EEL 4914 - Senior Design 1
Final Document
7/28/2020

Pet Connect

A remotely operated pet door system

Department of Electrical Engineering and Computer Science
University of Central Florida
Dr. Samuel Richie

Group 10

Michael Choi	EE
Graham Goerg	CpE
Joy Weaver	CpE
Ryan Flynn	CpE

Table of Contents

Figures Index

Tables Index

1.0 Executive Summary

According to the American Pet Products Associations (APPA), there are about 8.5 million families that own a pet. Many American pet family owners had to leave the home for work and were forced to decide either to find someone to look after their beloved pet or send them to pet care. Hiring someone to look after one's pet tends to be expensive and undesired as it requires to either invite another person in your household or a spot to and from work. Especially during the COVID-19 pandemic, families will be leaning towards a user-friendly, self-sufficient, and affordable option to take care of the pet's needs.

The Pet Connect system was designed with convenience and affordability in mind. Auto Pet Lock consists of three parts, a collapsible rod-like lock, multiple sensors unit, and mobile application. The camera and speaker unit will be placed at a high vantage point typically located near the top corners of the sliding door to maximize the area of surveillance. The camera and speaker unit will provide the user the capability to call out their pets. The collapsible rod-like lock unit installed on the railing of a sliding door will be able to open and close the sliding door through a motor. The collapsible rod-like lock will have a display to indicate the different configurations and modes that Pet Connect provides. It is critical to ensure the reliability of the collapsible rod-lock unit to keep the safety of the household.

This report documents the Pet Connect design process. It will first outline the motivation and goals of the product. The report will dive into details about specifications and requirements for the project. Specifications and requirements like having our system opening and closing a sliding glass door and having an application that allows remote use from a mobile phone will lay the foundation for the team to strive and work towards. The research section will include the ranges of choices for each component of the project and justify the reasoning for why the selected technology and part was chosen. The paper will outline the constraints and standards that influenced the team's design decisions and directions of the project. This following report will include block diagrams, flowcharts, and schematics to illustrate the hardware and software design details. Within the hardware design detail section, one will explain the PCB design and power management scheme. The software design detail section will dive in specific on the different modes the user or owner may utilize through the mobile application. It will explain the logic flowchart and state-machine chart to outline the various decisions the product will be making with the given data. Moving on, one will be presented a Bill of Materials to list out the necessary components and the manufacturing cost of one Pet Connect system. The paper will explain how the product will be put through a range of tests in order to validate all requirements and specifications declared have been met. The administrative section will present how the responsibility and task of this product was distributed, how the budget was divided, and a tentative schedule and milestone that the team will use to ensure the development of the project will be met.

2.0 Project Description

Pet Connect gives pet owners a convenient way to let their pets out when they are not at home. It allows people to experience the joy of having a pet even if they are working all day without worrying about the pet "going" in the house and being restless.

This project is a remotely operated pet door that will allow the owner to open their standard sliding glass door to let their pet out even when they are not at home. When the pet gets close to the door an alert will be sent to the owner's phone which will then give them the option to unlock and open the door. The pet will be wearing a small device that can be attached to a collar, which allows the sensors on the doorway to recognize the pet is close. For security reasons, a camera will show the user their doorway to make sure that their pet is trying to get out and an intruder is not trying to get in. Once the pet has come back inside the door will be closed and locked by the owner. Our design also includes a speaker and microphone that can be used by the owner to either call the pet inside or call the pet to the door if the owner desires to let the pet out without the pets initiation of the system. With an easy to use mobile app, the user would also be able to set an at home mode that enables the door to open automatically without the approval from the owner. This project description lays out our motivations, goals, objectives, intended functionality, and project operation manual for Pet Connect.

2.1 Motivation

Many people face the dilemma of having to be away from home for extended periods of time while leaving their pets at home alone. A prime example is going to work each day. With the usual work day starting at nine in the morning and ending at five at night, this leaves the pet stuck in the house for eight hours. Getting exercise and fresh air are an important part of keeping a pet healthy. This can be challenging for owners who come home late, tired from a long day of work. Pet Connect gives the pets a chance to run and enjoy the outdoors in their backyard whenever they feel like it. Not to mention the bigger factor of the pet having to relieve themselves. This would prevent the owner from coming home to a ruined carpet or couch simply because their pet could not help themselves while possibly saving them thousands at the same time.

