FAST AND FURRYOUS:

A Mobile Application to Assist in Locating Pets

Michelle T. Ho
Massachusetts Academy of Math and Science

Table of Contents

Acknowledgements

The author would like to thank Dr. Florian Michahelles, corresponding author of "Improving the Design of Track and Trace Products: Evidence from a Field Study on Pet Tracking Devices," for additional information about the study (see Appendix C). The author would also like to thank Ms. Angela Taricco for her guidance in debugging and using Firebase and Android Studio and for providing the ASUS Nexus 7 tablets used for testing. The author would finally like to thank Mrs. Julia Wildfong for being so helpful in STEM advising and making sure there were minimal mental breakdowns, and Ms. Curran for guiding this STEM project, assisting in every stage with her prior knowledge to make the project the best it could be.

Abstract

Of the 600,000 pets left behind during the 2005 Hurricane Katrina evacuations, only 15,000 were rescued. The recent Texas hurricanes also caused mass evacuations, and emergency personnel used a combination of two applications to locate, rescue, and return missing pets. Many pets are not tracked because existing pet trackers are expensive and inaccurate. Because pets are valued family members, an alternative solution is needed. Fast and Furryous is an Android mobile application designed to facilitate communication between pet owners and finders. It allows users to report a missing pet, send necessary information to others within a certain radius, and display the last known pet location on a map. Once a pet is located, the finder sends a "found pet" report and directly contacts the owner. Information is stored and retrieved from a database via user authentication. Market analysis was used to populate a weighted scoring matrix, which compared the app with competitive pet tracking devices. The app met pet owner expectations and compared favorably with the competition. In the future, the app can be used as a cheaper, more suitable pet tracking alternative.

Literature Review

The Problem Being Addressed

Everyday Life: Losing Pets

Evidence suggests that pets have been loyal companions to humans for at least 16,000 years. (Ault, 2016). Today, Americans own approximately 89.7 million dogs in a total of 60.2 million households, 94.2 million cats in a total of 47.1 million households, and 20.3 million birds in a total of 7.9 million households (APPA, 2017). In a survey conducted by the ASPCA (American Society for the Prevention of Cruelty to Animals), 15 percent of pet owners reported a missing pet in the last five years. Approximately 93 percent of dogs and 75 percent of cats reported missing were found, but only 15 percent of them were found because of microchip tracking technology or identification tags (Moore, 2012). Most people consider their pets to be a valued member of the family. Pet owners increase their chances of finding their beloved furry friend with the use of current technology. However, existing pet-finding devices can be improved to increase the likelihood of finding lost pets.

Losing Pets during Natural Disasters

In the event of a natural disaster, pet owners can be forced to evacuate quickly, and leave their pets behind. One example was Hurricane Katrina in 2005, which caused 1.5 million people to evacuate, leaving behind 600,000 pets. Only 15,000 of those were rescued (Dilonardo, 2017). Recently, Hurricane Harvey struck Texas, forcing 30,000 to evacuate. Thousands of pets were left behind, and some were trapped in flood waters. Many lacked tracking devices like microchips, making them difficult to locate. Rescuers used a combination of two mobile applications to find the pets. One app was called 6, which served as a walkie talkie between rescuers and owners, and the other was Google Maps, which helped rescuers locate the pets’ reported locations (Thompson, 2017). Although this system worked, it was inefficient, and could have been improved.

Existing Tracking Technology

The emergence of human and pet tracking technology has increased within the last decade. Examples of track and trace technology available today include mobile applications and websites that use global positioning system (GPS), implanted chips, and geofencing. Many consumers are still dissatisfied with the existing technology (Von Watzdorf, 2010).

Tracking Technology with Human Studies

Before GPS was used to track pets, it was used on human subjects for a variety of field studies. However, today GPS can be used by the public to determine a user's location, give directions, and track a user's movement. For example, the state of Wisconsin tracked sex offenders with GPS sensors to determine if they had violated their restraining orders. However, GPS often sent the police false alerts, sending law abiding ex-convicts back to jail. The system would also fail to track offenders in areas with poor weather, large buildings, and tall vegetation. Additionally, the program was incredibly expensive for its substandard performance. The state officials were displeased with this tracking technology (Hall, 2013). GPS was also used in a pedestrian tracking study which sought to use pedestrian walking patterns to determine the best layout for traffic reduction in a new city. The main reason that the researchers used GPS technology was because GPS is effective in determining orientation, collecting navigational data, and communicating that data back to the researchers. However, some of the collected data was inaccurate because the device was misused by the participant or lost signal during testing. That data was filtered out, and the remaining data was used to construct maps of traffic flows (Spek, 2014).

Another form of location tracking technology, geofencing, uses the Global Navigation Satellite System (GNSS) to set virtual boundaries where the user pleases. When a device leaves its boundaries, a notification is sent to an app or website to alert the user. The same event occurs when the device reenters its boundaries, and when the device moves within the boundary. The device typically has a GNSS chip; the satellites track the position of the chip. However, geofencing has its flaws. A quality chip can be incredibly expensive. Its battery life decreases greatly when in use, and, since the satellites are in space, its notification relay time can incorrectly report the device's position and speed. Geofencing technology must be tested thoroughly to ensure its accuracy before it can be integrated into an app (Fattepur, 2016).

Tracking Technology in Pet Studies

As human tracking technology has improved, marketers have expanded the technology for use with pets. GPS collars and microchip implementation have become increasingly but, they still have some flaws. A patent from 2004 described a collar for dogs which had a GPS device that sent latitude and longitude coordinates back to the owner's remote control. The collar also had a movable video camera so that the owner could see where the pet had gone, and a recording device that could play a recorded message to the person that found the pet. However, the device provides no way for the owner to contact the finder, depends on the user's knowledge of the landscape, and has a low battery life (Edwards, 2004).

Table 1: Comparison of existing pet tracking technology*

Criteria	Home Again Micro Chip	Loc8tor Pet Finder	Tractive GPS Pet Tracker	POD 2	Pixie	Tile	Gibi	Nuzzle	Garmin Astro Radio Tracker
Tracks a pet accurately in real time	No	No	Yes	Yes	No	No	Yes	Yes (can be wrong)	Yes
Alerts others in the area of lost pet	No	No	No	No	No	No	No	No	No
Allows others to tell owner when pet is found	Yes	No	No	No	No	No	No	No	No
Battery Life	N/A	Good 10+ days	Bad 2-5 days	Bad 6 days	Good 12+ month	Good 1 year	Bad 1 day	Bad 8 hrs	Good 10+ days
Lag time	N/A	High	Low (2-3 s)	Low	High	Low	Low	Low	Low (2.5 s)
Shows a map	No	No	Yes	Yes	No	Yes	Yes	No	Yes
Uses mobile application	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No
Can be used without GPS	Yes	No	No	No	No	No	No	No	No
Size (dimensions in mm)	12 x 2 x 2	30.5 x 19.5 x 8.5	50.8 x 40.64 x 15.24	22.9 x 50.8 x 0.5	76.2 x 114.3 x 45.72	33 x 33 x 5.1	76.2 x 203.2x 152.4	15.2 x 101.6 x 22.9	160 x 35.56 x 61.0
(g)	0.029	5	35	27.2	113.4	5.7	226.8	144.6	272.2
Cost	$20/ year	$89.95	$120.28 + $5/ month	$198.5	$30	$20	$130	$190	$650
Data is user friendly	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
For different kinds of pets	Dogs	All	Cats and dogs	All	All	All	All	Dogs	Dogs
Tracking range	N/A	120 m	Where cell service is	Where cell service is	45.7 m	100 m	Where cell service is	Where cell service is	14484 m
Shows user current location	No	No	No	No	No	No	No	No	Yes
User can enter location	No	No	No	No	No	No	No	No	No
Friendly user interface	N/A	N/A	Yes	Yes	Yes	Yes	Yes	Yes	Yes

information collected from Amazon.com

Table 1 compares nine of the top existing pet tracking technologies with twenty different criteria that customers would like to see in pet tracking technology. The devices were selected using a general internet search that provided a list of the best ranked pet tracking technologies on the market. People had ranked them the most helpful devices in 2017 for finding their lost pets. Then, a search was conducted to determine if the device was still on the market and whether customer reviews were available. This reduced the list from thirteen devices to nine. The criteria were chosen based on a study that found what people wanted in a pet tracking device. The data was collected by looking at the product summaries for each device and reading both positive and negative customer reviews. Based on the chart, it is evident that generally the devices are expensive; seven of the devices cost more than $100. The devices can also be large; four have a dimension greater than 100 mm. The devices also cannot alert others in the area of a lost pet and have small tracking ranges. Customers were generally dissatisfied with the difficult installation process, the lag time from when the owner refreshes the device location to when the owner receives the device location, the short battery life, the long charging process, and the excessive use of cellular data (Amazon, 2017).

The devices presented in the table are a sample of the various types of tracking devices available on the consumer market. General small tracking devices can be helpful to track material items like keys, phones, or wallets. However, the alert radius for such devices is restrained, which can be unhelpful if a pet travels far from home (Perez, 2015). Microchips are inserted into a pet through an injection, so no batteries are required. Still, microchips cannot track a pet in real time and only become useful to the owner if their pet is found by another person. GPS collars are commonly used to track pets in real time, but their drawbacks include expense, occasional location inaccuracy, and low battery life (Why, 2015).

Additionally, a study conducted on existing pet tracking technology determined that marketers struggle to make pet trackers that are of a reasonable weight, are precise and accurate, and have a prolonged battery life. The study aimed to determine what the current issues with pet tracking technology are and identify the most important aspects of an optimal solution to. After deciding on two devices to test, the researchers surveyed human participants to identify the qualities best executed by the devices. According to the surveys, fast charging, long battery life, a small weight/size, water resistance, accurate localization, trace functions, chase functions, tracking duration, and connection to a mobile app were the most important features for an optimal pet tracker to have. An organized display, buttons and web access were nice add-ons and an alarm function and SMS messaging alerts were unnecessary features. Although the study determined what features would be included in an optimal solution, researchers did not design or manufacture an optimal solution (Von Watzdorf, 2010).

Programming a Solution

With the rise in mobile technology, marketers have expanded device usage to be compatible with smartphones because of their availability and easy functionality. Many tracking devices display data to a user using a mobile application (Von Watzdorf, 2010).

Android Studio

One mobile application platform used by app developers is Android Studio. Approximately 650,000 mobile applications are available on the Android market out of the over 1 million in existence (Joorabchi, 2013). Apps made with this platform can be run on devices with the Android operating system, which make up 45% of the smartphone market. Android Studio is incredibly user friendly and offers many features to help code an app such as a user interface creator, a better memory monitor, sample code for beginners to manipulate, a debugger, and a built-in translator. It utilizes Google SDKs (software development kits) and developers write code with xml and java. A developer can install emulators for various Android devices to test the app within Android Studio or the code can be downloaded to a physical Android device. With Android Studio, it is possible to integrate Google Maps and user to user communication, features essential to a pet locator application (How to Developers Android, 2015).

Map Integration

One of Android Studio's built in features is an activity for Google Maps. Google Maps is available in the Google play services SDK which must be installed. This activity has several functions for developers to integrate into their apps. Examples of these functions include allowing users to enter specific addresses and displaying them on a map or identifying their current location on a map. Several basic features are required for this activity to function in an app: a user interface to allow user interaction, buttons to trigger events, a database to store user data, and labels to display text onscreen (How to Developers: Using Google, 2016). When the developer tests the activity on a device or emulator, they can view a map of a specified location and nearby roads and landmarks (Maps, 2017).

Firebase

Android Studio is compatible with another mobile platform called Firebase. Linking a mobile application project to Firebase provides the app with more functionality. For instance, Firebase offers a real-time database for developers to integrate into their apps. The database stores user data and allows the user, and others with permission, to access that data with a login. Users can store images and audio files in the database as well. The developer can put restrictions on how that data is released. Firebase also can integrate user to user communication in an application. This platform can also provide the developer with tools to measure the app's performance. Firebase includes a test lab to run the app on many Android devices, a crash report SDK, performance SDK. All provide insight into the latencies that users experience when running the app (Developers: Let's, 2017).

Testing

Quality testing must occur before app developers can release a product to the market. The app must satisfy customers by performing the desired task accurately and efficiently. How performance is quantified varies for every unique app. Developers generally outline the goals for their app then determine metrics to measure whether those goals were met by the app. For instance, certain apps need to connect to networks quickly to run, accurately track location with GPS, or use a portion of the device's battery. Goals for these metrics would be set and tested. Developers also survey the target demographic to acquire feedback on the target's general expectations for a particular app (AppInsight, 2012). Common app metrics used by developers include versatility across multiple devices and operating systems, quick app launch and load times, minimum lag times, alert range accuracy, low battery usage, and usability, including low crash and virus instances (Joorabchi, 2013).

Typically, developers have multiple means of testing these metrics. For example, to improve user experience, the app can simply be run through a crash report and debugger to identify and target errors. The app is also run on multiple devices of varying screen sizes and operating systems. Firebase provides services with a crash report and multiple device testing (Let's, 2017). Apps can also be run on emulators, which are virtual versions of the targeted device, to see if the app works on a simple device. For the full functionality experience, developers will allow the targeted demographic to test their app and gain feedback on its strengths and weaknesses (Joorabchi, 2013).

Mobile App Performance Metrics

It can be difficult for developers to determine app performance metrics because each app has unique goals and functions. In a study conducted in 2012, a group of app developers documented how they determined their performance metrics. They identified the possible issues future customers could face from their mobile applications. Examples of problems included network connectivity problems, variances in smart phone hard ware and GPS instruments, and changes in API behavior due to battery charge and platform changes. According to the study, although there are existing support tools such as analytics frameworks, which collect data on usage, and crash reports available, these tools did not help developers trace app performance once the app was released on the market. Developers targeted their main concerns with a unique system designed to measure their specific predetermined metrics (AppInsight, 2012).

One example of these app performance metric principles in action is a study conducted in 2013 that described a mobile application which recommended apps to users based recent app usage. To determine the algorithm's effectiveness, developers needed a measurable metric. They found it necessary to test the algorithm several times and record the probability that the recommended app and the user's preferred app would be used in the same situation. Not only did they record numerical values, they judged the algorithm's accuracy to match similar apps through logic and reasoning by qualitative observations (Wang, 2013).

According to research done in 2013, some of the main challenges for app developers include a need for more effective analysis tools to determine their app metrics. Current popular methods include distributing surveys to potential users, beta testing to monitor performance, security testing to monitor the protection of user data, and emulator testing to try the app on a virtual device (Joorabchi, 2013).

Engineering Plan

Phrase 1

Over 100 million households in the United States have pets. Approximately 15 percent of pets go missing annually, and only 15 percent of those are found through microchip tracking and identification tags. An alternative option is needed.

Phrase 2

The goal of the application was to help people locate lost pets by alerting nearby users of missing pets and by facilitating direct communication between owners and the people who find their lost pets.

Overview of Methodology

The app was coded using Android Studio, Google SDKs, OneSignal, and Firebase tools. The user interface was coded using XML and Java. Buttons were coded that triggered other activities. These other activities were maps, a submission for a lost pet report, a submission for a found pet report, create an account, messaging, and settings. Firebase capabilities were added to the create an account and report activities to store user information and allow others with permission to access that information. Google Maps was added to the app to get user location and display it on a map. Then, automated push notifications were coded which sent alerts to users within a certain radius of the person who submitted the original report. Then, messaging was added which allowed pet finders to contact pet owners about a missing pet. Throughout this process, video tutorials were consulted for assistance and every time a new function was added, the app was tested with an emulator or tablet. The app was tested by documenting loading times and alert sending accuracy, which were then compared to past versions and existing competitors.

Methodology and Materials

Materials and Programs

The mobile application was coded on a Dell laptop with a Microsoft Windows 10 pro operating system. Android Studio was used to code the app with Java and XML. The Google Maps API (Application Programming Interface), provided in Android Studio, was integrated into the app to identify location on a map, which was necessary for the app's functionality. Firebase, a mobile application development, was used for its authentication, database, storage, cloud messaging, and crash report features. OneSignal, a multiplatform push notification service, was used to send automatic push notifications between devices. Both Firebase and OneSignal were accessible via the internet and were integrated into the app with code. Two ASUS Nexus tablets were obtained from Mass Academy for testing purposes.

Design Criteria

Table 2: Decision Matrix to determine optimal solution

Criteria	Weight	Build Device	Program an app	Total Device	Total Program
Low cost	3	3	5	9	15
Low time	5	2	3	10	15
Many resources available	3	3	4	9	12
User friendly	5	3	5	15	25
			Total	43	67

The project needed to accomplish the goal of facilitating direct communication with pet owners and finders to more efficiently return lost pets. Two solutions were considered. A mobile application for smartphones could be developed or a hardware tracking device could be created. The decision matrix above was created to choose the optimal solution, based on the criteria of low cost, low time, many available resources, and user friendliness. The optimal solution ended up being a mobile application. It was coded with Android Studio because it is user friendly and able to target over 50% of smartphone users.

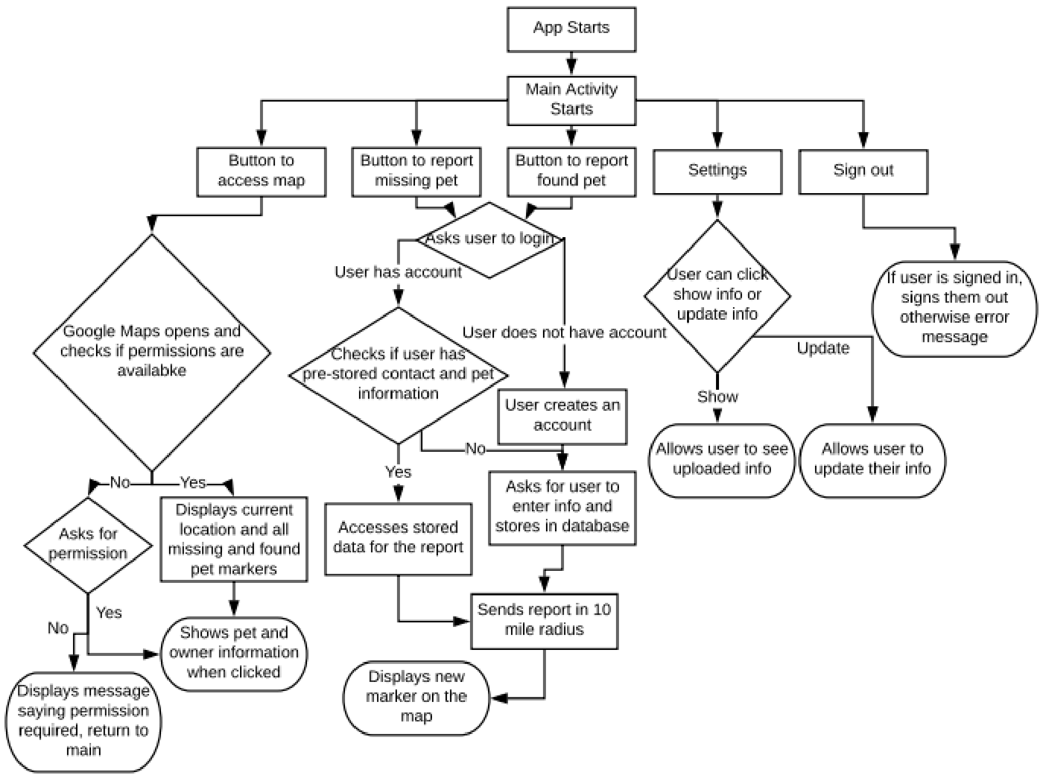
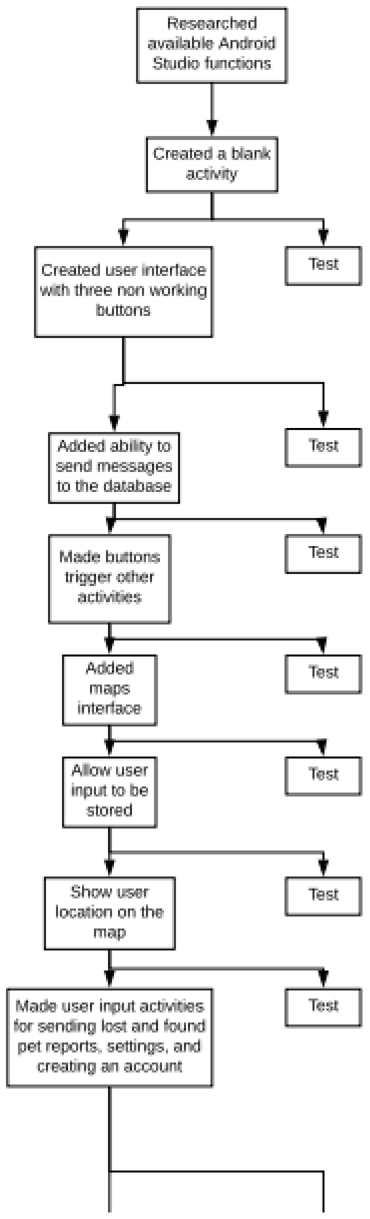

App Starts
Main Activity Starts
Button to access map
Button to report missing pet
Button to report found pet
Settings
Sign out
Asks user to login
User has account
User does not have account
Google Maps opens and checks if permissions are availabke
Checks if user has pre-stored contact and pet information
User creates an account
User can click show info or update info
If user is signed in, signs them out otherwise error message
Update
Show
Allows user to see uploaded info
Allows user to update their info
No
Yes
Asks for permission
Displays current location and all missing and found pet markers
Accesses stored data for the report
Asks for user to enter info and stores in database
Sends report in 10 mile radius
Shows pet and owner information when clicked
Displays message saying permission required, return to main
Displays new marker on the map

Figure 1: Flowchart depicting how the app was projected to run at the start of the project

The app needed to accomplish the specific goals of allowing a user to report a missing pet, storing that information in a database, sending an alert of the missing pet to others in the area, and allowing user to user communication, once the pet is found. To accomplish these goals, the app must run the process depicted in Figure 1. It is projected to start with the main home screen with five buttons. The first button shows Google Maps and checks if user has allowed the app location permissions. If so, a map is displayed with the user's current location and markers of lost and found pets. The second and third button on the main screen perform essentially the same functions, although they open different activities, but one is for reporting lost pets and the other is for found pets. They both ask the user to login and if they do not have an account, they make one. Then, if the database does not have prestored information for the user, the user is asked to upload information. A report is created, sent in a ten-mile radius, and a marker is created on the map. The fourth button triggers the settings activity which can allow a user to change the prestored information in the database or show that information. Lastly, the fifth button can allow a signed in user to sign out.

App Development


Researched available Android Studio functions
Created a blank activity
Created user interface with three non working buttons
Test
Added ability to send messages to the database
Test
Made buttons trigger other activities
Test
Added maps interface
Test
Allow user input to be stored
Test
Show user location on the map
Test
Made user input activities for sending lost and found pet reports, settings, and creating an account
Test

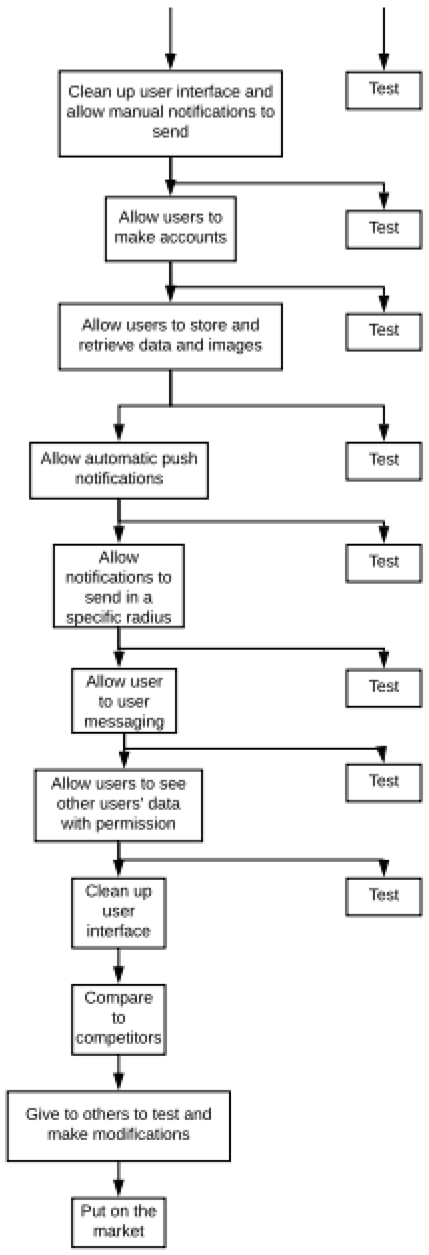

Clean up user interface and allow manual notifications to send
Test
Allow users to make accounts
Test
Allow users to store and retrieve data and images
Test
Allow automatic push notifications
Test
Allow notifications to send in a specific radius
Test
Allow user to user messaging
Test
Allow users to see other users' data with permission
Test
Clean up user interface
Test
Compare to competitors
Give to others to test and make modifications
Put on the market

Figure 2: Flowchart which shows the procedure to code the app

(note: flowchart layout is subject to change)

Figure 2 shows the procedure for coding the app. To examine the entire code for this mobile application, please refer to Appendix D. (Please note that when all my code is all done, I will place it all in the appendix for reference). Every time a new function was added, it was called a new app version, e.g. versions 1, 2, and 3, all had a new function. First, Android Studio was researched for its capabilities and functions. A blank activity was created and run on an emulator to test its ability to launch. Then, the main activity screen was coded with three non-functioning buttons using XML for the layout. Firebase database dependencies were added to the application to allow for the app to send information to the database. Methods with intents to start new activities were added to buttons to allow them to trigger a new screen when clicked. The Google Maps API was added to the app and set up with a unique app code. Another activity was coded to show the Google Maps interface upon the click of a button from the main screen. Code was added to allow users to input information and store their own data in the database. The Maps interface was programmed to ask for user location permissions then display current location on a map. Four more activities were made which could be triggered from the main screen: Lost Pet Report, Found Pet Report, Create an Account, and Settings. The first three activities could take user input and store it in a database and show the user the data entered. The main activity screen was updated, and Firebase Cloud Messaging dependencies were added to the app to allow the developer to send notifications to the app from the Firebase console. The app was updated with Firebase Authentication dependencies, so that new code could be added to allow users to make accounts. A new java class with a layout allowed users to pick images from the photo gallery and strings of text to store in the database under their specific user. Then, they could retrieve the data with the click of a button using a combination of two java classes with two XML layouts. After, OneSignal notification capabilities were added to the code that assigned specific tags to each user, so notifications could be sent to specific users on the click of a button, which checked if the recipient was within a certain radius to the sender. Using Firebase Cloud Messaging and OneSignal, direct messaging from device to device was coded. The user interface was once again updated, the app versions were compared to the competitors, and a small sample population was given the app, to see if the app is user friendly, and if they could use the app's functions. After, the app was slightly modified and will be placed on the Google Play Store.

