SMIDGE

The Smart Fridge System

Felipe Bernal, Arian Caraballo, Isabel Virag, Daniela Zicavo

School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This paper present the design and overview of a system developed for a smart fridge. Through the use of user friendly interfaces, the system allows the user to keep track of and add items to their inventory as well as maintain shopping lists and recipes. This smart fridge system is a system of subsystems. The main part of the system is the fridge client, which allows the user to interact with the system through a touch screen LCD and UPC scanner. The user can also access its information through a mobile application and a website. The fridge client and the mobile application are interfaces developed in Android. The web-based application can be accessed with any browser. Each application will contain its own database which will synchronize with the main database maintained in the server.**

***Index Terms* — UPC codes, barcode scanner, Android applications, ARM processor, smart fridge.**

I. Introduction

The team decided to engage in a smart fridge project for the technology challenges it represents and how beneficial it can be in ones everyday life. The project merges different technologies, both hardware and software, to come up with the final product. There have been some attempts at creating smart fridge systems but none have made a real breakthrough. This system will attempt to fulfill different areas of need. From the private sector point of view, this system can simplify homes, simplify life and give people back all that time that is wasted on dealing with kitchen inventories, shopping lists and complex recipes. From the industrial point of view, the creation of a database that will be linked to several hardware and software components, from which the database can be accessed and modified, is definitely an idea that can be used in future applications.

Although the team members had various options when it came to the design of the overall system, certain goals were really taken into account when selecting the optimal implementation technique. Some features that required a large amount of work and were thought of as extremely time consuming were not sacrificed for the sake of simplicity. The main goal of the project was to create a useful system that was very user friendly.

The team is comprised of four computer engineers and wanted to undertake in a project that was software heavy. The system contains three clients which contains a large amount of software. The development of web-based applications as well as the development of mobile phone applications has grown rapidly in demand within the past few years. For this reason, it was deemed important to undertake a project which would allow the team members to expand their technical foundation in these areas. The project also contains several hardware components to it which stood as a major challenge to the team members. Through self-study and research all of the tasks were tackled appropriately.

II. Motivation

What can you make that is a Caribbean dish in the seafood group category that is easy to make? How about an Italian recipe in the dairy category that is difficult to make? Questions like this were what inspired our group to create an application that would be able to put together a delicious meal from what you have in your fridge. In addition, instead of asking this question why not let the application know what you already have in stock at the moment of purchase or when you are running low of certain item, or if it is completely out of stock. Imagine this and some other features added to one of the most used appliances in any home.

The motivation for “Smidge” was to incorporate one of the most used appliances in the household to the era of smart devices. The idea came up when the group made the suggestion of a project that will make life a little bit easier for users. With “Smidge” you would not have to worry about running out of a particular item, or searching through your shelves looking for items that you need to buy to put them on a shopping list. “Smidge” is linked to your cell phone through an Android application so you will have access to your fridge and pantry inventory twenty four hour a day, seven days a week. Smidge database is also going to be synchronized to a website to make it easier to replace items or make adjustment to the inventory.

III. System Breakdown

The system can be described by the different components that make it up. There are three interfaces by which the user can interact with the system. The web-based application and the mobile application are both software only components of the system. The third interface, the fridge client, contains both software and

hardware components. The hardware consists of the printed circuit board which interfaces the scanner and the LCD display. The other main component is the database system. While this part might seem trivial, it consists of detailed design and various scripts needed to keep all of them synchronized. The system distribution can be seen in figure

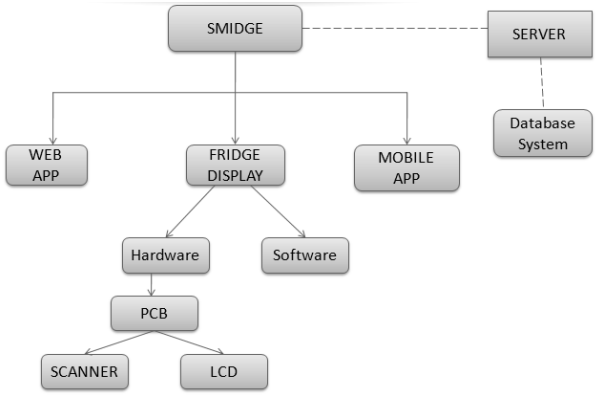

Fig. 1. System Breakdown Block Diagram.

