Smart Food Dispenser for Dogs

Pedro Anibarro[1*] Harry Miranda[2†*]

[1,2]Department of Computer & Electrical Engineering,
Universidad del Turabo, Gurabo PR

Abstract

Many dog owners do not feed their dogs in the correct way thus causing problems such as obesity. According to the Association for Pet Obesity Prevention, 54% of Dogs in the United States are overweight or obese[1]. Eukanuba explains that timed feeding methods are ideal for overeaters, large breeds, obesity, and healthy dogs in general. This project allows a dog owner to program feeding schedules and portion sizes[2]. In addition, the system includes a monitoring system that keeps track of the amount of food eaten by the dog as well as the food supply. The system also includes humidity and temperature sensors to monitor food freshness. This project is composed of three parts: the food dispenser, a server, and a mobile application. The dispenser collects data from the sensors and sends them to the server. The server stores the dispenser data in a database for later use. This collected data is used to give the dog's owner feedback about eating habits, food spoilage, and assist with dietary restrictions by giving a report to the owner. The smart dispenser also includes a proximity sensor to detect the presence of a designated dog by employing a smart name tag to protect the food if the dog is not near. This system gives control to the user over his/her dogs eating habits meaning a better and healthier life for the dog.

*panibarro1@email.suagm.edu
†hmiranda12@email.suagm.edu

Contents

List of Figures

List of Tables

1 Introduction

This report proposes a solution to control a dog eating habit and monitor the state of the food by designing a smart food dispenser system based on Redbear Duo microcontroller in conjunction with humidity, temperature, weight, and proximity sensors. This is a Capstone 2 report, meaning that some improvements were made compared to Capstone 1. The report shows the before, after and the why some changes were made.

To control and monitor the dispenser, a mobile application designed for iOS and a web application is used. Figure 1 shows the general architecture of the solution. The device proposed has the potential to monitor and protect the dog food by giving the user a solution to establish the portion sizes, eating hours and time period of a dispenser. Only a dog wearing an intelligent name tag will have access to the food.

1.1 Problem Definition

People think that they know the basics of keeping their pets safe, and yet each year there are more than 100,000 cases of pet poisoning just in the United States. Items that are safe to handle and ingest for humans, including certain foods and medications we may take on a daily basis, can cause huge problems to the dogs. Some of the contaminants that can reach the dogs foods plates are:

- Rodent Poison
- Household Chemicals
- Insecticides

Bromethalin rodenticide toxicity, more commonly referred to as rat poisoning, occurs when a dog becomes exposed to the chemical bromethalin, a toxic substance that is found in a variety of rat and mice poisons. This type of poison may occur when the rodent goes to eats to dog plate and urine on the food. Common household products can poison your pet by falling into the food after being applied for it designed use.

On the other hand, feeding a dog the right way is not as easy as people think. According to Eukanuba[2], the best method for feeding your dog

depends on her size and personality. The portions sizes my vary by age and the dog physical activity.

Having said this, there are some fundamentals principles to feed a dog that many owners do not know or do not follow. One is that the dog have to eat at the same hour for the same period of time every day.

For example, a puppys meal schedule must include three measured meals a day, preferably at the same time every day. The best time for your puppys first meal is around 7 a.m., noontime for lunch, and 5 p.m. for dinner. The last meal should always be around 5 p.m. so that he will have ample time to digest his food and eliminate one last time before bedtime.

1.2 System Description

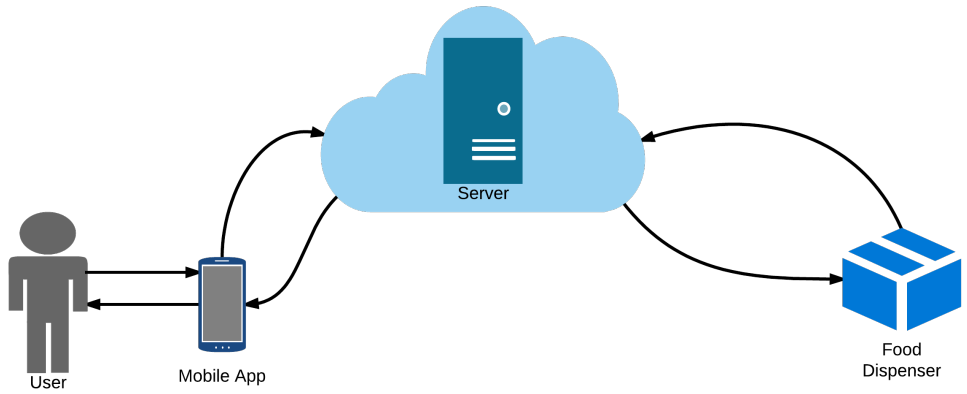

Figure 1: General Architecture

Knowing the facts all ready described on the problem definition, we designed a smart food dispenser for dogs capable of being programmed using an mobile application and a web application. Figure 1 shows the general architecture of the system. The user uses a mobile or web application to set the dog feeding hours and the time period that the dog will have access to the food if and only if the proximity sensor detects the name tag that the dog wears. After the period conclude, the dispenser will report the amount of food left by the dog if any, and the status of it (temperature and humidity) for quality control.

In order to protect the food and in case of the user owning more than one dog, our smart tag for a dog collar, communicates with the dispenser

using Radio Frequency IDentification (RFID) technology. This smart tag allows the dispenser to identify the correct dog that has permission to ingest the food. Also, this feature allows the dispenser to protect the food if the dogs is not near the plate when the meal time comes from rodents and other contaminants already mentioned.