Testing and Comparing

To test the app's functionality, every time a new function was coded, the app was run with either the emulator or an ASUS Nexus tablet, which is referred to as unit testing. Therefore, whenever a problem occurred, it was evident that the cause was a result of the new function. Generally, the websites of YouTube, Google Developers, Firebase, Android Studio, and OneSignal were consulted to find the methods and their syntax to be incorporated into the application. They also provided bug fixes for common coding errors. When the app crashed, the cause of the crash was logged on the console, and the code was fixed based on the error message produced.

Additionally, the app was compared to nine competitor devices, as shown in table 1, which were chosen from an article outlining the best pet tracking devices of 2017. The criteria chosen for comparison were adapted from a 2012 study on pet tracking devices. The study suggested that the best pet locators should be ideally small in size and weight, accurate in tracking, inexpensive, have large tracking range distances, short battery life, and low lag times, and be compatible with a mobile. (Von Watzdorf, 2010). Other criteria were considered that were necessary to locate a pet, such as communication from owner and finder, alerting others in the area of a lost pet, shows user current location, and allows a user to enter location. Furthermore, criteria that would make user experience overall more enjoyable with the product were also used in the comparison process, including having a mobile application that shows a map to the user, displays user friendly data, can be used on different pets, and has an easy-to-use interface. Data for eight competitor pet locating devices were gathered for these characteristics, as shown in Table 1, and the criteria was weighted based on what the pet owners deemed most necessary for such a device, as outlined by the study. The same data was gathered for all app versions and scored using the same system.

Results

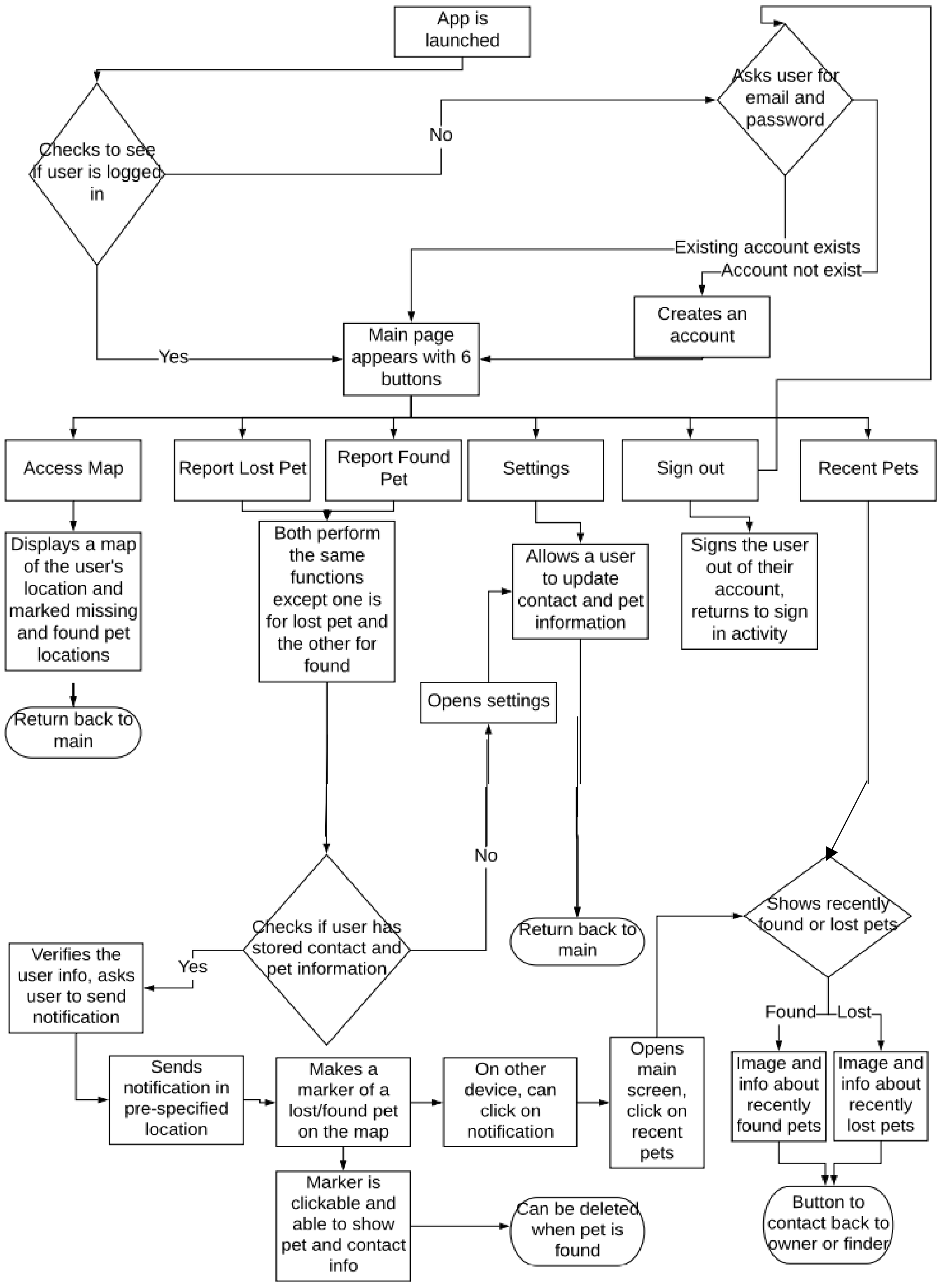
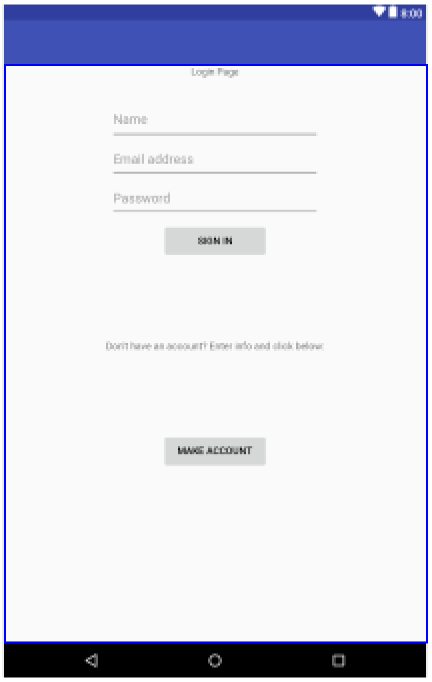
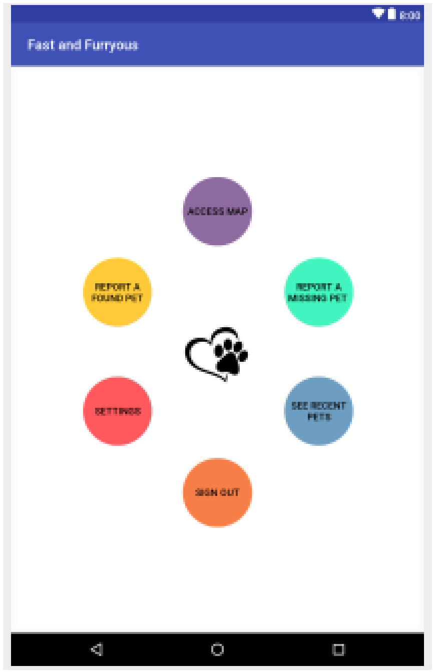

App is launched
Checks to see if user is logged in
No
Asks user for email and password
Existing account exists
Account not exist
Creates an account
Yes
Main page appears with 6 buttons
Access Map
Report Lost Pet
Report Found Pet
Settings
Sign out
Recent Pets
Displays a map of the user's location and marked missing and found pet locations
Return back to main
Both perform the same functions except one is for lost pet and the other for found
Allows a user to update contact and pet information
Signs the user out of their account, returns to sign in activity
Opens settings
No
Checks if user has stored contact and pet information
Yes
Return back to main
Shows recently found or lost pets
Found
Lost
Verifies the user info, asks user to send notification
Sends notification in pre-specified location
Makes a marker of a lost/found pet on the map
On other device, can click on notification
Opens main screen, click on recent pets
Image and info about recently found pets
Image and info about recently lost pets
Marker is clickable and able to show pet and contact info
Can be deleted when pet is found
Button to contact back to owner or finder

Figure 3: A flowchart depicting the app's progression

The app, named Fast and Furryous, was able to successfully perform its main features: storing user information with accounts, sending notifications to other devices in specified areas, marking the pet's last known location on a map, and allowing pet finders to contact pet owners. Figure 3 shows how the app runs upon launch, accurate of the latest app version. Upon launch, the app checked to see if the user was logged in. If the user was logged in, the Main Activity opened. If not, the Sign-In Activity opened (see Figure 4), which asked the user for an email and password to log in with. If the user did not have an account, the activity created one. The user was then signed in and the Main Activity opened.





Figures 4 and 5: The Sign-In Activity is shown on the left and the Main Activity is shown on the right. See Appendix E for other Activity Layouts.

As shown in Figure 5, the Main Activity had 5 buttons, which trigger different functions. The Access Map Button triggered the Map Activity, which showed a map with the user's current location and previously marked lost and found pet locations. The markers could be clicked to display pet and owner contact information and could be deleted by the marker's original creator. The Report a Found Pet and Report a Missing Pet buttons triggered essentially the same functions to send an alert out, except one was for found pets and the other was for lost pets. Both buttons triggered an activity that asked the user to verify their stored pet and contact information. Then a notification of the lost/found pet was sent to people in a prespecified radius of 10 miles and a marker for the pet's last known location was placed on the map. Selecting the notification on a receiving device initiates an activity which displays the pet and contact information. The activity had an option to send a message back to the sender to either alert the user that they had located the original sender's pet, or that the other user's pet had been found. The Recent Pets button shows another activity with a choice of seeing recently found or lost pets. The user can select either to see recent reports others have made, which include pet images, information, and contact. Users can then contact the owner or finder of a pet from that activity. The Settings button triggered an activity that allowed users to update pet and contact information for sending reports. The Sign Out button signed the user out of the account and returned them to the Sign-In Activity, as seen in Figure 4. Each function was tested throughout the entire process and performed as intended by the latest app version.

Conclusions

Table 2: A legend and scoring matrix which compares eight competitor devices alongside the final version of Fast and Furryous with weighted criteria.

A	B	C	D	E	F	G	H	I	F&F
Home Again Microchip	Loc8tor Pet Finder	Tractive GPS Pet Tracker	POD 2	Pixie	Tile	Gibi	Nuzzle	Garmin Astro Radio Tracker	Fast and Furryous

Criteria	Weight	A	B	C	D	E	F	G	H	I	F&F
Tracks a pet accurately in real time	4	1	1	5	5	1	1	5	3	5	1
Alerts others in the area of lost pet	4	1	1	1	1	1	1	1	1	1	5
Allows others to tell owner when pet is found	4	5	1	1	1	1	1	1	1	1	5
Long battery life	5	5	5	1	1	5	5	1	1	5	3
Low lag time	5	1	1	5	5	1	5	5	5	5	3
Shows a map	4	1	1	5	5	1	5	5	1	5	5
Uses mobile application	5	1	1	5	5	5	5	5	5	1	5
Can be used without GPS	4	5	1	1	1	1	1	1	1	1	5
Small size	5	5	4	2	4	1	4	1	1	1	5
Low cost	5	5	1	1	1	5	5	1	1	1	5
Data is user friendly	5	5	5	5	5	5	5	5	5	5	5
For different kinds of pets	2	3	5	3	5	5	5	5	3	3	5
Large tracking Range	5	1	1	5	5	1	1	5	5	4	5
Shows user current location	3	1	1	1	1	1	1	1	1	5	5
User can enter location	3	1	1	1	1	1	1	1	1	1	5
Friendly user interface	5	1	1	5	5	5	5	5	5	5	5
	Total	184	131	209	223	176	227	208	180	211	304

Table 3: A legend and scoring matrix which compares the fourteen previous app versions with weighted criteria.

1	2	3	4	5	6	7	8	9	10	11	12	13	14
Blank activity	Non-working buttons	Working buttons	User input	maps	Get user location	Message others pt 1	login	Sign out	Store images in data-base	Retrieve data	Message others pt 2	Report sending	Clean user interface + debug

Criteria	Weight	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Tracks a pet accurately in real time	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Alerts others in the area of lost pet	4	1	1	1	1	1	1	1	1	1	1	1	1	5	1
Allows others to tell owner when pet is found	4	1	1	1	1	1	1	5	5	5	5	5	1	1	1
Long battery life	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3
Low lag time	5	1	5	5	1	1	5	5	5	5	3	3	3	3	3
Shows a map	4	1	1	1	5	5	5	5	5	5	5	5	5	5	1
Uses mobile application	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Can be used without GPS	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Small size	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Low cost	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Data is user friendly	5	1	1	1	5	5	5	5	5	5	5	1	1	1	1
For different kinds of pets	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5

Large tracking Range	5	1	1	1	1	1	1	1	5	1	1	1	1	5	1
Shows user current location	3	1	1	1	1	1	5	5	5	5	5	5	5	5	1
User can enter location	3	1	1	1	1	1	1	1	1	1	1	1	1	1	5
Friendly user interface	5	1	1	1	1	1	5	1	5	5	5	5	1	1	5
	Total	172	192	192	208	208	260	256	296	276	266	246	196	232	200

According to tables 2 and 3, the device effectively addressed phrase 1 and helped solve the issue by meeting the goals set by phrase 2. Tables 2 and 3 depict scoring matrices with weighted criteria. The letters in table 2 represent the nine competitor tracking devices, and the numbers in table 3 represent the versions of the mobile application. The criteria were chosen based on a survey conducted in a study by Swiss professor, Dr. S. Von Watzdorf in 2012. A random sample of pet owners were asked about their complaints about current pet tracking devices. Those complaints included short battery life, high lag time, does not show data in a mobile application, not user friendly, large in size, small tracking range, and expensive cost. These complaints were addressed as the highest priorities for the mobile application, so the criteria of long battery life, low lag time, uses mobile application, friendly user interface, small in size, large tracking range, and low cost were weighted the highest at a 5. For more explanation about how the rest of the criteria were weighted, and how the app versions and competitors were scored, please see Appendix F.

The competitors were scored using raw data gathered from customer reviews on Amazon.com whereas the app versions were scored based on observations during testing for each app version. Table 3 shows that the first six app versions increased in score because more features were added that met the criteria on the matrix. However, the scores decreased with later versions due to more complex features being added which often caused crashes, high battery usage, and high lag times. Still, as shown by table 2, the final version of Fast and Furryous had the highest total score of 304 when compared to the nine competitors. Therefore, the application met pet owner expectations and compared favorably with the competition.

Resources

Ault, A. (2016, September 28). Ask Smithsonian: When Did People Start Keeping Pets? Retrieved from https://www.smithsonianmag.com/smithsonian-institution/ask-smithsonian-when-did-people-start-keeping-pets-180960616/

APPA: Pet Industry Market Size & Ownership Statistics (2017). Retrieved from http://www.americanpetproducts.org/press_industrytrends.asp

Developers - Let's Try: Power Your Mobile Applications with Firebase (2017, Aug 1). *Open Source FOR you.* Retrieved from libraries.state.ma. us/login?gwurl=http://go.gale group.com/ps/i.do?p=ITOF&sw=w&u=mlin_c_worpoly&id=GALE|A499749668&v=2.1&it=r&sid=summon&password=boushuperionos&ugroup=outside

Dilonardo, M. J. (2017, September 6). What's Next for the Lost and Rescued Animals of Hurricane Harvey? Retrieved from https://www.mnn.com/family/pets/stories/whats-next-lost-and-rescued-animals-hurricane-harvey

Edwards, M. (2004). US Patent No. US6720879B2.Washington DC: Google Patents.

Fattepur, M. B., S, S. G., & Huttangoudar, J. B. (2016, June 16). A Solution to Improve the Performance of Geofence Enabled GNSS Chipset. Retrieved September 18, 2017, from http://ieeexplore.ieee.org.ezproxy.wpi.edu/stamp/stamp.jsp?arnumber=7779405

Hall, D. J. (2013). *Lawmakers raise questions about GPS tracking program for sex offenders.* Retrieved from https://search-proquest- com.ezproxy.wpi.edu/docview/1323894851/abstract/628A2BB BBE36421BPQ/1?accountid=29120

How to Developers: Android Studio: A Platform for Android Development. (2015, Oct 1). *Open Source FOR You.* 1 Oct. 2015. Retrieved from libraries.state.ma.us/login?gwurl=http://go.galegroup. com/ps/i.do?p=ITOF&sw=w&u=mlin_c_worpoly&v=2.1&id=GALE%7CA430814302&it=r&asid=b e6d232c7ef2f97d127c7d19f056dfd0. Accessed 8 Oct. 2017.

How to Developers: Using Google Maps in App Inventor 2. (2016, May 1). *Open Source FOR You.* Retrieved from http://libraries.state.ma.us/Login?gwurl=http://go.galegroup.com/ps/i.do

Joorabchi, M. E., Mesbah, A., & Krutchten, P. (2013, October 10-11). Real Challenges in Mobile App Development. 15-24. Retrieved from DOI: 10.1109/ESEM.2013.9.

Lau, A., Tabay, O., Yuen, I. & Tarle, M. (2005). US Patent No. US20050108692A1.Washington DC: Google Patents.

Maps Android API. (2017, July 12). Google Map APIs. Retrieved from https://developers.google.com/maps/documentation/android-api/start

Moore, A (2012, July 11). Lost Pet Statistics: Survey Looks At Likelihood Of Finding A Missing Dog Or Cat Retrieved from https://www.huffingtonpost.com/2012/07/11/lost-pet-statistics-survey-dog-cat_n_1662860.html

Perez, S. (2015). *Lost-Item Tracker Tile Rolls Out a New Renewal Service Offering Early Adopters Discounted Replacements.* Retrieved from https://techcrunch.com/2015/07/22/lost-item-tracker-tile-rolls-out-a-renewal-service/

Ravindranath, L. et al. (2012, October 8). AppInsight: Mobile App Performance Monitoring in the Wild. 108-120. Retrieved from DOI: 10.1145/347057.347416

Spek, S. V., VanSchaick, J., Bois, P. D., & Haan, R. D. (2014). Sensing Human Activity: GPS Tracking. MDPI AG, 9(4), 3033-3055. Retrieved from DOI: 10.3390/s90403033

Thompson, J. (2017, September 1). Beverly Woman Uses App to Help Animal Rescuers in Texas. Retrieved from http://whdh.com/news/beverly-woman-uses-app-to-help-animal-rescuers-in-texas/

Von Watzdorf, S., & Michahelles, F. (2010, June 7-9). Improving the Design of Track and *Trace Products: Evidence from a Field Study on Pet Tracking Devices. 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing.* 177-180. Retrieved from http://ieeexplore.ieee.org.ezproxy.wpi.edu/document/5504688/

Wang, F., Zhang, Z., & Sun, H. (2013, November 7-10). A Cooperation Based Metric for Mobile Applications Recommendation. 13-16. Retrieved from DOI: 10.1109/WI-IAT.2013.142

Appendix

Appendix A: Limitations and Assumptions

One limitation for this project was that external funding was unavailable, which limited testing to one type of device, the ASUS Nexus 7 tablet, and the virtual emulator. The app development platform was also a limiting factor because the app was coded on Android Studio on a laptop running Windows, both of which cannot code apps for iOS. Another limitation was the time constraints, which limited the number of features that the app could include. The size of the files being uploaded to the database had to be limited because the database can only support strings of 10 MB or less and pictures of 1KB or less. The app was not given to a sample population for testing, so it is unclear how users will interact with the app.

It was assumed that all sources cited for this project are accurate. The competing technologies were assumed to be representative of the top pet tracking devices available on the market. The ASUS Nexus 7 tablets were assumed to perform as advertised. The app was assumed to continue to run in the same manner in the future as it did during testing. The Firebase, OneSignal, and Google API tools were assumed to perform functions as advertised.

Appendix B: Project Notes

Knowledge Gaps

Knowledge Gap	Resolved By	Information Location
What programs do I need to create an app?	Research	Developers - Let's Try: Power Your Mobile Applications with Firebase How to Developers: Android Studio: A Platform for Android Development

How do I integrate maps into my app?	Research	How to Developers: Using Google Maps in App Inventor 2
Statistics: On average, how far and where do pets travel when they go missing?		
Would creating an app that just has a map be easier/ be a better preliminary design than adding GPS, or can they be done simultaneously?	Creating an app first would be easier Research	How to Developers: Using Google Maps in App Inventor 2
What software allows me to store data?	Research, talking to people	Developers - Let's Try: Power Your Mobile Applications with Firebase
Would cellular data or Bluetooth be the best way to connect the phone to GPS?	Research, cellular data	Sensing Human Activity: GPS Tracking
Statistics: How often do pets go missing/ run away, depending on region?		
Statistics: How many people have smartphones and/or pets?		
How do I add a Firebase database into an Android studio created app?	Research	On Firebase's website under performance monitoring section

Literature Search Parameters

These searches were performed between September 12, 2017 and December 13, 2017.

Database/ Search Engine	**Keywords**	**Date**
WPI Summons	Tile Tracker, GPS Tracker, GPS Tracking Program, GPS, GPS program	September 12, 2017
Amazon	Pet GPS tracker	September 23, 2017
Google Patents	Pet tracker	September 30, 2017
WPI Summons	Android studio, firebase, basics of app development	10/8/17
Google patents	Pet app, pet mobile application, dog mobile application,	10/8/17
WPI Summons	Geofencing	10/16/17
WPI Summons	Map, google map, app	10/17/17
Google search	Android studio, google maps	10/17/17
YouTube	Android studio map integration	10/18/17
Google, Google scholar	App performance metrics	10/19/17
Google Search	Firebase	10/23/17

Sources

Source 1: Tracker Tile

Source Title	Lost-Item Tracker Tile Rolls Out A Renewal Service Offering Early Adopters Discounted Replacements
Citation	Perez, S. (2015). *Lost-Item Tracker Tile Rolls Out a New Renewal Service Offering Early Adopters Discounted Replacements.* Retrieved from https://techcrunch.com/2015/07/22/lost-item-tracker-tile-rolls-out-a-renewal-service/
Source Found By	WPI Summons
Source Type	News Article
Keywords	Tile Tracker
Summary	The small Tile tracker has done well during sales in the past few years. However, the battery does not recharge so it needs to be replaced often which can be expensive. Tile is not the only tracking device in existence. It uses Bluetooth and an app to locate an item that the Tile tracker is attached to, like keys or a wallet in a certain Bluetooth range. People have now determined how to locate lost items out of range, by working together with their apps. The battery only lasts a year and cannot be replaced. The company's reasoning for this design is they believe that people will not bother to buy a new battery, so they can do more with the design without including a changeable battery. Now the company is implementing a program for replacing used up Tiles, for lower prices. Tile is also getting a hardware upgrade, hopefully that helps make them recyclable, so they do not add to landfills.
Reason for Interest	This is a preexisting device, which can give ideas for a new app to find lost pets, since this helps people find their lost items. Perhaps the project can implement GPS, unlike this device.
Notes	This is just one example of a device that exists to help people find lost things. It uses Bluetooth and an app. It emits a sound when the user needs to find their lost item. What happens if the user is out of range or cannot hear the sound? The app cannot tell the user exactly where the item is since there is no GPS. The device is also easily broken.
Question	How was this device created? How does one implement Bluetooth in an app? How does Bluetooth even work? How did they program the app so that it was compatible with the device? What sensors or wiring does the device use? How can the device be improved so it is more sustainable and does not have to be replaced so often? How can a device be created that has infinite range?

Source 2: Lawmakers use GPS Tracking

Source Title	Lawmakers raise questions about GPS tracking program for sex offenders
Citation	Hall, D. J. (2013). *Lawmakers raise questions about GPS tracking program for sex offenders.* Retrieved from https://search-proquest-com.ezproxy.wpi.edu/docview/1323894851/abstract/628A2BBBBE36421BPQ/1?accountid=29120 ProQuest document ID 1323894851
Source Found By	WPI summons
Source Type	Journal Article
Keywords	GPS, GPS tracking program
Summary	There is a new state law that requires sex offenders or people who violated their restraining orders to wear ankle bracelets with GPS tracking. However, there were concerns expressed whether the GPS system could effectively track 600 sex offenders without setting off false alarms. At the current moment, the system is only being used for sex offenders. The state of Wisconsin plans on expanding the program to include people who violate domestic abuse restraining orders, but the cost will sky rocket. Data shows that the system has failed on multiple occasions, leaving offenders in jail for an unnecessary amount of time. The state is considering whether the program is worth all the money, with all the false alarms, even though tracking is cheaper than prison. The system also is unreliable in places with bad weather, large buildings, and areas with many trees. The alert is sent from the bracelet to an operator.
Reason for Interest	The article describes another method used to track, specifically now living subjects. Ideas from this program could be implemented into the developing STEM project.
Notes	An anklet with a GPS monitor is worn by an offender, and when the person is out of range, a signal is sent to a group of operators who either try and get in contact with the offender or call the police if that is not possible.
Question	Why did the system fail and say that offenders were out of range when they were not? Can the GPS anklet give an exact location? What program were the operators using to receive the signal? What wiring is in the GPS anklet?