The Barcode Scanner is in charge of reading the Universal Product Code located in any one of the products that will be placed inside the fridge, and delivering it as a sequence of numbers in the form of a keyboard input. This keyboard input will then be entered into a search for product recognition. The scanning system is responsible for linking this input read in by the scanner to the inventory API. The information is then retrieved and displayed in the LCD on the refrigerator system.

The touch Interface is provided through an LCD touch screen placed on top of the small refrigerator for convenient access. With this, the user is able to easily enter and delete products from the inventory on the system. Through this system, the user will have the option to enter a product or a batch of products; remove a product from the inventory; change the size or weight of the product as it progresses through its use; add expiration dates if desired with the option to notify the user when the expiration date is reached; add and modify shopping lists and finally, view any recipes already in the system.

The Web Interface resembles the touch interface very closely. It is responsible for offering yet another option for the user to check their current fridge inventory and make any changes; as well as create shopping lists, adjust shopping lists, create or delete recipes. The webpage plays a big part in the enhancement of user experience with the system and requires a personalized login account for complete privacy.

The Mobile Application provides one of the main advantages of the system, being able to access the inventory at any time, any place. Having the ability to access, create, and modify shopping lists while at the grocery store makes all the hard work of keeping track of groceries worthwhile. The contents of the mobile application are very similar to those of the website. Through the mobile application, the user is also able to access, modify and create new recipes on the go as well as modify items on the current inventory available.

Shopping Lists are created by the user in order to have a list of items desired to be purchased in the future. They are available for creation, viewing, modification and deletion through the three interfaces: mobile application, refrigerator system and website. The user should be able to add his personal shopping lists with different names. Also, there should be a shopping list automatically created by the system that provides an up-to-date list of items the system has analyzed and concluded the user shall need. This list will be based on items bought in the past, items the user has already disposed of, or items whose expiration date is approaching. Based on user preferences, the shopping lists will have the ability to get sent through email or mobile application notifications.

Recipes will be comprised of a list of ingredients and an itemized procedure divided into steps. The user will have to input these recipes into the system through the mobile application or website; however, the recipes should be viewable through all interfaces, including the refrigerator system. The system should be able to determine which recipes are available for creation based on the items currently available on the inventory. It should also be able to determine which items are missing in the inventory from a selected recipe.

The notification system will be implemented to alert the user about items soon to reach their expiration date, items soon to be discarded, undesirable temperature readings, or shopping lists selected through user preferences. It can be implemented to use email or mobile application notifications in a mobile phone.

IV. FRIDGE CLIENT

The fridge client is broken down into two parts; hardware and software. All of the hardware in the smart fridge system is contained in this subsystem of the overall system.

A. Fridge Client Hardware

The system needed a way to input the items being stored in the fridge by the user. Since most items purchased in a grocery store contain a UPC code, a barcode scanner

needed to be implemented in some sort of way. To provide the user with an interface that was easy to use and effective, several options were researched but ultimately it was decided that the system should run on Android. For these reasons mentioned, the system was ultimately implemented using an ARM processor. It was required that the system had a minimum speed of 667Mhz and had pop memory capacity of greater than 1 GB for both SDRAM and NAND. Additionally, the fridge client needed to be compact so this client needed to be implemented in a board of no more than four inches squared. After narrowing the processor choices down to three, OMAP 4430, OMAP 3530, and SC6410, it was determined that the OMAP 4430 and 3530 would be the best options available but the OMAP 3530 would be the only one that was easily accessible.

The system uses this processor to interface several components necessary for the functionality of the system. The TPS65950 Power Interface control is connected to the processor through several pins to interface the power supply, SD slot and USB host and connector. The processor also has pins which connect to the SD slot, USB host, reset buttons, and JTAG connector. [1] The board uses a 5V power connector to power the processor and components. This configuration can be seen in figure 1.

