**NFC ID: An NFC Authorized Database**

Angel Hernandez

[Angel.Hernandez.751@my.csun.edu](mailto:Angel.Hernandez.751@my.csun.edu)

**Abstract**

NFC ID is a NFC authorized database application. NFC, or ‘Near Field Communication’, is a type of computer communication that enables two devices to simply tap together and send information. NFC devices commonly share photos, website links, or simple contact info. Knowing the capabilities of NFC, I wanted to create an application which removed the need for a physical wallet. If all your information was accessible with just a touch of your phone, then the need to carry around extra items would not be needed. NFC ID allows users to send information to another device with just a tap and the use of a authorization key.

**Introduction and Related Work**

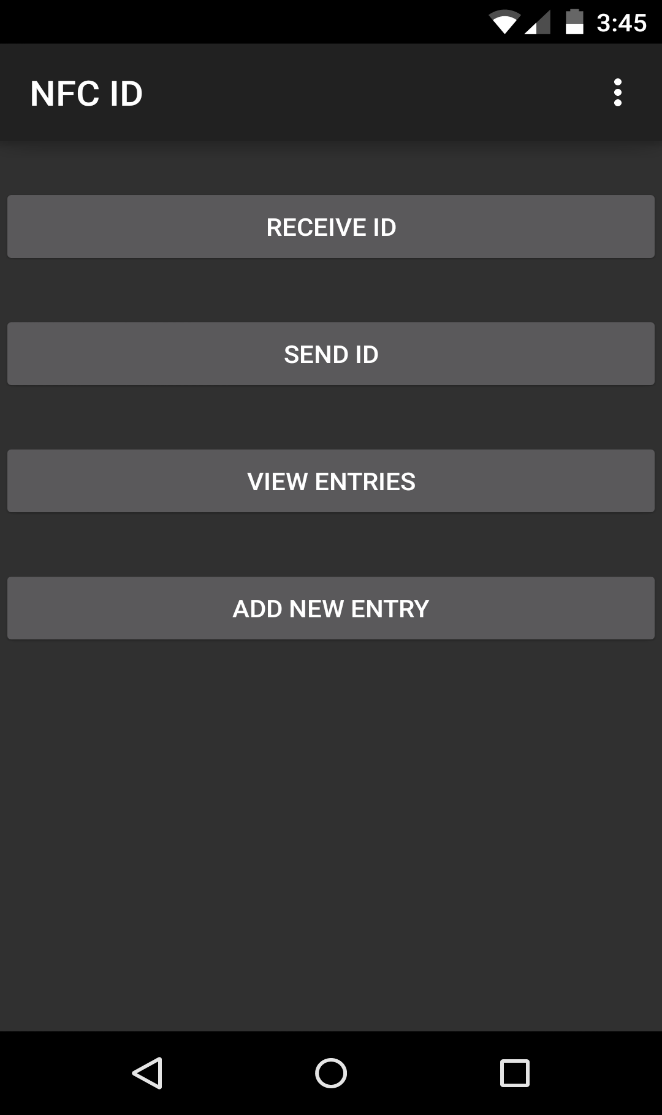
NFC ID uses two important capabilities of modern phones, ‘Near Field Communication’ and network access. NFC allows for the transfer of data from one device to another, while network access makes it possible to store information in a easy to access medium. Using NFC that is available in newer smartphones and a locally hosted database, NFC ID sends and receives information for various uses. The main goal of this application is to illustrate that a NFC Database is possible and that it can pave the way into a future in which no person would ever have the need to carry a wallet.