The system includes a battery backup system to allow proper functioning of the dispenser if the energy coming from the energy provider is interrupted. We will have a server that will contain the databases and serve as the middle man between the dispensers and the users mobile applications.

1.3 Contributions

The most remarkable contribution of this design is the huge capability of protect the food by storing the plate inside the dispenser. Thus the only way that the food get exposed is when it gets dispatched for the dog, this means that the programmed hour has arrived and the dog is near. This solution and the feature of controlling the portion size, time of eating and period are good tools for people that owns dogs and don't have enough time to attend their pet, have really busy lives, or are owners that several dogs.

Is important to mention that the reason of the interest on designing this solution came from people having the problems described and more related ones that can be added as future work. Thus dogs owners can have more control of their dogs care and feel more close with them.

Also, based on the research, this dispenser can help decrease the dogs mortality by poisoned food, help prevent obesity, and control the dogs that have bad eating habits by controlling the portions and eating hours.

1.4 Projected Activities

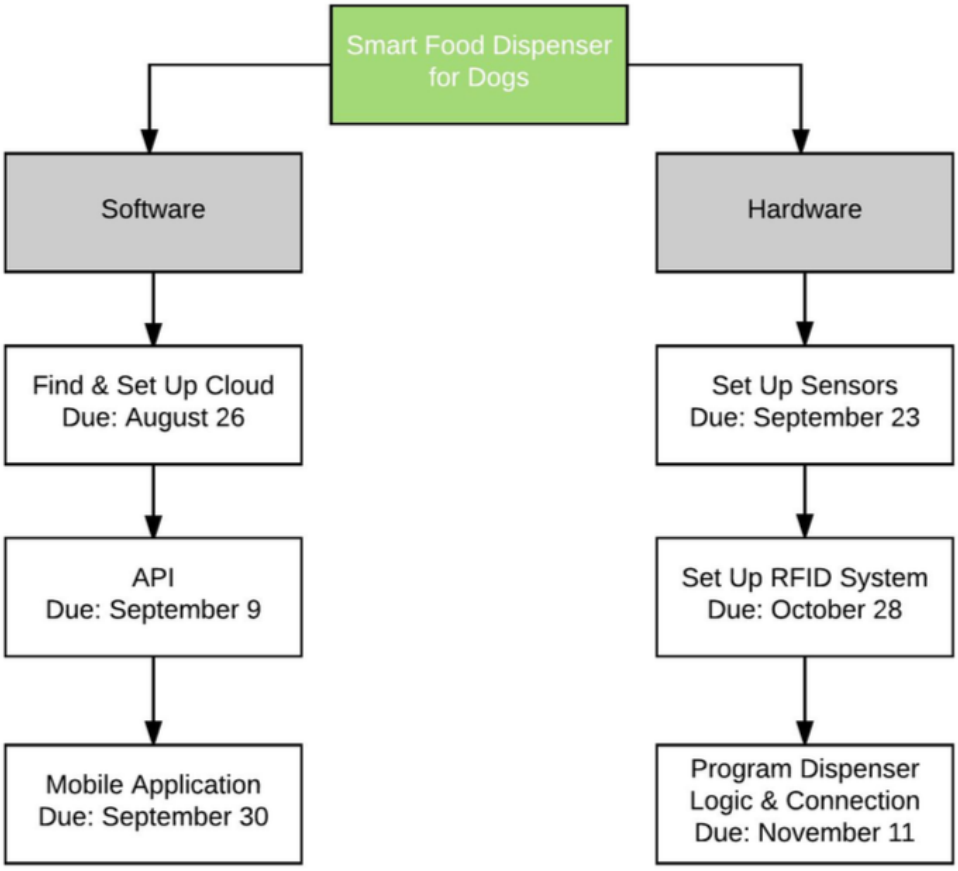

Figure 2: Projected Activities

1.5 Report Organization

This report contains all the information related to the design and development process of the dispenser. The information of the technologies where selected to use are described and explained on the Related Word section. The complete description of the system architecture and components selected to work on the plate are included on the System Description Section. Finally, in the Summary section we concluded all the information and provide a conclusion and future work.

2 Related Work

This section will explain every old and new technology used for Capstone 1 and 2 to give the reader some perspective and reasons of why we changed the technology.

2.1 Bluetooth Low Energy

Bluetooth Low Energy (BLE) is a wireless personal area network technology used for transmitting data over short distances[3]. As the name implies, it's designed for low energy consumption. The main purpose of BLE is for advertisements applications. The three main differences about BLE and Classic Bluetooth are:

- Power Consumption: BLE power consumption is very low. The battery can last up to 3 years on a single coin cell battery. This is ideal for some applications where you only need to advertise a device location.
- Cost: BLE is 60-80% cheaper than traditional Bluetooth.
- Application: BLE is ideal for simple applications requiring small periodic data transfers. Classic Bluetooth is preferred for more complex applications requiring consistent communication and more data throughput.