The inspiration for this project also comes from us personally having pets while no one is home for long periods of time. We also wanted to take a different approach to solving this problem as there are products that are automatic pet door flaps. We did not want the user to have to make a hole in their wall to use our product. Opening the entire door instead of a pet door flap gives the user the possibility of getting any size pet they want in the future without worrying about the size of the pet door flap. Thinking about these challenges gives the motivation of our project. Additionally, the motivation behind this project comes from wanting to demonstrate the skills and experience obtained along the way through our Electrical and Computer Engineering programs at the University of Central Florida.

2.2 Goals and Objectives

As with a lot of home technology, our overall goal is to make the user's life easier while still keeping their house safe. Pet Connect intends to take some of the stress away from pet owners who are away from their house for long periods of time for any reason. We hope to also provide a more convenient way of letting a pet out back when the owner is home, with the automatic opening feature. At night or for any other reason, the security of the door will also increase with Pet Connect as it will stay locked and prevent the door from even opening in the case of an intruder.

Going along with making the user's life easier, is the goal of providing an easy set up and installation. The user will be able to install this on any standard sliding glass door and will not need to cut any holes in the wall like they would with a classic pet flap. They will then place the required sensors by the door, and the camera in a place of their choosing and the hardware setup is complete. Finally, with the easy to use mobile app they can simply register and connect their specific pet to their app and they will be ready to start using the Pet Connect.

While our product aims to give the user more convenience, it also seeks to keep the pets healthier. Dogs specifically can develop health complications after years of waiting to relieve themselves for hours. This can include urinary tract infections, possible urinary cancer, and incontinence. Another thing that is important for a dog's health is exercise. Running around and exerting some energy in the fresh air everyday can improve the dog's hip joints, digestive system, and keeps them at a healthy weight. Our project tries to help solve these problems by simply giving the pet some more freedom to go in the backyard as they please.

2.2.1 General Project Goals

These are the general goals for the overall physical product and its functionality.

- Open and close a standard sliding glass door.
- Lock the door when not operating.
- Have an application that allows remote use from a mobile phone.
- Have a security component that includes audio and visual.
- Can be installed in 2 hours or less.
- Universal. Can be moved from one door to another without customization.
- No user maintenance after installation.
- Product lifetime of 5 or more years.
- Internet access for remote operation.

2.2.2 Hardware Goals

In order to achieve all goals and objectives for our product, hardware will play an important role. We have envisioned a main processing unit working with many peripherals, mostly housed within a singular, small footprint or design and tactfully connected externally. The hardware goals desired are as follows:

- All internal operations of the system will be done with a microcontroller.
- All power for the system and its peripherals shall be drawn from a single AC electrical outlet.
- Each power requirement will be split internally from the single input.
- All power design shall be consolidated to a single printed circuit board.
- The system needs to be able to send and receive signals from a mobile application in order for remote operations.
- The system also needs to be able to send and receive signals from all its physical peripherals.
- The system should detect any entity within one meter of the door via a sensor.
- Entities within one meter of the door should be identified as an owned pet or not with greater than 90% precision .
- One way video coverage of the door and entity when in operation is needed for visual security measures.
- Two way audio coverage of the door and entity when in operation is needed for communication purposes.
- When in operation the system shall unlock and open the door to a maximum clearance of two feet.
- When in operation the system shall close the door completely and lock when closed.
- The system shall have a screen to display operations, modes, and other information available.
- The system shall have an external visual representation of each operation and mode.

2.2.3 Software Goals

With a lengthy list of hardware goals and objectives, there must also be software components allowing for communication and compatibility between them. The following list includes all software goals and objectives our project needs to make every piece work together as an autonomous system as well as for a mobile application that integrates remote operation.

- Entity detection and verification logic based on data received from sensors.
- Peripheral communication and control logic.
- Android/iOS application for product owners to interact with the system remotely.
- Application shall receive notifications from the system when initiated by the pet sensors.
- Application shall be able to initialize the system remotely by the user.
- Application UI shall show system status.
- Application UI shall show system mode.
- Application UI shall display video streamed from the system camera.
- Application shall output audio received from the system microphone.
- Application shall have the ability to transmit audio to the system speaker.
- Application shall allow the user to open and close the door.
- Application shall allow the user to change the systems mode.

2.3 Function

The main function of the project is to allow pets to restlessly outdoors while a pet owner is away from home. Like people, pets get bored if there is not enough variety in their lives. In the majority of U.S homes today, pets spend the better part of everyday at home while the pet's owner at work. Often when pet parents come from their busy workday, they are often too exhausted to provide the kind of outdoor stimulation and exercise they need. In order to provide an outdoor environment that pets need and to accommodate a busy lifestyle for pet parents, therefore the important functionality of the system is mobility. The best approach for a pet's owner to be able to let their pet out is to use a wireless communication system that allows the pet owner to open and close doors for their pet. Mobile applications provide different modes to set up how the pet owner wants the door to be open/locked when the pet approaches the door. This function creates an easy and convenient way for pets to rest outside.