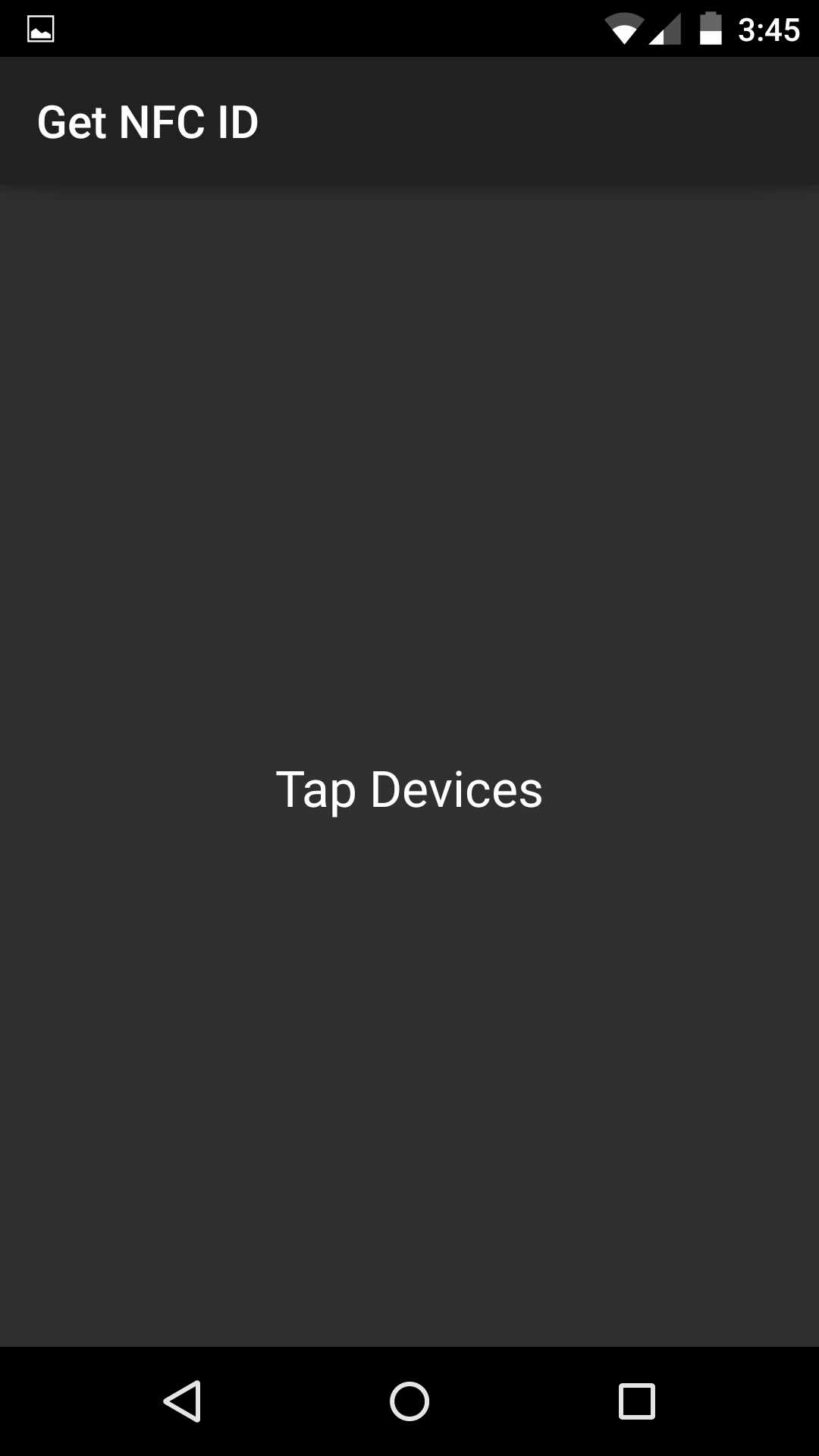
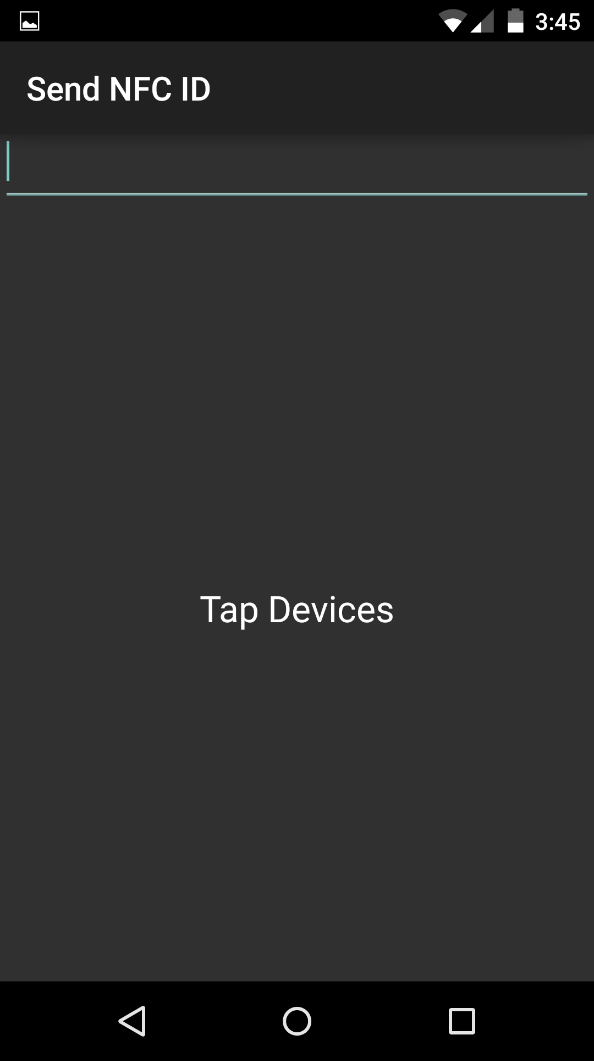
In order to make my ultimate goal of a ‘Wallet-Free’ future a reality I first had to research what other similar applications had to offer. Unfortunately there is not a wide variety of uses for NFC, but there are two implementations that show promise. Google Wallet and Metro Tap are two entities that share the use of NFC. Google Wallet is an application that allows the user to pay for items with the use of their phones. Metro Tap is an application that allows the user to pay with a refillable NFC card.