2.2 Beacons

Beacons are one-way transmitters that are used to mark important places and objects[4]. Typically, a beacon is visible to a user's device from a range

of a few meters, allowing for highly context-sensitive use cases. These devices are a small low-cost piece of hardware, that use battery friendly Bluetooth Low Energy connections to transmit messages or prompts directly to a smart phone or tablet. Some of the most use cases scenarios are:

- Marketing
- Place Tips
- Indoor location tracking

Figure 3 shows hows a beacon scenario. There are different standards of beacons available. These specifications defines the format of the advertisement message that BLE proximity beacons broadcast. Some of the most famous beacons standards are:

- iBeacon: Apple's own protocol specification
- Eddystone: Google open source protocol specification
- AltBeacon: Emerging open source protocol to compete with iBeacon from Radius Networks

2.3 Estimotes

Estimote Beacons an Stickers are small wireless sensors that can attach to any location or object. They broadcast tiny radio signals which a smartphone can receive and interpret, unlocking micro-location and contextual awareness[5]. Estimote offers a SDK so that applications can understand their proximity to nearby locations and objects, recognizing their type, ownership, approximate location, temperature and motion. This data can be used to build a new generation of magical mobile applications to connect the real world to a smart device. Estimote Beacons are certified Apple iBeacon compatible as well as support Eddystone. In this project we are using Estimote Beacon Sticker as a dog smart name tag.

Figure 3: Beacon Communication using Bluetooth Low Energy

2.4 Socket.IO

Socket.IO enables real-time bidirectional event-based communication[6]. It works on every platform, browser or device, focusing equally on reliability and speed. Also, it simplifies the WebSocket API and unifies the APIs of its fallback transports. Some of the main features that Socket.IO includes are:

- Uses HTTP
- Bi-directional communication
- Can declare what transports to use and the priority of the fallback
- Include WebSocket typical connect and disconnect events.
- It uses multiplexing
- Multiplexing enables namespaces
- Scalable by enabling multiple nodes

Fallback transports that Socket.IO includes[7]:

- WebSocket
- Flash Socket
- AJAX long-polling
- AJAX multi part streaming
- IFrame
- JSONP polling

2.5 RedBear Duo

The RedBear Duo is a thumb-size development board designed to simplify the process of building Internet of Things (IoT) products. It provides Wi-Fi, BLE and a powerful Cloud backend, all in a compact form factor that makes it ideal for prototyping, a finished product, and everything in between[8]. This microcontroller is cloud ready thanks to a partnership with Particle. A developer can push any changes and data via Cloud right from the start. This is helpful in the development phase and distribution phase because we can update all the microcontrollers at the same time remotely.

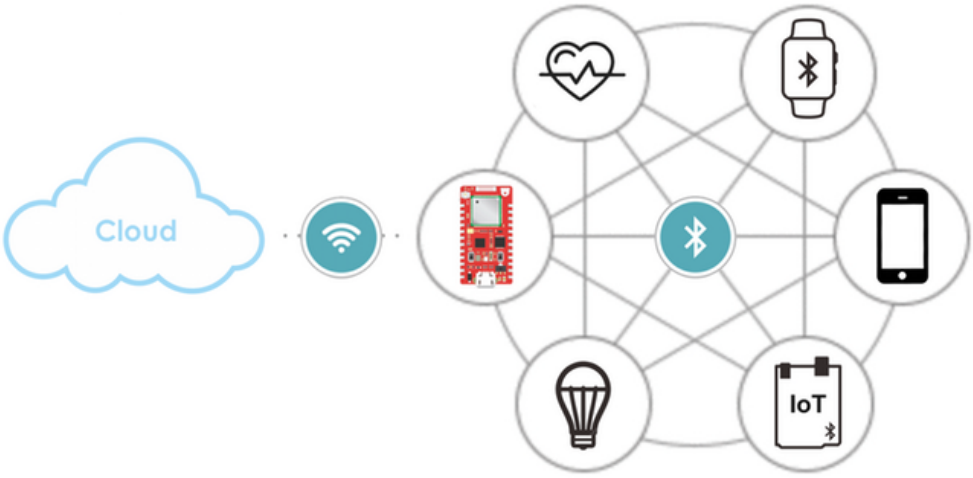

Figure 4: RedBear Duo Flow Diagram

2.6 Radio Frequency IDentifiers (RFID)

Radio or wireless is a way of transmitting energy through empty spacethat is, instead of using a wire cable. The energy is carried by invisible waves of electricity and magnetism that vibrate through the air at the speed of light. RFID tags is a technology where a tag returns an ID when its hit by a specified frequency.

Simple RFID tags are described as passive. Instead of containing batteries, they work entirely by responding to the incoming radio waves from the scanner or transmitter. There is just enough energy in those radio waves to activate the RFID chip. Passive tags typically send and receive signals only a few centimeters, but not much more. An alternative form of RFID technology, known as active tags, contain more advanced chips and tiny batteries to power them. They can send and receive signals over much greater distances[9].

Passive RFID tags contain just three components:

- The antennacatches incoming radio waves and sends them back out again.
- The chipgenerates a unique identifier code for the particular tag.
- The substratethe backing material (typically paper or plastic) to which the antenna and chip are fixed.

2.7 AngularJS

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly[10]. Angular's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.

3 System Description

There are many ways to attack the explained problem. The solution has to tackle the following main problems:

- Program feeding schedules
- Control portion sizes
- Send reports to user about how much food is left
- Monitor food freshness
- Aware of dog presence

These problems must be solved with usable and mobile-first as a priority. Having this in mind, there are some characteristics that are very important. Because we are working with dogs, safety is a very important aspect of the project. We must design with safety in mind so that the life of the animal is out of danger. Also, the ease of use, not just for the dog's owner, also for the dog itself is very important. In the decision matrices, from the appendix, we can see how the decisions of the design were taken with this characteristics in mind. From table 1-8 is shown all the decision matrices for the project.