Source 3: Sensing Human Activity: GPS Tracking

Source Title	Sensing Human Activity: GPS Tracking
Citation	Spek, S. V., VanSchaick, J., Bois, P. D., & Haan, R. D. (2014). Sensing Human Activity: GPS Tracking. MDPI AG, 9(4), 3033-3055. Retrieved from DOI: 10.3390/s90403033
Source Found By	Wpi Summons
Source Type	Journal Article
Keywords	GPS Tracking
Summary	GPS has been increasingly used for transportation research, where it has been effective for tracking pedestrian activity. described a study about tracking city traffic. goal was to find a method to improve traffic in urban city areas and to apply that method to new cities. Cities face problems of overcrowding- traffic because cities places of interest like business centers, malls, and landmarks. To determine how existing traffic acts, use GPS technology (new tech) GPS = orientation, navigation, and communication data visualization, such as maps and tables, help researchers to identify traffic patterns easily. collect temporal data as well. 1300+ pedestrians carry GPS devices in the major cities of Norwich, Rouen, Koblenz, for 7 days per location. The experiment needed many participants to get enough usable data, flyers and people were approached face to face interviewed to determine the city's most popular public spaces, also making note of the interviewee's gender, age, frequency of visits to the city, and route taken in the city, and average time and money spent in the city. new city of Almere in the Netherlands, where families carried GPS devices 24/7 for a week. Researchers aimed to determine the patterns of movement, to see how a new city could improve its layout based on current data. GARMIN MAP60Cx GPS receivers, recorded spatial and temporal data in the form of track logs. data converted into a series of gpx files and shapes files, then linked to the ArcGIS platform. participants required to take a questionnaire that determined whether they properly used the GPS device. The data from the questionnaires was put into the SPSS database then linked to the ArcGIS platform, with the track log data . From there, the data was analyzed for density analysis of the times spent at certain locations and the number of people at a location at a certain time.

	The data was also converted into visual representations including illustrated temporal maps, animated temporal maps, and a conclusion map, which was based on the patterns researchers found in the data. The maps made it easy to see what areas had the most pedestrian traffic. For the experiment involving the three older cities, the study was able to identify certain patterns in pedestrian activity, based on the social and demographic qualities. The article did not provide any further explanation for exactly what these patterns were, only saying that the conclusions from the study would be used to improve cities for pedestrians. For the Almere experiment, the article described how new towns are planned, based on social and economic patterns. The study found it difficult to track pedestrian activity of the families, since cars were the main mode of transportation. Still, there were maps provided showing the movement of families in Almere. Using the pedestrian density analysis from the first experiment and the visual representations of traffic flow from both experiments, the researchers determined a method to improve pedestrian traffic, although the method was not explicitly stated in the article. From this study, it was determined that GPS technology was incredibly useful for tracking pedestrian technology and should be used for transportation science research in the future.
Reason for Interest	GPS tracking may be integrated into my project
Notes	Not helpful for my project
Question	How was the data collected in this study used to improve new city development?

Source 4: Improving Pet Tracking Devices

Source Title	Improving the Design of Track and Trace Products: Evidence from a Field Study on Pet Tracking Devices
Citation	Von Watzdorf, S., & Michahelles, F. (2010, June 7-9). Improving the Design of Track and Trace Products: Evidence from a Field Study on Pet Tracking Devices. 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing. 177-180. Retrieved from http://ieeexplore.ieee.org.ezproxy.wpi.edu/document/5504688/ **DOI:** 10.1109/SUTC.2010.69
Source Found By	Wpi Summons
Source Type	Conference on June 7-9, 2010
Keywords	GPS, Pet tracking
Summary	• Providers struggle with device size, weight, precision requirements, battery life • App is more important than a website or remote control • 20,000 cats and dogs are lost each year in Switzerland • Can be used for elderly and children and items not just pets • Existing tracking technology includes GPS, Cell-ID, UMTS- and RFID • Goals: evaluate existing technology and create guidelines for future technology • Created a survey for people to complete after testing the different devices • Google searched providers and chose 11 that actually still sold the product • Those were split into groups of GSM tech vs VHF • GSM has infinite range, needs network infrastructure • VHF has limit of a few hundred meters but no need infrastructure • Chose GSM products: 2 products • Looked at charging time, display, battery life, water resistance, easy to use, not weight the pet down • 3 functionalities: SMS text, web, mobile app notifications • All complained about battery life (20-48 hours) and charging time (3 to 5 hours) • Weight, size, handling were sufficient • Size only suitable for dogs, not smaller animals • SMS text was unnecessary because it could not provide an accurate description for rural, unfamiliar areas • Web not always have service to app is best • Location preciseness could be improved

	• Optimal solution to have a battery of 2 weeks+, lighter small device, waterproof, uses an app, accuracy between 10-50 meters
Reason for Interest	This study analyzed what customers look for the most and consider most important in GPS tracking pets technology which will help me determine what criteria I should focus on more when creating my app.
Notes	They made a decision matrix, and this can be my justification for why I weighted some criteria higher than others.
Question	Was the accuracy of the devices quantified? Were there any extensions done? Like following up on criteria to make a more optimal solution or testing out more existing technology? GSM vs VHF? What was on the test scripts?

Source 5: Patent for an Animal Tracking Collar

Source Title	Animal collar including tracking and location device
Citation	Edwards, M. (2004). US Patent No. US6720879B2.Washington DC: Google Patents.
Source Found By	Google Patents
Source Type	Patent
Keywords	Pet tracker, GPS
Summary	• Animal collar with GPS • Has a video camera to see what the pet sees • Wireless communication • Legal status expired • The recorder has a pre-recorded message of the owner giving a phone number • Also has a telephone like device that calls the owner when a person finds the pet and pushes the button • GPS gives longitude and latitude and sends it back to a remote-control display • Camera controlled by a remote control • Camera mounted to the collar and can rotate • Can go into a sleep state to avoid wasting too much battery • Not an app • Conveys information about the immediate environment to find the pet based on landmarks • Can also be used for rescue personnel or zoologists studying animals in the wild • One remote can go to many collars and track many pets • Not water proof • Covered in plastic, reflective, glow in the dark material • Rechargeable • Only gives coordinates, not location on a map • Not just a collar: also harnesses, muzzles, saddles, etc.
Reason for Interest	This gives insight on existing technology and I read it to see how I can make my project better based on what already exists.
Notes	This device has a video camera which is novel. The collar allows communication between the owner and finder, which my app should help do.
Question	Is it ideal to record personal info and let anyone with access to your pet have access to that information?

Source 6: Android Studio Introduction

Source Title	How to Developers: Android Studio: A Platform for Android Development
Citation	How to Developers: Android Studio: A Platform for Android Development. (2015, Oct 1). *Open Source FOR You.* 1 Oct. 2015. Retrieved from libraries.state.ma.us/login?gwurl=http://go.galegroup.com/ps/i.do?p=ITOF&sw=w&u=mlin_c_worpoly&v=2.1&id=GALE%7CA430814302&it=r&asid=be6d232c7ef2f97d127c7d19f056dfd0. Accessed 8 Oct. 2017.
Source Found By	Wpi Summons
Source Type	Journal Article
Keywords	Android Studio, Java, App development
Summary	• Android studio easily helps create apps • Offered by Google, utilizes Java • Android shares 45% of the smartphone market • Android is open source and easy to learn • Cross platform integrated development environment (IDE) • Eclipse was used before Android studio, but replaced because unstable • Studio has gradle build environment, short cuts, improved user interface creator, better memory monitor, translator, faster speed • Runs from menu and command line, tracks memory and monitors memory use, supports google cloud, can integrate cloud messaging, and app engine, has inline debugging and performance analysis tools • Emulators for Nexus • Must have 4GB of RAM on laptop • SDK manager and update required tools for app building • How to set up proxy settings • Has samples of code to experiment with for newbies • Template wizards preset to manage projects • Layout editor to easily make a user interface by drag and drop and can be previewed with different screen orientations • Preview of app shows • Can implement fingerprint authentication • Can be run on an android device or an emulator with a USB cable • Debug, build, compiles code • Newbies should learn shortcuts to save time
Reason for Interest	Need to use Android studio to develop my app
Notes	• Easy to use, just learn the basics • Learn Java, available for windows • Tells you how to install and what to do after installing
Question	What are the necessary SDK tools needed for app development?

Source 7: Firebase Introduction

Source Title	Developers - Let's Try: Power Your Mobile Applications with Firebase
Citation	Developers - Let's Try: Power Your Mobile Applications with Firebase (2017, Aug 1). *Open Source FOR you.* Retrieved from libraries.state,ma. us/login?gwurl=http://go.gale group.com/ps/i.do?p=ITOF&sw =w&u=mlin_c_worpoly&id=GALE|A499749668&v=2.1&it=r&sid= summon&password=boushuperionos&ugroup=outside
Source Found By	WPI Summons
Source Type	Journal article
Keywords	Firebase, app, mobile app
Summary	• Affiliated with Google • Has many services to offer • It became more used when acquired by google • Can develop apps, test them, distribution, analytics • Many services under those areas and Google explains when to use them • Database provides SDKs on many platforms and easily integrates functionality • Allows use of cloud storage • Allows collaboration across devices • Performance monitoring such as network, latency, memory, app start up time, etc. before setting out the app and having users complain • Test lab lets you test the app on multiple devices • Cloud lets one upload media like files, videos, images, audio • Allows adding authentication into the app • Notifications • Search features for the app
Reason for Interest	Need to use Firebase in app development for global cloud
Notes	• Need cloud storage to upload images • “Say you are developing an imaging mobile app that allows people to upload their pictures. As part of the final image, you might need to generate thumbnails of the images. Instead of doing all this as part of the mobile app and making the user wait, you can simply upload the image into cloud storage. Once the file is uploaded, it can kick off a cloud function that will process the image data and generate multiple thumbnails. All this is done in the background and the user need not wait for it.” •
Question	How do I upload images to the cloud? Why do I need both Android studio and firebase? How do I put them both together?

Source 8: Patent for an App Program

Source Title	System and method of generating applications for mobile devices
Citation	Lau, A., Tabay, O., Yuen, I. & Tarle, M. (2005). US Patent No. US20050108692A1.Washington DC: Google Patents.
Source Found By	Google Patents
Source Type	Patent
Keywords	Mobile devices, app development
Summary	• Applies to apps coded in Java • The method unpacks references apps in class files • Transforms the reference apps into the optimal app by a device plug-in based on a combination between the reference mobile device and the optimal device • XML files and software code that modifies the first code to fit the optimal device • Uses bytecode during transformation • Popularity of mobile devices and the types increase • Customized with apps like games, mail, contact management written for a certain device and need them to be cross platform type apps and run just as well • Sometimes they do not because they have different operating systems or device constraints like display sizes or different libraries, even when both devices use the same code • Several methods to convert apps from one device to another exist by running the code through a compiler for the target device • But it takes too long and needs human intervention and there needs to be different codes for every device in existence • Another method is interpreting the code and rewriting it for the target device, but the interpreter, an emulator, must be downloaded on every device which is not possible for small mobile devices • The invention is a java app that executes the code on the first and the second device with a plug in • Sample of the code of the invention
Reason for Interest	I need to develop an app, so I wanted to see how others did it.
Notes	• This would be useful if the code already existed and would need to be run on a different platform, does not help me make the first code
Questions	How was this application created? With what programs? How can I use it?

Source 9: Geofencing Improvement

Source Title	A Solution to Improve the Performance of Geofence Enabled GNSS Chipset.
Citation	Fattepur, M. B., S, S. G., & Huttangoudar, J. B. (2016, June 16). A Solution to Improve the Performance of Geofence Enabled GNSS Chipset. Retrieved September 18, 2017, from http://ieeexplore.ieee.org.ezproxy.wpi.edu/stamp/stamp.jsp?arnumber=7779405 **DOI:** 10.1109/CSITSS.2016.7779405
Source Found By	Given to me by Vishnu Penubarthi
Source Type	Journal article
Keywords	Geofencing, GNSS
Summary	• Global Navigation Satellite System (GNSS) improves redundancy and availability of info from satellites • Geofencing tracks moving objects within set boundaries • It also sends notifications as to when something leaves or enters the boundaries • The GNSS chipset has geofencing capabilities • Issues with that include incorrectly saying an object has left or entered the boundaries, heavily affects battery life of the device, decreases accuracy of the object's speed and position • The paper aims to explain a study that addresses the above issues • The solution is in the form of a matrix • The solution assumes that the user is moving with varying signal strengths • Many apps use geofencing, which allows the user to decide on an area to be enclosed by a geofence and in that area, the app determines the user's location and reports specific events in the area • There are many shapes of Geofences and multiple events can be triggered by the same GNSS receiver for different fences • The GNSS system determines geographical location of the receiver • The chipset is tested for signal to noise ratio, which determine signal strength • The following aspects are used to determine performance of the GNSS chip: • 1. Dwell time is the minimum time the use is in or out of the fence before the chip tells the app of this event • 2. Notification latency is the minimum time which the event has, to be triggered to the application • 3. Unknown timer is the minimum time limit after which the uncertain event should be reported to the app • 4. Fix interval is the reporting interval between the chip and the app

	• Because geofencing is used in many apps in everyday life, it is important to make sure the accuracy and speed of the chip are efficient • Fences can be circular, polygonal, or elliptical shaped • Proposed solution = interface with different statuses like unknown, uncertain, entering, inside, leaving, outside to determine user position relative to the fence • Rigorous checks of current a previous status of user with respect to the fence and reports it back to the app • Gives an algorithm with pseudocode in Java • Notifications • Performance evaluated with configured, fully charged device with the app • Two fences circles of 100m and 200m • Geofencing apps can be difficult and must be tested thoroughly for accuracy b/c predicted and obtained results differ
Reason for Interest	To see if this is a technique I could implement into my own app to see when pets leave their set boundaries
Notes	• Could use geofencing to say when a pet goes missing • Pet must be wearing a device though • But no way to track the pet after it leaves the boundary
Questions	Could I use geofencing to send an alert to people within a certain range when a pet goes missing? Do I need a sensor for that?

Source 10: Integrating Maps Into an App

Source Title	How to Developers: Using Google Maps in App Inventor 2
Citation	How to Developers: Using Google Maps in App Inventor 2. (2016, May 1). *Open Source FOR You.* Retrieved from http://libraries.state.ma.us/ Login?gwurl=http://go.galegroup.com/ps/i.do
Source Found By	Wpi summons
Source Type	Journal article
Keywords	Google maps, app
Summary	• App inventor 2 can create android apps • Android has a built-in feature for google maps • It is reliable to anyone new in the area • This study focuses on an application that would help a user locate a car parked in an unfamiliar area • The app will show you the route to get to your care • GUI requirements (graphical user interface) • The GUI helps the user interact with the onscreen features • The response of the features is defined in the block editor section • Components needed: • Activity starter: triggers processes outside of the app (need to launch the map app within the app) • Label: displays headings and markings on the screen • Button: triggers an event • Horizontal arrangement (self-explanatory) • Tiny DB (saves address in database to be accessed later, so TinyDB keeps data persistently so the data will be there when you close and reopen the app) • Location sensor: provides geographical position using coordinate system • To make the app, drag and drop those components (refer to figure for layout) • By pressing the Location_Save_Butoon, it saves coordinates of location to the tiny database and overrides any previous stored location • Show_Directions_Butoon shows directions between 2 points • Must set data URI of the starter activity • Download the app to your computer and connect to phone via Bluetooth or push to test
Reason for Interest	I need to integrate maps into my pet finder app
Notes	This article also gave beginning code to use I can copy and paste in.
Question	Is there an equivalent to TinyDB in android studio or do I have to figure out how to import firebase?

Source 11: Integrating Google maps Into Android Studio

Source Title	Maps Android API: Getting Started
Citation	Maps Android API. (2017, July 12). Google Map APIs. Retrieved from https://developers.google.com/maps/documentation/android-api/start
Source Found By	Google search
Source Type	Web article
Keywords	Google maps, app, android studio
Summary	• To get android studio to implement google maps you must: • Download android studio • Install the google play services software development kit (SDK) • Create a google maps project by: • Starting android studio • Creating a new project, enter app name, company domain, project location • Select form factors (phone and tablet) • Select google maps activity in the 'Add an activity to Mobile' • Enter the activity name, layout name and title (default ones work) • Android studio then starts gradle and build the project • When the build is finished, java file and xml file open • The xml file has directions for getting a google maps API key before trying to run the app • To get a google maps API key, (needed to access google maps servers) (restricted for android apps) • Use the link in the xml file and put into a browser which takes you to the Google API console and gives the required info • Follow instructions to make a project on or select a new project for google API console • Create an API key, copy it, go back to android studio and paste into the string element in the xml file • Then examine the code supplied by the template (the xml file and the java file) and copy the code in this article if it does not appear to look like this • To test the app, connect to an android device (must enable develop options on the device) • Or use an emulator
Reason for Interest	I need to integrate maps into my pet finder app
Notes	• Follow these instructions to make sure you can get google maps in app • Try over October break
Question	How can I add geofencing to this map feature?

Source 12: Metric Research on How Apps Perform in the Real World

Source Title	AppInsight: Mobile App Performance Monitoring in the Wild
Citation	Ravindranath, L. et al. (2012, October 8). AppInsight: Mobile App Performance Monitoring in the Wild. 108-120. Retrieved from DOI: 10.1145/347057.347416
Source Found By	Google Scholar
Source Type	Journal Article
Keywords	Mobile App Performance
Summary	• Developers need data about how well app is doing in real world • Asynchronous, multi-threaded nature of mobile apps makes tracing difficult • AppInsight, a system that instruments mobile app binaries to automatically identify the critical path in user transactions • Need to get this data to improve the app • Network connectivity (wi-fi or cell data), GPS, phone hardware vary • Some platform APIs change behavior depending on battery level • Difficult to produce all these conditions in a lab • Little support today for tracing app performance in the field • Can report crash logs (not identify performance problems) • Flurry and Preemptive (analytics frameworks) collect usage analytics rather than performance data • Need to write custom trace code then • AppInsight help diagnose problems • No extra code needed in app, or changes to OS or runtime • Tested on Windows Phone • For user transactions, determined root cause • “The graph contains five types of nodes, namely: (M) User Manipulation, (S) Upcall start, (E) Upcall end, (A) sync call start, and (L) Layout updated.” • The system was able to tell a developer that the cause of crashing in his app was due to the URL where the line was fetched
Reason for Interest	Need to determine how to measure app performance
Notes	• This paper told me about a system that tells you what is wrong, but not necessarily the data that was retrieved from the app that was measurable
Question	What data did they specifically collect from the app to determine where the error occurred?

Source 13: Real Challenges in Mobile App Development

Source Title	Real Challenges in Mobile App Development
Citation	Joorabchi, M. E., Mesbah, A., & Krutchten, P. (2013, October 10-11). Real Challenges in Mobile App Development. 15-24. Retrieved from DOI: 10.1109/ESEM.2013.9. http://ieeexplore.ieee.org.ezproxy.wpi.edu/xpls/icp.jsp?arnumber=6681334
Source Found By	Google Scholar
Source Type	Journal Article
Keywords	App performance
Summary	• Goal of study was to gain understanding of what developers face • Qualitative human study • Hard to develop across multiple platforms, lack of robust monitoring, analysis, testing tools, slow emulators or miss major device features • 800,000 mobile apps on the AppStore • 650,000 on the Android Market • Mobile apps = 3 categories: native (run on device's operating system and are required to be adapted for other devices, web based (require web browser) and hybrid (mix of the two) • Developers like native apps because they can use the device's specific features (i.e. camera, sensors) • Goal was to understand how native apps are developed and challenges that occur • Interviewed app developers from iOS, Android, windows, blackberry • Developers need better analysis tools to determine metrics • Grounded theory approach: used when the goal is to learn how people react in difficult situations, gaining popularity in software engineering research • Asked in a survey what is missing and what is redundant • Android is an open source • Difficult to store a lot of data on the device and syncing offline to online sometimes does not work • Beta testing is used to determine performance • Automated UI testing, unit testing, integration testing, system testing, regression testing, GUI testing, performance testing, usability testing, security testing—what data is gathered from these tests? What numbers? • Use emulators but lack real mobile device features • Most developers test their mobile apps manually. • Unit testing is not common within the mobile community and current testing frameworks do not provide the same level of support for different platforms.

	• testing tools are weak and unreliable and do not support important features for mobile testing such as mobility (e.g., changing network connectivity), location services, sensors, or different gestures and inputs.
Reason for Interest	I read this thinking that the testing section would give me some examples of data that has been collected on apps during testing
Notes	• several different tests were mentioned, might be worthwhile to research some and see if they return any number data
Question	What number data is gathered from the various types of tests?

Source 14: Recommender System Mobile App Metric

Source Title	A Cooperation Based Metric for Mobile Applications Recommendation
Citation	Wang, F., Zhang, Z., & Sun, H. (2013, November 7-10). A Cooperation Based Metric for Mobile Applications Recommendation. 13-16. Retrieved from DOI: 10.1109/WI-IAT.2013.142 http://ieeexplore.ieee.org.ezproxy.wpi.edu/xpls/icp.jsp?arnumber=6690685
Source Found By	Google Scholar
Source Type	Journal Article
Keywords	Metric mobile application
Summary	• hard for users to find an app that meets their needs • researches done to design and implement a system that recommends apps • need to evaluate an app's effectiveness • Existing metrics include MAE, precision, recall, Fl, P@N, • Most recommender systems fail to evaluate the recommendation algorithms • Need to determine apps with a correlation • Used the Jaccard Similarity coefficient to determine a correlation relation • Deployed the algorithm and saw it was more accurate • They determined it was more accurate by reasoning, not numbers • The metric they used only applies to relationships between apps • Propose a new user centered metric that judges the quality of recommended results • The metric's meaning is the probability that the pair of apps is used in the same situation
Reason for Interest	I needed to read on metrics people have used to quantify app performance.
Notes	The paper discussed metrics used that I should probably read more up on to see if they can apply to my app.
Question	What number data did their metric return? How can they prove their app worked without displaying the numbers that prove it so?

Source 15: Microchip Tracking Vs. GPS Tracking

Source Title	Why a microchip can't track my dog in real time and what to use instead
Citation	Why a microchip can't track my dog in real time and what to use instead (2015, October 2015). *Tractive.* Retrieved from https://tractive.com/blog/en/tech/microchip-for-dogs-and-tractive-gps.
Source Found By	Google Search
Source Type	Website article
Keywords	Microchip, track
Summary	• Microchips are ineffective because they only are useful once the pet is found by someone who takes them to a vet • They do not track the pet's real-time location • A microchip is about the size of a grain of rice
Reason for Interest	I need to determine why microchip tracking is not the most effective method for pet tracking.
Notes	• Does not give owner information to finder • No one might know a pet has gone missing in the area
Question	How does the finder know that a pet is microchipped and should take them to the vet?

Source 16: History of Pets

Source Title	When Did People Start Keeping Pets?
Citation	Ault, A. (2016, September 28). Ask Smithsonian: When Did People Start Keeping Pets? Retrieved from https://www.smithsonianmag.com/smithsonian-institution/ask-smithsonian-when-did-people-start-keeping-pets-180960616/
Source Found By	Google search
Source Type	Article
Keywords	Pets
Summary	• Evidence of pets has been found from 16,000 years ago • Pets have always been a companion to humans • People have always cared about their pets; there is evidence of burying them
Reason for Interest	I needed perspective on how important pets are to human beings and for how long.
Notes	• Pets are family to humans, and it is important to help someone find their lost family member.
Question	Do people care more about their pets today than in ancient times or vice versa?

Source 17: Pet Statistics

Source Title	Pet Industry Market Size & Ownership Statistics
Citation	APPA: Pet Industry Market Size & Ownership Statistics (2017). Retrieved from http://www.americanpetproducts.org/press_industrytrends.asp
Source Found By	Google search
Source Type	Statistics
Keywords	Pet ownership
Summary	• In 2017, people in the US have spent 69.36 billion dollars on pets • 7.9 million households that own birds, 47.1 million households that own cats, and 60.2 million households that own dogs in the US • There are 20.3 million birds, 94.2 million cats, and 89.7 million dogs owned as pets in the US
Reason for Interest	Needed statistics on how many pets are owned in the United States
Notes	• There are over 100 million households that own pets in the United States.
Question	How many of these households own more than one pet?

Source 18: Lost Pet Statistics

Source Title	Lost Pet Statistics: Survey Looks at Likelihood of Finding A Missing Dog or Cat
Citation	Moore, A (2012, July 11). Lost Pet Statistics: Survey Looks at Likelihood of Finding A Missing Dog or Cat Retrieved from https://www.huffington post.com/ 2012/07/11/lost-pet-statistics-survey-dog-cat_n_1662860.html
Source Found By	Google Search
Source Type	News Article
Keywords	Lost pet statistics
Summary	• 15% of pets went missing in the last year • 14 percent missing were dogs and 15 percent were cats. • 93 percent of lost dogs and 75 percent of lost cats were reported returned home. • 6 percent of dog owners and 2 percent of cat owners found their pets at animal shelters. • 15 percent of dogs were found because of microchip tracking and identification tags

Reason for Interest	Need missing pet statistics to provide a scope for the problem.
Notes	• 15 percent out of 100 million total pets go missing • Only 15 percent were found from tracking, not a great percentage
Question	How many pets are GPS tracked?

Source 19: Pet locating during Hurricane Harvey

Source Title	Beverly Woman Uses App to Help Animal Rescuers in Texas.
Citation	Thompson, J. (2017, September 1). Beverly Woman Uses App to Help Animal Rescuers in Texas. Retrieved from http://whdh.com/news/beverly-woman-uses-app-to-help-animal-rescuers-in-texas/
Source Found By	News search
Source Type	News article
Keywords	Missing pets, app, hurricane
Summary	• During Hurricane Harvey, people had to evacuate quickly, leaving pets behind • They wanted to find them again, so they got in touch with rescuers in Texas • It took a combination of two apps: Google Maps and Zello, for communication for the rescuers to locate the pets
Reason for Interest	Real world application for missing pet problem on a much larger scope
Notes	• This two-app combination method could be combined into 1 app to locate pets, so it is not as inefficient
Question	Were most of these pets not GPS tracked? How is Zello communication programmed?

Appendix C: Expert Email

From: Michahelles, Florian (CT RDA NEC WOS-US) <florian.michahelles@siemens.com>
Sent: Saturday, October 12, 2017 13:35 PM
To: Ho, Michelle T [mailto:mtho@wpi.edu]

Subject: Questions regarding pet tracking devices

Hi Michelle,

Thanks for your interest:

- The studies has been conducted quite some years ago. Back then we simply included all device that were available on the market back then, thus we didn't really include customer ratings.

- The choice of devices was indeed more qualitatively based on own experience and testing. We tested the devices and compared to some benchmark measurements, GPS in phone or separate GPS device back then.
- No, we didn't really do a follow up. I'm sure technology has advanced a lot after the study has been conducted.

