An Android Application: Rent-A-Dog

Armando Silva

Abstract- An essential problem for Santa Cruz localist is that they are too busy to own a dog; however, they would want one during vacations or available times. By developing **Rent-A-Dog**, an Android mobile application, it offers a solution to Santa Cruz locals. This research paper will discuss the underlying structure of building the mobile application that offers the solution to people's problems.

Index Terms- Rent-A-Dog, OEMs, Intents, Activitiy

Table of Contents

1 Objective

The objective of this research report is to discuss the newly develop Android mobile application, **Rent-A-Dog**. Through these discusses, we will understand in detailed how the application affects people's daily lives. Through a graphical user interface and a database, data is able to be collected and stored in order for people to use the application.

2 Introduction to Android Mobile Application Structure

Android mobile applications have been developed to store, secure, and access information. These capabilities help structure my research, Rent-A-Dog. In order to achieve this, the application consist of programming languages written in Java, XML, and MySQL; however, other languages may be used by importing them through the Android library. Through these programming languages, we are able to develop Android mobile applications that are further underlying structure that allows for the application to be fully functional.

2.1 The Linux Kernel

One of the underlying structures of an Android application is the Linux kernel. To further specify, "the kernel system service provided by Android inner layer is based on Linux 2.6 kernel" [2]. Through the Linux 2.6 kernel, memory addressing, protocols, drivers, interrupts, and other operating system commands are handled. By allowing the Linux kernel to handle these functions, it "makes it possible for a large number of **OEMs** (Original Equipment Manufacturers, such as phone manufactures) to port it on their devices and allow application developers to write applications" [2]. This structure has been utilized by developers to adjust the Linux kernel structure in order for their mobile application to run more efficiently on handling information and security.

2.2 Android’s Library and Runtime

The second part of an Android system architecture is its libraries and Android runtime. Through this layer, there is "an important link connecting between application framework and Linux kernel" [2]. It is here that the library layer allows for the functionalities that programmers call in their code for their applications. It is here that the libraries are developed in C or C++ to provide most functions for the mobile applications, along with the Dalvik virtual machines, which is a "register virtual machine and makes some specific improvements for mobile devices" [3]. With the Dalvik virtual machine, Android application allows for a consistent and portable runtime which are "designed for embedded environments" [3]. This structure, the layer provides powerful and functional designs to a programmer; they are provided with data structures, file access, network access and more.

2.3 Android’s Application Framework

The third layer of an Android system is its application's framework. The framework consists of the developer's ability to "access all the API framework of the core program" [3]. By having the frame work, a user has the ability to reuse libraries and components from other APIs to their use. However, the layer enforces Android security and provides management to the owners. As well, according to the Research on Development of Android Applications, the Android application framework includes, "a rich and extensible set of Views that can be used to build an application with beautiful user interface and an activity manager that manages the life cycle of application to access data from other applications" [3]. Having these features in the framework layer allows developers to produce more efficient functions to their mobile application by using the built-in interfaces and APIs. Thus, the framework provides these developers powerful functionalities to design their custom applications.

2.4 Android's Application Layer

Lastly, in an Android system architecture, there is the application layer. At this layer, an Android device specifies the set of applications that are to be run. To programmers, they use four different types of application components to communicate with the Android system. These four components: activity, service, content providers, and broadcast receivers, each have their own purpose and are used by the developer in different ways.

2.4.1 Four Components for Application Layer

For an activity, it is a representative user interface of a page on the android device. The developer specifies one of the activates as the main activity, which signals the Android system to start the application. It is here, where the developer uses Java and XML programming languages to perform functionalities as the user performs actions. The next component is its service; here the Android service runs in the background, without any interface, and starts new threads or **intents**, an abstract description of an operation to be performed. These services help link the activity interface with the Linux Kernel by using some functions in the Android library. Another component is the content provider, which is important to store information about the mobile application. The last component to the application layer is the broadcast receivers. This is where “broadcast receivers are in charge of the reception of system wide broadcast and take response aiming at the information that a broadcast transmitted” [4]. By having such a component in the application layer, this allows the user to manage the hardware through software broadcasts. An example, given by the Research on Development of Android Applications, was to broadcast an announcement that the screen has turned off. These broadcast may be useful for developers creating mobile applications that want to manage the broadcast and notifications of devices. Through these components, it allows the developer to have access to more functionalities in Android application.