There are some components that were change due to imporovements that needed to be done on the design of the dispenser. The most significant change is the microcontroller. Because of how cost-efective is and the low power consumption.

Like we can see in the decision matrices, we have several categories of components. Here is a summary of the component that we choose according to the tables. Components:

- Controller: Raspberry Pi 3
- Proximity: Bluetooth Low Energy
- Sensors: DHT11
- Load Cell: Straight Bar (TAL220)
- Connection: Wi-Fi Huzzah
- Server: Node.js
- Database: PostgreSQL

The decisions were made for all categories. However, we will see in the next subsection how we combine the controller, proximity and connection in just one component thanks to the capabilities of the Raspberry Pi 3.

3.1 Improvements

Because we are in Capstone 2, we needed to improve some aspects of our project compared to Capstone 1. The list of improvements are:

- SmartTag based on RFID
- Control Unit Upgrade
- Measuring amount of food
- Battery Backup
- Web Application

3.2 Food Dispenser

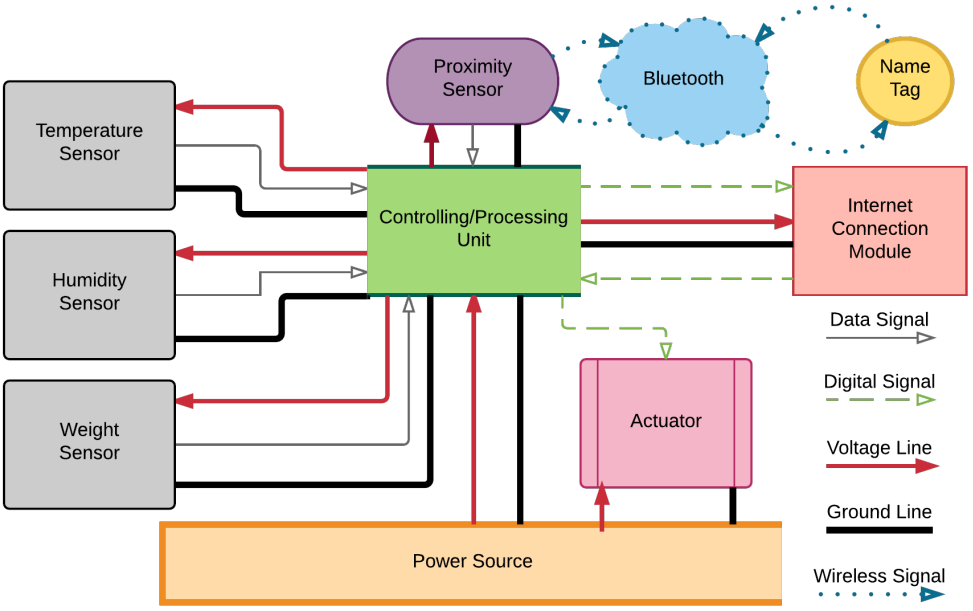

Figure 5: Dispenser Architecture

3.2.1 Controller/Proximity/Connection Unit

The Controller/Processing Unit is the component or that will control every process the may be required. This basically is the brain of the dispenser. In table 2 we can see that the Raspberry Pi 3 won in the decision process.

We choose the Raspberry Pi 3 because of all the features that includes this computer in a small package for its economic price of $35. We are going to use the built in Wi-Fi and the Bluetooth Low Energy as our Connection and Proximity module respectively. This controlling unit will be in charge of controlling:

- Being aware of dog presence using the BLE
- Read temperature and humidity from both, food bowl and food container
- Read how much weight is in both the food bowl and the food container
- Connect to the Internet
- Send data collected to the server

As for the Proximity Unit, we are using an Estimote[11] beacon as the smart name tag that the dispenser will detect by using BLE. If the smart name tag is 1 feet nearby and is time to eat according to the schedule that the owner programed, the dispenser will open the bowl of food. As soon as the dog is not nearby or the time of eating is due, the bowl will close the bowl of food. The smart name tag beacon has an approximately battery life of one year. The battery life status is displayed to the user in the mobile application. The battery is not interchangeable meaning that each year, the owner would need to buy a new smart name tag. The name tag cost only $10.

The Connection Unit of our system is solved using the built in WiFi module that the Raspberry Pi 3 has. We are going to use this module to connect to the server via Internet using Socket.io.

3.2.2 Sensors

The design includes three types of sensors to measure important characteristics of the food. We included temperature and humidity sensors to measure

food freshness and a weight sensor to know how much food is left on both the food bowl and food container.

For the temperature and humidity sensor, as table 4 shows, we selected the DHT11 sensor. The DHT11 sensor includes both temperature and humidity sensor in a single body. Also, the output of the module is digital. This is perfect as the Raspberry Pi 3 do not include any analog input.

The system should report how the dog is eating referring to his habits. This is possibly in our design thanks to the weight sensor. The dispenser will measure how much food is left in the bowl. With the collected data, the system should report to the user mobile application if the dog did eat or not. Using this information, we can generate weekly reports about the eating habits of the animal. This way, the dog's owner can know how his/her dog is eating and can monitor the dog's health.

