SpikeBanner

By: Tom Parry, Will Denne, and Jared Parrish
Client: Enrique Cubillo
6/19/2018

Table of Contents

Introduction

Enrique Cubillo is the inventor of a new mode of urban transportation and new endurance sport called SpikeBoarding™ as seen below in Figure 1. He has big dreams to see SpikeBoarding become the next big racing sport and to see the rise of SpikeBoarding in an urban environment. He came to Mines with a project to create two apps and a website to track and promote non-profits as well as SpikeBoarding in general. For the non-profit organizations, Enrique uses banners that he can connect to the back of his SpikeBoard to advertise for their causes. He calls this enterprise SpikeBanner™. The goal for us was to create one private app that could be used by Enrique and the rest of his team to track themselves as they use their SpikeBanners to promote the different non-profits. From there a public Android application and website was created to retrieve and show data that has been collected by the private app. If a member of the SpikeBanner team is currently tracking themselves, the users will be able to track each team member live and see which banner they are currently SpikeBoarding with on the public app. On the website, Mr. Cubillo wanted an elegant display of data for each of his sponsors as well as an easy way to add and remove the non-profits. Finally, all applications that write to the database must be secure so that no one can change the data without being a member of the SpikeBanner team. With the new SpikeBanner applications, the SpikeBoarding and SpikeBanner team will have an easy way for people to follow along with their journey and watch SpikeBoarding rise to fame.

Figure 1: Picture of Dan Bowen Spikeboarding around New York. Photo taken by Enrique Cubillo

Requirements

High Level Vision of the Product

The SpikeBanner project is made up of three different components: a public website, a public Android application, and a private Android application. Each component has a different role and needed to be completed by June 18th, 2018. The website has two main subcomponents. The first is a basic table that displays the distance and elevation accumulated under each sponsor's banner. It looks nice and fits in with the client's other websites. The other subcomponent is an administrative page so the client can add or remove sponsors as they come and go. Next is the public Android application. Like the website, it is aimed at the public and displays the data that was collected under the sponsor's banner. Finally, the private Android application collects the elevation and distance data from the athlete (similar to Strava or

Sno Cru). It will have a limited release and only be available to selected SpikeBanner athletes. It saves the data collected to a database for use by the public applications.

Functional Requirements

The website displays the data collected by the private application. Each sponsor has picture to be displayed in the table of data. Additionally, a login page is available, and upon logging in the authorized user can add or remove organizations from the database. The public Android application must be similar to the website. Each sponsor is selectable from the main screen. The next screen will display the data that was collected and entered into the database, along with a picture or two of the banner. The public application was deployed to Google Play Store on Friday, June 15. Finally, the private app must track the user's location and other relevant data points. A basic login screen determines which athlete is using the app and which banner they are riding on. The app then collects data using a foreground service and uploads the data to the database when the athlete is finished.

Non-Functional Requirements

Both the public app and website must be elegant and simple to operate. While the private app does not need to look good, it still is easy to operate and understand. The client is the administrator on all accounts to make sure they can change the application in the future. The website also works with the client's current website infrastructure. The applications uses affordable and easy to manage third party libraries and software.

Nice to Haves

The client's minimum requirements are listed above, but he did have a variety of other optional requests. On the public app and website, each individual trip could be selected, leading to a screen displaying the data being and a map of the trip. Another request was to implement live tracking of the athletes using the private app. There is also a lot of possible expansion on the administration end. While adding, removing, editing organizations and users is possible through the Firebase console, it is not easy to do so for someone who is unfamiliar with Firebase. (Note, the Firebase database and why we choose it will be explained later in this document.) We could have made it easy for the SpikeBanner team to do all this on the website without having to figure out how Firebase works. We had only enough time to implement the live-tracking feature and administrative features for the website.

System Architecture

Public Application

The top left image in Figure 2 shows the home screen for the public Android application. It shows the list of companies, which are all clickable buttons. Upon selecting an organization, the user is directed to the screen on the right, which displays all distance, elevation, and time data collected under the corresponding banner. Pressing the back button takes the user back to the home screen. If a SpikeBoarder is actively riding, a notification bar appears at the bottom of the screen. Upon clicking this notification, the user is directed to the lower screen in the diagram. This is a map, with all current SpikeBoarders' locations marked on it with the SpikeBanner logo. The user can tap each of these markers to see which rider it indicates and which organization they represent.