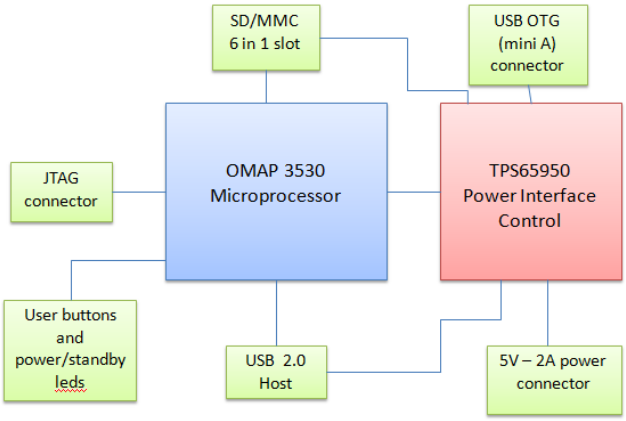

Fig. 2. Hardware Overview Block Diagram

To retrieve the information for each item, the system needed to be able to interface a UPC scanner with the processor. Several scanners were researched, some which were connected via USB. The best option, the MetroLogic's Scanglove, was selected due to various reasons. It is a durable, lightweight, wearable automatic-single line 1D bar code scanner that improves productivity as it increases scanning flexibility. It can be used as a back of hand scanner or a stationary desktop scanner or also as

Fig. 3. Metrologic ScanGlove.
(Reprinted with permission from Metrologic Intruments, Inc)

a compact, fixed mount presentation scanner. Its most important aspect is that as soon as it is plugged into a PC, and the barcode is presented to the scanner, the numbers are automatically entered into the computer, similar to a keyboard input; no special drivers are needed to be able to use the device.

The scanner can read up to 80 data characters and has a visible laser diode of 650mm. It has three indicators: if the LED is red, the laser is on and ready to scan. If it is green, it means it just read a barcode successfully. When it comes to power, the ScanGlove comes with its own USB power cord attached and gets power as well as outputs the code through this same cable. The unit also features an infrared sensor (IR), if a specified time has elapsed without any scanning, the unit will enter a "standby" mode. The unit reactivates automatically when it detects an object in front of the IR sensor. The last feature is that the scanner can be programmed to emit three beeps when a timeout occurs between the host and the scanner. [3]

An internet connection is required to connect to the UPC database and also to be able to keep all the databases synchronized. With this system in mind, wireless network connection allows for better system integration. The main concern in selecting a Wi-Fi module was selecting a module which was compatible with Android without the need of custom drivers. The Belkin Wireless G USB Network Adapter F5D7050 was selected for system implementation. Its 400 feet range, USB connectivity, and 5V operating voltage made it an optimal fit for the hardware system.

The fridge client will be equipped with an LCD touch screen display. The IMO Pivot Touch was selected as the best option. This device adds an element of speed and ease to the user experience when using the system. This touch screen display is also powered through the USB. It is a 7 inch display with a resistive touch-screen.

It has a resolution of 800 x 400 and its power consumption is 100 – 240V. [2] The IMO Pivot Touch has generic drivers and custom ones needed to be built for the display to work on the Android operating system.

B. Fridge Client Software

The fridge client is the main point of interaction with the user. This is the only client application in the system that allows the user to input items into its inventory. Since, the fridge display will be the main interaction point between the user and the system, it is crucial that this display is as user friendly as possible. There are five different areas which have been kept in mind when enhancing usability of this application. These include the learning curve, efficiency, ability to memorize, errors, and satisfaction. With these factors optimized, the fridge display contains an extremely easy to use interface.

This application, as well as the mobile phone application, is developed in Android. It makes use many of the features available to Android application developers. The fridge client application was developed in Eclipse Helios IDE with an Android Development Tools plugin. The target build version of Android used in development is 2.2 and additional emulator plugins, such as the Galaxy Tab emulator, were installed to give the developers a better idea of what the application would look like once installed on the board.