Another function of the system is to protect the door from scratching or biting from pets. Another behavior of most pets is scratching objects. For example dogs often bite or scratch the door if they need attention or want to be outside. As a result, it can cause significant damage to the door panel. Prices for repair or replacing new doors can cause homeowners a hundred of dollars. Our system includes sensors detected which alert the pet owners when the pet is closed to the doorway, the system will notify the pet owner to open the door before the pet attacks and does any damage to the door.

Moreover, Security plays a big part in this project. Adding the camera in the system allows the pet owner to see pet movement at the doorway, or if the pet is already back inside the home as well as observe when and how the door is operated. If the system is not armed, and the door has been opened. The owner will be noticed through the camera so they can close the door to protect any intruders from entering the home.

2.3.1 Related Work

There is a similar product launch in early 2018 made by Company name Wayzn. This product is currently active in the market. The product has similar functionalities which allow dogs owners to convert existing slide glass doors into app-controlled pet doors. However, our goal is to have the security component included in the system instead of just compatible with existing one. Moreover, we tend to use different sensors and implement the design that can lower the cost to provide affordability. Figure 1 shows a similar product from Wayzn.

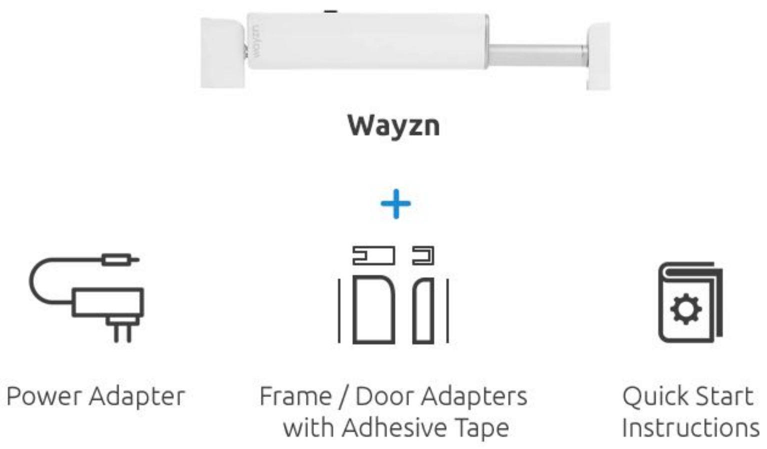


Figure 1: Similar product Wayzn (Pending Permission)

Another similar product called AutoSlide ® - Sliding Glass Door Opener. This product ranges from $450 to $750 due to the various selection of sensors. This product has very functionalities in which our product will strive towards but this product also will respond to people in addition to pets. The interesting function of this product is its capability to open the door at various distances. Figure 2 below shows a product from AutoSlide.

Figure 2: Similar product Autoslide (Pending Permission)

2.4 Project Operation Manual

Pet automatic door opener is the system that provides the ability to operate the door through mobile application. The device allows dog owners to open and close the door anywhere so considering portable products that are easy for pet owners in case they are not home. No need to hire a dog sitter to let pets outside or worry about pets does any damage to the door panel when they need to restless. This system is very easy to install and low cost of installation. Mechanical bars can easily attach to existing door panels. Motion sensors shall detect the presence of objects. The sensor shall measure objects no farther than 4 feet. Targets can be monitored over the full sensor range or restricted within user defined distance range. The object also must be large enough for the sensor to detect. RFID tag assists for security to provide accurate data and real-time tracking objects. Sensors are placed into two locations. First inside the mechanical door bar that attaches to the bottom of the door panel and second sensor is attached to target objects. In order to meet requirements of compatibility, components selection and common elements are vital.

Signaling between RFID tags and readers is done depending on the frequency band used by the tag. Often more than one tag will respond to the tag reader. Collision detection is important to allow reading data, Study shows that "A group of objects, all of them RFID tagged, are read completely from one single reader position at one time." Therefore, bulk reading can be possible, yet intuitive, method for sensor identification. However, under operational conditions, bulk reading is not reliable and may lack sufficient precision. This issue is related to the lock system, if the user has multiple pets and plans to use an RFID key to help secure the lock system. One of our requirements is to reduce the cost. It is not guaranteed that using RFID tags can properly read and accommodate multiple keys. Without proper understanding of the use of sensor limitation, functionality and security will be inhibited.