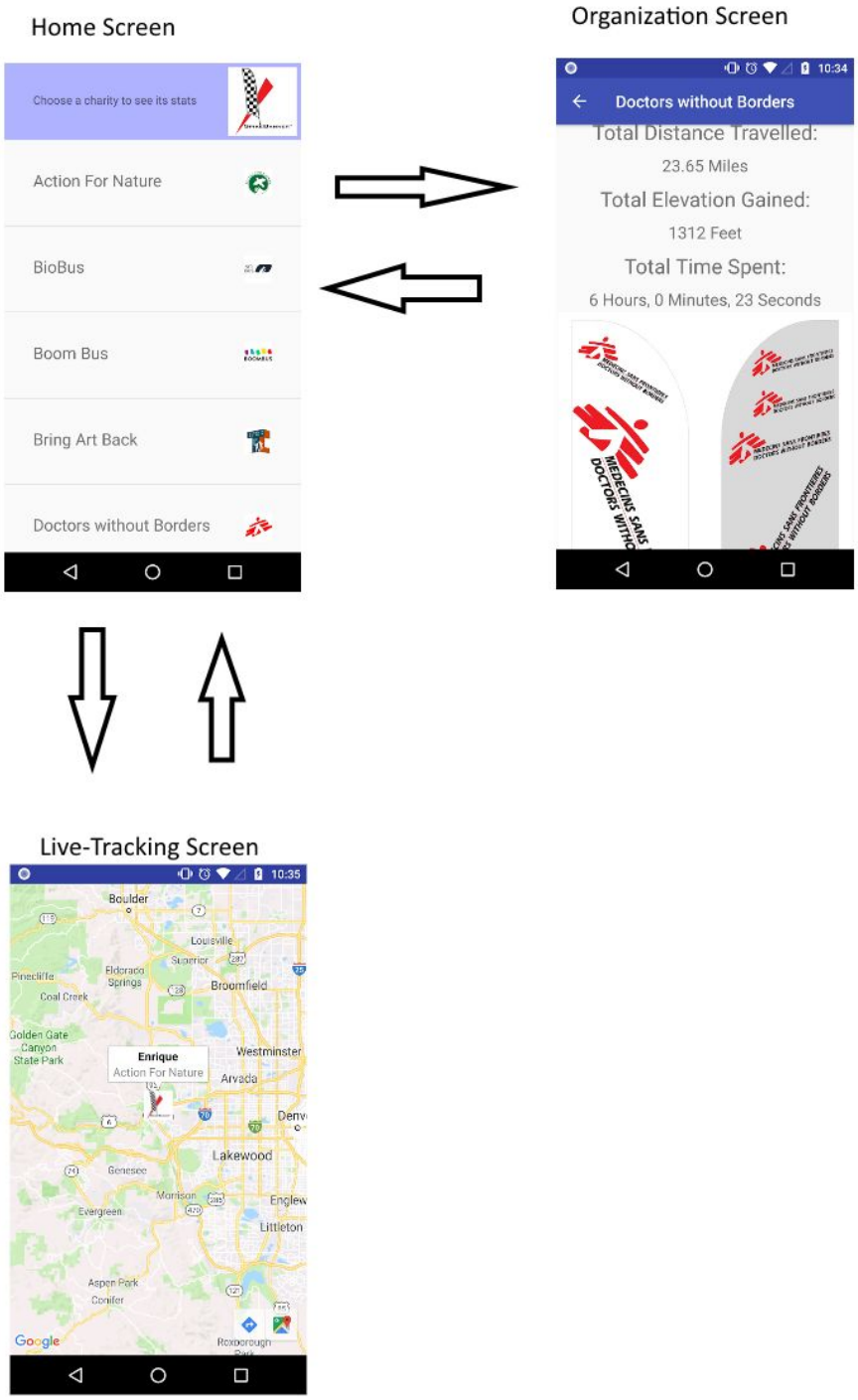

Figure 2: Public Application Storyboard

Private Application

The home screen of the private Android application, shown in Figure 3, requires the user to log in. It has text entry fields for email, password, name, and organization they are representing. When an email and password are submitted that are recognized by the database, the user is directed to the data collection screen, shown on the right. The three buttons across the top allow the user to start data collection, pause collection, and stop and store the data. Below the buttons, the user is shown their elevation, distance, time, and organization for the current ride. When the stop button is pressed, these values are saved to the database.

