Splitr

Final Report

Sam Hauck & Stefan Barac

Group Members

- Sam Hauck
    - Responsible for designing the UI, creating mockups, and implementing the UI using Cordova and Onsen
    - Responsible for Camera plugin and functionality
    - Contributed to weekly Trello Board assignments
- Stefan Barac
    - Responsible for backend/database implementation
    - Contributed to Application design
    - Contributed to weekly Trello Board assignments
    - Documentation

Abstract

The goal of our project shifted partway through the semester from creating a platform for students to organize study sessions to creating an app that allows people to split a check and display the total for each person. We decided to change our project following Dev Week after having trouble with implementing a map into our app and seeing that another group in the class had a similar idea. We had discussed a few different ideas when initially deciding the topic for our project and chose to continue the semester creating Splitr.

Our reasoning for choosing to create this app comes mainly from experience. Having to determine how much each person owes after eating at a restaurant is a fairly common occurrence, and we wanted to create a way to simplify this process and create a digital location to store receipts.

Project Narrative

The goal of the project was to develop a lightweight tools that served as a solution to splitting checks or bills with multiple people. The usability goal was to create an easy to use bill splitting tool that could be navigated easily and used quickly or on the fly. One example would be at the end of a dinner with friends in college, everyone wants to split the bill and pay for only what they ordered, this tool would allow someone to do that easily. We wanted to also build a tool that could charge the people who were attached to a bill. That way only one person would have to pay and the application would charge all the other people brought into the bill, this way payment at the restaurant would be easy and quick and getting repaid would be automatic and hassle-free. We wanted to build this tool for users because we have been in multiple situations where a bill is too large to split, or there are too many people to split the bill correctly. We hoped to create a tool that will replace any time spent fumbling through calculations.

Design Specifications

For our initial design for Study Buddies, we wanted to keep the design as simple as possible and the give the user easy access to all of the features of the app. As the purpose of the app changed, our design changed with it, but we tried to keep the goal of simplicity in mind when designing the new app. We started the design process by determining what datasets would be used and how we wanted to both obtain the data from the user and display it back to them. Our data consists of mainly numeric and text information from receipts such as each food item, prices, and the names of each person in the party. Due to having multiple form inputs (item name and price) for each item on the receipt, we decided to use the Onsen grid system to organize the layout of the input page. The grid system allowed us to put two Onsen Input

elements on a line without it looking overcrowded. For displaying the data, we combined the Onsen grid with an Onsen List element to create a similar look to the data entry page in the app.

Because we used OnsenUI, our styling for the app was mainly dependant on the CSS provided in the Onsen package, but we looked to popular apps like Facebook and Reddit for inspiration to implement the included UI features. One thing that we felt fit our needs for the app was the tabbar along the bottom of the page. This tabbar truly fit our goal of creating a easy to use app that still allowed the user to see all relevant data.

During discussion for the design of our original app, Study Buddies, we wanted to use a hamburger menu that is commonly seen in many popular apps today. After numerous unsuccessful attempts to use the Splitter Menu from Onsen UI, we decided to seek out other options for navigation beyond the hamburger menu. The one-click access to different pages of the app was something that immediately grabbed our attention when thinking critically about the design of apps that we use on a daily basis, and upon doing research on UI design practice, we discovered that there has been a recent shift from hamburger menus to using tabbars. The tabbar allows the user to easily navigate through the app without needed to actively search through the app to find the feature they are looking for because the tabbar is always present. On top of this, the navigation option does not block any of the content of the app, which is not the case with a hamburger menu.

Our initial design concepts came to fruition through first hand drawing prototypes to hash out a simple design. From there we used the program Balsamiq to make actual mockups for Study Buddies. We used Balsamiq's mobile-specific templates to get a better idea of how our design would look after implementation. When we switched the focus of our project, we repeated this process of taking hard drawn prototypes and recreating them in Balsamiq to have a solid foundation on which to base the design.