The unit can be mounted to the upper top or lower of the existing door with a motor to pull and push the door to closed or open. Door opening force is the measurement of how many pounds of force are required to open the door. This information can be found in the Americans with Disabilities Act Accessibility Guidelines (ADAAG), ICC/ANSI A117.1. The research stated that "The maximum forces allowed by the IBC (2003) for an egress door are 15 pounds to release the latch,". Most of the sliding doors in the residence building have about 12 pounds of weight. Which we are considering using the motor and power that can swing the door to close and open. If there is a custom glass door that has a weight greater than 15 lbs. The system will not operate to open and close the door properly. Our design will comply with section 1010.1.4.2 Power-operated doors, meaning that the door will be operated by the power. The system shall be designed in the event of power failure. This is for users to be aware that in the case of power cut off the lock will become deactivated and the door can be operated in a manual mode.

Mobile application is the center control for the users to operate the device. The pet owners shall mainly have capabilities to use mobile user interface to control activities, view, and manage the device. Research showed that "The significance of the Internet of Things application development is the real-time transmission of data" Therefore connectivity is challenging when it comes to monitor, process data, and supply information. Push notification and real-time video camera are essential for users. Poor connectivity will be a major issue to connect mobile devices. Therefore, users required strong internet connectivity to proper operate the door and received efficient data transmission.

2.4.1 Steps to operate the system:

1) Install the unit on the top or lower on existing sliding doors. The unit should be covered and lined up along with the non-leading edge of the door then secure it with the door jam.

The frame should be tall enough to have room to mount long gear to drive the door back and forward. Position mounted on the unit based on type of the doors.

2) Installed the gear rack extending from the unit. Measure and cut the rack to match the size of single door panel

3) Attach the rack with the unit, align the rack to the cogwheel and make sure the rack is engaged with the cogwheel. Sliding door back and forward for testing. Connect the power supply to the nearest power outlet.

4) Installed the lock system at the side bar to control lock and unlock of the door. This is the wire split from the main unit.

5) Place an RFID tag to the pet collar and push the bottom to activate the device.

6) Installed the application in the mobile phone by first creating an account for login and password. Second, create pet data that connects to the RFID tag to link the device. Third, connect wifi to active device connectivity.

3.0 Requirements and Specifications

Throughout the brainstorming and research process of our design, we have uncovered numerous requirements that our system should satisfy and certain specifications consumers would be interested in for such a product. Table 1 is a working list of hardware requirements and specifications we have put together for the system's core functionality, including hardware and software.

Requirement	Description	Value
Cost	The total cost we want to spend on the project.	≤ $500
Weight	The total weight of the door opening mechanism. (Not including peripherals)	≤ 10 lbs

Requirement	Description	Value
Dimensions	The total weight of the door opening mechanism. (Not including peripherals, arm retracted)	10” x 10” x 18”
Sensor Range	Desired distance from door for detecting and verifying a pet.	24” for detection 18” for verification
Power Usage	The power needed to operate the system when active.	5v