3.2.3 Battery Backup

To calculate the system power consumption we used the Raspberry Pi 3 and DC Motors peak Voltages and Currents. This way we can know the how much energy we need to supply to the system in the case it needs it. Also, this is helpful information to select the best miliAmps per Hour battery for the Battery backup that the system will have.

The sensors power will be provided from the Raspberry Pi and this values are assumed by the energy consumed from it. The Raspberry Pi peak voltage and amperage are 5.2V and 2.5A, making it to dispel $P = IV$ around 13 Watts of electric energy. The same way, the DC motors selected for this project works with a peak voltage and current of 6V and 1.7 Amps with load, meaning a that booth will consume 20.4 Watts on their peaks values.

Knowing that additive in any configuration of resistive circuit, the total peak power of the circuit is approximately 34 Watts. The system works with 6V and the for calculate the current that the circuits needs for its peak values the same formula of power[12] has been used as follows:

- $P = IV$
- $I = P/V$
- $I = 34W/6V$
- $I = 5.667A$

From doing some research for choosing the correct battery for backup, the type selected is a Li-Ion one for various reasons. Then . Some of them are:

1. Usable Capacity

    - Unlike with lead acid batteries, it is considered practical to regularly use 85% or more of the rated capacity of a lithium battery bank, and occasionally more.

2. Fast & Efficient Charging

    - Lithium-ion batteries can be fast charged to 100% of capacity. Unlike with lead acid, there is no need for an absorption phase to get the final 20% stored.

To meet peak values system we need a battery of about 1.2 Amps per hour to keep the circuit in operation for two hours at peak values. these values can be improved by working more efficiently the circuit is to be implemented.

3.3 Server

To be able to connect the dispenser with the mobile application no matter where the user is, we need a server. As we can see in table 7, the server is a Node.js server. We selected to develop our server in Node because for its ease of use, yet powerful framework. In Node, using the Express framework, we develop an RESTful API. We decided to go with an HTTP API because of how compatible is.

Our server will be our intermediate between a dispenser and a user mobile application. It will have 2 databases. One PostgreSQL database hosted Amazon Web Services (AWS) with all the information about dispensers, sensors, feeding schedule, data about food temperature and humidity, and a second one hosted in Google Firebase Authentication Cloud for security purposes. Figure 7 shows the entity relationship diagram of the database.

3.4 Mobile Application

To complete the general architecture as shown in figure 1, we designed an iOS application for the dog's owner. This application will let the user log in into

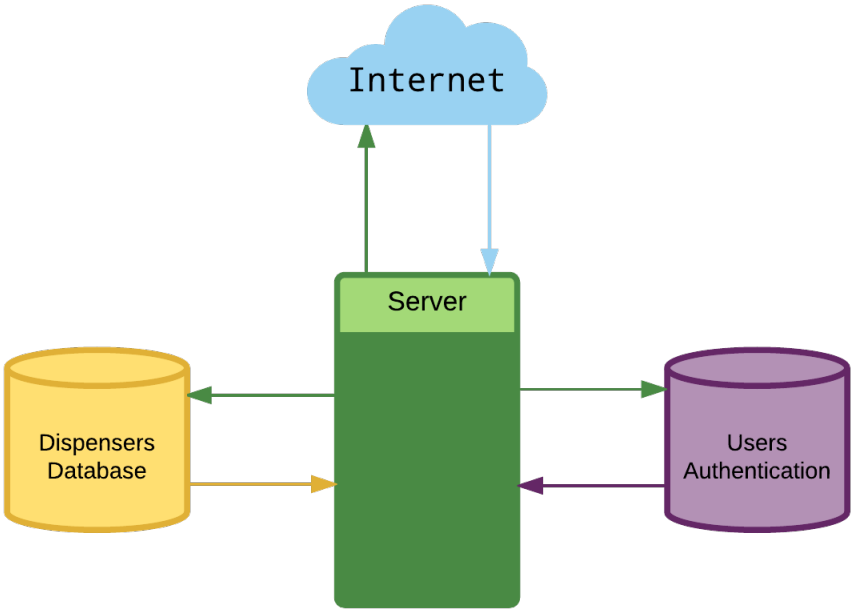

Figure 6: Server Architecture

an account, view all user's dispensers, program schedules, view eating reports and monitor food freshness. The application communicates with the server using our API. The application is divided in three parts. Figure 9 shows the mobile application flowchart with the three parts indicated. Appendix B shows all the images of the scenes.

The first one is the Sign Up/Log In scenes. This is where the user need to authenticate before having access to the dispensers configurations and main features. The authentication process will use Google Firebase Authentication Cloud. This service allows the developer to add a simplified authentication process where you can create a custom account or sign up/ log in with a Facebook account.