Best

Florian

From: Ho, Michelle T [mailto:mtho@wpi.edu]
Sent: Saturday, October 07, 2017 4:28 PM
To: Michahelles, Florian (CT RDA NEC WOS-US) <florian.michahelles@siemens.com>
Subject: Questions regarding pet tracking devices

Dear Dr. Michahelles,

Hello, my name is Michelle Ho and I am a high school student at the Massachusetts Academy of Math and Science at the Worcester Polytechnic Institute. I was conducting background research for my intensive five-month science fair project, when I came across your paper, "Improving the Design of Track and Trace Products: Evidence from a Field Study on Pet Tracking Devices." I had acquired a lot of information from the reading, such as the other applications for pet GPS device, the differences between GMS and VHF devices, and what people found most important to a pet tracking device. The paper was written well, and in such a way that it was quite clear and easy to comprehend with the background information given. Your description of data prioritization was particularly helpful. If you have the time in your busy schedule, I had a few questions as well about the paper. For instance, when you chose the two devices to use, both GSM devices, what was the full process you underwent to narrow your choices to the two devices? Did customer ratings or usability play a role in the decision along with novelty and availability? Also, how were you able to distinguish what devices used GSM versus VHF radio technology? You also mentioned that the devices was not entirely accurate, so how were you able to measure the accuracy and provide a range of accuracy in distance for the devices? Lastly, were there any follow up studies done on other devices to see if they met the criteria for an optimal solution, or even an extension that created the optimal pet tracking device? Thank you so much for your time and any help.

Kind regards,

Michelle Ho

Massachusetts Academy of Math and Science mtho@wpi.edu

Appendix D: Final Version of Application Code

Manifest File

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.miche.fastandfurryous">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <!--
         The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
         Google Maps Android API v2, but you must specify either coarse or fine
         location permissions for the 'MyLocation' functionality.
    -->
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <!-- To auto-complete the email text field in the login form with the user's
emails -->
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.READ_PROFILE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />

    <application
        android:name=".MyApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <!--
             The API key for Google Maps-based APIs is defined as a string resource.
             (See the file "res/values/google_maps_api.xml").
             Note that the API key is linked to the encryption key used to sign the
APK.
             You need a different API key for each encryption key, including the
release key that is used to
             sign the APK for publishing.
             You can define the keys for the debug and release targets in src/debug/
and src/release/.
        -->
        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="@string/google_maps_key" />
        <!--
            -<meta-data
            android:name="com.onesignal.NotificationOpened.DEFAULT"
            android:value="DISABLE"/>
        -->

        <activity
            android:name=".CreateAccount"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
```

```xml
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:launchMode="singleTop" />
        <activity
            android:name=".Maps"
            android:label="@string/app_name" />
        <activity android:name=".UserLocation" />
        <activity
            android:name=".Found"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity android:name=".Lost" />
        <activity
            android:name=".LoginActivity"
            android:label="@string/app_name" />
        <activity
            android:name=".SignIn"
            android:label="@string/app_name" />

        <service android:name=".MyFirebaseInstanceIDService">
            <intent-filter>
                <action android:name="com.google.firebase.INSTANCE_ID_EVENT" />
            </intent-filter>
        </service>
        <!--
        - <service
            android:name=".MyFirebaseMessagingService"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="com.google.firebase.MESSAGING_EVENT" />
            </intent-filter>
        </service>
        -->

        <activity
            android:name=".SettingsActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity android:name=".UploadInfo" />
        <activity android:name=".ShowData" />
        <activity android:name=".ShowFoundData" />
        <activity android:name=".ShowLostData" />
        <activity android:name=".ShowFoundDataNotificationOpen" />
        <activity android:name=".ShowLostDataNotificationOpen" />
        <activity android:name=".Messaging" />
    </application>

</manifest>
```

Java

For Restarting Application if Crashes

```java
package com.example.miche.fastandfurryous;

import android.app.Application;
import android.content.Context;

public class Crash extends Application {

    public static Crash instance;

    @Override
    public void onCreate() {
        super.onCreate();
        instance = this;
    }

    @Override
    public Context getApplicationContext() {
        return super.getApplicationContext();
    }

    public static Crash getInstance() {
        return instance;
    }
}
```

Sign in: Launcher Activity Code

```java
package com.example.miche.fastandfurryous;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import android.view.View;
import android.Manifest;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseAuthException;
import com.google.firebase.auth.FirebaseAuthInvalidCredentialsException;
import com.google.firebase.auth.FirebaseAuthUserCollisionException;
import com.google.firebase.auth.FirebaseAuthWeakPasswordException;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.UserProfileChangeRequest;
import com.onesignal.OneSignal;

public class CreateAccount extends AppCompatActivity {
    private FirebaseAuth mAuth;
    String name1, email;
    String name2, password;
    String name3, name;
    EditText nameInput, emailInput, passwordInput;
    Button makeaccountbut, signinbut;
    private static final String TAG = "TESTING";
    FirebaseUser user;
    static String LoggedIn_User_Email;
    public int openNotification;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        Intent mIntent = getIntent();
        int intValue = mIntent.getIntExtra("intVariableName", openNotification);

        if(getIntent().hasExtra("lost")) {  //do I need to clear this hasExtra?
            Intent showLost = new Intent (CreateAccount.this,
ShowLostDataNotificationOpen.class);
            startActivity(showLost);
        }
        else if (getIntent().hasExtra("found"))
        {
            Intent showFound = new Intent (CreateAccount.this,
ShowFoundDataNotificationOpen.class);
            startActivity(showFound);
        }

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_account);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        mAuth = FirebaseAuth.getInstance();

        nameInput = (EditText) findViewById(R.id.name);
        emailInput = (EditText) findViewById(R.id.createemail);
        passwordInput = (EditText) findViewById(R.id.createpassword);

        makeaccountbut = (Button) findViewById(R.id.makeaccountbut);
        signinbut = (Button) findViewById(R.id.signinbut);

        // Call syncHashedEmail anywhere in your app if you have the user's email.
        // This improves the effectiveness of OneSignal's "best-time" notification
scheduling feature.
        // OneSignal.syncHashedEmail(userEmail);

        //String Loggedin_user_email = mAuth.getEmail();
        //OneSignal.sendTag("User_ID",Loggedin_user_email);

        //check if user is already logged in
        if(mAuth.getCurrentUser()!= null)
        {
            //user is logged in, goes to home screen
            finish();
            Intent Home = new Intent (CreateAccount.this,MainActivity.class);
            startActivity(Home);
        }

        OneSignal.startInit(this)
                .inFocusDisplaying(OneSignal.OSInFocusDisplayOption.Notification)
                .unsubscribeWhenNotificationsAreDisabled(true)
                .setNotificationOpenedHandler(new ExampleNotificationOpenedHandler())
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler())
                .init();
       // Toast.makeText(CreateAccount.this, "MAKE SURE YOU ARE CONNECTED TO WIFI!!!",
         //       Toast.LENGTH_SHORT).show();

        /*signinbut.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                email = emailInput.getText().toString().trim();
                password = passwordInput.getText().toString().trim();
                callsignin(email,password);

            }
        });

        makeaccountbut.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                email = emailInput.getText().toString().trim();
                password = passwordInput.getText().toString().trim();
                callsignup(email,password);
            }
        }); */

    }

    public void signin (View view){
        email = emailInput.getText().toString();
        password = passwordInput.getText().toString();

        if (email.equals(null) || password.equals(null)){

            showToast("Please enter your email and password to sign in.");
        }
        else {
            callsignin(email,password);
        }
    }

    public void signup (View view){
        email = emailInput.getText().toString();
        password = passwordInput.getText().toString();
        callsignup(email,password);
    }

    private void showToast (String text)
    {
        Toast.makeText(CreateAccount.this,text,Toast.LENGTH_LONG).show();
    }

   private void callsignup (String email, String password)
    {
        mAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        Log.d(TAG, "createUserWithEmail:onComplete:" +
task.isSuccessful());
                        if (task.isSuccessful()) {
                            // Sign in success, update UI with the signed-in user's
information
                            Log.d(TAG, "createUserWithEmail:success");
                            FirebaseUser user = mAuth.getCurrentUser();
                            Intent intent = new Intent
(CreateAccount.this,MainActivity.class);
                            startActivity(intent);
                        } else {
                            // If sign in fails, display a message to the user.

                            Log.d(TAG, "onComplete: Failed=" +
task.getException().getMessage()); //ADD THIS
                            Toast.makeText(CreateAccount.this, "Authentication
failed.",
                                    Toast.LENGTH_SHORT).show();
                         //String exception =  (task.getException()).toString();
                           // Toast.makeText(CreateAccount.this,exception,
Toast.LENGTH_SHORT).show();
                        }

                        // ...
                    }
                });
       /* mAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        Log.d("TESTING", "Sign up successful" + task.isSuccessful());

                        //if sign up fails, display message to user
                        //if sign in succeeds, auth listener will be notified
                        //logic to handle signed in user can be handled in the
listener

                        if (!task.isSuccessful()) {
                            Toast.makeText(CreateAccount.this,"Sign up failed.",
Toast.LENGTH_SHORT).show();
                            // If sign in fails, display a message to the user.

                        } else {
                            //if successful, call another method to display user
information
                                userProfile();
                            Toast.makeText(CreateAccount.this,"Account created.",
Toast.LENGTH_SHORT).show();
                            Log.d("TESTING", "Account created.");
                        }


                    }
                }); */
    }

    //set User display name
    private void userProfile ()
    {
        FirebaseUser user = mAuth.getCurrentUser();
        if (user!=null)
        {
            UserProfileChangeRequest profileUpdates = new
UserProfileChangeRequest.Builder().setDisplayName(nameInput.getText().toString().trim(
)).build();

            //.setPhotoUri(Uri.parse("https://example.com/jane-q-user/profile)) here
you can set image link also
```

```
            user.updateProfile(profileUpdates).addOnCompleteListener(new
OnCompleteListener<Void>() {
                @Override
                public void onComplete(@NonNull Task<Void> task) {
                    if (task.isSuccessful()){
                        Log.d("TESTING", "User profile updated");
                    }
                }
            });
        }
        }

        //Now start sign in process
        //signin process
    private void callsignin (String email, String password)
    {
        mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(this,
new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                Log.d("TESTING", "Sign in successful." + task.isSuccessful());

                //if sign in fails, display message to user
                //if sign in succeeds, auth listener will be notified
                //logic to handle signed in user can be handled in the listener

                if (!task.isSuccessful()){
                    String exception = task.getException().toString();
                    Log.v("TESTING", "signInWithEmail:failed", task.getException());
                    Toast.makeText(CreateAccount.this,exception,
Toast.LENGTH_SHORT).show();
                }
                else
                {
                    //starts home screen if sign in successful
                    Intent i = new Intent (CreateAccount.this,MainActivity.class);
                    finish();
                    startActivity(i);
                }
            }
        });

    }
        //main activity shows after succesful sign in, sign out shows upong clicking
sign out button
}
```

Notification Opened Handler

```
package com.example.miche.fastandfurryous; //code adapted from OneSignal.com

import android.content.Context;
import android.content.Intent;
import android.util.Log;

import com.onesignal.OSNotificationAction;
import com.onesignal.OSNotificationOpenResult;
import com.onesignal.OneSignal;

import org.json.JSONObject;
import android.Manifest;
import android.view.View;


class ExampleNotificationOpenedHandler implements OneSignal.NotificationOpenedHandler
{
        // This fires when a notification is opened by tapping on it.

        @Override
        public void notificationOpened(OSNotificationOpenResult result) {
                OSNotificationAction.ActionType actionType = result.action.type;
                JSONObject data = result.notification.payload.additionalData;
                String customKey;

                if (data != null) {
                        customKey = data.optString("customkey", null);
                        if (customKey != null)
                                Log.i("OneSignalExample", "customkey set with value: "
+ customKey);
                }

                if (actionType == OSNotificationAction.ActionType.ActionTaken)
                        Log.i("OneSignalExample", "Button pressed with id: " +
result.action.actionID);

                // The following can be used to open an Activity of your choice.
                // Replace - getApplicationContext() - with any Android Context.
              //  Intent intent = new Intent(getApplicationContext(),
ShowFoundDataNotificationOpen.class);
              //  intent.setFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT |
Intent.FLAG_ACTIVITY_NEW_TASK);
               // startActivity(intent);

                // Add the following to your AndroidManifest.xml to prevent the
launching of your main Activity
                //   if you are calling startActivity above.
     /*
        <application ...>
          <meta-data android:name="com.onesignal.NotificationOpened.DEFAULT"
android:value="DISABLE" />
        </application>
     */
        }
}
```

Notification Received Handler

```
package com.example.miche.fastandfurryous;  //code adapted from OneSignal.com

import android.util.Log;
import android.Manifest;

import com.onesignal.OSNotification;
import com.onesignal.OneSignal;

import org.json.JSONObject;

public class ExampleNotificationReceivedHandler implements
OneSignal.NotificationReceivedHandler {
    @Override
    public void notificationReceived(OSNotification notification) {
        JSONObject data = notification.payload.additionalData;
        String customKey;
        if (data != null) {
            customKey = data.optString("customkey", null);
            if (customKey != null)
                Log.i("OneSignalExample", "customkey set with value: " + customKey);
        }
    }
}
```

Exception Handler

```
package com.example.miche.fastandfurryous;
import android.app.Activity;
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;

public class ExceptionHandler implements Thread.UncaughtExceptionHandler {           //
code adapted from http://www.ssaurel.com/blog/how-to-auto-restart-an-

//android-application-after-a-crash-or-a-force-close-error/
    private Activity activity;

    public ExceptionHandler(Activity a) {
        activity = a;
    }

    @Override
    public void uncaughtException(Thread thread, Throwable ex) {
        Intent intent = new Intent(activity, MainActivity.class);
        intent.putExtra("crash", true);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP
                | Intent.FLAG_ACTIVITY_CLEAR_TASK
                | Intent.FLAG_ACTIVITY_NEW_TASK);

        PendingIntent pendingIntent =
PendingIntent.getActivity(Crash.getInstance().getBaseContext(), 0, intent,
PendingIntent.FLAG_ONE_SHOT);

        AlarmManager mgr = (AlarmManager)
Crash.getInstance().getBaseContext().getSystemService(Context.ALARM_SERVICE);
        mgr.set(AlarmManager.RTC, System.currentTimeMillis() + 100, pendingIntent);

        activity.finish();
        System.exit(2);
    }
}
```

Found Report

```
package com.example.miche.fastandfurryous;

import android.app.Application;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.annotation.NonNull;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.client.Firebase;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.analytics.FirebaseAnalytics;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.iid.FirebaseInstanceId;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;
import com.onesignal.OSPermissionObserver;
import com.onesignal.OSPermissionStateChanges;
import com.onesignal.OSPermissionSubscriptionState;
import com.onesignal.OneSignal;

import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Scanner;

import static com.example.miche.fastandfurryous.CreateAccount.LoggedIn_User_Email;

public class Found extends AppCompatActivity implements OSPermissionObserver {

    private FirebaseAnalytics mFirebaseAnalytics;
    private FirebaseAuth mAuth;
    String fail = "You have been signed out.";
    Button foundreport, markerfound,select_image_found;
    ImageView found_image;
    TextView pet_name_f, location_f, description_f, owner_name_f, owner_contact_f;
    public static final int READ_EXTERNAL_STORAGE = 0;
    private static final int GALLERY_INTENT = 2;
    private ProgressDialog mProgressDialog;
    private Firebase mRoofRef;
    private Uri mImageUri = null;
    private DatabaseReference mdatabaseRef;
    private StorageReference mStorage;
    private Application application;
    public String found;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_found);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
```

```
       /** if (mAuth.getCurrentUser() == null) {
            //user not logged in
            showToast(fail);
            finish();
            Intent intent = new Intent(Found.this, CreateAccount.class);
            startActivity(intent);
        } */
        /** FirebaseDatabase database = FirebaseDatabase.getInstance();
         DatabaseReference myRef = database.getReference("message");
         myRef.setValue("Hi");
         DatabaseReference myRef1 = database.getReference("message1");
         myRef1.setValue("Cool look what I can do"); */

        mFirebaseAnalytics = FirebaseAnalytics.getInstance(this);

        Firebase.setAndroidContext(this);
        foundreport = (Button) findViewById(R.id.foundreport);
        markerfound = (Button) findViewById(R.id.markerfound);
        select_image_found = (Button) findViewById(R.id.select_image_found);
        found_image = (ImageView) findViewById(R.id.found_image);
        pet_name_f = (TextView) findViewById(R.id.pet_name_f);

        location_f = (TextView) findViewById(R.id.location_f);
        description_f = (TextView) findViewById(R.id.description_f);
        owner_name_f = (TextView) findViewById(R.id.owner_name_f);
        owner_contact_f = (TextView) findViewById(R.id.owner_contact_f);

        //initialize progress bar (shows progress while uploading image to firebase
storage)
        mProgressDialog = new ProgressDialog(Found.this);

        //Initialize Firebase Database paths for database and Storage

        mdatabaseRef = FirebaseDatabase.getInstance().getReference();
//change details name
        mRoofRef = new Firebase("https://fastandfurryous-
48c3f.firebaseio.com/").child("Found_Pet_Details").push();  // Push will create new
child with unique name every time upload data in database
        mStorage =
FirebaseStorage.getInstance().getReferenceFromUrl("gs://fastandfurryous-
48c3f.appspot.com");


        OneSignal.startInit(this)
                .autoPromptLocation(true)       //gets user location
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler()) //I hope I can put all of these in here
                .setNotificationOpenedHandler(new LostNotificationOpenedHandler(this))
//change this to open the other activity
                .init();

        //OneSignal.addSubscriptionObserver(this);
        OneSignal.promptLocation();


        OneSignal.sendTag("User_ID", LoggedIn_User_Email);
        OneSignal.addPermissionObserver(this);


        /**if (mAuth.getCurrentUser() == null) {
         //user not logged in
         showToast("You have been signed out.");
         finish();
         Intent intent = new Intent(Messaging.this, CreateAccount.class);
         startActivity(intent);
         } */

        OSPermissionSubscriptionState status =
OneSignal.getPermissionSubscriptionState();
        status.getPermissionStatus().getEnabled();
        status.getSubscriptionStatus().getSubscribed();
        status.getSubscriptionStatus().getUserSubscriptionSetting();
        status.getSubscriptionStatus().getUserId();
        status.getSubscriptionStatus().getPushToken();


    }

    private void showToast(String text) {
        Toast.makeText(Found.this, text, Toast.LENGTH_SHORT).show();
    }

    //Select image from External Storage...
    //set runtime permission to access image from external memory
    public void selectimage(View view) {
        //check runtime permission
        if (ContextCompat.checkSelfPermission(getApplicationContext(),
android.Manifest.permission.READ_EXTERNAL_STORAGE)
                != PackageManager.PERMISSION_GRANTED) {
            Toast.makeText(getApplicationContext(), "Call for Permission",
Toast.LENGTH_SHORT).show();
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                requestPermissions(new
String[]{android.Manifest.permission.READ_EXTERNAL_STORAGE}, READ_EXTERNAL_STORAGE);
            }
        } else {
            callgallery();
        }
    }

    public void reportFound(View view) {
        //Click on Upload Button Title will upload to Database
        final String petName_f = pet_name_f.getText().toString().trim();

        final String petLocation_f = location_f.getText().toString().trim();
        final String petDescription_f = description_f.getText().toString();     //does
there need to be a trim?
        final String ownerName_f = owner_name_f.getText().toString().trim();
        final String contact_f = owner_contact_f.getText().toString().trim();


        if (petName_f.isEmpty() || petLocation_f.isEmpty() ||
petDescription_f.isEmpty() || ownerName_f.isEmpty() || contact_f.isEmpty()) {
            Toast.makeText(getApplicationContext(), "Fill all Fields",
Toast.LENGTH_SHORT).show();
            return;
        }

        else {
            Firebase childRef_name6 = mRoofRef.child("Pet_Name_F");
            childRef_name6.setValue(petName_f);
            Firebase childRef_name7 = mRoofRef.child("Pet_Last_Known_Location_F");
            childRef_name7.setValue(petLocation_f);
            Firebase childRef_name8 = mRoofRef.child("Pet_Description_F");
            childRef_name8.setValue(petDescription_f);
            Firebase childRef_name9 = mRoofRef.child("Owner_Name_F");
            childRef_name9.setValue(ownerName_f);
            Firebase childRef_name10 = mRoofRef.child("Contact_Information_F");
            childRef_name10.setValue(contact_f);

            Toast.makeText(getApplicationContext(), "Updated Info",
Toast.LENGTH_SHORT).show();

            String found = "found";
            Intent intent = new Intent(Found.this, CreateAccount.class);
            intent.putExtra("found", found);      //quotes is the string I need
            startActivity(intent);
            sendNotification();

            showToast("Notification was sent.");
        }

    }

    public void mapfound (View view)
    {
        Intent intent = new Intent (Found.this, Maps.class);
        startActivity(intent);
        /**  mMap = googleMap;                            //code adapted from Google
Developers
         LatLng holden = new LatLng(42.3518, -71.8634);   //changing these latitude
longitude puts a marker somewhere else
         Marker marker = mMap.addMarker(new
MarkerOptions().position(holden).alpha(0.8f)    sets transparency) ; */   //custom
color markers
        //marker.remove();
        //marker.showInfoWindow or hide with .title and .snippet either when create
marker or added later
        //hash map of markers
    }

    public void onOSPermissionChanged(OSPermissionStateChanges stateChanges) {
        if (stateChanges.getFrom().getEnabled() &&
                !stateChanges.getTo().getEnabled())
        {new AlertDialog.Builder(this)
                .setMessage("Notifications Disabled!")
                .show(); }

        Log.i("Debug", "onOSPermissionChanged: " + stateChanges);
    }

    //Check if user granted runtime permissions to access storage
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode) {
            case READ_EXTERNAL_STORAGE:
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)
                    callgallery();
                return;
        }
        Toast.makeText(getApplicationContext(), "...", Toast.LENGTH_SHORT).show();
    }

    //If Access Granted photo gallery will open
    private void callgallery() {
        Intent intent = new Intent(Intent.ACTION_PICK);
        intent.setType("image/*");
        startActivityForResult(intent, GALLERY_INTENT);
    }

    //After Selecting image from gallery image will directly uploaded to Firebase
Database
    //and Image will Show in Image View
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == GALLERY_INTENT && resultCode == RESULT_OK) {

            mImageUri = data.getData();
            found_image.setImageURI(mImageUri);           //change
            StorageReference filePath =
mStorage.child("FoundPet_Images").child(mImageUri.getLastPathSegment());

            mProgressDialog.setMessage("Uploading Image....");
            mProgressDialog.show();

            filePath.putFile(mImageUri).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

                    Uri downloadUri = taskSnapshot.getDownloadUrl();  //Ignore This
error


mRoofRef.child("Found_Pet_Image_URL").setValue(downloadUri.toString());     //change
this

                    Glide.with(getApplicationContext())
                            .load(downloadUri)
                            .crossFade()
                            .placeholder(R.mipmap.loading)
                            .diskCacheStrategy(DiskCacheStrategy.RESULT)
                            .into(found_image);
//change this
                    Toast.makeText(getApplicationContext(), "Updated.",
Toast.LENGTH_SHORT).show();
                    mProgressDialog.dismiss();
                }
            });
        }
    }

    private void sendNotification() {
        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                int SDK_INT = android.os.Build.VERSION.SDK_INT;
                if (SDK_INT > 8) {
                    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
                            .permitAll().build();
                    StrictMode.setThreadPolicy(policy);
                    String send_email;

                    //This is a Simple Logic to Send Notification to a specific
devices
                    /** if
```

```
(CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
mAuth email username  //this is a hard coded user id, need to change this probably --
can use these to
                     {

                     send_email = "spacelover824@gmail.com";            //hard coded
user id    //target specific people

                     }
                     else
                     {
                     send_email = "michelleho24@aol.com";
                     } */

                    try {                               //this code was adapted from
https://documentation.onesignal.com/reference#section-example-code-create-notification
                        String jsonResponse;

                        URL url = new
URL("https://onesignal.com/api/v1/notifications");
                        HttpURLConnection con = (HttpURLConnection)
url.openConnection();
                        con.setUseCaches(false);
                        con.setDoOutput(true);
                        con.setDoInput(true);

                        con.setRequestProperty("Content-Type", "application/json;
charset=UTF-8");
                        con.setRequestProperty("Authorization", "Basic
MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                        con.setRequestMethod("POST");

                        String strJsonBody = "{"
                                + "\"app_id\": \"cc133b06-34bc-49bc-a340-
2dd3de16aa92\"," //provided in OneSignal settings

                                +   "\"included_segments\": [\"Location Holden\"],"
//sends to users in this segment
                                +   "\"data\": {\"foo\": \"bar\"},"
                                +   "\"contents\": {\"en\": \"A pet has been
found!\"}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST)
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });          //user unique ids for each person to target a specific device,
provided in OneSignal

    }                            //--try to use this to filter out the ideas I don't
want to send to then send notification to the rest?