The fridge client application follows the framework described by Android applications. The diagram shown in figure 3 shows how an Android application is organized.

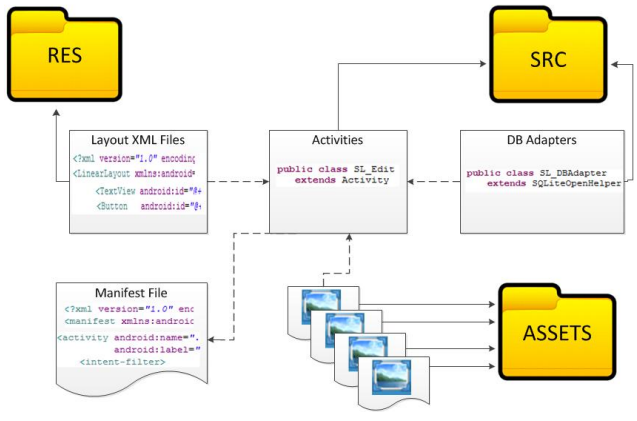

Fig. 4. Android application Framework

The application contains three main files which hold most of the files of the application. The source file contains all of the java files with all the source code which the application is made of. These files are usually one of three kinds. The first is an activity class. This is a java class which extends activity. Each page in an android application is an activity and is declared in these files. Another type of file in the source file is an adapter file. In the case of the fridge client application, most of these are adapters that abstract the connection to the database from the interface. These are java files that extend SQLiteOpenHelper which aid in getting and setting information into the local database. The last can be a simple java object used by any activity.

The assets folder contains items that can be referenced by other files in the application such as images, strings, and the like. The resources folder contains all of the XML files of the application. These files define the layout of each page in the application. Components such as textboxes, lists, titles, and so on can be modified in these files. Another major part of the framework is the Manifest file. This file declares all the activities used in the application as well as the main intent.

There are five main activities that reference back to main activity. Each of them contain the basic methods found in a common activity such as onCreate, linkbutton, onKeyDown, The inventory activity allows the user to access the information regarding what items are currently stored in the fridge. This activity uses a listview adapter to set display the values appropriately. It also uses the database adapter to get the information regarding the inventory table. The items activity allows the user to input any items into the system. This can be done with the scanner or through the produce item activity. This activity also uses a listview adapter and database adapter. This database adapter however accesses the items table as well as the inventory table. The structure of the database will be discussed in a future section. The other activities in this application consist of recipes activity and shopping list activity. These activities use their own adapters to properly display data and provide a useful interface for the user. These activities and the structure of the application can be seen in the basic class diagram displayed in figure 5. Although this diagram displays most of the classes involved in the fridge client, it does not include several classes that were required during development to facilitate the coding of the activities mentioned. There is an item object class which keeps track of the information of an item in an object. These objects are used on a list in the add items activity to keep track of what has been scanned. Additionally a parceable item object had to be created to be able to pass the items from one activity to the next. Specifically, when a user goes to the produce item page, the information from the add items page must be stored until the user comes back to that activity. The