Both of these companies store information for their individual uses, and this is where my application differs. There are three things that NFC ID offers in which the competition does not. The availability of more information, increase in convenience and ease of use is were NFC ID prospers. NFC ID is not limited to just one ability, it stores information such as Names, ID number, and current work status. All this information is also conveniently located on one device, there is no need to have redundancies such as multiple identification cards or credit cards. NFC ID is also easier to use in the sense that the UI is simplified for the user by not overwhelming them with options, as well as not containing convoluted instructions to use the application.

**Software Interface**

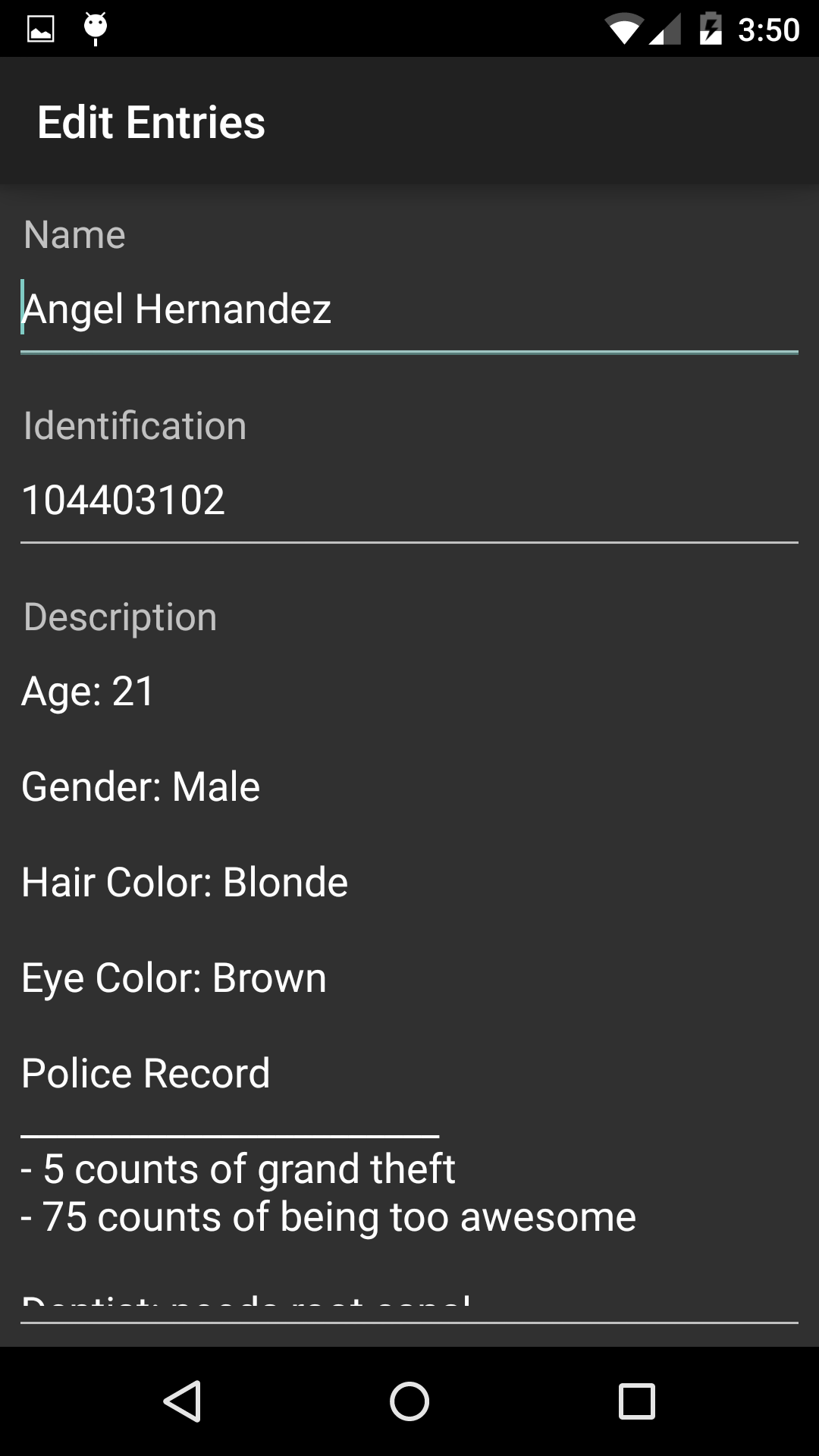
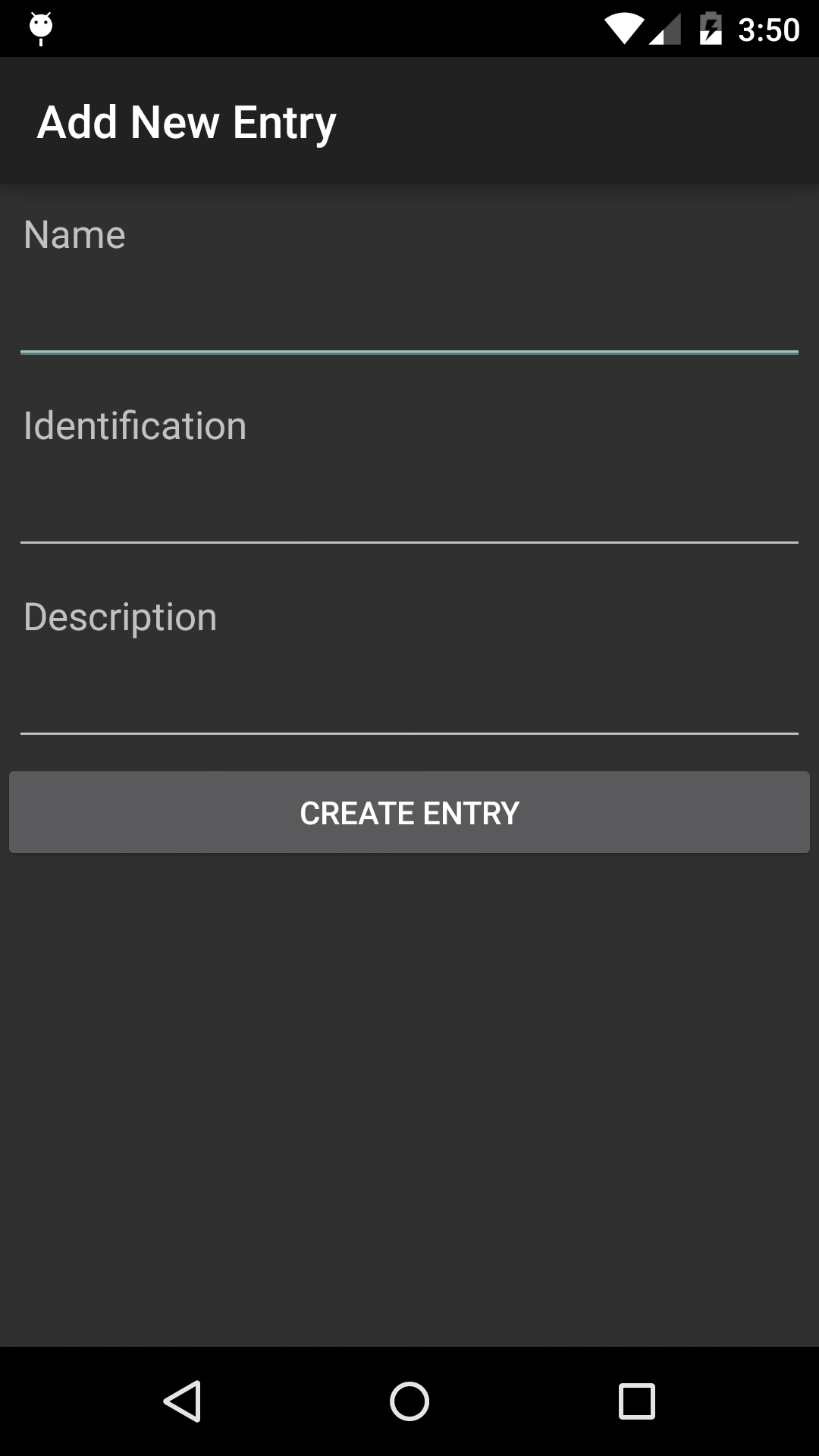
A major aspect of this application and project as a whole is to allow for a simple and straight forward system that anyone can navigate. This is something that is very important to me, an application can be the most versatile thing in the world but if it’s not user friendly, then it won’t make a difference. With this ideology my application consists of a UI that is clearly labeled and takes to account device functionality to reduce unneeded redundancies.