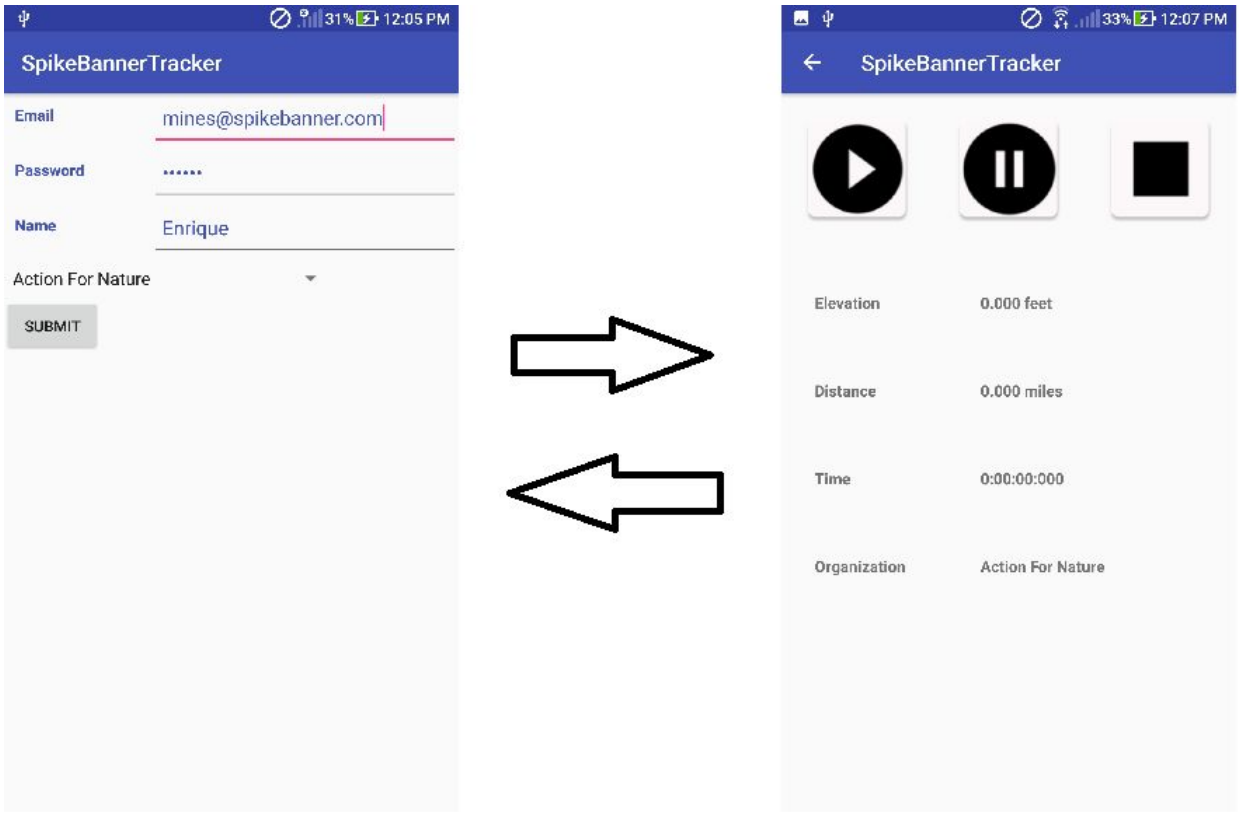

Figure 3: Private Application Storyboard

Data Flow

Our database is a NoSQL database hosted by Firebase. As seen in Figure 4, all parts of our project communicate with and rely on the database. All communications are in the form of JSON objects. The private application receives data from the database so that it knows what organizations can be represented, and sends data to record distance, elevation, and time. The public application does not send data to the database, but receives composite distance, elevation, and time for each organization. The website receives the same data as the public application, but it can also send data by adding or removing sponsors. Anyone can view the website or download the public app, which is currently available on the Google Play Store. However, the private app has not been published and is only for use by SpikeBoarders.