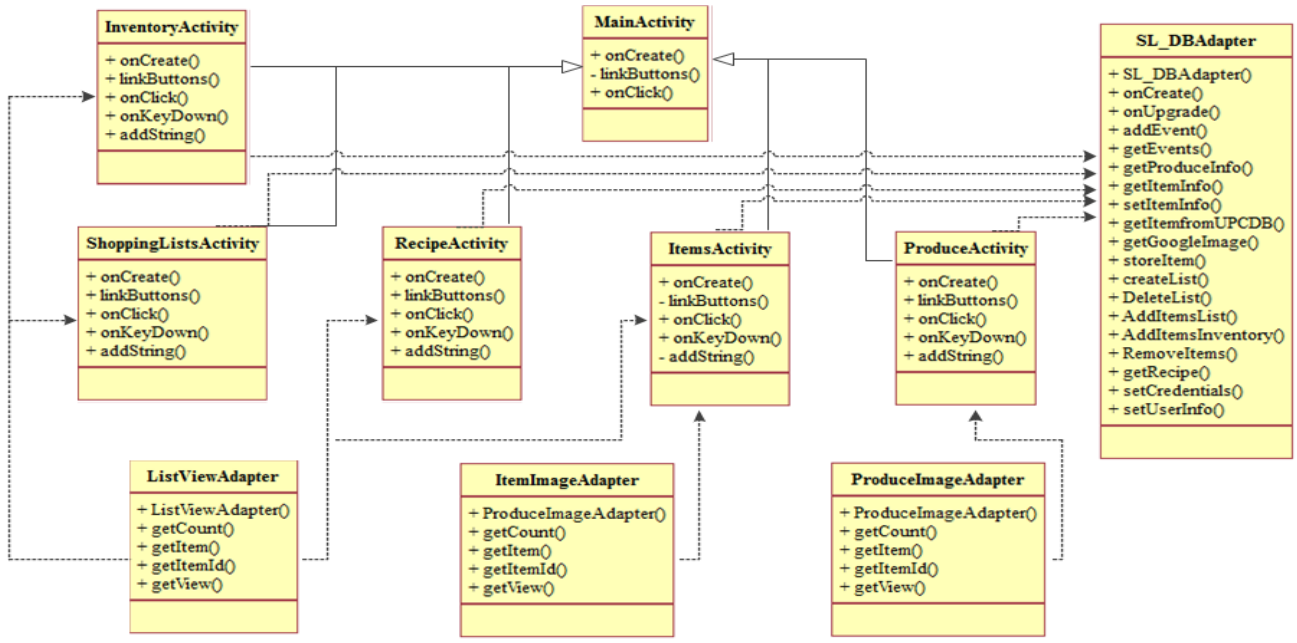

Fig. 5. Fridge client basic class diagram

implementation of these parceable objects was implemented using several classes. Without these, the information passed back and forth between add items activity and produce activity would have been significant, and the device would have experienced unexpected crashes due to memory latency.

The main process within the fridge client is that of adding items to the system. This process is illustrated in figure 6. Note that there are detailed steps which are not specified by this diagram and will be illustrated with

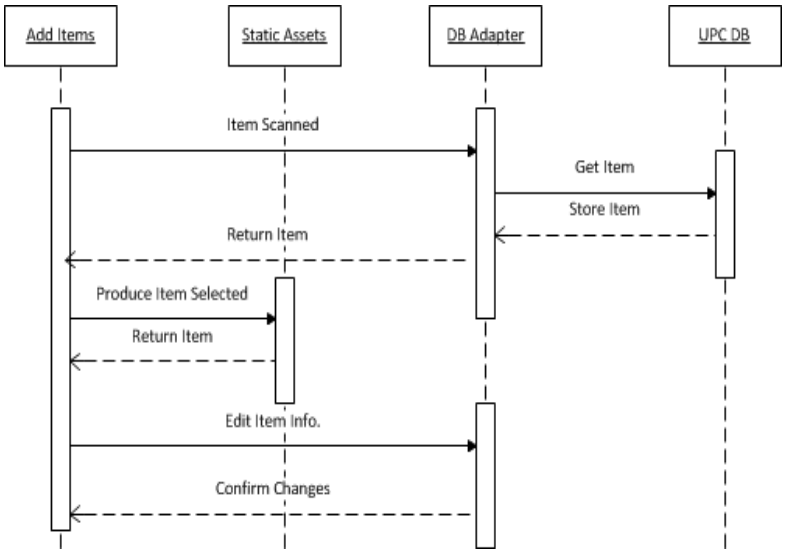

Fig. 6. Add Items sequence diagram

the actual display in a later section. When an item gets scanned, the static assets are bypassed. This is due to the fact that the amount of UPC codes is too large to be able to store it in this folder. The database adapter is accessed to see if that item is stored in the local database by using the appropriate function calls. If the item is stored in the local database then the item is returned to add items activity. If the item is not in the local database then the UPC database is accessed to retrieve the item given by this barcode.