3 Android’s Database Structure

As precisely described, one of the components in the application layer consisted of being a content provider, which allows users to store or access important data from the mobile application. By using a database platform that is supported by Android devices, a developer is able to efficiently store and access the important information. Firebase, “a real-time database platform that supports syncing database across multiple connected devices at the same time,” [6] contributes to programmers by accessing theses requests for data. In order for Firebase to do so, it implements AngularJS framework that helps connect all devices to the same information. As a developer, this is important when programming a mobile application that stores information because they would want devices to be compatible with each other. Therefore, by providing the programmer with a database platform, this helps store and access information of the mobile application much more efficiently.

Through the Android System architecture and Firebase database structure, a programmer is provided efficient and full functional components to create an Android mobile application. Through the Linux Kernel, library and runtime calls, application framework, and the application itself, a developer is able to manage a mobile application such as Rent-A-Dog.

4 Discussion about the Idea of Rent-A-Dog

In recent time, as described by the Research and Development of Mobile Application for Android Platform, "phone is no longer a communication tool, but also an essential part of people's communication and daily life," [1]. From this, the demand for Android mobile applications and influenced my research to develop a solution to people's daily life problems. Using all the compatible features of Android, I am developing an android mobile application that is intended for Santa Cruz localist, ages twenty-one and over, with credible background, and desiring a dog to rent for a period of time.

4.1 Setting Up Rent-A-Dog on Android's Platform

To begin, there was prior knowledge on the software languages I used to develop the application. Using the courses taken during my undergraduate degree at the University of California Santa Cruz, I have developed a solid foundation using Java, C, MySQL, and XML; which will be further discussed.

In order to start programming, I downloaded Android Studio IDE along with the Java Development Kit 8. Next, I allowed for 2 GB of available disk space to provide Android's application layers enough memory to hold its features. Once Android Studio was downloaded and ran, I opened a new project file in order to hold my application structure together. Soon after, Android studio opened up a graphical user interface to code. Here, in **Figure 1**, we are shown the left of the platform consists of Android's files and libraries, which was described earlier sections. In the middle, we are shown a virtual image of what the **activity** looks like, a screen where a user interacts with. Lastly, on the right, there are functionalities that can be added on to the styling of the activity; which helps to capture user's interaction with the Android device.

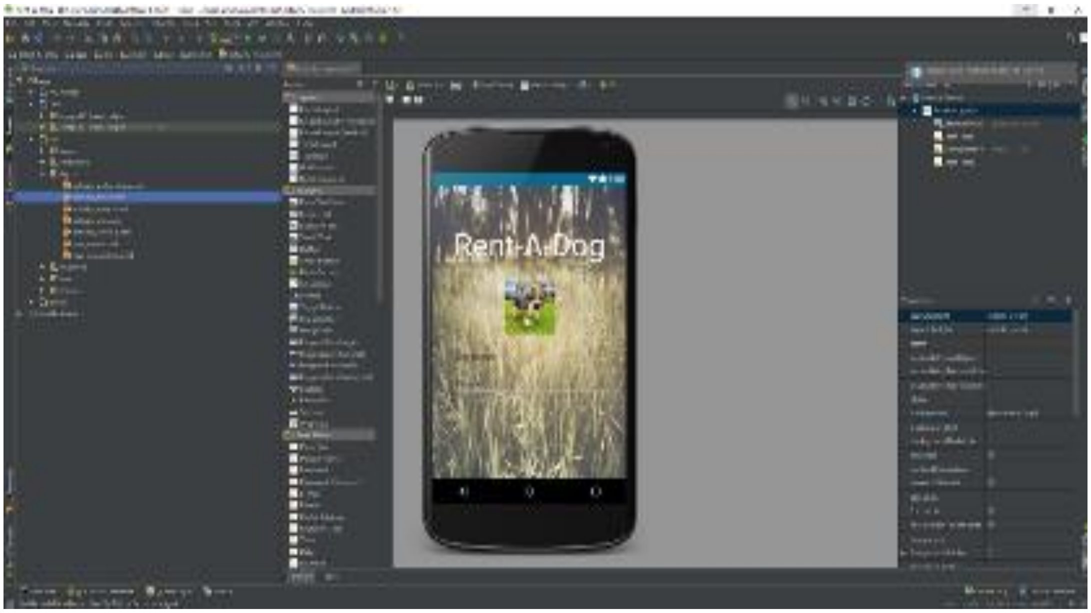

Figure 1. Rent-A-Dog on Android Studio

Once the environment has been set up, I started with the main activity layout page, also known as the main activity.xml file. Here, Android offers a nice graphical user interface, to drag and drop "buttons" or functionalities to the page. These functions at first do not do anything unless handled by a higher order programming language, such as Java; however, it was the start to structuring my application.