The next part of the application is the dispensers list. This scenes will allow the user to see all the proper registered dispensers. Additionally to that, the user will have the option to configure a new dispenser. This process will consist of scanning for near dispensers using Bluetooth. After detecting a dispenser, this link will be saved in the server database. During the configuration process, the user will need to configure the WiFi network of the

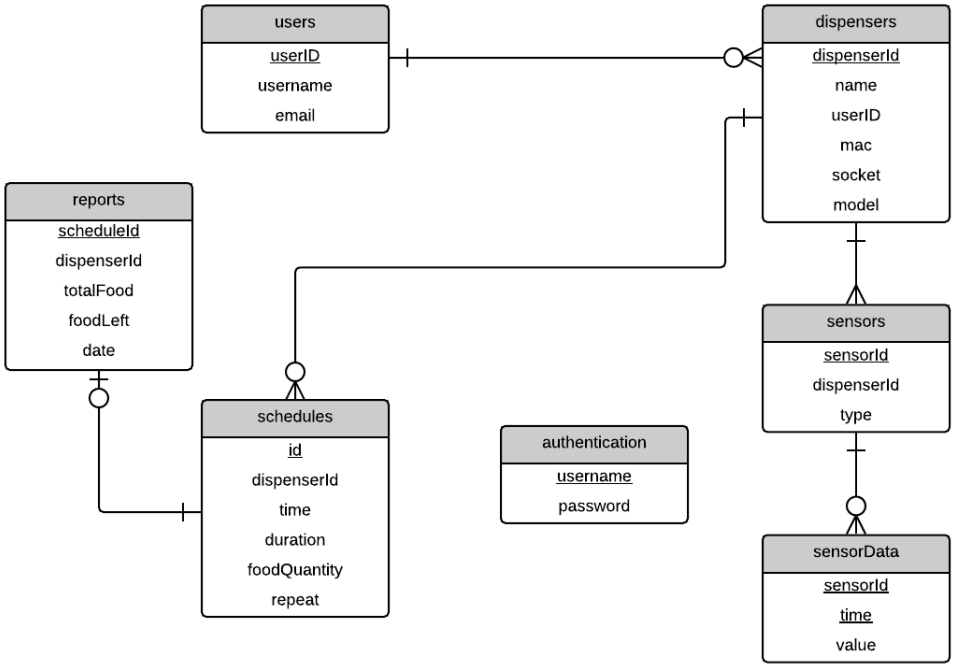

Figure 7: Entity Relationship Diagram

dispenser. The user will see a list of the available WiFis networks that the dispenser detected, and will have to select one. The user must input the corresponding network password and send it back to the dispenser. After making this configuration, all the communication between the dispenser and the mobile application will be by using the server as the middle man.

The last part of the mobile application consist of the main features. The user will be able to schedule, monitor, and see reports of eating habits of a dog. In the scene of scheduling eating times, the user will need to input from what hour to what hour, the portion size, and if it repeats daily. In the monitor scene, the user will have a table with the data collected from the temperature and humidity sensors. And last, in the report scene the user will see graphics about how the dog is eating and a table with all the collected data from the dispenser.

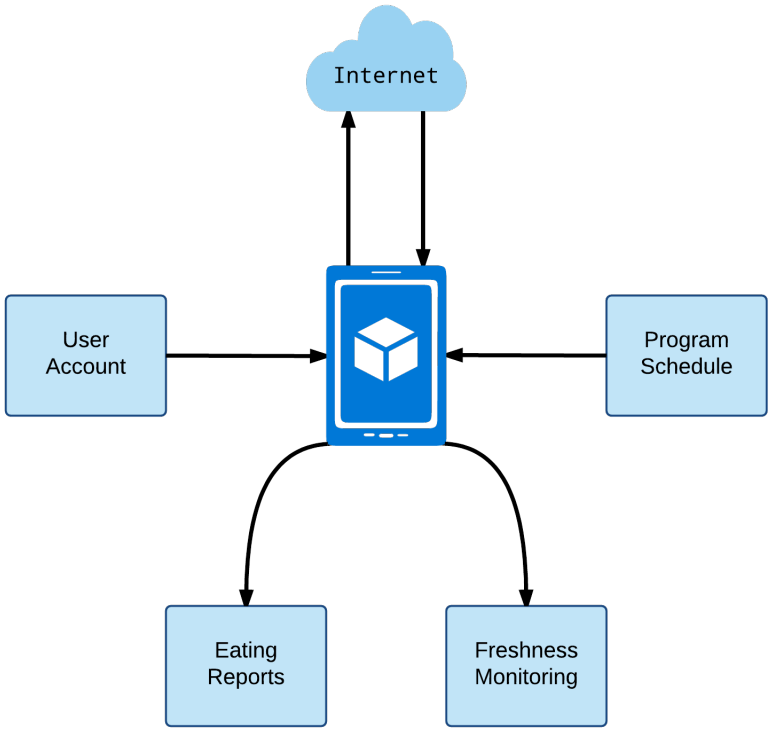

Figure 8: Mobile Application Architecture

3.5 Web Application

As an improvement, we develop a web application that clones all the features of the mobile application. The reason to develop this technology, is to give options to the customer. The web application will let the user log in into an account, view all user's dispensers, program schedules, view eating reports and monitor food freshness. We used AngularJS 2 as our front end framework. AngularJS 2 let the developer to create single page applications that are fully cross platform with speed and high performace. Appendix C shows all the web pages.