The UPC database that is being used by the system is upcdatabase.com. This website provides an api to be able to connect to their system through our own Android application. Since this service charges for every item that is searched in their database, although it is very inexpensive, the items are stored in the local database in order to be able to retrieve that item again without having to go to the upc database to do so. This in essence works like a cache in that if the item has been called recently, the look up time for this item will be much faster the second time around.

When a produce item is selected, the static assets are accessed. All of the produce items, along with the PLU codes, images, and names are stored in the static assets folder. This procedure is in fact done in a separate activity but it is accessed from the add items activity. Once the produce item is retrieved, it is sent back to add items activity for processing. If the information of an item is edited, it is then forwarded to the database through the database

adapters. That information regarding an item is saved in the database and the changes are confirmed.

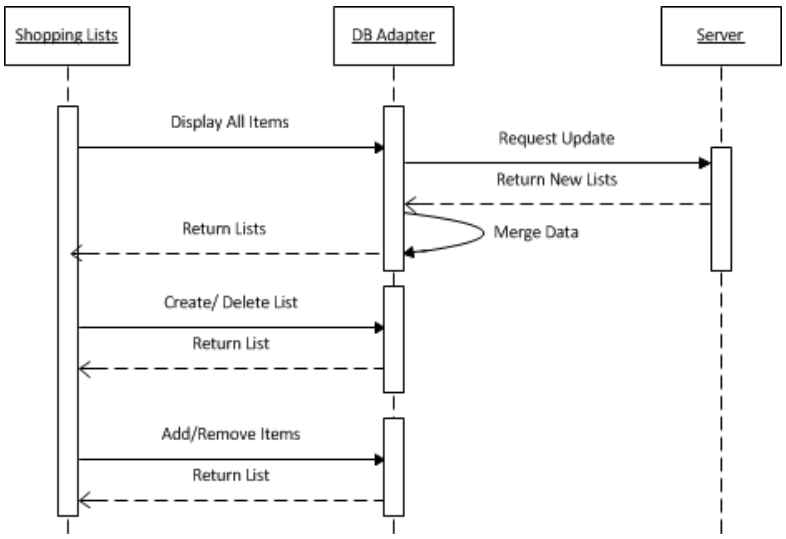

Fig. 7. Shopping List sequence diagram

The process of interacting with the shopping list feature of the fridge client is illustrated in figure 7. The shopping list activity requires all of the shopping lists to be synchronized in order to not only get the shopping lists that have been created in the fridge client but also any shopping lists that may have been created in the mobile phone application as well as the ones created in the website. This part of the process can be seen on the first arrow in the sequence diagram. The activity first requests the shopping list to the database adapter. Then the synchronization takes places when the adapter sends a request to the server to get the updated shopping lists from the server. The server gets that request and returns the updated list of shopping lists. Once the adapter gets this response from the server, it merges the shopping lists with the ones that were already on the local database. Now that the database adapter has the updated information regarding shopping list it can return the lists to the activity so they can be displayed. To create and delete a shopping list, the activity calls the database adapter to be able to perform the necessary data exchange with the local database. The same procedure is used for adding and removing an item.

The layout of the pages for the fridge client can be seen by looking at figures 8 and 9. Figure 8 shows a screenshot of what the main page looks like. Although functionality was the main priority of this system, user friendliness as well as look and feel were also taken into account. In the main page, the user can go to add items when new items need to be stored in the fridge, or it can go to inventory to view the items that are already stored in the fridge. Additionally, the main page has buttons to go to the recipes section of the fridge client as well as the shopping list section of the fridge client. The other two buttons seen in the main client are the help button and the settings button. The settings button allows the user to enter its credentials to be able to access their information in the website.

The picture seen on figure 9 shows a screenshot of the add items page. When a user scans an item, it is added to the list shown. Once the item is retrieved either from the local database or upcdatabase.com, this activity does a google image search for the item. That image is stored in the application and the path to the file is stored in the database as well. That image is then displayed as shown in the figure below. The user then has the option to edit the expiration date, quantity, and amount left. From this page the user can go to the Produce Items page.