4.2 Functionalities to the Main Activity

For the main log in screen, I had developed an activity page that handles the user's log in information. Generally, mobile applications have sign up and login page at the same screen; therefore, by following the same structure, the page follows as seen in **Figure 2**.

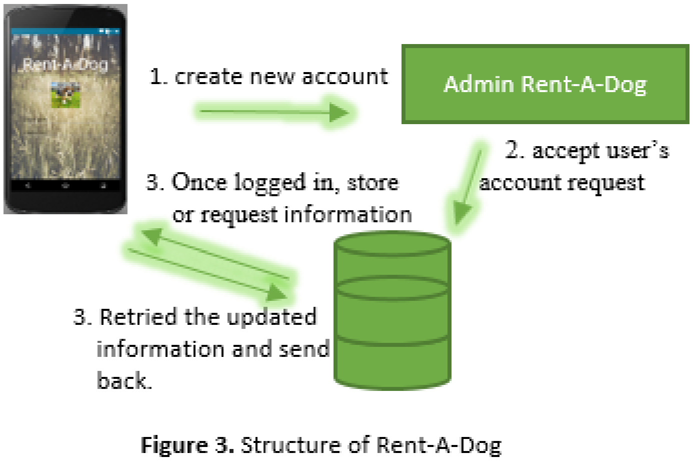
Rent-A-Dog
1. create new account
Admin Rent-A-Dog
2. accept user's account request
3. Once logged in, store or request information
3. Retried the updated information and send back.
Figure 3. Structure of Rent-A-Dog

Figure 2. The main activity page's structure

Here, we have a state that handles if the user would want to sign up to use the Rent-A-Dog application. If they were to sign up, the information would be stored via Firebase, as described earlier, and the administrator would handle whether to allow the user access. The reason for the administrator handling a user's request to sign up is strictly on security purposes. Here, I propose for a team that handles background checks to determine whether one is eligible to rent out or rent a dog; however, for technical purposes, I will not go further in detail.

Once, a user's account is accepted, then they may log in using their username and password. Here, we query the information from the database and ensure that it matches. Using some of the libraries provided to use, we use some of the functionalities through Java programming in order to handle the user's information. Through **Figure 2**, querying happens in the third and fourth step. Thus, after finalizing these steps, the user will then proceed to the next activity page.

4.3 User's Account Page

Once the user is logged in, the user is prompted by an option to rent or rent out dogs. By using a MySQL database, it allows for easy and efficient search up of the dogs depending on the criteria. However, in order to display that, there is Java functionalities that handle the information and displays them on the XML style page.

One of the Java functionalities on this activity is the enabling global positioning system in order for a dog to be searched based on the location of the Android device. Here, Android Studio provides the developer an API framework that allows for such a feature. In the Android Manifest XML file, one may enable Android permission using the ACCESS_FINE_LOCATION to enable the device's location. After that feature has been enabled, there are more functionalities that are programmed in Java that allows for settings on searching a location for the closest dog. As a developer, we used location listener class that are used to set up the constructors and variables for the class. Once set, we provide functions that get the latitude and longitude of the device and asks the user to set the radius for a search or rent out dog.

4.4 Rent-A-Dog Listing

Once the GPS has been enabled and the user has specified the search radius, we query all results from the database with dogs that are posted in that region. Here, we use a dynamic list that obtains the dog that are available and price for each to rent, as seen on **Figure 3**.

