojave which eventually worked. I originally did not update the operating system because of the problems i heard other Mac users were experiencing. It was a very time consuming process because I had to do some research on the issue and the updates took a really long time.

Discuss your overall experience as an iOS developer so far.

* My overall experience as an iOS developer has been good. I will admit that it was difficult learning Swift in the beginning because of the syntax and learning the different functionalities that go into creating an app. There’s so much to Swift and XCode that allows for users to create an app to their liking that I have yet to discover. Each week has been really interesting working on the assignments, using the new material to build on from the previous assignments. I really like being able to take control with the front-end visuals and getting them to work on the back end with a functionality that interacts with other parts of the application. I hope to continue learning swift and xcode, and create side projects that have more complex components to them. For my next personal project, I really want to create an iOS app that incorporates what we’ve learned this quarter.

**4. Team Contributions**

Jess worked on the front-end of the app specifically creating all the screen views, design, and connecting the buttons to go to each screen. Jennifer created the detail/master view screens and created alert boxes for the other lines that were not in function yet. Lastly, Jackson worked on the back-end of the application using the APIs to get the spots and maps working. We came together on Sunday and worked on the project as a team as well.  We all equally put in work into the final documentation.