Fig. 8. Fridge Client Main Page

Fig. 9. Fridge Client Add Items Page

The page shown in figure 10 allows the user to enter any produce items that need to be stored in the system. This page retrieves all of the produce items from the static assets and displays them. Since there are quite a few produce items to be displayed, the page takes a while to load all of the paths for the images for each produce item. This only happens the first time the page is accessed. To accommodate this inconvenience and increase the user friendliness of the page, a loading dialog is displayed and lets the user know. Once all the items have been loaded to the page, the dialog disappears and the page can be used as desired.

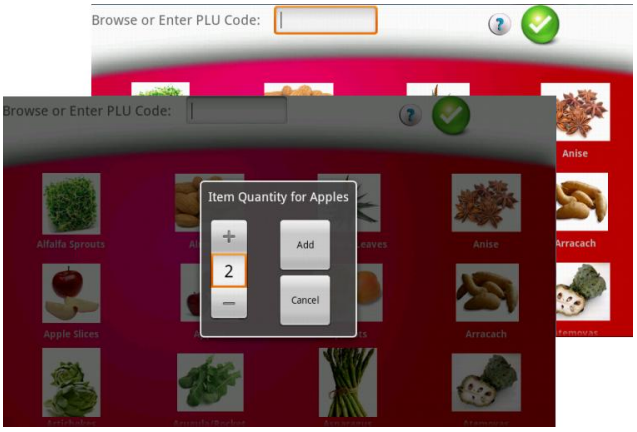

Fig. 10. Fridge Client Add Items Page

The shopping list page and the recipes page follow similar layout formats as the ones seen on figures 9 and 10. The recipe page allows you to add an item, which has been scanned at some point, to the list. It also allows the user to enter an item manually or to choose an item from the produce page. The recipe page allows the user to view the recipes in the database but not alter them. This process is done in the web based application or the mobile phone application.

V. Remote Applications

A. Mobile Phone Application

The mobile application is also an Android platform application. This application contains a similar functionality as the fridge client but varies in the layout of the application. The same goals are carried throughout the remote application especially having an application that is user friendly. From the mobile application, the user is able to view the information regarding the system from anywhere as long as they have internet connection. The mobile application allows the user to access the inventory, and perform any necessary procedures in recipes and shopping lists. One of the main advantages of having a mobile phone application is that it provides the user with and interface which he or she can use at any time. Specifically, the mobile phone application becomes very useful if the user wants to view the shopping list in the grocery store, as is usually the case, and wants to mark off the items that he or she has acquired. This functionality can be seen in figure 11. When the item is touched in the screen, it is checked of and the name is crossed off. The user can also add or remove items to the list as shown.

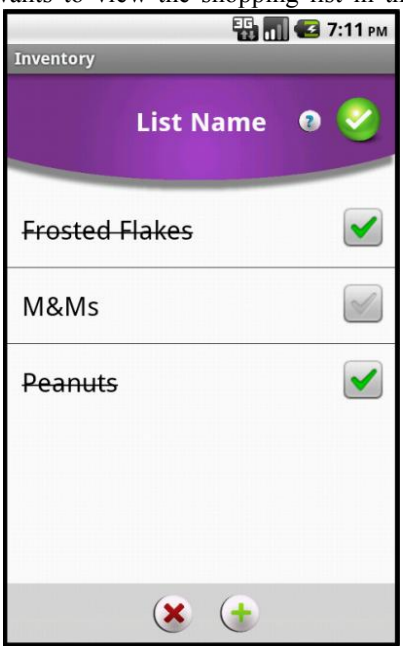

Fig. 11. Shopping List Mobile Application

B. Web-based Application

The web-based application is in place to allow the user to have an interface accessible through his or her computer and allow simple data manipulation. Since it's easier to type with a keyboard than inputting data through a touch screen, this application gives the user the ability to manage all of the information in the system quickly. The system also allows the user to manage recipes as desired.