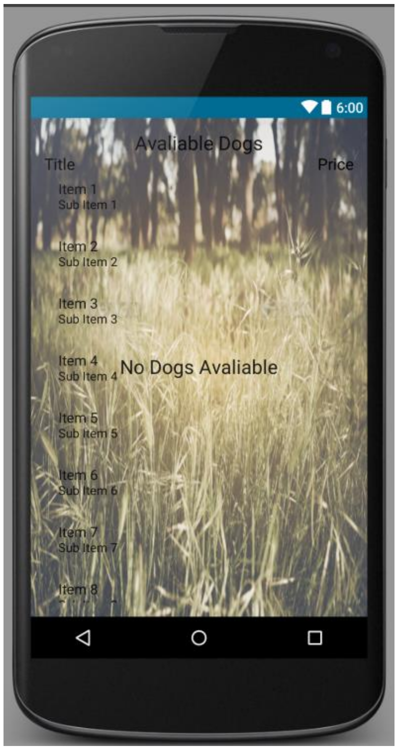

Figure 3. The listing activity that displays dogs available

Using ListView function from Android Studio and the database from Firebase, we obtain all the stored dogs from the database and fill the information onto a ListView to display each item. Here if there are no animals, then we just display that there are no dogs available; however, if there are then we use an **adapter**, which displays a collection of items, using a ListView. For each item in the ListView, we use a Java based function to handle the user's interaction with the dog's post, called ". setOnItemClickListener ()". The way the user interacts with the dog’s post is that they click on the specified dog’s price and starts a new activity, where there, they can set up a payment to rent out a dog.

4.5 Future Research

Although there is a great amount of research done in developing and planning an application for users to rent out dogs, there is still flexibility between changing the Android mobile application. Some proposed improvements to develop on moving forward are, but not constrain to: establishing an online payment system and allowing SMS messaging.

4.5.1 Online Payment System

As describe, a way of developing the application even further is through an online payment system. Through an Android API, using Android Pay, we are able to develop functionality for users to pay through the application, as seen in **Figure 4**

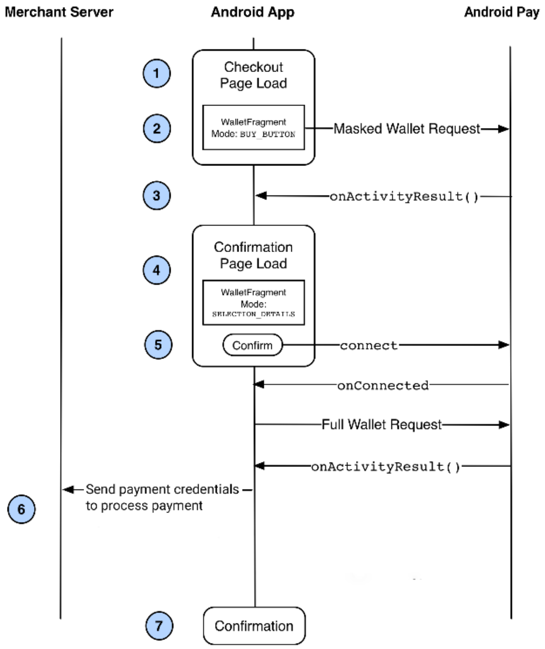

Figure 4. Android Pay Process Flow [5]

As stated in the Integration Overview for Android Pay, there would be requirements in order to be capable of using these functionalities. To start, we would need the Android device's SHA1 fingerprint in the developer's key to generate a client ID. Next, we would add some following tags to the Android Manifest.xml file that enables for the Android Pay API. Once we have set up the following, then we are able to use the isReadyToPay method to check whether the user has installed the Android Pay app and then have a handler for the button where the user pays.

4.5.2 SMS Messaging

To further explore for ways to improve on the mobile application, there has been suggestions for SMS messaging; a way to contact the renter or owner of the dog. In previous Android applications, I have implemented SMS messaging functionalities. Through Android.content.Intent class, I had used the sendIntent.putExtra() in order to determine what media one would want to extract information to. Through these functions, I would be able to extract a renter’s or owner’s information and allow for the two users to allow for messaging.

5 Conclusion

By developing a mobile application that allows users to rent out dogs, we are serving the community and stray dogs together. The application consists of Java functionalities through Android Studio; which is then displayed on an Android device using XML, and stores the information querying data from a database hosted in a cloud by Firebase. Through these functionalities, there will be a form of access to users about dogs that are allowed to be rented.

References

[1] “Research and Development of Mobile Application for Android Platform.” Web. 17 May 2016.

[2] “A Look Inside the Android Kernel with Automated Code Testing.” Web. 17 May 2016.

[3] “Research on Development of Android Applications.” Web. 17 May 2016.

[4] “DEVELOPING GEOLOCATION CHAT BASE APPLICATION WITH IONIC FRAMEWORK.” Web. 17 May 2016.

[5] (n.d) Retrieved from https://developers.google.com/android-pay/images/detailed-android_rev.png

[6] (n.d.). Retrieved Jun 7, 2016, from https://firebase.google.com/docs/database/