}
```

Lost Report

```
package com.example.miche.fastandfurryous;

import android.*;
import android.app.Application;
import android.app.ProgressDialog;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Build;
import android.os.StrictMode;
import android.support.annotation.NonNull;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.client.Firebase;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;
import com.onesignal.OSNotificationAction;
import com.onesignal.OSNotificationOpenResult;
import com.onesignal.OSPermissionObserver;
import com.onesignal.OSPermissionStateChanges;
import com.onesignal.OSPermissionSubscriptionState;
import com.onesignal.OneSignal;

import org.json.JSONObject;

import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Scanner;

import static android.provider.Contacts.SettingsColumns.KEY;
import static com.example.miche.fastandfurryous.CreateAccount.LoggedIn_User_Email;

public class Lost extends AppCompatActivity implements OSPermissionObserver {

    private FirebaseAuth mAuth;
    String fail = "You have been signed out";
    Button lostreport, markerlost, select_image_lost;
    ImageView lost_image;
    TextView pet_name, location, description, owner_name, owner_contact;
    public static final int READ_EXTERNAL_STORAGE = 0;
    private static final int GALLERY_INTENT = 2;
    private ProgressDialog mProgressDialog;
    private Firebase mRoofRef;
    private Uri mImageUri = null;
    private DatabaseReference mdatabaseRef;
    private StorageReference mStorage;
    public String lost;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lost);

       /** if (mAuth.getCurrentUser()==null)
        {
            //user not logged in
            showToast(fail);
            finish();
            Intent intent = new Intent (Lost.this,CreateAccount.class);
            startActivity(intent);
        } */

        //boolean lost = false; //if equals true (which changes with submit button)
log as lost pet event

        Firebase.setAndroidContext(this);
        lostreport = (Button)findViewById(R.id.lostreport);
        markerlost = (Button)findViewById(R.id.markerlost);
        select_image_lost = (Button)findViewById(R.id.select_image_lost);
        lost_image = (ImageView) findViewById(R.id.lost_image);
        pet_name = (TextView) findViewById(R.id.pet_name);

        location = (TextView) findViewById(R.id.location);
        description = (TextView) findViewById(R.id.description);
        owner_name = (TextView) findViewById(R.id.owner_name);
        owner_contact = (TextView) findViewById(R.id.owner_contact);

        //initialize progress bar (shows progress while uploading image to firebase
storage)
```

```
        mProgressDialog = new ProgressDialog(Lost.this);

        //Initialize Firebase Database paths for database and Storage

        mdatabaseRef = FirebaseDatabase.getInstance().getReference();
//change details name
        mRoofRef = new Firebase("https://fastandfurryous-
48c3f.firebaseio.com/").child("Lost_Pet_Details").push();  // Push will create new
child with unique name every time upload data in database
        mStorage =
FirebaseStorage.getInstance().getReferenceFromUrl("gs://fastandfurryous-
48c3f.appspot.com");

        OneSignal.startInit(this)
                .autoPromptLocation(true)       //gets user location
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler()) //I hope I can put all of these in here
                .setNotificationOpenedHandler(new LostNotificationOpenedHandler(this))
//change this to open the other activity
                .init();


        OneSignal.addPermissionObserver(this);
        //OneSignal.addSubscriptionObserver(this);
        OneSignal.promptLocation();

        OneSignal.sendTag("User_ID", LoggedIn_User_Email);


        OSPermissionSubscriptionState status =
OneSignal.getPermissionSubscriptionState();
        status.getPermissionStatus().getEnabled();
        status.getSubscriptionStatus().getSubscribed();
        status.getSubscriptionStatus().getUserSubscriptionSetting();
        status.getSubscriptionStatus().getUserId();
        status.getSubscriptionStatus().getPushToken();

    }

    //Select image from External Storage...
    //set runtime permission to access image from external memory
    public void selectimage (View view){
        //check runtime permission
        if (ContextCompat.checkSelfPermission(getApplicationContext(),
android.Manifest.permission.READ_EXTERNAL_STORAGE)
                != PackageManager.PERMISSION_GRANTED)
        {
            Toast.makeText(getApplicationContext(), "Call for Permission",
Toast.LENGTH_SHORT).show();
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M)
            {
                requestPermissions(new
String[]{android.Manifest.permission.READ_EXTERNAL_STORAGE}, READ_EXTERNAL_STORAGE);
            }
        }
        else
        {
            callgallery();
        }
    }

    public void reportLost (View view){
        //Click on Upload Button Title will upload to Database
        final String petName = pet_name.getText().toString().trim();

        final String petLocation = location.getText().toString().trim();
        final String petDescription = description.getText().toString();     //does
there need to be a trim?
        final String ownerName = owner_name.getText().toString().trim();
        final String contact = owner_contact.getText().toString().trim();


        if(petName.isEmpty()
||petLocation.isEmpty()||petDescription.isEmpty()||ownerName.isEmpty()||contact.isEmpt
y())
        {
            Toast.makeText(getApplicationContext(), "Fill all Fields",
Toast.LENGTH_SHORT).show();
            return;
        }

        else {
            Firebase childRef_name1 = mRoofRef.child("Pet_Name");
            childRef_name1.setValue(petName);
            Firebase childRef_name2 = mRoofRef.child("Pet_Last_Known_Location");
            childRef_name2.setValue(petLocation);
            Firebase childRef_name3 = mRoofRef.child("Pet_Description");
            childRef_name3.setValue(petDescription);
            Firebase childRef_name4 = mRoofRef.child("Owner_Name");
            childRef_name4.setValue(ownerName);
            Firebase childRef_name5 = mRoofRef.child("Contact_Information");
            childRef_name5.setValue(contact);

            Toast.makeText(getApplicationContext(), "Updated Info",
Toast.LENGTH_SHORT).show();

          String lost = "lost";
            Intent intent = new Intent(Lost.this, CreateAccount.class);
            intent.putExtra("lost", lost);
          startActivity(intent); //do I need this line?

            sendNotification();

            showToast("Notification was sent.");
        }

    }

    public void maplost (View view)
    {
      Intent intent = new Intent (Lost.this, Maps.class);
      startActivity(intent);
        /**  mMap = googleMap;                             //code adapted from Google
Developers
        LatLng holden = new LatLng(42.3518, -71.8634);   //changing these latitude
longitude puts a marker somewhere else
        Marker marker = mMap.addMarker(new
MarkerOptions().position(holden).alpha(0.8f)    sets transparency) ; */   //custom
color markers
      //marker.remove();
        //marker.showInfoWindow or hide with .title and .snippet either when create
marker or added later
        //hash map of markers
    }

    public void onOSPermissionChanged(OSPermissionStateChanges stateChanges) {
        if (stateChanges.getFrom().getEnabled() &&
                !stateChanges.getTo().getEnabled()) new AlertDialog.Builder(this)
                .setMessage("Notifications Disabled!")
                .show();

        Log.i("Debug", "onOSPermissionChanged: " + stateChanges);
    }

    //Check if user granted runtime permissions to access storage
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode) {
            case READ_EXTERNAL_STORAGE:
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)
                    mAuth.getCurrentUser().reload();
                    callgallery();
                return;
        }
        Toast.makeText(getApplicationContext(), "...", Toast.LENGTH_SHORT).show();
    }

    //If Access Granted photo gallery will open
    private void callgallery() {
        Intent intent = new Intent(Intent.ACTION_PICK);
        intent.setType("image/*");
        startActivityForResult(intent, GALLERY_INTENT);
    }

    //After Selecting image from gallery image will directly uploaded to Firebase
Database
    //and Image will Show in Image View
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == GALLERY_INTENT && resultCode == RESULT_OK) {

            mImageUri = data.getData();
            lost_image.setImageURI(mImageUri);          //change
            StorageReference filePath =
mStorage.child("Lost_Pet_Images").child(mImageUri.getLastPathSegment());

            mProgressDialog.setMessage("Uploading Image....");
            mProgressDialog.show();

            filePath.putFile(mImageUri).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

                    Uri downloadUri = taskSnapshot.getDownloadUrl();  //Ignore This
error


mRoofRef.child("Lost_Pet_Image_URL").setValue(downloadUri.toString());     //change
this

                    Glide.with(getApplicationContext())
                            .load(downloadUri)
                            .crossFade()
```

```
                                .placeholder(R.mipmap.loading)
                                .diskCacheStrategy(DiskCacheStrategy.RESULT)
                                .into(lost_image);
//change this
                        Toast.makeText(getApplicationContext(), "Updated.",
Toast.LENGTH_SHORT).show();
                        mProgressDialog.dismiss();
                    }
                });
            }
        }

    private void showToast (String text)
    {
        Toast.makeText(Lost.this,text,Toast.LENGTH_SHORT).show();
    }

    private void sendNotification() {
        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                int SDK_INT = android.os.Build.VERSION.SDK_INT;
                if (SDK_INT > 8) {
                    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
                            .permitAll().build();
                    StrictMode.setThreadPolicy(policy);
                    String send_email;

                    //This is a Simple Logic to Send Notification to a specific
devices
                    /** if
(CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
mAuth email username  //this is a hard coded user id, need to change this probably --
can use these to
                     {

                     send_email = "spacelover824@gmail.com";            //hard coded
user id    //target specific people

                     }
                     else
                     {
                     send_email = "michelleho24@aol.com";
                     } */

                    try {                               //this code was adapted from
https://documentation.onesignal.com/reference#section-example-code-create-notification
                        String jsonResponse;

                        URL url = new
URL("https://onesignal.com/api/v1/notifications");
                        HttpURLConnection con = (HttpURLConnection)
url.openConnection();
                        con.setUseCaches(false);
                        con.setDoOutput(true);
                        con.setDoInput(true);

                        con.setRequestProperty("Content-Type", "application/json;
charset=UTF-8");
                        con.setRequestProperty("Authorization", "Basic
MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                        con.setRequestMethod("POST");
                        String strJsonBody = "{"
                                + "\"app_id\": \"cc133b06-34bc-49bc-a340-
2dd3de16aa92\"," //provided in OneSignal settings

                                +   "\"included_segments\": [\"Location Holden\"],"
//sends to users in this segment
                                +   "\"data\": {\"foo\": \"bar\"},"
                                +   "\"contents\": {\"en\": \"A pet has gone
missing!\"}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST)
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });          //user unique ids for each person to target a specific device,
provided in OneSignal

    }                               //--try to use this to filter out the ideas I don't
want to send to then send notification to the rest?


}
```

Main Activity/Home Screen

```
package com.example.miche.fastandfurryous;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.BottomNavigationView;
import android.support.v7.app.AppCompatActivity;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import android.Manifest;
import com.google.firebase.auth.FirebaseUser;
import com.onesignal.OneSignal;
import com.google.firebase.analytics.FirebaseAnalytics;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import static com.example.miche.fastandfurryous.CreateAccount.LoggedIn_User_Email;

public class MainActivity extends AppCompatActivity {

    private TextView mTextMessage;
    private FirebaseAnalytics mFirebaseAnalytics;
    private FirebaseAuth mAuth;
    private StorageReference mStorageRef;
    private static boolean activityStarted;
    FirebaseUser user;
    Button signoutbut;
    String fail = "You have been signed out.";

    public void recent (View view)
    {
        Intent recent = new Intent (MainActivity.this,RecentPetsActivity.class);
        startActivity(recent);
    }

    public void lostActivity (View view)
    {
        Intent lostActivity = new Intent (MainActivity.this,Lost.class);
        startActivity(lostActivity);
    }

    public void startMaps (View view)
    {
        Intent startMaps = new Intent (MainActivity.this, Maps.class);
        startActivity(startMaps);
    }

    public void foundActivity (View view)
    {
        Intent foundActivity = new Intent (MainActivity.this, Found.class);
        startActivity(foundActivity);
    }

    public void signOutActivity (View view)
    {
        Intent signoutActivity = new Intent(MainActivity.this, SignIn.class);
        startActivity(signoutActivity);
    }

    public void settingsActivity (View view)
    {
        Intent settingsActivity = new Intent(MainActivity.this,
SettingsActivity.class);
        startActivity(settingsActivity);
    }

    public void messaging (View view)
    {
        Intent messageactivity = new Intent(MainActivity.this, Messaging.class);
        startActivity(messageactivity);
    }

    private void showToast (String text)
    {
        Toast.makeText(MainActivity.this,text,Toast.LENGTH_SHORT).show();
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_activity);

        Bundle extras = getIntent().getExtras();

        if (extras != null)
        {
            if (extras.containsKey("found"))
            {
                extras.remove("found");
                Intent i = new Intent (MainActivity.this,
ShowLostDataNotificationOpen.class);
                startActivity(i);
            }
            else if (extras.containsKey("lost"))
            {
                extras.remove("lost");
                Intent i = new Intent (MainActivity.this,
ShowFoundDataNotificationOpen.class);
                startActivity(i);
            }
        }

        activityStarted = true;


        mTextMessage = (TextView) findViewById(R.id.message);
        // Obtain the FirebaseAnalytics instance.
        mFirebaseAnalytics = FirebaseAnalytics.getInstance(this);
        mAuth = FirebaseAuth.getInstance();

        mStorageRef = FirebaseStorage.getInstance().getReference();

        signoutbut = (Button) findViewById(R.id.signoutbut); //important call

        if (mAuth.getCurrentUser()==null)
        {
            //user not logged in
            showToast(fail);
            finish();
            Intent intent = new Intent (MainActivity.this,CreateAccount.class);
            startActivity(intent);
        }

        //Setting the unique tags for Current User
        user = mAuth.getCurrentUser();
        LoggedIn_User_Email =user.getEmail();
        OneSignal.sendTag("User_ID", LoggedIn_User_Email); //makes a tag for the user

        OneSignal.startInit(this)
                .setNotificationOpenedHandler(new ExampleNotificationOpenedHandler())
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler())
                .init();

        /*signoutbut.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

            }
        }); */
    }



}
```

Maps

```
package com.example.miche.fastandfurryous;

import android.*;
import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationManager;
import android.os.Build;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import android.support.v4.content.ContextCompat;
import android.util.Log;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.firebase.auth.FirebaseAuth;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class Maps extends FragmentActivity
        implements OnMapReadyCallback,
        GoogleMap.OnMarkerClickListener,
        GoogleMap.OnMarkerDragListener {
```

```
    private GoogleMap mMap;
    private final static int MY_PERMISSION_FINE_LOCATION = 101;
    private FirebaseAuth mAuth;
    private static final String TAG = "Map Activity";
    private static final float DEFAULT_ZOOM = 15f;
    private EditText mSearchText;
    TextView pet_name, location, description, pet_name_f, location_f, description_f;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be
used.
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        pet_name = (TextView) findViewById(R.id.pet_name);

        location = (TextView) findViewById(R.id.location);
        description = (TextView) findViewById(R.id.description);
        location_f = (TextView) findViewById(R.id.location_f);
        pet_name_f = (TextView) findViewById(R.id.pet_name_f);
        description_f = (TextView) findViewById(R.id.description_f);

        LocationManager locationManager;

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        //be able to delete markers and add info window and make geolocate work

        // Add a marker in Sydney and move the camera
        /** LatLng holden = new LatLng(42.3518, -71.8634);   //changing these latitude
longitude puts a marker somewhere else
         mMap.addMarker(new MarkerOptions().position(holden).title("Marker in
Holden"));
         mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(holden, 5.0f)); */


        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
            // TODO: Consider calling
            //    ActivityCompat#requestPermissions
            // here to request the missing permissions, and then overriding
            //   public void onRequestPermissionsResult(int requestCode, String[]
permissions,
            //                                          int[] grantResults)
            // to handle the case where the user grants the permission. See the
documentation
            // for ActivityCompat#requestPermissions for more details.
            mMap.setMyLocationEnabled(true);
            LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
            Location location =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
```

```
            if (location != null) {
                LatLng latlng = new LatLng(location.getLatitude(),
location.getLongitude());
                mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latlng, 12.0f));
            }

        } else {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                requestPermissions(new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, MY_PERMISSION_FINE_LOCATION);
            }
        }
        MarkerOptions markerOptions = new MarkerOptions();

        LatLng holden = new LatLng(42.3518, -71.8634);
        LatLng mams = new LatLng(42.2760, -71.7999);
        mMap.addMarker(new
MarkerOptions().position(holden).title("Bailey").snippet("Pet Lost")); //add info
window for description
        mMap.addMarker(new MarkerOptions().position(mams).title("Magnus").snippet("Pet
Found"));

        //        geoLocate();


            //.title()
            //.position()
           /* LatLng holden = new LatLng(42.3518, -71.8634);
            LatLng mams = new LatLng(42.2760, 71.7999);
            mMap.addMarker(new MarkerOptions(). position(holden).title(petName + "
last seen " + locPet)); //add info window for description
            mMap.addMarker(new MarkerOptions(). position(mams).title(petName_f + "
last seen " + locPet_f)); } */


        }


    private void geoLocate() {
        Log.d(TAG, "geoLocate: geolocating");

        String searchString = mSearchText.getText().toString();

        Geocoder geocoder = new Geocoder(Maps.this);
        List<Address> list = new ArrayList<>();
        try {
            list = geocoder.getFromLocationName(searchString, 1);
        } catch (IOException e) {
            Log.e(TAG, "geoLocate: IOException: " + e.getMessage());
        }

        if (list.size() > 0) {
            Address address = list.get(0);

            Log.d(TAG, "geoLocate: found a location: " + address.toString());
            //Toast.makeText(this, address.toString(), Toast.LENGTH_SHORT).show();

            LatLng latLng = new LatLng(address.getLatitude(), address.getLongitude());

            mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng)); //should move
camera to current location
        }
        /**
         * private void moveCamera(LatLng latLng, float zoom, String title){
         Log.d(TAG, "moveCamera: moving the camera to: lat: " + latLng.latitude + ",
lng: " + latLng.longitude );
         mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, zoom));
         +
         +        if(!title.equals("My Location")){
         +            MarkerOptions options = new MarkerOptions()
         +                    .position(latLng)
         +                    .title(title);
         +            mMap.addMarker(options);
         +        }
         +
         +        hideSoftKeyboard();
         }
         */
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions,
                                           @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode) {
            case MY_PERMISSION_FINE_LOCATION:
                if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                    if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
                        mMap.setMyLocationEnabled(true);
                    }
                    mMap.setMyLocationEnabled(true);
                } else {
                    Toast.makeText(getApplicationContext(), "This application requires
location permissions to be granted.", Toast.LENGTH_LONG).show();
                    finish();
                }
                break;

        }
    }

    @Override
    public boolean onMarkerClick(Marker marker) {
        return false;
    }

    @Override
    public void onMarkerDragStart(Marker marker) {

    }

    @Override
    public void onMarkerDrag(Marker marker) {

    }

    @Override
    public void onMarkerDragEnd(Marker marker) {

    }


}
```

My Application Class

```java
package com.example.miche.fastandfurryous;

import android.app.Application;
import android.content.Context;

import com.onesignal.OneSignal;

/**
 * Created by miche on 2/4/2018.
 */

public class MyApplication extends Application {
    private static Context mContext;

    @Override
    public void onCreate() {
        super.onCreate();
        mContext = getApplicationContext();
        OneSignal.startInit(this)
                .setNotificationOpenedHandler(new NotificationOpenedHandler(this))
                .init();
    }

    public static Context getContext() {
        return mContext;
    }
}
```

Firebase Instance ID Service

```java
package com.example.miche.fastandfurryous;


import android.util.Log;

import com.google.firebase.iid.FirebaseInstanceId;
import com.google.firebase.iid.FirebaseInstanceIdService;


public class MyFirebaseInstanceIDService extends FirebaseInstanceIdService {
    @Override
    public void onTokenRefresh() {
        // Get updated InstanceID token.
        String refreshedToken = FirebaseInstanceId.getInstance().getToken();


        // If you want to send messages to this application instance or
        // manage this apps subscriptions on the server side, send the
        // Instance ID token to your app server.
        sendRegistrationToServer(refreshedToken);
    }

    private void sendRegistrationToServer (String token){

    }
}
```

```
Settings
```

```java
package com.example.miche.fastandfurryous;
```

```java
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.auth.FirebaseAuth;

public class SettingsActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;
    private TextView tvSettings;

    Button updateInfobut, showInfobut;
    String fail = "You have been signed out.";
    String owner, petname, address, description;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_settings);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

       mAuth = FirebaseAuth.getInstance(); //important call

       updateInfobut = (Button) findViewById(R.id.updateInfobut);
       showInfobut = (Button) findViewById(R.id.showInfobut);

        if (mAuth.getCurrentUser()==null)
        {
            //user not logged in
            showToast(fail);
            finish();
            Intent intent = new Intent (SettingsActivity.this,CreateAccount.class);
            startActivity(intent);
        }

    }

    public void update (View view){
        /*owner = ownerInput.getText().toString();
        petname = petnameInput.getText().toString();
        address = addressInput.getText().toString();
        description = descriptionInput.getText().toString(); */

        //then do something with the info, like store it

        Intent i = new Intent(SettingsActivity.this,UploadInfo.class);
        startActivity(i);

    }

    public void showuploaded (View view){
        Intent i = new Intent(SettingsActivity.this, ShowData.class);
        startActivity(i);
    }
    private void showToast (String text)
    {
        Toast.makeText(SettingsActivity.this,text,Toast.LENGTH_SHORT).show();
    }

}
```

ShowData Settings

```
package com.example.miche.fastandfurryous;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import android.Manifest;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import static android.R.attr.direction;
import static java.security.AccessController.getContext;

public class ShowData extends AppCompatActivity {

    RecyclerView recyclerView;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference myRef;
    private FirebaseRecyclerAdapter<ShowDataItems, ShowDataViewHolder>
mFirebaseAdapter;        //change
    private static Context mContext;

    public ShowData() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show_data_layout);

        firebaseDatabase = FirebaseDatabase.getInstance();

        myRef = FirebaseDatabase.getInstance().getReference("User_Details"); //change


        recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
        recyclerView.setLayoutManager(new LinearLayoutManager(ShowData.this));
        Toast.makeText(ShowData.this, "Wait !  Fetching List...",
Toast.LENGTH_SHORT).show();

        Context mContext = this;
    }

    //retrieve data from firebase with help of database
    @Override
    public void onStart() {
        super.onStart();
        //Log.d("LOGGED", "IN onStart "); //change
        mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowDataItems,
ShowDataViewHolder>(ShowDataItems.class, R.layout.show_data_single_item,
ShowDataViewHolder.class, myRef) {
            public void populateViewHolder(final ShowDataViewHolder viewHolder,
ShowDataItems model, final int position) { //change

                viewHolder.Image_URL(model.getImage_URL()); //change
                viewHolder.Image_Title(model.getImage_Title()); //change and add


                //OnClick Item
                viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                    @Override
                    public void onClick(final View v) {
                        AlertDialog.Builder builder = new
AlertDialog.Builder(ShowData.this); //change
                        builder.setMessage("Do you want to Delete this data
?").setCancelable(false)
                                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        int selectedItems = position;

mFirebaseAdapter.getRef(selectedItems).removeValue();

mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                        recyclerView.invalidate();
                                        onStart();
                                    }
                                })
                                .setNegativeButton("No", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {

                                        dialog.cancel();
                                    }
                                });
                        AlertDialog dialog = builder.create();
                        dialog.setTitle("Confirm");
                        dialog.show();
                    }
                });

            }
        };

        recyclerView.setAdapter(mFirebaseAdapter);
    }
    //View Holder For Recycler View
    public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
        private final TextView image_title; //change
        private final ImageView image_url; //change and add



        public ShowDataViewHolder(final View itemView) {
            super(itemView);
            image_url = (ImageView) itemView.findViewById(R.id.fetch_image); //change
            image_title = (TextView) itemView.findViewById(R.id.fetch_image_title);
//change and add

            if (image_url==null || image_title ==null) //update
            {
                goToLost(mContext);                             //if there is no image
or words stored in settings, the user input activity will show up
            }
        }

        private void Image_Title(String title) {
            image_title.setText(title);
        } //change

        private void Image_URL(String title) {
            // image_url.setImageResource(R.drawable.loading);
            Glide.with(itemView.getContext())
                    .load(title)
                    .crossFade()
                    .placeholder(R.mipmap.loading)
                    .thumbnail(0.1f)
                    .diskCacheStrategy(DiskCacheStrategy.ALL)
                    .into(image_url); //change
        }



    }
    public static void goToLost(Context mContext) {
        Intent login = new Intent(mContext, Lost.class);
        mContext.startActivity(login);
    }

}
```

ShowData Items Settings

```
package com.example.miche.fastandfurryous;

public class ShowDataItems {
    //retrieve data from database

    private String Image_URL, Image_Title; //make sure these fields have the same name
in the database

    public ShowDataItems(String image_URL, String image_Title) {
        Image_URL = image_URL;
        Image_Title = image_Title;
    }

    public ShowDataItems (){
        //require an empty constructor
    }
    public String getImage_URL() {
        return Image_URL;
    }

    public void setImage_URL(String image_URL) {
        Image_URL = image_URL;
    }

    public String getImage_Title() {
        return Image_Title;
    }

    public void setTitle(String title) {
        Image_Title = title;

    }
}
```

ShowFoundData

```
package com.example.miche.fastandfurryous;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class ShowFoundData extends AppCompatActivity {

    RecyclerView recyclerView;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference myRef;
    private FirebaseRecyclerAdapter<ShowFoundItems, ShowDataViewHolder>
mFirebaseAdapter;
    private static Context mContext;

    public ShowFoundData() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show_data_layout);

        firebaseDatabase = FirebaseDatabase.getInstance();
        myRef = FirebaseDatabase.getInstance().getReference("Found_Pet_Details");


        recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
        recyclerView.setLayoutManager(new LinearLayoutManager(ShowFoundData.this));
        Toast.makeText(ShowFoundData.this, "Wait !  Fetching List...",
Toast.LENGTH_SHORT).show();

        Context mContext = this;
    }

    //retrieve data from firebase with help of database
    @Override
    public void onStart() {
        super.onStart();
        //Log.d("LOGGED", "IN onStart ");
        mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowFoundItems,
ShowDataViewHolder>(ShowFoundItems.class,
R.layout.show_data_single_item_found_notification_open, ShowDataViewHolder.class,
myRef) {
            public void populateViewHolder(final ShowDataViewHolder viewHolder,
ShowFoundItems model, final int position) {

                viewHolder.Found_Pet_Image_URL(model.getFound_Pet_Image_URL());
                viewHolder.Pet_Name_f(model.getPet_Name_F());

viewHolder.Pet_Last_Known_Location_f(model.getPet_Last_Known_Location_F());
                viewHolder.Pet_Description_f(model.getPet_Description_F());
                viewHolder.Owner_Name_f(model.getOwner_Name_F());
                viewHolder.Contact_Information_f(model.getContact_Information_F());


                //OnClick Item
                viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                    @Override
                    public void onClick(final View v) {
                        AlertDialog.Builder builder = new
AlertDialog.Builder(ShowFoundData.this);
                        builder.setMessage("Do you want to Delete this data
?").setCancelable(false)
                                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        int selectedItems = position;

mFirebaseAdapter.getRef(selectedItems).removeValue();

mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                        recyclerView.invalidate();
                                        onStart();
                                    }
                                })
                                .setNegativeButton("No", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        dialog.cancel();
                                    }
                                });
                        AlertDialog dialog = builder.create();
                        dialog.setTitle("Confirm");
                        dialog.show();
                    }
                });