This application was developed using PHP and javascript. Separate modules control the different sections of the page and it connects with the database in the server. Each user can access the website through the use of its username and password saved in the fridge client. The information displayed in the webpage is unique to that user's system information. The layout of this application can be seen in figure 12.

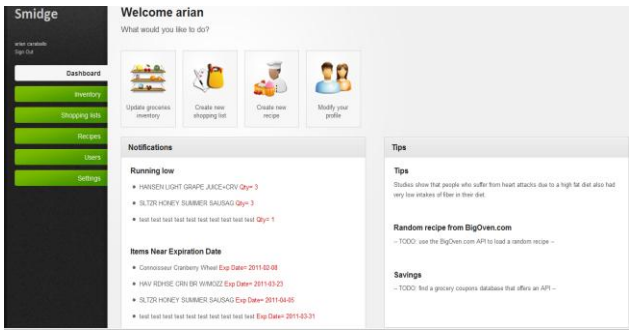

Fig. 10. Fridge Client Add Items Page

C. Database Systems & Database Connection API

The database section of this system is more cumbersome than most systems. While the data being used by the different applications is similar throughout, each application contains a separate database. It is critical to be able to successfully synchronize data when needed as needed. The web based application contains a separate functionality that allows the Android applications to synchronize their data with the database in the server. This is done through several forms stored in the server which allow the android applications to access the central database stored in the server. Each form performs a specific task that allows each application to synchronize their own database. This allows the applications to get, put, delete, and update any form of data being altered in the local database. To send the information to the form, a JSON object is created with the necessary data to be transferred to the main database. The form then gathers this information, parses it and performs the necessary operation according to the form. If there is data to be sent back to the android applications, the form too packages the information in a JSON object and sends the response back to the android application as needed. Any time there is a change in the local database of any of the android applications, a form is called to synchronize the main database accordingly. When the applications launch a specific activity, each table related to that activity is synchronized to ensure any changes to that table is done on the most up-to-date information. In essence, the database in the server will always have the latest data available. The database management system used for the web-based application is mySQL and the ones used for the Android applications is SQLite.

V.CONCLUSION

A system was developed to accomplish the concept of a smart fridge system. This system contained several subsystems which were comprised of both hardware and software. Many interconnections between database systems made several aspects of this system complex. The group took advantage of several features of the android framework to come up with a user friendly result. System properties such as layering, modularity, hierarchy, and abstraction, were all taken into account when designing the system to come up with an effective result.

REFERENCES

[1] Texas Instruments TPS 65950 Integrated Power Management IC Information Page, Website: http://focus.ti.com/docs/prod/folders/print/tps65950.html

[2] IMO Pivot Touch review information page, Website: http://www.hanselman.com/blog/ReviewMimoMonitorsMoPivot.aspx

[3] Metrologic IS4225 ScanGlove Wearable Scanner Product Information, Website: http://www.semicron.com/is4225.html

Felipe Bernal is a senior computer engineering student at the University of Central Florida and is scheduled to graduate in May 2011. Upon graduation, Felipe will join the consulting team at Citrix Systems where he has interned for the past three summers. He is a former Google Scholar and two-time Inroader of the year. Eventually, he plans on getting his masters in business administration.

Isabel Virag is currently a senior majoring in Computer Engineering at the University of Central Florida. Aside from being a full time engineering student, she participated on a lung cancer research project as part of the RAMP program and worked at the CAH Technology department before then. She currently holds a position in product development modeling as an entrepreneurship scholar of the YES program at Mydea Technologies and wishes to continue to work for this company in the future.

Arian Caraballo is currently a senior at the University of Central Florida and will receive his Bachelor's in Computer Engineering in May of 2011. After graduation he plans to join the workforce and has already accepted a position as a Software Engineer with his current employer.

Daniela Zicavo is a senior Computer Engineering student at UCF graduating in the spring 2011. She attended UCF from 2009 to 2011. Prior to that she attended Edison State College where she got her Associate in Arts Degree. For the last year she has work as a Research Assistant in the Computer Vision Lab at UCF. After graduating from UCF she plans on start working as a software engineer and prepare for her postgraduate studies.