Figure 1 is a snap shot of the main user interface for NFC ID, in order to have a simple application a complimentary UI had to be made. By only having four buttons it makes the application easier to use. Having them clearly labeled also distinguishes to the user the options they currently have. Receive ID and Send ID make it clear to the user that these options are for NFC communication, while View Entries and Add New Entry have a clear indication of information lookup. To accompany these buttons I have also implemented toast bubbles that tell the user if they have received a valid entry key. If the user does not have a valid authorization key, then a short toast appears with the text, “You do not have a valid ID”.  
**Fig. 1 Main Menu**

**Fig. 2 Get NFC**  **Fig. 3 Send NFC**

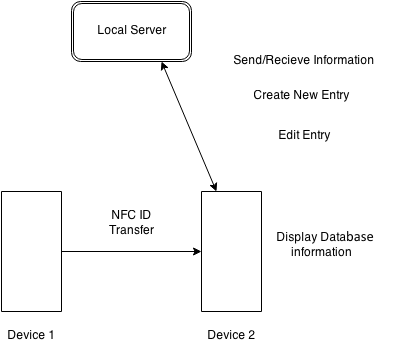
Next up is the Get and Send NFC UI. These two intents help transfer the authorization key from one device to another. The sender types in the key and then simply taps the device to the receiving application. These two menus display simple instructions with text visuals. This application also takes advantage of androids built in buttons. The bottom row is fully functional with the application, so there was no need to have a back button in the application itself.



**Fig. 4 New Entries Fig. 5 View Entries Fig. 6 Editing Entries**

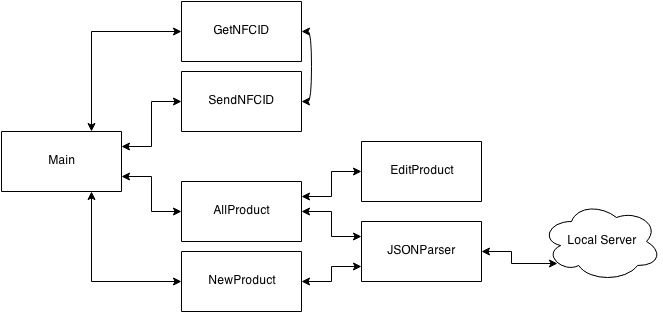
Figures 4 to 6 are the main structures of this application. Once the user receives a valid key, the device is now allowed to connect to the local server. The user can either view all entries (Fig. 5) or create a new entry (Fig. 4). In All Entries the user can locate what information is currently on the server, and when the user clicks on an entry it opens up the Edit Entries intent (Fig. 6). From here the user can see the entry, with all subset information. The user is now allowed to edit the information, if needed, and then save their changes to the local server.