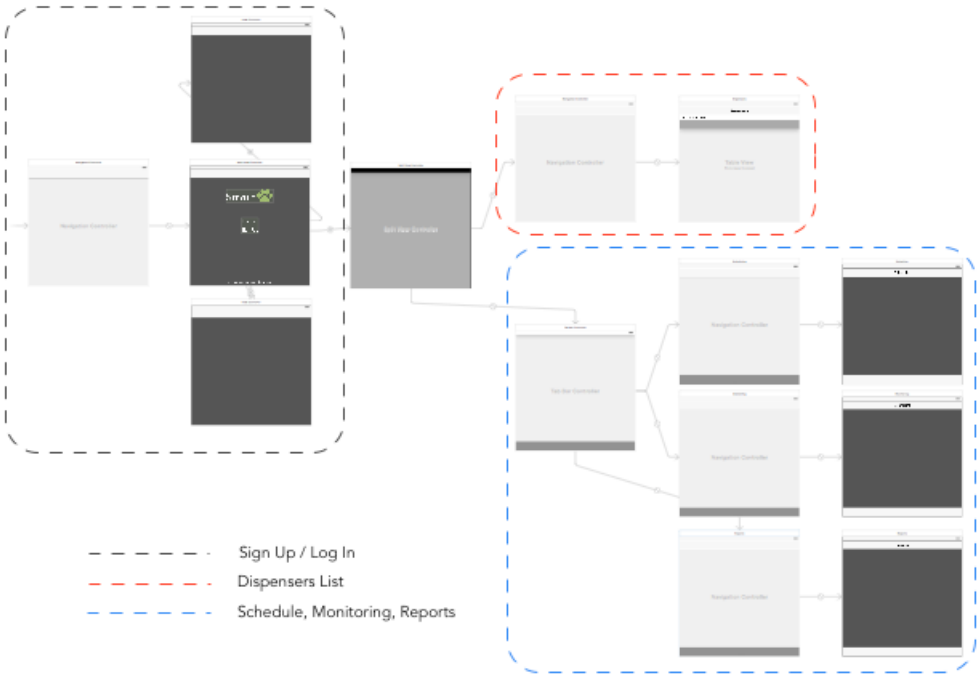

Figure 9: Mobile Application Flowchart

4 Conclusions and Future Work

This report proposes a system that solve some of the main problems that dog owners have when feeding their pets. The system uses a Raspberry Pi 3 as the brain of the dispenser and protects the food when a smart name tag is not nearby. This smart name tag uses the beacon technology using BLE. Also, the dispenser monitor and reports to the user food freshness and dog's eating habits. All this is reported and programmed by using a mobile application. Our system is in the early stages of development. We were able to program the dispenser to detect the smart name tag. Also, temperature and humidity data is sent to the server using Socket.IO. In order to complete the design, we need to work the aesthetics of the dispenser and how the mechanical aspects of the dispenser should work.

For future work, we need to implement the proposed design, consider water integration in the dispenser, and research more if RFID is a more suitable technology for the smart name tag. For this design we choose BLE

because of all the advantages it has over RFID but we need to consult more the safety aspect of the battery in a dog's collar with a veterinary or an expert in a related area. Also, we should consider if a single dispenser can be used with several dogs.

The most remarkable contribution of this design is the huge capability of protecting the food by retracting the plate inside the dispenser. Is important to mention that the interest on designing this solution came from people having the problems described in the problem definition. We are contributing and simplifying peoples live by empowering them to have control over his/her dog's eating habits.

Acknowledgment

Special thanks to Dr. Vergara for supporting this project with his mentorship and guidance.

References

[1] APOP, "Association for Pet Obesity Prevention (APOP)," 2016.

[2] Eukanuba, "Healthy Eating Habits for Dogs — Eukanuba," 2016.

[3] IBeacon.com, "What is an iBeacon?," 2016.

[4] Google, "Beacons — Google Developers," 2016.

[5] Estimote, "Estimote Beacons real world context for your apps," 2016.

[6] G. Rauch, "Introducing Socket.IO," 2014.

[7] F. Kelleher, "Understanding Socket.IO," 2014.

[8] R. Labs, "redbear:Duo: Resources for the RedBear Duo IoT development board," 2016.

[9] E. that Stuff, "How do RFID and RF tags work? - Explain that Stuff."

[10] AngularJS, "AngularJS: Developer Guide: Introduction," 2016.

[11] E. K, "Beacon tracking with Node.js," 2014.

[12] All About Circuits, *Power Calculations : Series And Parallel Circuits - Electronics Textbook*.

A Decision Matrices

Legend	Scale 1-10
Not care	1
Care	2
Interested	3
Important	4
Crucial	5

Table 1: Legend of Decision Matrix

	Microcontroller	Architecture	Operating Voltage	Input Voltage (recommended)	Input Voltage (limit)	Digital I/O Pins	PWM Digital I/O Pins	Analog Input Pins	DC Current per I/O Pin	DC Current for 3.3V Pin	Flash Memory	SRAM	EEPROM	Clock Speed	Features	Length	Width	Price	Total
Arduino Uno	7	6	7	8	8	7	7	7	7	7	5	5	5	4	1	7	7	7	347
Arduino 101	7	8	8	8	8	7	6	7	7	7	6	6	5	5	5	7	7	6	375
Arduino Zero	7	8	8	8	8	8	8	7	6	7	7	7	5	6	1	7	7	5	371
Teensy 2.0	7	6	7	6	8	9	7	8	6	7	8	8	5	7	1	8	8	8	397
Teensy++ 2.0	7	6	7	6	8	10	8	7	6	7	9	8	7	4	1	7	7	7	391
Teensy 3.0	7	8	7	6	8	9	8	9	6	7	9	8	6	4	1	7	7	7	397
Teensy 3.1	7	8	7	6	8	9	9	10	6	7	9	9	6	7	1	7	7	7	417
Raspberry Pi 3	7	10	8	7	8	8	9	1	6	7	10	10	7	10	10	7	7	10	448