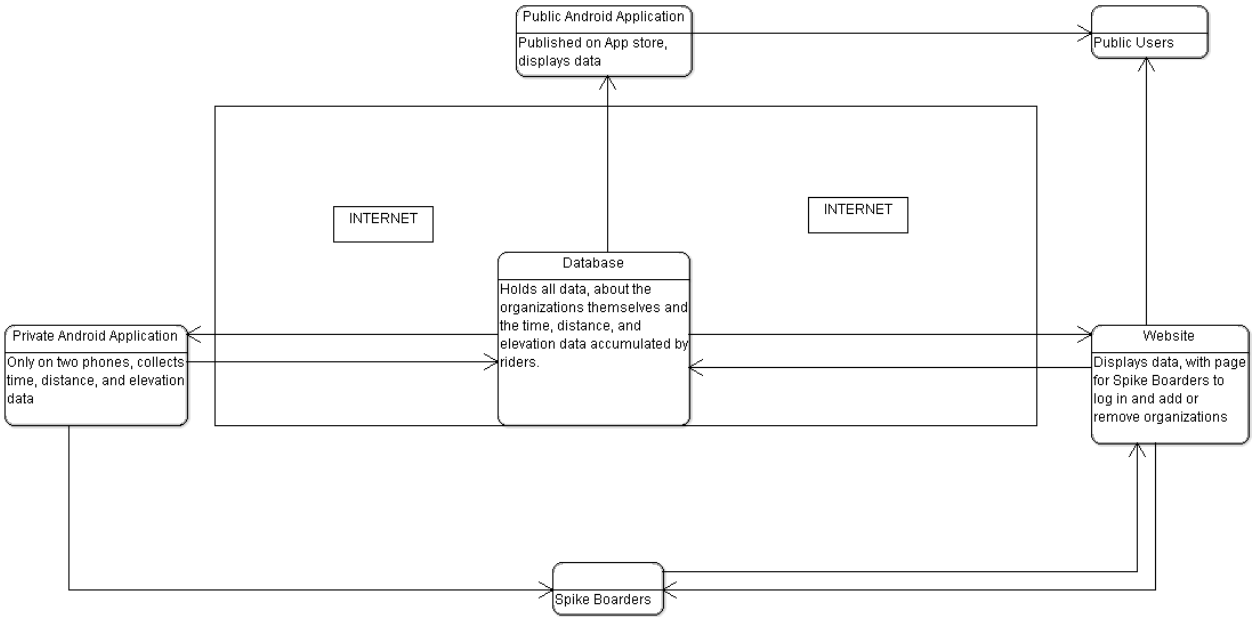

Figure 4: Data Flow diagram

Database

This entity-relationship diagram shows (figure 5) how data is correlated in the database. Any number of riders can be riding for 1 organization at a time, and each rider has fields to store their name, current organization, current latitude and longitude, and whether they are active for live tracking. Each organization is composed of fields to hold their name, logo URL, and the total distance, elevation, and time accumulated by riders displaying the sponsor's banner.

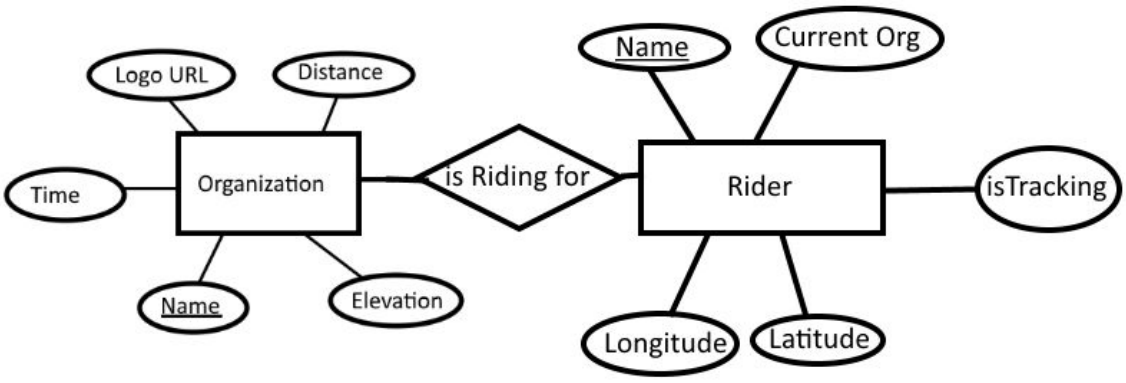

Figure 5:Database Entity-Relationship Diagram

Technical Design

Database

All applications share the same database with each other. Each application reads from the database in some way, and the private Android application and website also write to the database.