            }
        };

        recyclerView.setAdapter(mFirebaseAdapter);
    }

    //View Holder For Recycler View
    public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
        private final TextView pet_name_f, location_f, description_f, owner_name_f,
owner_contact_f;
        private final ImageView found_image;



        public ShowDataViewHolder(final View itemView) {
            super(itemView);
            found_image = (ImageView) itemView.findViewById(R.id.fetch_found_image);
            pet_name_f = (TextView) itemView.findViewById(R.id.fetch_pet_name_f);
            location_f = (TextView) itemView.findViewById(R.id.fetch_pet_location_f);
            description_f = (TextView)
itemView.findViewById(R.id.fetch_pet_description_f);
            owner_name_f = (TextView) itemView.findViewById(R.id.fetch_owner_name_f);
            owner_contact_f = (TextView) itemView.findViewById(R.id.fetch_contact_f);

            if (found_image == null || pet_name_f == null || location_f == null ||
description_f == null || owner_contact_f== null || owner_name_f== null)
            {
                goToFound(mContext);                                //if there is no
image or words stored in settings, the user input activity will show up
            }
        }

        private void Pet_Name_f(String petName) {
            pet_name_f.setText(petName);
        }

        private void Pet_Last_Known_Location_f(String petLocation) {
            location_f.setText(petLocation);
        }

        private void Pet_Description_f(String description) {
            description_f.setText(description);
        }

        private void Owner_Name_f(String ownerName) {
            owner_name_f.setText(ownerName);
        }

        private void Contact_Information_f(String info) {
            owner_contact_f.setText(info);
        }


        private void Found_Pet_Image_URL(String title) {
            // image_url.setImageResource(R.drawable.loading);
            Glide.with(itemView.getContext())
```

```
                        .load(title)
                        .crossFade()
                        .placeholder(R.mipmap.loading)
                        .thumbnail(0.1f)
                        .diskCacheStrategy(DiskCacheStrategy.ALL)
                        .into(found_image);
            }



        }
        public static void goToFound(Context mContext) {
            Intent login = new Intent(mContext, Found.class);
            mContext.startActivity(login);
        }

    }

        public ShowFoundDataNotificationOpen() {
            // Required empty public constructor
        }

        @Override
        public void onCreate(@Nullable Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);

    setContentView(R.layout.show_data_single_item_found_received_notification_open);
    //might have to make another items for lost and found received classes

            firebaseDatabase = FirebaseDatabase.getInstance();

            myRef = FirebaseDatabase.getInstance().getReference("Lost_Pet_Details");


            recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
            recyclerView.setLayoutManager(new
    LinearLayoutManager(ShowFoundDataNotificationOpen.this));
            Toast.makeText(ShowFoundDataNotificationOpen.this, "Wait !  Fetching List...",
    Toast.LENGTH_SHORT).show();

            Context mContext = this;

            receivedlostbut_f = (Button) findViewById(R.id.receivedlostbut_f);
            contactInfoFR = (TextView) findViewById(R.id.contactInfoFR);
            String contInfoFR = contactInfoFR.toString();

            OneSignal.sendTag("contact",contInfoFR);

            OneSignal.startInit(this)
                    .inFocusDisplaying(OneSignal.OSInFocusDisplayOption.Notification)
                    .unsubscribeWhenNotificationsAreDisabled(true)
                    .setNotificationOpenedHandler(new ExampleNotificationOpenedHandler())
                    .setNotificationReceivedHandler(new
    ExampleNotificationReceivedHandler())
                    .init();
        }

        //retrieve data from firebase with help of database
        @Override
        public void onStart() {
            super.onStart();
            //Log.d("LOGGED", "IN onStart ");
            mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowLostItems,
    ShowDataViewHolder>(ShowLostItems.class,
    R.layout.show_data_single_item_lost_notification_open, ShowDataViewHolder.class,
    myRef) {
                public void populateViewHolder(final ShowDataViewHolder viewHolder,
    ShowLostItems model, final int position) {

                    viewHolder.Lost_Pet_Image_URL(model.getLost_Pet_Image_URL());
                    viewHolder.Pet_Name(model.getPet_Name());

    viewHolder.Pet_Last_Known_Location(model.getPet_Last_Known_Location());
                    viewHolder.Pet_Description(model.getPet_Description());
                    viewHolder.Owner_Name(model.getOwner_Name());
                    viewHolder.Contact_Information(model.getContact_Information());


                    //OnClick Item
                    viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                        @Override
                        public void onClick(final View v) {
                            AlertDialog.Builder builder = new
    AlertDialog.Builder(ShowFoundDataNotificationOpen.this);
                            builder.setMessage("Do you want to Delete this data
    ?").setCancelable(false)
                                    .setPositiveButton("Yes", new
    DialogInterface.OnClickListener() {
                                        @Override
                                        public void onClick(DialogInterface dialog, int
    which) {
                                            int selectedItems = position;

    mFirebaseAdapter.getRef(selectedItems).removeValue();

    mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                            recyclerView.invalidate();
                                            onStart();
                                        }
                                    })
                                    .setNegativeButton("No", new
    DialogInterface.OnClickListener() {
                                        @Override
                                        public void onClick(DialogInterface dialog, int
    which) {
                                            dialog.cancel();
                                        }
                                    });
                            AlertDialog dialog = builder.create();
                            dialog.setTitle("Confirm");
                            dialog.show();
                        }
                    });

                }
            };

            recyclerView.setAdapter(mFirebaseAdapter);
        }

        public void notifyBack (View view)
        {
            sendNotification ();
        }
        //View Holder For Recycler View
        public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
            private final TextView pet_name_rf, location_rf, description_rf,
    owner_name_rf, owner_contact_rf;
            private final ImageView found_image_rf;



            public ShowDataViewHolder(final View itemView) {
                super(itemView);
                found_image_rf = (ImageView)
    itemView.findViewById(R.id.fetch_lost_image_rf);
                pet_name_rf = (TextView) itemView.findViewById(R.id.fetch_pet_name_rf);
                location_rf = (TextView)
    itemView.findViewById(R.id.fetch_pet_location_rf);
                description_rf = (TextView)
    itemView.findViewById(R.id.fetch_pet_description_rf);
                owner_name_rf = (TextView)
    itemView.findViewById(R.id.fetch_owner_name_rf);
                owner_contact_rf = (TextView)
    itemView.findViewById(R.id.fetch_contact_rf);

                if (found_image_rf == null || pet_name_rf == null || location_rf == null
    || description_rf == null || owner_contact_rf== null || owner_name_rf== null)
                {
                    showToast ("No data was sent.");                            //if there
    is no image or words stored in settings, the user input activity will show up
                }
            }

            private void Pet_Name(String petName) {
                pet_name_rf.setText(petName);
            }

            private void showToast(String text) {
                Toast.makeText(mContext, text, Toast.LENGTH_SHORT).show();
            }

            private void Pet_Last_Known_Location(String petLocation) {
                location_rf.setText(petLocation);
            }

            private void Pet_Description(String description) {
                description_rf.setText(description);
            }

            private void Owner_Name(String ownerName) {
                owner_name_rf.setText(ownerName);
            }

            private void Contact_Information(String info) {
                owner_contact_rf.setText(info);
            }


            private void Lost_Pet_Image_URL(String title) {
                // image_url.setImageResource(R.drawable.loading);
                Glide.with(itemView.getContext())
                        .load(title)
                        .crossFade()
                        .placeholder(R.mipmap.loading)
                        .thumbnail(0.1f)
                        .diskCacheStrategy(DiskCacheStrategy.ALL)
                        .into(found_image_rf);
            }



        }
        public static void goToLost(Context mContext) {
            Intent login = new Intent(mContext, Lost.class);
            mContext.startActivity(login);
        }

        private void sendNotification() {
            AsyncTask.execute(new Runnable() {
                @Override
                public void run() {
                    int SDK_INT = android.os.Build.VERSION.SDK_INT;
                    if (SDK_INT > 8) {
                        StrictMode.ThreadPolicy policy = new
    StrictMode.ThreadPolicy.Builder()
                                .permitAll().build();
                        StrictMode.setThreadPolicy(policy);
                        String send_email;

                        //This is a Simple Logic to Send Notification to a specific
    devices
                        /** if
    (CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
    mAuth email username  //this is a hard coded user id, need to change this probably --
    can use these to
                         {

                         send_email = "spacelover824@gmail.com";            //hard coded
    user id    //target specific people

                         }
                         else
                         {
                         send_email = "michelleho24@aol.com";
                         } */

                        try {                               //this code was adapted from
    https://documentation.onesignal.com/reference#section-example-code-create-notification
                            String jsonResponse;

                            URL url = new
    URL("https://onesignal.com/api/v1/notifications");
                            HttpURLConnection con = (HttpURLConnection)
    url.openConnection();
                            con.setUseCaches(false);
                            con.setDoOutput(true);
                            con.setDoInput(true);

                            con.setRequestProperty("Content-Type", "application/json;
    charset=UTF-8");
                            con.setRequestProperty("Authorization", "Basic
    MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                            con.setRequestMethod("POST");

                            String strJsonBody = "{"
                                    + "\"app_id\": \"cc133b06-34bc-49bc-a340-
    2dd3de16aa92\"," //provided in OneSignal settings
```

```
                                + "\"included_segments\": [\"Location Holden\"],"
//sends to users in this segment
                                + "\"data\": {\"foo\": \"bar\"},"
                                + "\"contents\": {\"en\": \"I think I have found
your pet! Contact me at  {{ contact | default: \"default\" }}\"}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST)
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });         //user unique ids for each person to target a specific device,
provided in OneSignal

    }

}
```

ShowFound Items

```
package com.example.miche.fastandfurryous;

public class ShowFoundItems {
    //retrieve data from database

    private String Pet_Name_F, Pet_Last_Known_Location_F, Pet_Description_F,
Owner_Name_F, Contact_Information_F, Found_Pet_Image_URL; //make sure these fields
have the same name in the database

    public ShowFoundItems(String petName_f, String petLocation_f, String
petDescription_f, String ownerName_f, String contact_f, String foundImageURL) {
        Pet_Name_F = petName_f;
        Pet_Last_Known_Location_F = petLocation_f;
        Pet_Description_F = petDescription_f;
        Owner_Name_F = ownerName_f;
        Contact_Information_F = contact_f;
        Found_Pet_Image_URL = foundImageURL;

    }

    public ShowFoundItems(){
        //require an empty constructor
    }

    public String getFound_Pet_Image_URL () {return Found_Pet_Image_URL;}
    public void setFound_Pet_Image_URL (String foundImageURL) {Found_Pet_Image_URL =
foundImageURL;}

    public String getPet_Name_F() {
        return Pet_Name_F;
    }
    public void setPet_Name_F(String petName) {
        Pet_Name_F = petName;
    }

    public String getPet_Last_Known_Location_F() {
        return Pet_Last_Known_Location_F;
    }
    public void setPet_Last_Known_Location_F(String petLocation)
{Pet_Last_Known_Location_F = petLocation;}

    public String getPet_Description_F() {
        return Pet_Description_F;
    }
    public void setPet_Description(String petDescription) {Pet_Description_F =
petDescription;}

    public String getOwner_Name_F() {
        return Owner_Name_F;
    }
    public void setOwner_Name_F(String ownerName) {Owner_Name_F = ownerName;}

    public String getContact_Information_F() {
        return Contact_Information_F;
    }
    public void setContact_F(String contact) {Contact_Information_F = contact;}
}
```

ShowFound Data Notification Opened

```
package com.example.miche.fastandfurryous;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.onesignal.OneSignal;

import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Scanner;

public class ShowFoundDataNotificationOpen extends AppCompatActivity {

    RecyclerView recyclerView;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference myRef;
    private FirebaseRecyclerAdapter<ShowFoundItems, ShowDataViewHolder>
mFirebaseAdapter;
    private static Context mContext;
    Button receivedfoundbut_f;
    TextView contactInfoFR;

    public ShowFoundDataNotificationOpen() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show_found_data_layout);       //might have to make
another items for lost and found received classes

        firebaseDatabase = FirebaseDatabase.getInstance();

        myRef = FirebaseDatabase.getInstance().getReference("Found_Pet_Details");

        recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
        recyclerView.setLayoutManager(new
LinearLayoutManager(ShowFoundDataNotificationOpen.this));
        Toast.makeText(ShowFoundDataNotificationOpen.this, "Wait !  Fetching List...",
Toast.LENGTH_SHORT).show();

        Context mContext = this;

        receivedfoundbut_f = (Button) findViewById(R.id.sendFound);


        OneSignal.startInit(this)
                .inFocusDisplaying(OneSignal.OSInFocusDisplayOption.Notification)
                .unsubscribeWhenNotificationsAreDisabled(true)
                .setNotificationOpenedHandler(new ExampleNotificationOpenedHandler())
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler())
                .init();
    }
    //retrieve data from firebase with help of database
    @Override
    public void onStart() {
        super.onStart();
        //Log.d("LOGGED", "IN onStart ");
        mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowFoundItems,
ShowDataViewHolder>(ShowFoundItems.class,
R.layout.show_data_single_item_found_notification_open, ShowDataViewHolder.class,
myRef) {
            public void populateViewHolder(final ShowDataViewHolder viewHolder,
ShowFoundItems model, final int position) {

                viewHolder.Found_Pet_Image_URL(model.getFound_Pet_Image_URL());
                viewHolder.Pet_Name_F(model.getPet_Name_F());

viewHolder.Pet_Last_Known_Location_F(model.getPet_Last_Known_Location_F());
                viewHolder.Pet_Description_F(model.getPet_Description_F());
                viewHolder.Owner_Name_F(model.getOwner_Name_F());
                viewHolder.Contact_Information_F(model.getContact_Information_F());


                //OnClick Item
                viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                    @Override
                    public void onClick(final View v) {
                        AlertDialog.Builder builder = new
AlertDialog.Builder(ShowFoundDataNotificationOpen.this);
                        builder.setMessage("Do you want to Delete this data
?").setCancelable(false)
                                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        int selectedItems = position;

mFirebaseAdapter.getRef(selectedItems).removeValue();

mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                        recyclerView.invalidate();
                                        onStart();
                                    }
                                })
                                .setNegativeButton("No", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        dialog.cancel();
                                    }
                                });
                        AlertDialog dialog = builder.create();
                        dialog.setTitle("Confirm");
                        dialog.show();
                    }
                });

            }
        };

        recyclerView.setAdapter(mFirebaseAdapter);
    }

    public void notifyBack (View view)
    {
        String found = "found";
        Intent intent = new Intent(ShowFoundDataNotificationOpen.this,
CreateAccount.class);
        intent.putExtra("found", found);      //quotes is the string I need
        startActivity(intent);
        sendNotification();

        showToast1("Notification was sent.");
    }
    private void showToast1 (String text)
    {

Toast.makeText(ShowFoundDataNotificationOpen.this,text,Toast.LENGTH_SHORT).show();
    }



    //View Holder For Recycler View
    public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
        private final TextView pet_name_rf, location_rf, description_rf,
owner_name_rf, owner_contact_rf;
        private final ImageView found_image_rf;



        public ShowDataViewHolder(final View itemView) {
            super(itemView);
            found_image_rf = (ImageView)
itemView.findViewById(R.id.fetch_found_image);
            pet_name_rf = (TextView) itemView.findViewById(R.id.fetch_pet_name_f);
            location_rf = (TextView) itemView.findViewById(R.id.fetch_pet_location_f);
            description_rf = (TextView)
itemView.findViewById(R.id.fetch_pet_description_f);
            owner_name_rf = (TextView) itemView.findViewById(R.id.fetch_owner_name_f);
            owner_contact_rf = (TextView) itemView.findViewById(R.id.fetch_contact_f);

            if (found_image_rf == null || pet_name_rf == null || location_rf == null
|| description_rf == null || owner_contact_rf== null || owner_name_rf== null)
            {
                showToast ("No data was sent.");                         //if there
is no image or words stored in settings, the user input activity will show up
            }


            String contInfoFR = owner_contact_rf.toString();

            //OneSignal.sendTag("contact",contInfoFR);
        }

        private void Pet_Name_F(String petName) {
            pet_name_rf.setText(petName);
        }

        private void showToast(String text) {
            Toast.makeText(mContext, text, Toast.LENGTH_SHORT).show();
        }

        private void Pet_Last_Known_Location_F(String petLocation) {
            location_rf.setText(petLocation);
        }

        private void Pet_Description_F(String description) {
            description_rf.setText(description);
        }

        private void Owner_Name_F(String ownerName) {
            owner_name_rf.setText(ownerName);
        }

        private void Contact_Information_F(String info) {
            owner_contact_rf.setText(info);
        }


        private void Found_Pet_Image_URL(String title) {
            // image_url.setImageResource(R.drawable.loading);
            Glide.with(itemView.getContext())
                    .load(title)
                    .crossFade()
                    .placeholder(R.mipmap.loading)
                    .thumbnail(0.1f)
                    .diskCacheStrategy(DiskCacheStrategy.ALL)
                    .into(found_image_rf);
        }



    }
    public static void goToLost(Context mContext) {
        Intent login = new Intent(mContext, Lost.class);
        mContext.startActivity(login);
    }

    private void sendNotification() {
        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                int SDK_INT = android.os.Build.VERSION.SDK_INT;
                if (SDK_INT > 8) {
                    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
                            .permitAll().build();
                    StrictMode.setThreadPolicy(policy);

                    String send_email;

                    //This is a Simple Logic to Send Notification to a specific
devices
                   if
(CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
mAuth email username  //this is a hard coded user id, need to change this probably --
can use these to
                     {

                     send_email = "spacelover824@gmail.com";             //hard coded
user id    //target specific people

                     }
                     else
                     {
                     send_email = "michelleho24@aol.com";
                     }
                    try {                               //this code was adapted from 
https://documentation.onesignal.com/reference#section-example-code-create-notification
                        String jsonResponse;

                        URL url = new 
URL("https://onesignal.com/api/v1/notifications");
                        HttpURLConnection con = (HttpURLConnection) 
url.openConnection();
                        con.setUseCaches(false);
                        con.setDoOutput(true);
                        con.setDoInput(true);

                        con.setRequestProperty("Content-Type", "application/json; 
charset=UTF-8");
                        con.setRequestProperty("Authorization", "Basic 
MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                        con.setRequestMethod("POST");

                        String strJsonBody = "{"
                                + "\"app_id\": \"cc133b06-34bc-49bc-a340-
2dd3de16aa92\"," //provided in OneSignal settings

                                +   "\"included_segments\": [\"Location Holden\"]," 
//sends to users in this segment
                                +   "\"data\": {\"foo\": \"bar\"},"

                                +   "\"contents\": {\"en\": \"The owner has been 
found! See recently lost pets for their contact information!\"}"
                                //+   "\"contents\": {\"en\": \"The owner has been 
found! Contact them at \" + send_email}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST) 
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ? 
scanner.next() : "";
                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ? 
scanner.next() : "";
                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });         //user unique ids for each person to target a specific device,
provided in OneSignal

    }

}
```

ShowLostData

```
package com.example.miche.fastandfurryous;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.onesignal.OSPermissionObserver;
import com.onesignal.OSPermissionStateChanges;
import com.onesignal.OSPermissionSubscriptionState;
import com.onesignal.OneSignal;

import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Scanner;

import static com.example.miche.fastandfurryous.CreateAccount.LoggedIn_User_Email;
import static com.example.miche.fastandfurryous.Found.READ_EXTERNAL_STORAGE;

public class ShowLostData extends AppCompatActivity implements OSPermissionObserver {

    RecyclerView recyclerView;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference myRef;
    private FirebaseRecyclerAdapter<ShowLostItems, ShowDataViewHolder>
mFirebaseAdapter;
    private static Context mContext;
    public ShowLostData() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show_data_layout);

        firebaseDatabase = FirebaseDatabase.getInstance();

        myRef = FirebaseDatabase.getInstance().getReference("Lost_Pet_Details");


        recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
        recyclerView.setLayoutManager(new LinearLayoutManager(ShowLostData.this));
        Toast.makeText(ShowLostData.this, "Wait !  Fetching List...",
Toast.LENGTH_SHORT).show();

        Context mContext = this;

        OneSignal.startInit(this)
                .autoPromptLocation(true)       //gets user location
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler()) //I hope I can put all of these in here
                .setNotificationOpenedHandler(new LostNotificationOpenedHandler(this))
//change this to open the other activity
                .init();


        OneSignal.addPermissionObserver(this);
        //OneSignal.addSubscriptionObserver(this);
        OneSignal.promptLocation();

        OneSignal.sendTag("User_ID", LoggedIn_User_Email);


        OSPermissionSubscriptionState status =
OneSignal.getPermissionSubscriptionState();
        status.getPermissionStatus().getEnabled();
        status.getSubscriptionStatus().getSubscribed();
        status.getSubscriptionStatus().getUserSubscriptionSetting();
        status.getSubscriptionStatus().getUserId();
        status.getSubscriptionStatus().getPushToken();
    }

    //retrieve data from firebase with help of database
    @Override
    public void onStart() {
        super.onStart();
        //Log.d("LOGGED", "IN onStart ");
        mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowLostItems,
ShowDataViewHolder>(ShowLostItems.class,
R.layout.show_data_single_item_lost_notification_open, ShowDataViewHolder.class,
myRef) {
            public void populateViewHolder(final ShowDataViewHolder viewHolder,
ShowLostItems model, final int position) {

                viewHolder.Lost_Pet_Image_URL(model.getLost_Pet_Image_URL());
                viewHolder.Pet_Name(model.getPet_Name());

viewHolder.Pet_Last_Known_Location(model.getPet_Last_Known_Location());
                viewHolder.Pet_Description(model.getPet_Description());
                viewHolder.Owner_Name(model.getOwner_Name());
                viewHolder.Contact_Information(model.getContact_Information());


                //OnClick Item
                viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                    @Override
                    public void onClick(final View v) {
                        AlertDialog.Builder builder = new
AlertDialog.Builder(ShowLostData.this);
                        builder.setMessage("Do you want to Delete this data
?").setCancelable(false)
                                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        int selectedItems = position;

mFirebaseAdapter.getRef(selectedItems).removeValue();

mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                        recyclerView.invalidate();
                                        onStart();
                                    }
                                })
                                .setNegativeButton("No", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        dialog.cancel();
                                    }
                                });
                        AlertDialog dialog = builder.create();
                        dialog.setTitle("Confirm");
                        dialog.show();
                    }
                });

            }
        };

        recyclerView.setAdapter(mFirebaseAdapter);
    }

    //View Holder For Recycler View
    public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
        private final TextView pet_name, location, description, owner_name,
owner_contact;
        private final ImageView lost_image;



        public ShowDataViewHolder(final View itemView) {
            super(itemView);
            lost_image = (ImageView) itemView.findViewById(R.id.fetch_lost_image);
            pet_name = (TextView) itemView.findViewById(R.id.fetch_pet_name);
            location = (TextView) itemView.findViewById(R.id.fetch_pet_location);
            description = (TextView)
itemView.findViewById(R.id.fetch_pet_description);
            owner_name = (TextView) itemView.findViewById(R.id.fetch_owner_name);
            owner_contact = (TextView)
itemView.findViewById(R.id.fetch_owner_contact);

            if (lost_image == null || pet_name == null || location == null ||
description == null || owner_contact== null || owner_name== null)
            {
                goToLost(mContext);                              //if there is no image
or words stored in settings, the user input activity will show up
            }
        }

        private void Pet_Name(String petName) {
            pet_name.setText(petName);
        }

        private void Pet_Last_Known_Location(String petLocation) {
            location.setText(petLocation);
        }

        private void Pet_Description(String pet_description) {
            description.setText(pet_description);
        }

        private void Owner_Name(String ownerName) {
            owner_name.setText(ownerName);
        }

        private void Contact_Information(String info) {
            owner_contact.setText(info);
        }


        private void Lost_Pet_Image_URL(String title) {
            // image_url.setImageResource(R.drawable.loading);
            Glide.with(itemView.getContext())
                    .load(title)
                    .crossFade()
                    .placeholder(R.mipmap.loading)
                    .thumbnail(0.1f)
                    .diskCacheStrategy(DiskCacheStrategy.ALL)
                    .into(lost_image);
        }



    }
    public static void goToLost(Context mContext) {
        Intent login = new Intent(mContext, Lost.class);
        mContext.startActivity(login);
    }

    public void sendLostNot (View view)
    {
        sendNotification();
        showToast("Notification was sent.");
    }

    public void mapLos (View view)
    {
        Intent intent = new Intent (ShowLostData.this, Maps.class);
        startActivity(intent);
        /**  mMap = googleMap;                            //code adapted from Google
Developers
```

```
         LatLng holden = new LatLng(42.3518, -71.8634);   //changing these latitude
longitude puts a marker somewhere else
         Marker marker = mMap.addMarker(new
MarkerOptions().position(holden).alpha(0.8f)    sets transparency) ; */   //custom
color markers
        //marker.remove();
        //marker.showInfoWindow or hide with .title and .snippet either when create
marker or added later
        //hash map of markers
    }

    private void showToast (String text)
    {
        Toast.makeText(ShowLostData.this,text,Toast.LENGTH_SHORT).show();
    }
    public void onOSPermissionChanged(OSPermissionStateChanges stateChanges) {
        if (stateChanges.getFrom().getEnabled() &&
                !stateChanges.getTo().getEnabled()) new AlertDialog.Builder(this)
                .setMessage("Notifications Disabled!")
                .show();

        Log.i("Debug", "onOSPermissionChanged: " + stateChanges);
    }

    private void sendNotification() {
        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                int SDK_INT = android.os.Build.VERSION.SDK_INT;
                if (SDK_INT > 8) {
                    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
                            .permitAll().build();
                    StrictMode.setThreadPolicy(policy);
                    String send_email;

                    //This is a Simple Logic to Send Notification to a specific
devices
                    /** if
(CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
mAuth email username  //this is a hard coded user id, need to change this probably --
can use these to
                     {

                     send_email = "spacelover824@gmail.com";            //hard coded
user id    //target specific people

                     }
                     else
                     {
                     send_email = "michelleho24@aol.com";
                     } */

                    try {                               //this code was adapted from
https://documentation.onesignal.com/reference#section-example-code-create-notification
                        String jsonResponse;

                        URL url = new
URL("https://onesignal.com/api/v1/notifications");
                        HttpURLConnection con = (HttpURLConnection)
url.openConnection();
                        con.setUseCaches(false);
                        con.setDoOutput(true);
                        con.setDoInput(true);

                        con.setRequestProperty("Content-Type", "application/json;
charset=UTF-8");
                        con.setRequestProperty("Authorization", "Basic
MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                        con.setRequestMethod("POST");

                        String strJsonBody = "{"
                                + "\"app_id\": \"cc133b06-34bc-49bc-a340-
2dd3de16aa92\"," //provided in OneSignal settings

                                +   "\"included_segments\": [\"Location Holden\"],"
//sends to users in this segment
                                +   "\"data\": {\"foo\": \"bar\"},"
                                +   "\"contents\": {\"en\": \"A pet has gone
missing!\"}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST)
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");

                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";

                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");

                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";

                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });          //user unique ids for each person to target a specific device,
provided in OneSignal

    }                               //--try to use this to filter out the ideas I don't
want to send to then send notification to the rest?