Table 2: Controller Decision Matrix

	Frequency	Operating Volts	Current	Reading Range	Dimensions	Pins	Output	Encoding	Cost	Total
ID-12LA	6	7	7	3	6	6	7	7	5	160
ID-20LA	6	7	6	5	5	6	7	7	4	159
nRF8001	7	9	10	10	10	7	7	7	7	237

Table 3: Proximity Decision Matrix

	Sensing Data	Operating Volts	Current	Reading Range	Sampling Rate	Dimensions	Pins	Output	Operating Temperature	Cost	Total
LM35	6	5	8	10	8	8	5	7	8	7	240
DHT11	10	7	8	7	5	6	10	8	7	8	284
DHT22	10	7	8	9	6	6	10	8	8	6	283
MCP9700A	6	9	10	9	8	8	5	7	8	9	271
808H5V5	6	7	9	9	3	7	5	7	7	2	215
TMP36	6	9	10	9	7	8	5	7	8	7	260

Table 4: Sensors Decision Matrix

	Technology	Range	Dimensions	Pins	Cost	Total
Load Cell - 200kg, Disc (TAS606)	7	10	8	7	2	133
Force Sensitive Resistor - Square	5	7	9	9	8	149
Load Cell - 10kg, Straight Bar (TAL220)	7	7	7	7	9	150

Table 5: Load Cell Decision Matrix

	Frequency	Protocol	Antenna	Operating Volts	Current	Range	Sensitivity	Dimensions	Pins	Output	Encryption	Cost	Total
WiFi: HUZZAH ESP8266	8	8	10	7	7	7	7	8	7	7	7	9	311
XBee: Zigbee / 802.15.4	8	8	7	8	8	8	7	7	7	7	7	6	298
WiFi: RN-XV Data Sheet	8	8	7	7	8	7	7	6	7	8	7	5	285

Table 6: Connection Decision Matrix

	Multithread	Socket.io	Performance	Community	Modules	Common Usage	Total
Java Server	8	1	8	8	7	8	130
Node	8	10	10	7	8	10	187
Apache	8	1	8	8	7	1	123

Table 7: Server Decision Matrix

B Mobile Application

C Web Application

	Multi-Users	Portable	Secure	Scalable	Performance	Total
MySQL	10	7	8	8	8	156
Postgres	10	7	10	8	10	176
SQLite	1	10	8	8	8	135

Table 8: Database Decision Matrix

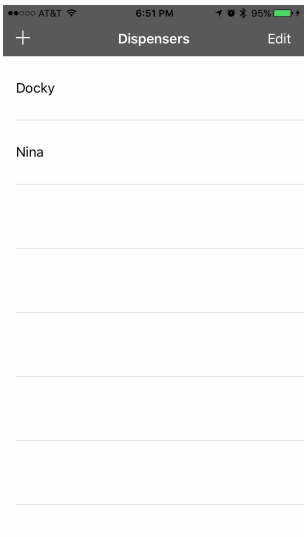

Figure 10: Mobile Application Dispensers

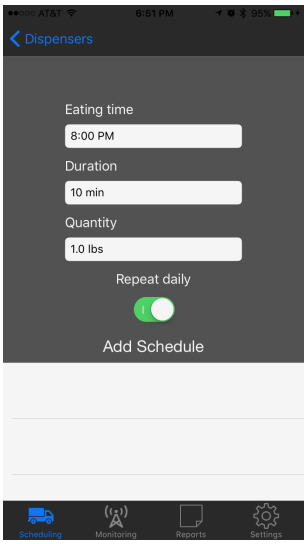

Figure 11: Mobile Application Schedule

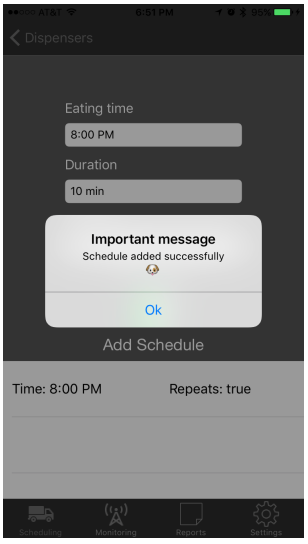

Figure 12: Mobile Application Schedule Confirmation

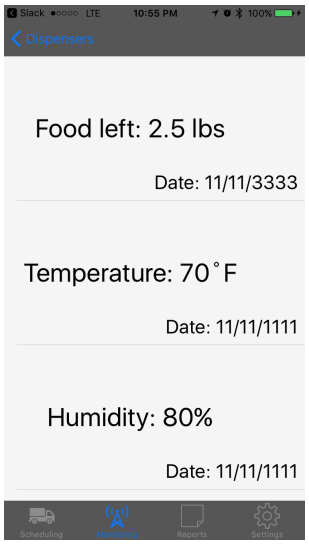

Figure 13: Mobile Application Monitoring

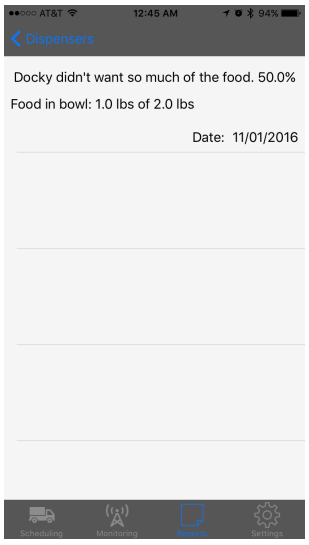

Figure 14: Mobile Application Reports