Because we changed our project after Dev Week, the feedback we got from our peers was not fully applicable to our new design. Instead, we analyzed the presentations that were given during Dev Week to find ways that we could improve our design. Once the design for Splitr came to life, I asked friends and family to give me critical feedback on the usability and theoretical functionality of the app. Listening to their feedback, we ended up making a few changes to the UI. First, we rearranged the order of the tabbar options for a more logical workflow. Originally, we had the results tab to the immediate right of the data entry tab with the camera tab on the far right, but someone suggested that the camera tab should be moved to the left. Following this change, we decided to incorporate all navigation into the tabbar, moving access to the settings page from a button in the toolbar to a final tab. We, and the makeshift testing team we had available, thought this streamlined the functionality of the app.

Testing and iterative design:

Throughout the development of the project, we utilized usability testing by eliciting the help of family and friends. While some were either involved in Computer Science or tech-savvy, another portion only used apps necessary for basic phone functionality. We wanted to make sure that anyone who used Splitr could open the app and understand its purpose and how to use it.

Beyond usability testing, we made use of Google's debugging tool to fix any errors in our project that we were able to resolve. Through testing, we encountered an HTTP request error on one of the pages in our app. It threw an exception status of 404 Not Found, despite the page in question (results.html) being readily available as an Onsen template, like all other pages in the app. We initially thought it was due to not having a function to catch any exceptions, as we were also plagued by a "Uncaught (in promise) Cannot GET /result.html" error. After added

catches for any exceptions, the error was not resolved and we were left in a state of uncertainty on how to resolve this issue.

Restrictions:

From the beginning of the project, we started running into restrictions. The first being issues using Cordova on a Windows machine. After being unable to utilize Cordova through the command line, we moved to Visual Studio and used Cordova through that environment, but it never functioned fully. Luckily, we also had a Mac in the group so most of the development took place on that machine through the command line. This difference in development environments made it much more difficult to properly test our code. Beyond these issues, we also ran into restrictions due to OnsenUI. The first issue we came across was using the Onsen navigator to display a home screen before moving to the form completion page. Because we were using the tabbar element containing four different HTML pages, the stack functionality of the navigator was not working properly. The tabbar continued to cause issues, now preventing the app from opening the camera to take a photo.

We also ran into issues regarding database storage for the user submitted receipt information. Having had experience with SQL, we chose to use SQLite and were able to successfully create the database and the table needed to store the data. Unfortunately, we had issue connecting the SQLite database to the Onsen Input elements to pull the data from the app. Being unable to display saved data left much to be desired from the results page. In an ideal situation, the user would enter the ID of their receipt and the database world return Onsen grid elements with each person on the bill, amount owed, and the saved image of the receipt.

Looking beyond what we accomplished, or tried to accomplish, in this class, there are some features that we feel would greatly improve the functionality of the app if implement. The

first idea is to create a user account system to connect people through the app. This user account system would then allow a payment method to be associated with the account, using the Braintree or Stripe APIs for example. Once getting the camera working, we would also then implement Optical Character Recognition technology to streamline the process even more, allowing the user to simply take a photo of the receipt the the result would be then generated automatically.

Another feature that we would like to implement in the future is a custom plugin that determines the tax rates depending on the GPS location of the device. As of now, the tax is entered manually and divided equally, dependent on the number of people on the bill. The plugin would look at what each person owed individually and calculate the amount of tax they owe on that number rather than an equal share of the total tax.

Conclusion

The experience creating a Cordova mobile application is not one we will quickly forget. On the surface, Cordova appears to provide a streamlined process for creating a fully functional mobile application, but our experience proved otherwise. From trouble with setting up the development environment to compatability issues with OnsenUI, Cordova has given us a taste of all of the tedious work required to create mobile applications. This project has given us a stable starting ground to create mobile apps in the future without being overwhelmed by simply the thought of developing them. By separating the app into different parts, either mentally or through code, it becomes much more manageable to lay out a coherent development lifecycle. This project allowed us to make use of the steps of development that we have learned through classes and work experience.