This Firebase database is a NoSQL database. Therefore all data is stored as a JSON document, as opposed to tables such as in SQL databases. JSON objects can be thought of as nested dictionaries, where each 'key' has a corresponding 'value' that can be either a single value or another JSON object. While the database is set up similar to that of a SQL database, it is slightly different in its design and how you read and write data. In the SpikeBanner applications the Organization and Rider sections act as tables. They each have different tuples with different attributes. The Organization table keeps track of each organization's stats over the lifetime of the partnership. The Rider table keeps track of location, name, and whether the rider is active for the live-tracking aspect of the app.

Location API

Another interesting aspect of our project was getting the location objects to track location change. We chose to use Google's FusedLocationProvider API to collect and track our location. We chose this API because it was easy to implement and use and there was a lot of information on how to use it online. Using this API we were able to get the last known location of our phone and we were able to request location updates at an interval that we set.

Decisions

Application Deployment Operating System

We were given the option to develop the public application for either just Android or both Android and iOS. While there are a few options to write and develop code for both operating systems and app stores, we elected to develop only for the Android OS and Google Play Store. This decision was made mostly to keep costs down. The cost to publish an application for Android is $25, while Apple costs close to $99. This does not include obtaining MacOS computers to work on the iOS app. Cross-Platform tools were

also considered, but we would still need to pay $99 for the Apple App Store and most cross-platform tools are expensive or have limited free functionality.

Development Environments

Two members of our team were already familiar with Android Studio to use as our Android IDE. Android Studio is the official IDE for Android development, is free and easy to use, and is easy to test and deploy both on physical and emulated devices. Due to our previous experience, this was an easy choice. One team member had some experience designing websites using the Atom IDE. Atom is an open source text editor made by GitHub. It allows the developer to view a preview of the website without having to open a browser, so this feature combined with one team member's previous experience led us to choose this IDE for the web development portion of our project.

Location Service

A key component of the private application is tracking the distance, elevation, and path taken by each rider participating in SpikeBanner. There are many code bases and services that allow apps to track the phone's location, but our options were already decreased due to the fact that we chose to make an Android app. Our team researched the best options to retrieve device location based off what was most commonly used, and we eventually narrowed our choices down to the android.location library or the Google Location Services API. The android.location library is considered deprecated, so this decision was simple. Using the Google Location Services API allows our application to easily retrieve the location of the device at a set interval. For getting elevation we first looked into also using the Google Location Services API but found that over time, the inaccuracy from android.location is negligible. The location services API would have been more accurate, but also could have cost our client more in the long run.

Database Host

We decided to use Firebase as our online database of choice. We made a table to show our decision making process:

Database	Cost per GB stored	Database Type	Easy to implement?	Other Comments
Firebase	Free under 1, $.026/GB after	NoSQL	The easiest	Used by a lot of others, made by Google
Amazon WS	$.115/GB	PostgreSQL (or other SQL-type)	Kinda	AWS is an industry standard
Azure	Free under 2, $0.085/Gb after	SQL	Yes	Made by Microsoft
Bluemix	Free under 1, $1/GB after	NoSQL	Yes	Not really made for beginners

Firebase was the cheapest and easiest to use out of the ones we considered. There is a lot of documentation about using Firebase for both Android and web applications. NoSQL can be easier to work with than SQL especially in small, basic databases like the project's. Firebase also can be used for iOS devices if our client decides to have an iOS application made.

Code Sharing

Being able to share code amongst the team is obviously important, and we had several options. Github, Bitbucket, and SourceForge are three of the most popular sites that host code for developers, and they are all free to use. Thus, the main factor our team took into account was familiarity. All team members have completed the Software Engineering course offered by Mines, and this course contains a segment using Github. Therefore, all three of us were familiar with this service and had all of the necessary components (such as the Github command line, Git Bash) and knowledge to easily incorporate Github into our project.

Results

Our goals were to create a private Android application which would track the user's location, another Android app and a website which displayed statistics gathered from the private app. We were able to implement all requested features and a few small extras. This including an organization manager for the client, pictures of the banners on the app, and a few others.