}
```

ShowLost Data Notification Opened

```
package com.example.miche.fastandfurryous;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.onesignal.OneSignal;

import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Scanner;

public class ShowLostDataNotificationOpen extends AppCompatActivity {

    RecyclerView recyclerView;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference myRef;
    private FirebaseRecyclerAdapter<ShowLostItems, ShowDataViewHolder>
mFirebaseAdapter;
    private static Context mContext;
    Button receivedlostbut;
    TextView contactInfoLR;

    public ShowLostDataNotificationOpen() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

setContentView(R.layout.show_data_single_item_lost_received_notification_open);

        firebaseDatabase = FirebaseDatabase.getInstance();

        myRef = FirebaseDatabase.getInstance().getReference("Lost_Pet_Details");


        recyclerView = (RecyclerView) findViewById(R.id.show_data_recycler_view);
        recyclerView.setLayoutManager(new
LinearLayoutManager(ShowLostDataNotificationOpen.this));
        Toast.makeText(ShowLostDataNotificationOpen.this, "Wait !  Fetching List...",
Toast.LENGTH_SHORT).show();
```

```
        Context mContext = this;

        receivedlostbut = (Button) findViewById(R.id.receivedlostbut);
        contactInfoLR = (TextView) findViewById(R.id.contactInfoLR);
        String contInfoLR = contactInfoLR.toString();

        OneSignal.startInit(this)
                .inFocusDisplaying(OneSignal.OSInFocusDisplayOption.Notification)
                .unsubscribeWhenNotificationsAreDisabled(true)
                .setNotificationOpenedHandler(new ExampleNotificationOpenedHandler())
                .setNotificationReceivedHandler(new
ExampleNotificationReceivedHandler())
                .init();

        OneSignal.sendTag("contact",contInfoLR);
    }

    //retrieve data from firebase with help of database
    @Override
    public void onStart() {
        super.onStart();
        //Log.d("LOGGED", "IN onStart ");
        mFirebaseAdapter = new FirebaseRecyclerAdapter<ShowLostItems,
ShowDataViewHolder>(ShowLostItems.class,
R.layout.show_data_single_item_lost_notification_open, ShowDataViewHolder.class,
myRef) {
            public void populateViewHolder(final ShowDataViewHolder viewHolder,
ShowLostItems model, final int position) {

                viewHolder.Lost_Pet_Image_URL(model.getLost_Pet_Image_URL());
                viewHolder.Pet_Name(model.getPet_Name());

viewHolder.Pet_Last_Known_Location(model.getPet_Last_Known_Location());
                viewHolder.Pet_Description(model.getPet_Description());
                viewHolder.Owner_Name(model.getOwner_Name());
                viewHolder.Contact_Information(model.getContact_Information());


                //OnClick Item
                viewHolder.itemView.setOnClickListener(new View.OnClickListener() {

                    @Override
                    public void onClick(final View v) {
                        AlertDialog.Builder builder = new
AlertDialog.Builder(ShowLostDataNotificationOpen.this);
                        builder.setMessage("Do you want to Delete this data
?").setCancelable(false)
                                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        int selectedItems = position;

mFirebaseAdapter.getRef(selectedItems).removeValue();

mFirebaseAdapter.notifyItemRemoved(selectedItems);
                                        recyclerView.invalidate();
                                        onStart();
                                    }
                                })
                                .setNegativeButton("No", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int
which) {
                                        dialog.cancel();
                                    }
                                });
                        AlertDialog dialog = builder.create();
                        dialog.setTitle("Confirm");
                        dialog.show();
                    }
                });

            }
        };

        recyclerView.setAdapter(mFirebaseAdapter);
    }

    public void notifyBack (View view)
    {
        sendNotification ();
    }



    //View Holder For Recycler View
    public static class ShowDataViewHolder extends RecyclerView.ViewHolder {
        private final TextView pet_name_r, location_r, description_r, owner_name_r,
owner_contact_r;
        private final ImageView lost_image_r;



        public ShowDataViewHolder(final View itemView) {
            super(itemView);
            lost_image_r = (ImageView) itemView.findViewById(R.id.fetch_lost_image_r);
            pet_name_r = (TextView) itemView.findViewById(R.id.fetch_pet_name_r);
            location_r = (TextView) itemView.findViewById(R.id.fetch_pet_location_r);
            description_r = (TextView)
itemView.findViewById(R.id.fetch_pet_description_r);
            owner_name_r = (TextView) itemView.findViewById(R.id.fetch_owner_name_r);
            owner_contact_r = (TextView)
itemView.findViewById(R.id.fetch_owner_name_r);

            if (lost_image_r == null || pet_name_r == null || location_r == null ||
description_r == null || owner_contact_r== null || owner_name_r== null)
            {
                showToast ("No data was sent.");                           //if there
is no image or words stored in settings, the user input activity will show up
            }
        }


        private void Pet_Name(String petName) {
            pet_name_r.setText(petName);
        }

        private void Pet_Last_Known_Location(String petLocation) {
            location_r.setText(petLocation);
        }
        private void showToast(String text) {
            Toast.makeText(mContext, text, Toast.LENGTH_SHORT).show();
        }
        private void Pet_Description(String description) {
            description_r.setText(description);
        }

        private void Owner_Name(String ownerName) {
            owner_name_r.setText(ownerName);
        }

        private void Contact_Information(String info){owner_contact_r.setText(info);
        }


        private void Lost_Pet_Image_URL(String title) {
            // image_url.setImageResource(R.drawable.loading);
            Glide.with(itemView.getContext())
                    .load(title)
                    .crossFade()
                    .placeholder(R.mipmap.loading)
                    .thumbnail(0.1f)
                    .diskCacheStrategy(DiskCacheStrategy.ALL)
                    .into(lost_image_r);
        }

    }

    private void sendNotification() {
        AsyncTask.execute(new Runnable() {
            @Override
            public void run() {
                int SDK_INT = android.os.Build.VERSION.SDK_INT;
                if (SDK_INT > 8) {
                    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
                            .permitAll().build();
                    StrictMode.setThreadPolicy(policy);
                    String send_email;

                    //This is a Simple Logic to Send Notification to a specific
devices
                    /** if
(CreateAccount.LoggedIn_User_Email.equals("michelleho24@aol.com")) //corresponds with
mAuth email username  //this is a hard coded user id, need to change this probably --
can use these to
                     {

                     send_email = "spacelover824@gmail.com";            //hard coded
user id    //target specific people

                     }
                     else
                     {
                     send_email = "michelleho24@aol.com";
                     } */

                    try {                               //this code was adapted from
https://documentation.onesignal.com/reference#section-example-code-create-notification
                        String jsonResponse;

                        URL url = new
URL("https://onesignal.com/api/v1/notifications");
                        HttpURLConnection con = (HttpURLConnection)
url.openConnection();
```

```
                        con.setUseCaches(false);
                        con.setDoOutput(true);
                        con.setDoInput(true);

                        con.setRequestProperty("Content-Type", "application/json;
charset=UTF-8");
                        con.setRequestProperty("Authorization", "Basic
MzNlYzk5YmEtYTVmNS00NjgwLWFjOWItOTNhZmVjYTExMmJi");  //provided in OneSignal settings
                        con.setRequestMethod("POST");

                        String strJsonBody = "{"
                                + "\"app_id\": \"cc133b06-34bc-49bc-a340-
2dd3de16aa92\"," //provided in OneSignal settings

                                +   "\"included_segments\": [\"Location Holden\"],"
//sends to users in this segment
                                +   "\"data\": {\"foo\": \"bar\"},"
                                +   "\"contents\": {\"en\": \"I think I have found
your pet! Contact me at  {{ contact | default: \"default\" }}\"}"
                                + "}";


                        System.out.println("strJsonBody:\n" + strJsonBody);

                        byte[] sendBytes = strJsonBody.getBytes("UTF-8");
                        con.setFixedLengthStreamingMode(sendBytes.length);

                        OutputStream outputStream = con.getOutputStream();
                        outputStream.write(sendBytes);

                        int httpResponse = con.getResponseCode();
                        System.out.println("httpResponse: " + httpResponse);

                        if (httpResponse >= HttpURLConnection.HTTP_OK
                                && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST)
{
                            Scanner scanner = new Scanner(con.getInputStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        } else {
                            Scanner scanner = new Scanner(con.getErrorStream(), "UTF-
8");
                            jsonResponse = scanner.useDelimiter("\\A").hasNext() ?
scanner.next() : "";
                            scanner.close();
                        }
                        System.out.println("jsonResponse:\n" + jsonResponse);

                    } catch (Throwable t) {
                        t.printStackTrace();
                    }
                }
            }

        });          //user unique ids for each person to target a specific device,
provided in OneSignal

    }

}
```

ShowLost Items

```
package com.example.miche.fastandfurryous;

public class ShowLostItems {
    //retrieve data from database

    private String Pet_Name, Pet_Last_Known_Location, Pet_Description, Owner_Name,
Contact_Information, Lost_Pet_Image_URL; //make sure these fields have the same name
in the database

    public ShowLostItems(String petName, String petLocation, String petDescription,
String ownerName, String contact, String lostImageURL) {
        Pet_Name = petName;
        Pet_Last_Known_Location = petLocation;
        Pet_Description = petDescription;
        Owner_Name = ownerName;
        Contact_Information = contact;
        Lost_Pet_Image_URL = lostImageURL;

    }

    public ShowLostItems(){
        //require an empty constructor
    }

    public String getLost_Pet_Image_URL () {return Lost_Pet_Image_URL;}
    public void setLost_Pet_Image_URL (String lostImageURL) {Lost_Pet_Image_URL =
lostImageURL;}

    public String getPet_Name() {
        return Pet_Name;
    }
    public void setPet_Name(String petName) {
        Pet_Name = petName;
    }

    public String getPet_Last_Known_Location() {
        return Pet_Last_Known_Location;
    }
    public void setPet_Last_Known_Location(String petLocation)
{Pet_Last_Known_Location = petLocation;}

    public String getPet_Description() {
        return Pet_Description;
    }
    public void setPet_Description(String petDescription) {Pet_Description =
petDescription;}

    public String getOwner_Name() {
        return Owner_Name;
    }
    public void setOwner_Name(String ownerName) {Owner_Name = ownerName;}

    public String getContact_Information() {
        return Contact_Information;
    }
    public void setContact(String contact) {Contact_Information = contact;}
}
```

Sign Out

```
package com.example.miche.fastandfurryous;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.design.widget.BottomNavigationView;
import android.support.v7.app.AppCompatActivity;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class SignIn extends AppCompatActivity{

    Button signoutbut2;
    Button backtohomebut;
    private FirebaseAuth mAuth;
    TextView username;
    String fail = "You have been signed out";

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.signout);

        mAuth = FirebaseAuth.getInstance(); //important call
        signoutbut2 = (Button)findViewById(R.id.signoutbut2);
        username = (TextView) findViewById(R.id.tvName);

        //again check if user is logged in or not
        if (mAuth.getCurrentUser()==null)
        {
            //user not logged in
            showToast(fail);
            finish();
            Intent intent = new Intent (SignIn.this,CreateAccount.class);
            startActivity(intent);
        }

        //Fetch the display name of the user
        FirebaseUser user = mAuth.getCurrentUser();

        if (user != null) {
            username.setText("Welcome, " + user.getDisplayName());
        }

       /* signoutbut2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                mAuth.signOut();
                finish();
            }
        }); */
    }

    public void signout (View view){
        mAuth.signOut();
        //finish();
        Intent signout = new Intent (SignIn.this,CreateAccount.class);
        startActivity(signout);
    }
    public void backtohome (View view){
        Intent backtohome = new Intent (SignIn.this,MainActivity.class);
        startActivity(backtohome);
    }
    private void showToast (String text)
    {
        Toast.makeText(SignIn.this,text,Toast.LENGTH_SHORT).show();
    }



}
```

Upload Info Settings

```
package com.example.miche.fastandfurryous;

import android.app.ProgressDialog;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Build;
import android.support.annotation.NonNull;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import android.Manifest;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import com.firebase.client.Firebase;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

public class UploadInfo extends AppCompatActivity {

    Button select_image, upload_button;
    ImageView user_image;
    TextView title;
    public static final int READ_EXTERNAL_STORAGE = 0;
    private static final int GALLERY_INTENT = 2;
    private ProgressDialog mProgressDialog;
    private Firebase mRoofRef;
    private Uri mImageUri = null;
    private DatabaseReference mdatabaseRef;
    private StorageReference mStorage;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_upload_info);

        Firebase.setAndroidContext(this);
        select_image = (Button)findViewById(R.id.select_image);
        upload_button = (Button)findViewById(R.id.upload_bttn);
        user_image = (ImageView) findViewById(R.id.lost_image);
        title = (TextView) findViewById(R.id.etTitle);

        //initialize progress bar (shows progress while uploading image to firebase
storage)
        mProgressDialog = new ProgressDialog(UploadInfo.this);

        //Initialize Firebase Database paths for database and Storage

        mdatabaseRef = FirebaseDatabase.getInstance().getReference();
//does the details have to change?
        mRoofRef = new Firebase("https://fastandfurryous-
48c3f.firebaseio.com/").child("User_Details").push();  // Push will create new child
with unique name every time upload data in database
        mStorage =
FirebaseStorage.getInstance().getReferenceFromUrl("gs://fastandfurryous-
48c3f.appspot.com");

    }

    //Select image from External Storage...
    //set runtime permission to access image from external memory
    public void selectimage (View view){
        //check runtime permission
        if (ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.READ_EXTERNAL_STORAGE)
                != PackageManager.PERMISSION_GRANTED)
        {
            Toast.makeText(getApplicationContext(), "Call for Permission",
Toast.LENGTH_SHORT).show();
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M)
            {
                requestPermissions(new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE}, READ_EXTERNAL_STORAGE);
            }
        }
        else
        {
            callgallery();
        }
    }

    public void upload (View view){
        //Click on Upload Button Title will upload to Database
        final String mName = title.getText().toString().trim();


        if(mName.isEmpty())
        {
            Toast.makeText(getApplicationContext(), "Fill all Fields",
Toast.LENGTH_SHORT).show();
            return;
        }
        Firebase childRef_name = mRoofRef.child("Image_Title");
        childRef_name.setValue(mName);

        Toast.makeText(getApplicationContext(), "Updated Info",
Toast.LENGTH_SHORT).show();
    }

    //Check if user granted runtime permissions to access storage
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode) {
            case READ_EXTERNAL_STORAGE:
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)
                    callgallery();
                return;
        }
        Toast.makeText(getApplicationContext(), "...", Toast.LENGTH_SHORT).show();
    }

    //If Access Granted photo gallery will open
    private void callgallery() {
        Intent intent = new Intent(Intent.ACTION_PICK);
        intent.setType("image/*");
        startActivityForResult(intent, GALLERY_INTENT);
    }

    //After Selecting image from gallery image will directly uploaded to Firebase
Database
    //and Image will Show in Image View
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == GALLERY_INTENT && resultCode == RESULT_OK) {

            mImageUri = data.getData();
            user_image.setImageURI(mImageUri);
            StorageReference filePath =
mStorage.child("User_Images").child(mImageUri.getLastPathSegment());

            mProgressDialog.setMessage("Uploading Image....");
            mProgressDialog.show();

            filePath.putFile(mImageUri).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

                    Uri downloadUri = taskSnapshot.getDownloadUrl();  //Ignore This
error

                    mRoofRef.child("Image_URL").setValue(downloadUri.toString());

                    Glide.with(getApplicationContext())
                            .load(downloadUri)
                            .crossFade()
                            .placeholder(R.mipmap.loading)
                            .diskCacheStrategy(DiskCacheStrategy.RESULT)
                            .into(user_image);
                    Toast.makeText(getApplicationContext(), "Updated.",
Toast.LENGTH_SHORT).show();
                    mProgressDialog.dismiss();
                }
            });
        }
    }

}
```

Recent Pets Activity

```
package com.example.miche.fastandfurryous;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

public class RecentPetsActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_recent_pets);
    }

    public void lostRecent (View view)
    {
        Intent recent = new Intent
(RecentPetsActivity.this,ShowLostDataNotificationOpen.class);
        startActivity(recent);
    }

    public void foundRecent (View view)
    {
        Intent recent = new Intent
(RecentPetsActivity.this,ShowFoundDataNotificationOpen.class);
        startActivity(recent);
    }
}
```

XML

Rounded Button

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <corners android:bottomRightRadius="100dp"
        android:bottomLeftRadius="100dp"
        android:topRightRadius="100dp"
        android:topLeftRadius="100dp"/>
</shape>
```

Create Account

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.CreateAccount">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <include layout="@layout/content_create_account" />

</android.support.design.widget.CoordinatorLayout>
```

Found Report

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.Lost">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="122dp"
        android:layout_alignParentTop="true"
        android:id="@+id/linearLayout">

        <ImageView
            android:id="@+id/found_image"
            android:layout_width="200dp"
            android:layout_height="150dp"
            android:layout_marginLeft="8dp"
            android:layout_marginRight="8dp"
            android:layout_marginTop="8dp"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintRight_toRightOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:srcCompat="@mipmap/loading" />

        <Button
            android:id="@+id/select_image_found"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="selectimage"
            android:text="Select image" />

        <EditText
            android:id="@+id/pet_name_f"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Pet Name"
            android:inputType="text" />

        <EditText
            android:id="@+id/location_f"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Location Found"
            android:inputType="textPostalAddress" />

        <EditText
            android:id="@+id/description_f"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Description"
            android:inputType="text" />

        <EditText
            android:id="@+id/owner_name_f"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Finder Name"
            android:inputType="text|textPersonName" />

        <EditText
            android:id="@+id/owner_contact_f"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Contact email or phone"
            android:inputType="text|textPersonName" />

        <Button
            android:id="@+id/foundreport"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="reportFound"
            android:text="Report" />

        <Button
            android:id="@+id/markerfound"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="mapfound"
            android:text="make a marker" />
    </LinearLayout>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignStart="@+id/linearLayout"
        android:layout_marginTop="46dp"
        android:text="Found Pet Report" />
</RelativeLayout>
```

Lost Report

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.Lost">


    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="122dp"
        android:layout_alignParentTop="true"
        android:id="@+id/linearLayout">

        <ImageView
            android:id="@+id/lost_image"
            android:layout_width="200dp"
            android:layout_height="150dp"
            android:layout_marginLeft="8dp"
            android:layout_marginRight="8dp"
            android:layout_marginTop="8dp"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintRight_toRightOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:srcCompat="@mipmap/loading" />

        <Button
            android:id="@+id/select_image_lost"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="selectimage"
            android:text="Select image" />

        <EditText
            android:id="@+id/pet_name"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Pet Name"
            android:inputType="text" />

        <EditText
            android:id="@+id/location"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Last Known Location"
            android:inputType="textPostalAddress" />

        <EditText
            android:id="@+id/description"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Description"
            android:inputType="text" />

        <EditText
```

```
            android:id="@+id/owner_name"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Owner Name"
            android:inputType="text|textPersonName" />

        <EditText
            android:id="@+id/owner_contact"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Contact email or phone"
            android:inputType="text|textPersonName" />

        <Button
            android:id="@+id/lostreport"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="reportLost"
            android:text="Report" />

        <Button
            android:id="@+id/markerlost"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="mapMarker"
            android:text="make a marker" />
    </LinearLayout>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignStart="@+id/linearLayout"
        android:layout_marginStart="13dp"
        android:layout_marginTop="46dp"
        android:text="Lost Pet Report" />
</RelativeLayout>
```

Main Activity

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.MainActivity">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="#ffffff"
        app:layout_constraintBottom_toBottomOf="parent"
```

```
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.0">

<Button
    android:id="@+id/button7"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_weight="1"
    android:background="@drawable/roundedfound"
    android:onClick="foundActivity"
    android:text="Report a found pet"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.13"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.39"
    tools:ignore="MissingConstraints" />

<ImageView
    android:id="@+id/home_image"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_marginBottom="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginTop="328dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0"
    app:srcCompat="@mipmap/pawheart" />

<Button
    android:id="@+id/mapsbut2"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_weight="1"
    android:background="@drawable/roundedmaps"
    android:onClick="startMaps"
    android:text="Access Map"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.20999998"
    tools:ignore="MissingConstraints" />

<Button
    android:id="@+id/button6"
    android:layout_width="100dp"
    android:layout_height="100dp"
```

```
            android:layout_marginBottom="8dp"
            android:layout_marginEnd="8dp"
            android:layout_marginStart="8dp"
            android:layout_marginTop="8dp"
            android:layout_weight="1"
            android:background="@drawable/roundedlost"
            android:onClick="lostActivity"
            android:text="Report a missing pet "
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.85"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.39"
            tools:ignore="MissingConstraints" />

        <Button
            android:id="@+id/signoutbut"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_marginBottom="8dp"
            android:layout_marginEnd="8dp"
            android:layout_marginStart="8dp"
            android:layout_marginTop="8dp"
            android:background="@drawable/roundsignout"
            android:onClick="signOutActivity"
            android:text="Sign out"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.73"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/home_image"
            app:layout_constraintVertical_bias="0.06999999" />


        <Button
            android:id="@+id/button5"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_marginBottom="8dp"
            android:layout_marginEnd="8dp"
            android:layout_marginStart="68dp"
            android:layout_marginTop="8dp"
            android:layout_weight="1"
            android:background="@drawable/roundedsettings"
            android:onClick="settingsActivity"
            android:text="Settings"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.13"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/home_image"
            app:layout_constraintVertical_bias="0.050000012"
            tools:ignore="MissingConstraints" />

    </android.support.constraint.ConstraintLayout>


</android.support.constraint.ConstraintLayout>
```

Maps

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:map="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/map"
android:name="com.google.android.gms.maps.SupportMapFragment"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="com.example.miche.fastandfurryous.Maps" />
```

Settings

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.SettingsActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <include layout="@layout/content_settings" />

</android.support.design.widget.CoordinatorLayout>
```

Upload Info Settings

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:background="#ffffff"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">


    <ImageView
        android:id="@+id/lost_image"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@mipmap/loading" />

    <Button
        android:id="@+id/select_image"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginTop="8dp"
        android:onClick="selectimage"
        android:text="SELECT IMAGE"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/lost_image" />


    <EditText
        android:id="@+id/etTitle"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginTop="24dp"
        android:ems="10"
        android:hint="Title"
        android:inputType="textPersonName"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/select_image"/>

    <Button
        android:id="@+id/upload_bttn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Upload"
        android:layout_marginTop="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginBottom="8dp"
        android:onClick="upload"
        app:layout_constraintTop_toBottomOf="@+id/etTitle"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintVertical_bias="0.69"/>

</android.support.constraint.ConstraintLayout>
```

ShowData Settings

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#EEEEEE"
    android:orientation="vertical">

    <android.support.v7.widget.RecyclerView
```

```
        android:id="@+id/show_data_recycler_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="2dp"
        android:layout_marginRight="2dp"
        android:layout_marginBottom="2dp"/>

</LinearLayout>
```

ShowData Single Item Settings

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp"
        app:contentPadding="3dp"
        card_view:cardCornerRadius="5dp"
        card_view:cardElevation="2dp"
        android:id="@+id/card_view2"
        app:cardBackgroundColor="#FAFAFA">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/fetch_image_title"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Image Title"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

            <View
                android:layout_width="match_parent"
                android:layout_height="2dp"
                android:id="@+id/view1"
                android:layout_marginTop="2dp"
                android:layout_marginBottom="2dp"
                android:background="#ff0aad"/>
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:orientation="vertical">

                <ImageView
                    android:id="@+id/fetch_image"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:layout_margin="5dp"
                    android:src="@mipmap/loading"
                    android:textStyle="normal|bold" />

                <TextView
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="Click on the image to delete/replace" />

            </LinearLayout>

        </LinearLayout>
    </android.support.v7.widget.CardView>

</LinearLayout>
```

ShowFound Single Item Notification Open

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp"
        app:contentPadding="3dp"
        card_view:cardCornerRadius="5dp"
        card_view:cardElevation="2dp"
        android:id="@+id/card_view2"
        app:cardBackgroundColor="#FAFAFA">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/fetch_pet_name_f"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

            <View
                android:layout_width="match_parent"
                android:layout_height="2dp"
                android:id="@+id/view1"
                android:layout_marginTop="2dp"
                android:layout_marginBottom="2dp"
                android:background="#ff0aad"/>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:orientation="vertical">

                <ImageView
                    android:id="@+id/fetch_found_image"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:layout_margin="5dp"
                    android:src="@mipmap/loading"
                    android:textStyle="normal|bold" />

                <TextView
                    android:id="@+id/fetch_pet_location_f"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Last Known Location"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_pet_description_f"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Description"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/textView9"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
```

```
                    android:id="@+id/textView10"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="Belongs to:"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
                    android:id="@+id/fetch_owner_name_f"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Owner Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_contact_f"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Contact Information"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>



        </LinearLayout>


    </android.support.v7.widget.CardView>

    <Button
        android:id="@+id/sendFound"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="sendLostNot"
        android:text="Send Notification" />

    <Button
        android:id="@+id/mapMark_f"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Make Map Marker" />

</LinearLayout>
```

ShowFound Single Item Received Notification

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
```

```
android:orientation="vertical">

<android.support.v7.widget.CardView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    app:contentPadding="3dp"
    card_view:cardCornerRadius="5dp"
    card_view:cardElevation="2dp"
    android:id="@+id/card_view2"
    app:cardBackgroundColor="#FAFAFA">

<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:gravity="center">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/colorPrimary"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/fetch_pet_name_rf"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Pet Name"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

        </LinearLayout>

        <View
            android:layout_width="match_parent"
            android:layout_height="2dp"
            android:id="@+id/view1"
            android:layout_marginTop="2dp"
            android:layout_marginBottom="2dp"
            android:background="#ff0aad"/>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/fetch_lost_image_rf"
                android:layout_width="150dp"
                android:layout_height="150dp"
                android:layout_margin="5dp"
                android:src="@mipmap/loading"
                android:textStyle="normal|bold" />

            <TextView
                android:id="@+id/fetch_pet_location_rf"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Last Known Location"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/fetch_pet_description_rf"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Pet Description"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/textView9"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/textView10"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Belongs to:"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/fetch_owner_name_rf"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Owner Name"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/fetch_contact_rf"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Contact Information"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