**System Architecture**

****The physical system layout of NFC ID is simplistic in nature. The application uses two android devices that are NFC enabled, once they come in contact with each other a prompt appears for an NFC key transfer. The key to connect to the local database is the local ipv4 address of the hosted computer, this was due to the issue of hosting a server from an external source, the extra cost associated with it was not a feasible option. Once the key transfer is completed Device 2 gains access to the local database on the network. This “local server” is defined by a laptop that runs WAMP, a software which

**Fig. 7 System Diagram**

allows local hosting of a MYSQL database. The second device is capable of viewing information on the server, as well as have the option to create or edit existing entries.

****

**Fig. 8 Software Diagram**

Figure 8 illustrates the way NFC ID works. The main class contains four calls to other classes, GetNFCID and SendNFCID are accessed by the UI buttons on the main class. This is where the keys are sent from one application to another. This was also a design choice, by having these two classes, there was no need for a second standalone application. The sending and receiving could all be done within one app and as such, items associated with updating and testing were made easier. AllProduct and NewProduct are available once the user has input a key. Both of these classes connect to another class call JSONParser. JSONParser is where all the communication to the server takes place, it verifies if the key is correct, connects to the server, and then grabs the needed information. The main difference from AllProduct and NewProduct is the EditProduct class, which can only be accessed by AllProduct, in order to change information that was already stored within the server.

**Approach**

NFC ID was written in Java using the Eclipse IDE with an extension to the android SDK library. When working with this platform it was important to choose what device NFC ID would work for. Thus the required version of android was Android 2.3 minimum for the use of NFC and network access.

The NFC aspect of my application also saw two iterations before settling in using a local database to store information. The information was going to be available within the device itself as well as it being hosted on a local server. The ability to send all this information locally was extremely tedious, as each thing would have to be sent as a NFCEF (Near Field Communication Exchange Format), which meant multiple variables would have to store long character of information and would also cause a worry due to file size. If the file was too big, then the convenience of my application would be in jeopardy. Therefore everything was migrated onto a local database.

Network access is the second major aspect of my application. The use of a JSONParser class enabled me to quickly implement server connection. While everything seemed to work together, because of the database being only locally available, it became difficult to create a unique ID so that the user can authorize access to their information. For this version of the application a solution was found by using the ipv4 address of the host computer and making that the key. It was an unfortunate solution, but the core concept of the project was still intact.

**Testing and Verification**

NFC ID is the first application I have created that utilized network access. I had no prior knowledge to networking and felt that this was a great project for me to work on. Not only did I learn more about networking as a whole, but also realized the complex structure of devices that connect to a network, as well as all the associated properties that are involved with sending and receiving data.

The majority of testing was done in-house by means of constantly testing the network aspect of my code. Troubleshooting bugs such as failed network connections, or general local device connection seemed to take up most of the debugging and testing time.

The other main concern was over the UI. This second major testing phase helped me optimize the UI for ease of use and efficiency. Testers were given the application with only the description of what the application did. This black box test helped pinpoint locations in the application that gave users the most trouble. Buttons were resized and equally spaced in order to appear much cleaner, sending and receiving intents were given “Tap Devices” text to signify when NFC could be used, and the visualization of the database received a massive overhaul of how it visually represented information.

**References**

1. Dodson, Ben, Hristo Bojinov, and Monica S. Lam. "Touch and Run With Near Field Communication." *Dhwani: Secure Peer-to-Peer Acoustic NFC - Microsoft Research*. Stanford University, 10 Nov. 2010. Web. 11 Dec. 2014. <http://mobisocial.stanford.edu/papers/nfc.pdf>.
2. Nandakumar, Rajalakshmi, Krishna Kant Chintalapudi, Venkata Padmanabhan, and Ramarathnam Venkatesan. "Dhwani: Secure Peer-to-Peer Acoustic NFC." Dhwani: Secure Peer-to-Peer Acoustic NFC - Microsoft Research. Microsoft, 12 Aug. 2013. Web. 11 Dec. 2014. <http://research.microsoft.com/apps/pubs/default.aspx?id=192134>.