Testing

For the private application, there are only two identical devices that need to run it, so compatibility testing was not necessary. We have conducted basic data collections tests with the private app which have yielded good results. These included short distance tests around campus and extended driving tests in the mountains. For the public Android application, we have tested a variety of emulators, Android versions, and device screens. As of yet, we have not experienced any issues with device compatibility. We still need to test on more devices, especially those with larger or smaller screens and devices which run older versions of Android. For the website, we've tested on Chrome, Firefox, and Edge. We've also tested on Chrome for Android and Safari for iPhone. The website appearance and

function is satisfactory for all browsers we have tested so far. We tested the private application by driving to Windy Saddle with the app open. The distance and elevation data returned were fairly accurate, and we believe any inaccuracy can be explained by the fact we were driving in a car and not on a SpikeBoard. We also tested the private app with Mr. Cubillo himself on a SpikeBoard, and discovered one issue. After approximately 5 minutes, the operating system on the phone stops our location tracking service. We solved this by making it a foreground service, which causes the operating system to recognize that even though the app is not open, the user still wants it to run.

For the Future

There are many possible extensions for the SpikeBanner app. One possible extension that we did not have time to look into is mapping the route that is tracked and displaying that route to the user of the private app and possibly store routes to the public app so users can look at different routes where data was collected from. Another extension could be for the live tracking map to mark each rider with the logo of the sponsor they are currently representing rather than the SpikeBanner logo.

Lessons Learned

- We learned how to research and implement Google and Android location APIs.
- We learned about how some functions can cause memory to leak and we had to problem solve to work around these functions to avoid taking up too much memory.
- We learned how to work with databases stored in the cloud.
- We learned client relations and how to work with a client who has no computer science background.
- We learned how to implement third-party authorization functions in both a website and mobile application context.
- We learned about how the operating system saves memory, and how this affects services in the background and in the foreground.

Appendices

Below is a user manual that will be given to Enrique so that he can understand and use the app to the best of his ability. Below we describe how each application and website works and explain how each is supposed to be used.

Section 1: The Private App

The private application starts with a login screen where it will accept a username in the form of an email and a password to login the user. Also on this screen you will be able to choose what banner you would like to select. From here the app will log you in if the username and password are correct. On the next screen the you will see three buttons across the top. The play button will start a tracking route. The pause button pauses a route in process until the play button is pressed again to resume the route. The stop button will stop the route entirely and send data to the database. After the stop button is pressed the next time you press play an entirely new route will begin. Below the buttons are displays of the data of the route. You will see the elevation in feet, the distance in miles, the time, and the company name of the organization you are representing for the route.

Section 2: The Public App

The public application's first screen takes names and logos from the database. You tap an organization to look at its stats. You also get a picture of the banner. If someone is actively SpikeBannering, a button will appear allowing you to track their location on google maps. There is not much you need to do with this. Just keep everything in good standing.

Section 3: The Website

The initial page is spikebanner.com/app, and the login page is {Removed}. The only way to access the login page is with a direct link. This is for security reasons. Login with your Firebase credentials to add or remove charities.

Adding Charities: After selecting "Enter a new Organization", Enter the name, logo, and banner picture for the organization. The logo picture must already be hosted another website. It must be a direct link to the picture! The URL must end in .png, .jpg, etc. Same goes for the banner picture. For the best results, use a 100 x 100 sized image for the logo and a 1000 x 563 sized banner picture. The banner picture can be a differently sized, but a 16:9 or similar aspect ratio with a similar size will work the best. If you have already SpikeBoarded with a banner, just click on the check for each piece of data you want to enter. This is optional.

Removing Charities: This is straightforward, just select an organization and click "remove selected". This cannot be undone, but it is easy to add it again if you remember the numbers.

Section 4: Firebase

This is the URL for managing Firebase: https://console.firebase.google.com/ Here you can add/remove SpikeBoarders, change data, check analytics, etc. I don't think you'll ever have to use this section. We tried to make it so you didn't have to. But things change, so we want you to be able to control everything if you need to.

Adding/Removing Users: Click on "Authentication" under Develop on the left side. Click "add user" to add a user. To remove a user, hover over their user info and click the three dots that appear on the right. Then click "delete".

Changing data: You might realize that some data is wrong, or you need to change a picture, logo, etc. Click on "Database" on the left under Develop. This will show you the active database. Anything you change here will immediately affect all applications. Click the black plus to view an org's stats, logos, etc. Click the x to delete stuff, plus to add, and click on the data to change it. If you delete an individual piece of data, it will likely break a bunch of stuff.