        </LinearLayout>

    </LinearLayout>
</android.support.v7.widget.CardView>

<EditText
    android:id="@+id/contactInfoFR"
    android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Enter your contact info here if you have seen this pet"
        android:inputType="text" />

    <Button
        android:id="@+id/receivedlostbut_f"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="notifyBack"
        android:text="Lost this pet? Enter info above then click to contact" />

</LinearLayout>

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp"
        app:contentPadding="3dp"
        card_view:cardCornerRadius="5dp"
        card_view:cardElevation="2dp"
        android:id="@+id/card_view2"
        app:cardBackgroundColor="#FAFAFA">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/fetch_pet_name"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

            <View
                android:layout_width="match_parent"
                android:layout_height="2dp"
                android:id="@+id/view1"
                android:layout_marginTop="2dp"
                android:layout_marginBottom="2dp"
                android:background="#ff0aad"/>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:orientation="vertical">

                <ImageView
                    android:id="@+id/fetch_lost_image"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:layout_margin="5dp"
                    android:src="@mipmap/loading"
                    android:textStyle="normal|bold" />

                <TextView
                    android:id="@+id/fetch_pet_location"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Last Known Location"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_pet_description"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Description"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/textView9"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
                    android:id="@+id/textView10"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="Belongs to:"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
                    android:id="@+id/fetch_owner_name"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Owner Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
```

```
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_owner_contact"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Contact Information"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

        </LinearLayout>
    </android.support.v7.widget.CardView>

    <Button
        android:id="@+id/sendLost"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Send Notification" />

    <Button
        android:id="@+id/mapMark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="mapLos"
        android:text="Make Map Marker" />

</LinearLayout>

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp"
        app:contentPadding="3dp"
        card_view:cardCornerRadius="5dp"
        card_view:cardElevation="2dp"
        android:id="@+id/card_view2"
        app:cardBackgroundColor="#FAFAFA">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/fetch_pet_name"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

            <View
                android:layout_width="match_parent"
                android:layout_height="2dp"
                android:id="@+id/view1"
                android:layout_marginTop="2dp"
                android:layout_marginBottom="2dp"
                android:background="#ff0aad"/>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:orientation="vertical">

                <ImageView
                    android:id="@+id/fetch_lost_image"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:layout_margin="5dp"
                    android:src="@mipmap/loading"
                    android:textStyle="normal|bold" />

                <TextView
                    android:id="@+id/fetch_pet_location"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Last Known Location"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_pet_description"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Description"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/textView9"
                    android:layout_width="match_parent"
```

```
                android:layout_height="wrap_content"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/textView10"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Belongs to:"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/fetch_owner_name"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Owner Name"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/fetch_owner_contact"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Contact Information"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

        </LinearLayout>

      </LinearLayout>
    </android.support.v7.widget.CardView>

    <Button
        android:id="@+id/sendLost"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Send Notification" />

    <Button
        android:id="@+id/mapMark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="mapLos"
        android:text="Make Map Marker" />

</LinearLayout>
```

ShowLost Single Item Notification Open

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
android:background="#ffffff"
android:orientation="vertical">

<android.support.v7.widget.CardView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    app:contentPadding="3dp"
    card_view:cardCornerRadius="5dp"
    card_view:cardElevation="2dp"
    android:id="@+id/card_view2"
    app:cardBackgroundColor="#FAFAFA">

<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:gravity="center">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/colorPrimary"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/fetch_pet_name"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Pet Name"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

        </LinearLayout>

        <View
            android:layout_width="match_parent"
            android:layout_height="2dp"
            android:id="@+id/view1"
            android:layout_marginTop="2dp"
            android:layout_marginBottom="2dp"
            android:background="#ff0aad"/>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/fetch_lost_image"
                android:layout_width="150dp"
                android:layout_height="150dp"
                android:layout_margin="5dp"
                android:src="@mipmap/loading"
                android:textStyle="normal|bold" />

            <TextView
                android:id="@+id/fetch_pet_location"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Last Known Location"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/fetch_pet_description"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Pet Description"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/textView9"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/textView10"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Belongs to:"
                android:textAlignment="center"
                android:textSize="22dp" />

            <TextView
                android:id="@+id/fetch_owner_name"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Owner Name"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

            <TextView
                android:id="@+id/fetch_owner_contact"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                android:text="Contact Information"
                android:textAlignment="center"
                android:textColor="@color/colorAccent"
                android:textSize="20dp"
                android:textStyle="bold|italic" />

        </LinearLayout>

    </LinearLayout>
</android.support.v7.widget.CardView>

<Button
    android:id="@+id/sendLost"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Send Notification" />

    <Button
        android:id="@+id/mapMark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="mapLos"
        android:text="Make Map Marker" />

</LinearLayout>
```

ShowLost Single Item Received Notification

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp"
        app:contentPadding="3dp"
        card_view:cardCornerRadius="5dp"
        card_view:cardElevation="2dp"
        android:id="@+id/card_view2"
        app:cardBackgroundColor="#FAFAFA">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/fetch_pet_name_r"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Name"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

            <View
                android:layout_width="match_parent"
                android:layout_height="2dp"
                android:id="@+id/view1"
                android:layout_marginTop="2dp"
                android:layout_marginBottom="2dp"
                android:background="#ff0aad"/>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:orientation="vertical">

                <ImageView
                    android:id="@+id/fetch_lost_image_r"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:layout_margin="5dp"
                    android:src="@mipmap/loading"
                    android:textStyle="normal|bold" />

                <TextView
                    android:id="@+id/fetch_pet_location_r"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Last Known Location"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_pet_description_r"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Pet Description"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/textView9"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
                    android:id="@+id/textView10"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="Belongs to:"
                    android:textAlignment="center"
                    android:textSize="22dp" />

                <TextView
                    android:id="@+id/fetch_owner_name_r"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Owner Name"
                    android:textAlignment="center"
```

```
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

                <TextView
                    android:id="@+id/fetch_contact_r"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_margin="5dp"
                    android:text="Contact Information"
                    android:textAlignment="center"
                    android:textColor="@color/colorAccent"
                    android:textSize="20dp"
                    android:textStyle="bold|italic" />

            </LinearLayout>

        </LinearLayout>
    </android.support.v7.widget.CardView>

    <EditText
        android:id="@+id/contactInfoLR"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Enter your contact info here if you have seen this pet"
        android:inputType="text" />

    <Button
        android:id="@+id/receivedlostbut"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="notifyBack"
        android:text="Found the pet? Enter info above then click to contact" />

</LinearLayout>
```

Sign Out

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/tvName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Are you sure you want to sign out?"
        app:layout_constraintBottom_toTopOf="@+id/backtohomebut"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/signoutbut2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:onClick="signout"
        android:text="Sign out"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.595" />

    <Button
        android:id="@+id/backtohomebut"
        android:layout_width="wrap_content"
        android:layout_height="47dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="232dp"
        android:onClick="backtohome"
        android:text="Back to Home"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.187" />
</android.support.constraint.ConstraintLayout>
```

Recent Pets

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.miche.fastandfurryous.RecentPetsActivity">

    <Button
        android:id="@+id/button3"
        android:layout_width="225dp"
        android:layout_height="299dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="#42f4bf"
        android:onClick="lostRecent"
        android:text="Show Lost Pets"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.09"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button4"
        android:layout_width="225dp"
        android:layout_height="300dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="#FFCA3A"
        android:onClick="foundRecent"
        android:text="Show Found Pets"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.91"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Most recent lost and found pets"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.18"
        android:textSize="25dp"/>
</android.support.constraint.ConstraintLayout>
```

Gradle Files

Project Gradle File

```
// Top-level build file where you can add configuration options common to all sub-
projects/modules.

buildscript {

    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.0.1'


        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
        classpath 'com.google.gms:google-services:3.1.1'

    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}
```

```
task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Application Gradle File

```
plugins {
    id 'com.onesignal.androidsdk.onesignal-gradle-plugin' version '0.8.0'
}
apply plugin: 'com.onesignal.androidsdk.onesignal-gradle-plugin'
apply plugin: 'com.android.application'


android {
    compileSdkVersion 26
    defaultConfig {
        applicationId "com.example.miche.fastandfurryous"
        minSdkVersion 21
        targetSdkVersion 26
        multiDexEnabled true
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
        vectorDrawables.useSupportLibrary = true
        manifestPlaceholders = [onesignal_app_id               : "cc133b06-34bc-49bc-
a340-2dd3de16aa92",
                                // Project number pulled from dashboard, local value
is ignored.
                                onesignal_google_project_number: "REMOTE"]

    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-
rules.pro'
        }
    }
    packagingOptions {
        exclude 'META-INF/LICENSE'
        exclude 'META-INF/NOTICE'
    }
    productFlavors {
    }
}

dependencies {
    implementation fileTree(include: ['*.jar'], dir: 'libs')
    //noinspection GradleCompatible
    implementation 'com.android.support:appcompat-v7:26.1.0'
    implementation 'com.google.android.gms:play-services-maps:11.8.0'
    implementation 'com.android.support:design:26.1.0'
    implementation 'com.android.support.constraint:constraint-layout:1.0.2'
    implementation 'com.android.support:support-vector-drawable:26.1.0'
    implementation 'com.google.firebase:firebase-core:11.8.0'
    implementation 'com.google.firebase:firebase-auth:11.8.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.1'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
    compile 'com.google.android.gms:play-services:11.8.0'
    compile 'com.onesignal:OneSignal:[3.6.5, 3.99.99]'
    compile 'com.google.firebase:firebase-core:11.8.0'
    compile 'com.google.firebase:firebase-database:11.8.0'
    compile 'com.google.firebase:firebase-storage:11.8.0'
    compile 'com.google.firebase:firebase-messaging:11.8.0'
    compile 'com.google.firebase:firebase-auth:11.8.0'
    compile 'com.firebase:firebase-client-android:2.4.0'
    compile 'com.firebaseui:firebase-ui-storage:1.2.0'
    compile 'com.firebaseui:firebase-ui-database:1.2.0'
    compile 'com.github.bumptech.glide:glide:3.7.0'
    compile 'com.android.support:cardview-v7:26.1.0'
    compile 'com.android.support:recyclerview-v7:26.1.0'
    compile 'com.android.support:appcompat-v7:26.1.0'
    compile 'com.android.support:multidex:1.0.2'
    compile 'com.squareup.okhttp3:okhttp:3.6.0'
}



repositories {
    maven { url 'https://maven.google.com' }
}


apply plugin: 'com.google.gms.google-services'
```

Appendix E: Application Activity Layouts

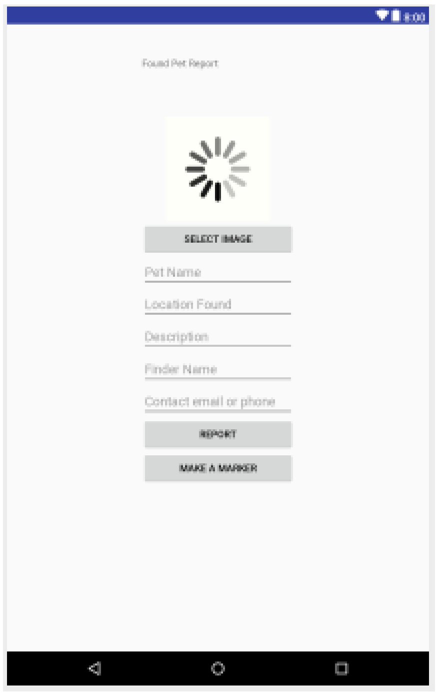


Figure 1E: Activity XML layout for found pet report

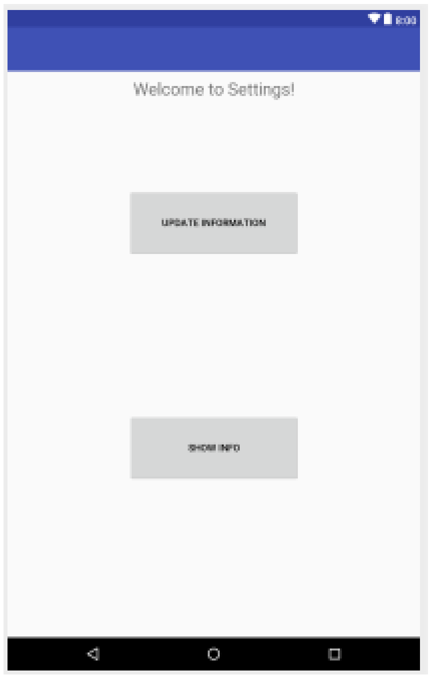


Figure 2E: Activity XML layout for settings

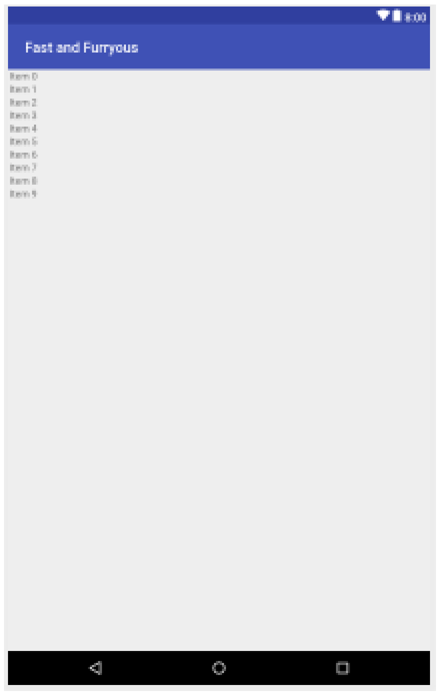


Figure 3E: Activity XML layout for show data in settings

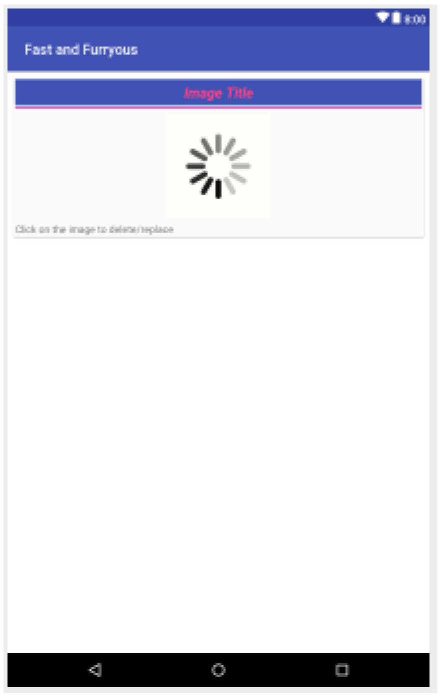


Figure 4E: Activity XML layout for show data single item in settings

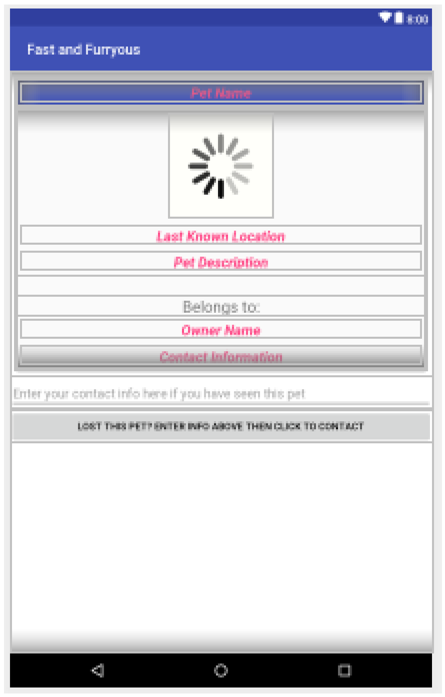

8:00
Fast and Furryous
Pet Name
Last Known Location
Pet Description
Belongs to:
Owner Name
Contact Information
Enter your contact info here if you have seen this pet
LOST THIS PET? ENTER INFO ABOVE THEN CLICK TO CONTACT

Figure 5E: Activity XML layout for received notification for found pet

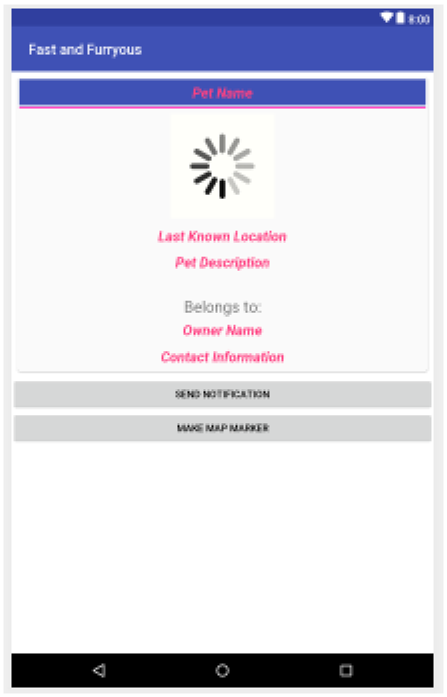


Figure 6E: Activity XML layout for making a found pet report from stored data

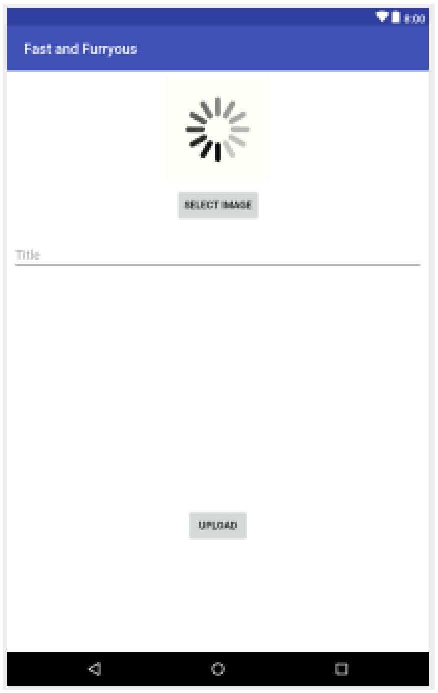


Figure 7E: Activity XML layout for upload data in settings

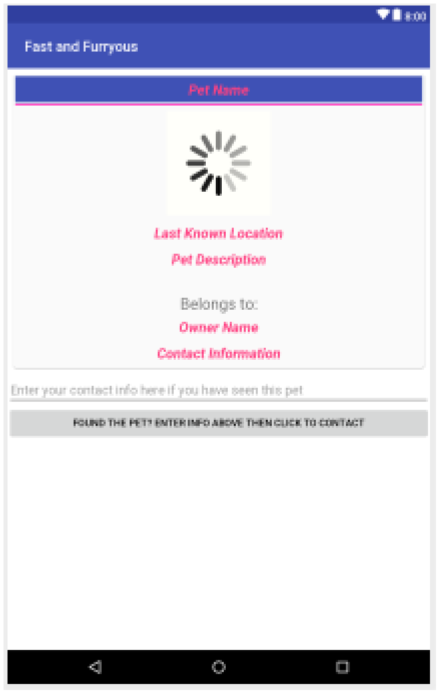

8:00
Fast and Furryous
Pet Name
Last Known Location
Pet Description
Belongs to:
Owner Name
Contact Information
Enter your contact info here if you have seen this pet
FOUND THE PET? ENTER INFO ABOVE THEN CLICK TO CONTACT

Figure 8E: Activity XML layout for received notification of a lost pet

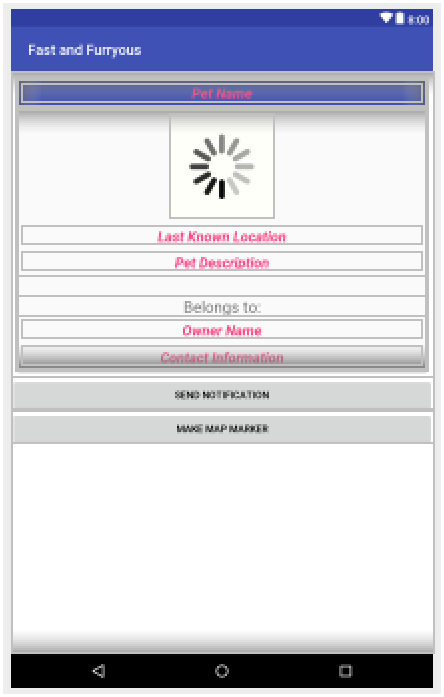


Figure 9E: Activity XML layout for making lost pet report from stored data

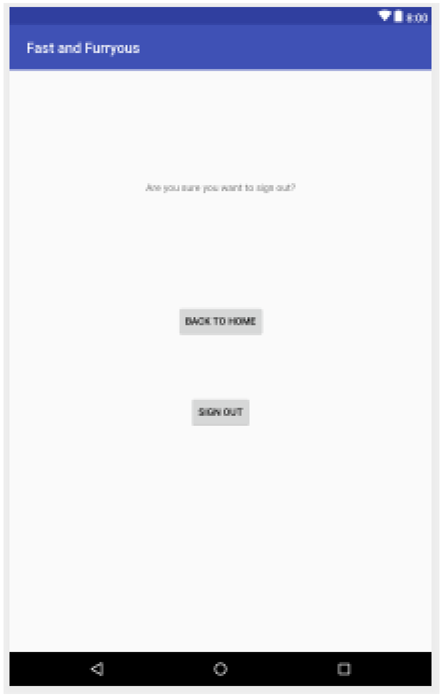


Figure 9E: Activity XML layout for signing a out

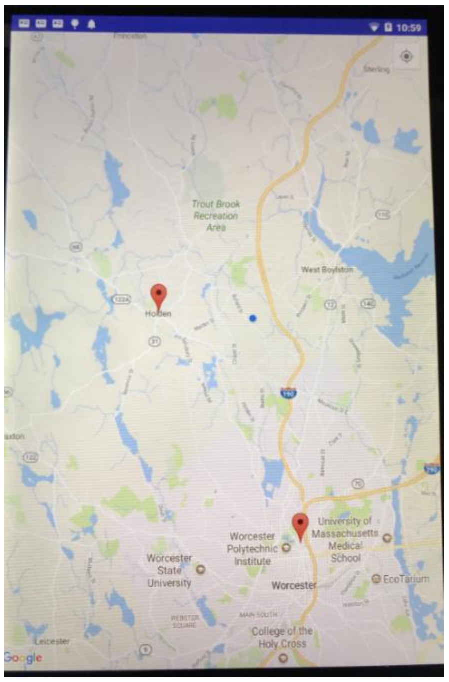

Trout Brook Recreation Area
West Boylston
Holden
Worcester Polytechnic Institute
University of Massachusetts Medical School
Worcester State University
Worcester
EcoTarium
College of the Holy Cross
Leicester
Google

Figure 9E: Activity XML layout for maps

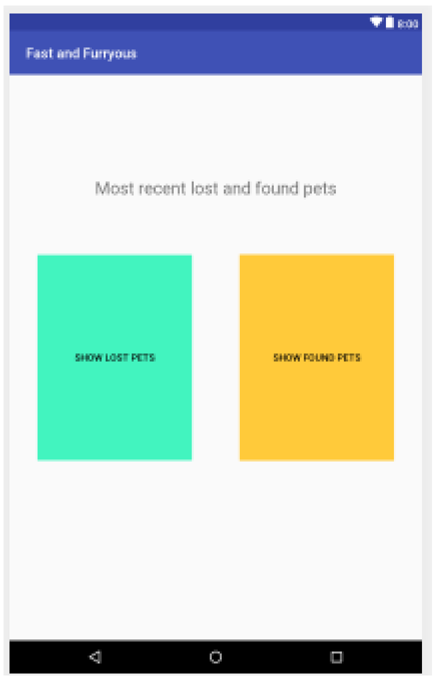


Figure 10E: Activity XML Layout for recent pets

Appendix F: Scoring System

As said in conclusions, the highest priority criteria (determined from the survey done in 2012) were weighted as 5's. The criteria, can be used without GPS, shows a map, tracks a pet accurately in real time, alerts others in the area of lost pet, and allows others to tell owner when pet is found were weighted as 4's. They were chosen because devices with these criteria would efficiently be able to locate and return pets lost and found during natural disasters. The criteria, showing user location and user can enter in a last known location were weighted lower, as 3's, because they were deemed criteria that would assist in locating and returning pets lost and found on an ordinary basis. Lastly, the criteria, can track different pets was weighted the lowest at 2 because it is not a main concern for most pet tracking devices because it can be easily addressed, but it reflects poorly on a device if the device is not adaptable for multiple pets.

The raw data gathered on competitors was found on Amazon.com. App versions were scored using observations of app performance during testing for each version. All criteria except for battery life, lag time, size, cost, tracking range, and adaptability for different pets, were scored on a yes-no basis. If the device had the feature stated by the criteria, its score for that criteria would be a 5; if not, its score would be a 1. For the size criteria, a device was awarded a 5 if the device was smaller than 25 mm in its longest dimension and a 1 if the device was larger than 100 mm in its longest dimension. The ranges for the following three criteria (size, cost, tracking range) were chosen based on the survey mentioned above, which listed what customers felt was an appropriate pet tracker size, cost, and tracking range. For the size criteria, devices were scored a 5 if the largest dimension was less than 25 mm, scored a 4 if the largest dimension was between 25 mm and 62.5 mm, scored a 3 if the largest dimension was 62.5 mm, scored a 2 if the largest dimension was between 62.5 mm and 100 mm, and a 1 if the largest dimension was greater than 100 mm. For the cost criteria, devices were scored a 5 if the cost was less than $40, scored a 4 if the cost was between $40 and $57.50, scored a 3 if the device was $57.50, scored a 2 if the cost was between $57.50 and $75, and a 1 if the cost was greater than $75. For the tracking range criteria, devices were scored a 5 if its tracking range was greater than 10 miles maximum, scored a 4 if the tracking range was between 6 and 10 miles, scored a 3 if the tracking range was 6 miles, scored a 2 if the tracking range was between 2 and 6 miles, and scored a 1 if the tracking range was less than 2 miles. For the criteria of adaptability for different pets, the device scored a 1 if the device was made for one specific pet, scored a 3 if the device was made for multiple specific pets, and a 5 if the device could be used by all pets, according to Amazon.com. The ranges for the following two criteria (battery life, lag time) were chosen based on customer reviews of the device. Generally, customers were unhappy with a device that had a battery life of less than a week, so devices that had a battery life of over a week scored a 5 and under a week scored a 1. Generally, if there were more than five complaints about lag time for a device, the device scored a 1 for lag time. If the app took over three seconds to load a screen, it also scored a 1 for lag time. Using this scoring system and the weights defined for each criterion, each device was scored by multiplying the weight of a criteria by the device's score for that criteria, then adding the sum of these products for a singular device.