Mallory Leppelmeier

Independent Study Report

December 7th, 2021

Southland Christian Church App

This semester I designed and implemented an app for Southland Christian Church. The app was designed for members of the church to easily access resources the church provides on their website on both IOS and Android devices. Southland has a good website that is designed for people new to the church to find resources and information. However, some of the resources that are more important to regular attenders are sometimes hard to find. I decided to make an app for members of the church so they can conveniently access the resources that Southland provides for them.

I chose to implement the app using React Native platform. One reason I choose to use React Native was so the app was compatible on both IOS and Android devices to make it more versatile for Southland’s members. This takes out popular IOS development in Swift or Android development in Android11. Another reason I chose React Native was because it combines native development with React’s JavaScript library. This gives the app the look and feel of a native app unlike platforms such as Ionic or Apache Cordova. Lastly, I choose React Native because it was released 8 years ago. This means that there was a lot of documentation and tutorials available. App development in newer platforms such as Flutter or Xamarin don’t have as many resources available for first time app developers. More information about React Native can be found on their website, <https://reactnative.dev/>.

The firsts part of the semester was focused on designing the app. To develop a design, I first focused on learning the ins and outs of Southland’s website. I wanted to make sure I included all the links from the website that are most frequently used by members of the church. I then took the links and laid them out in a logical format that matched the layout of the website. I designed the app to be clean and simple so that users could quickly find Southland’s resources. One way I implemented this was by the settings page. This design allows users to customize their home screen to only show links that are most useful to them. From these layouts, I created wireframes in BuilderX. BuilderX is a browser-based design tool which helps developers put their designs on paper. After I finished working on the layout of the app, I added the esthetic of the southland website. I wanted the app to look and feel like an extension of the church’s website, so I included fonts, logos, colors, and pictures found on their website. My final design can be seen in the figure below.

Graphical user interface, application

Description automatically generated

After my design was completed, I met with the Creative Director at Southland, Neil Gregory, to see if he had any changes or possible improvements to my design. Overall, he said he was very impressed with my design and was excited to see the app when it was finished. Neil liked that the app matched the look and the feel of the website, but he did have some minor changes he would like to see. These changes were mostly small uniformity issues he found in my design. He wanted to see all the edges have the same level of rounding, all the buttons be the same color and contain the same icon, and all the text in front of pictures to be the same color. Another suggestion he had was to change a few of the links to ones he felt were more useful to members of the church. This was great feedback that allowed me to make the app more user friendly and better reflect the Southland website.

The next part of the semester was focused on learning react native and how to use the react library. I did this by watching the following series of YouTube videos created by The Net Ninja, <https://www.youtube.com/playlist?list=PL4cUxeGkcC9ixPU-QkScoRBVxtPPzVjrQ>. These videos taught on topics including stack navigation, drawer navigation, custom header, card components, background images, and forms. These tutorials were very helpful in getting familiar with the library and the environment, which helped me prepare for app implementation.

The last part of the semester was focused on implementing the app. I first started by creating the bottom tab navigation and the drawer navigation. I was able to use what I learned from that tutorial and create my design. Then, I went through and implemented each screen. Once I got familiar with the react native environment most of the screens were straightforward to implement. React native has good documentation that can be found at <https://reactnative.dev/docs/getting-started>. I referenced this documentation frequently while working on the app.

During implementation I ran into some hurdles that required more research to overcome them. Most of these obstacles occurred during the implementation of the settings and notes screens. The first problem was learning how to pass information from a child screen to a parent screen. When closing the notes or settings screen, information from those child screens is passed back to the previous parent screens. To solve this problem, I used React’s useEffect component. This component takes two parameters, the first being a function and the second being an array of dependencies. React compares the current value of the dependency with the previous rendered value and if the values are not the same, the function is invoked. Using this component, I could create a dependency on the parent screen and then pass it to the child screen. When the child screen changed the dependency, the parent screen would know and update its information. The second problem I ran into with the settings and the notes screens was the need for react native’s asynchronous storage. The information on these pages needed to be saved if the app was closed. I watched the tutorial <https://www.youtube.com/watch?v=1BXx39IMqEY>, created by ToThePointCode to learn how to store and get information from react native’s asynchronous storage. The last big obstacle I ran into during implementation included the speed at which pictures were loading. When switching between screens, the pictures were pulling from a local folder each time. This significantly increased the time it took for a picture to load on the screen. To fix this issue, I used a react native import called appLoading. AppLoading is a tool from Expo, which keeps track of loading and allows actions to be performed once the loading is complete. I used this import to download all the images and the fonts from a local folder, and once the information is loaded, the app could then open. On average, this speed up the time it took for images to load when switching between screens by 0.4 seconds. This is because the pictures were all stored in a local cache and could be grabbed quickly.

My favorite part of implementation was figuring out how to get the app to look exactly like my design. I liked making sure all the buttons line up correctly, spacing was even, the text was the right size and the icons where in the correct place. It was a very time-consuming process, but I believe getting the little details right are important to make a product look complete. While creating the app, there were a few changes to the design I made to make the app more functional and polished. One change I made was moving the ‘add a new note’ button to the header on the noteslist screen and making it a single icon. This allows more space to list notes and gives the page a cleaner look. I also changed the settings button to only be accessed from the home screen. I believe this design makes more sense because the settings only change what is displayed on the home screen.

Some other cool features I implemented include the external links. Most of the external links on the app lead back to the Southland website, but others open different apps. For example, on the prayer page the submit button will open an email app on your phone and auto fill the email with the information provided in the form and a southland email address to send it to. Another cool external link is on the podcasts page. The links on this page will open Spotify to the selected podcast. If Spotify is not downloaded on the phone, it will link them to a url that contains the podcasts. Similarly, the Facebook link on the locations page will open the Facebook app if it is downloaded on the phone, and the Facebook website if it is not. Lastly is the map button on the location page. This button will open maps on apple or google maps app on android and drop a pin at that southland location.

My final code can be found on GitHub at <https://github.com/mnle239/SouthlandChurchApp>, the repository is 5.87 MB. The repository contains around 1,500 lines of code, 16 pictures and 3 fonts. To run the code on your own device, follow the terminal commands below. The first and second command will download the repository from GitHub and enter the downloaded folder. The third command will download Expo CLI for the project. Expo CLI is a React Native developer tool that aids in creating projects, viewing logs, and opening simulators. More info on Expo CLI can be found at <https://docs.expo.dev/>. The last command will start Expo CLI. A Webpage GUI will open and show logs along with options to open simulators.

git clone <https://github.com/mnle239/SouthlandChurchApp.git>

cd SouthlandChurchApp

npm install

expo start -c

I really enjoyed working on this project this semester. It was fun to take an idea and create a finished product based on that idea. Along the way I learned a lot about design and how to make screens look uniform. I also learned a lot about the react native environment and had ample opportunity to utilize its many features, including stack navigation, app loading, and asynchronous storage. I am very grateful for the opportunity to create this app and I am very proud of the finished product.