Table 1: Hardware Requirements and Specifications

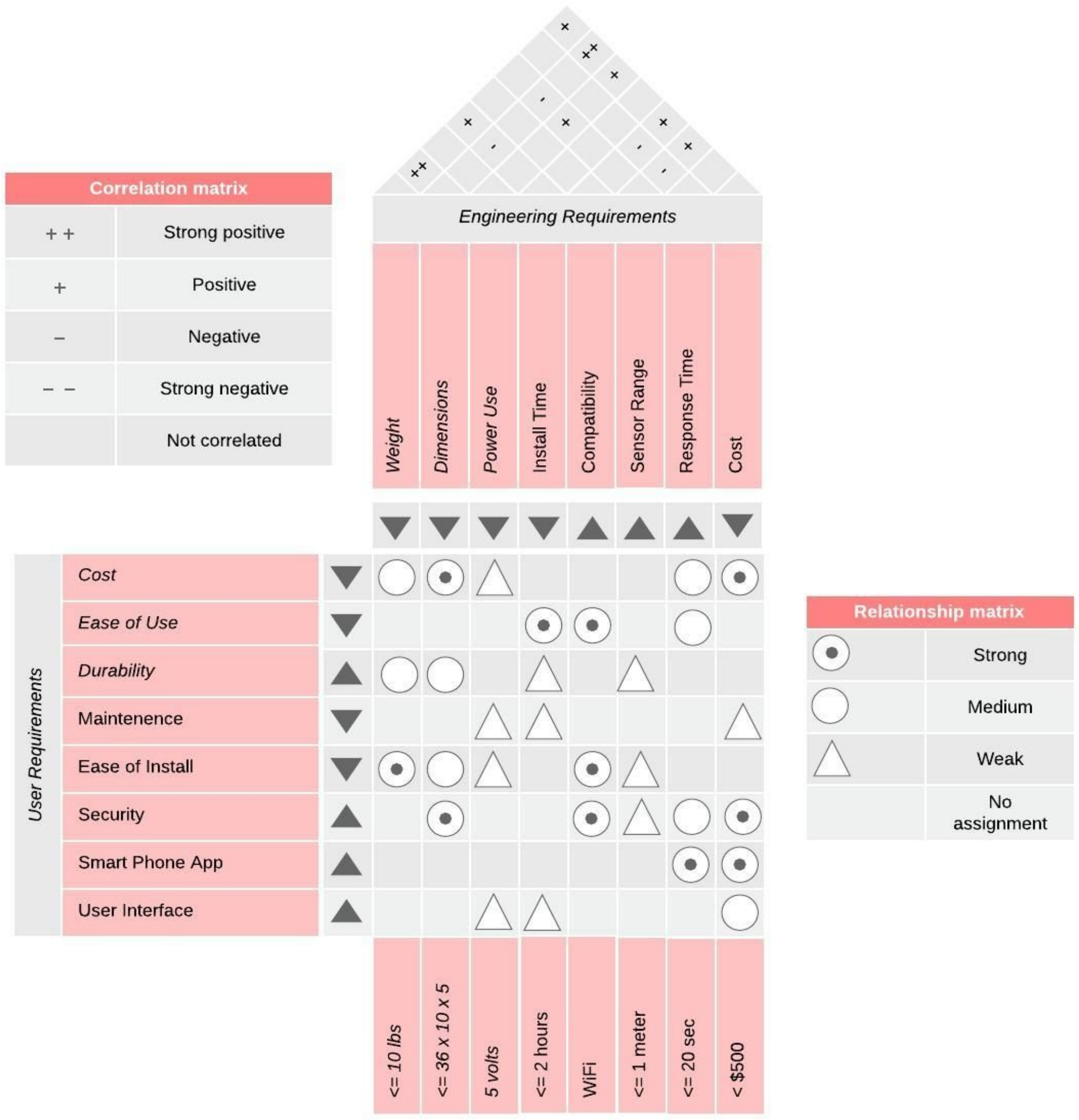
In the same way table 2 shows the software specific requirements for our project design. These are separated out as we look at different aspects and constraints when creating hardware and software requirements.

Requirement	Description	Value
Data Upload	The time it takes for the database to reflect the data that was just written to it.	≤ 3 s
Data Download	The time it takes when the software is reading data from the database	≤ 2 s
Mode Options	The system will give the user three different modes to operate on	3 Modes of Operation
Video Display	The time it takes for the user to see the live video display through the mobile application.	≤ 3 s
Android API Level	The minimum API level that the mobile application will run on	Level 16 (Jelly Bean) or later
Controllability	The time it takes for the home system to correctly reflect new commands from the mobile app.	≤ 5 s

Table 2 : Software Requirements and Specifications

3.1 House of Quality

With the use of focus groups and considering based on the user's need, we design target values for each design to meet necessary requirements. The matrix below establishes concrete goals for defining customers/users requirements and for the design engineer. Shown below in figure 3 the House of Quality for our project categorizes the user requirements financially, by usability, and by security. Cost, durability, and maintenance are considered up front financially. Where as ease of use, install, smartphone app, and user interface all fall under the usability of the product. Lastly, security, smartphone app, and user interface all pertain to the security requirement.



Correlation matrix	
+ +	Strong positive
+	Positive
–	Negative
– –	Strong negative
	Not correlated

User Requirements		Weight	Dimensions	Power Use	Install Time	Compatibility	Sensor Range	Response Time	Cost
		▼	▼	▼	▼	▲	▲	▲	▼
Cost	▼	○	◉	△				○	◉
Ease of Use	▼				◉	◉		○	
Durability	▲	○	○		△		△		
Maintenence	▼			△	△				△
Ease of Install	▼	◉	○	△		◉	△		
Security	▲		◉			◉	△	○	◉
Smart Phone App	▲							◉	◉
User Interface	▲			△	△				○
		<= 10 lbs	<= 36 x 10 x 5	5 volts	<= 2 hours	WiFi	<= 1 meter	<= 20 sec	< $500

Relationship matrix	
◉	Strong
○	Medium
△	Weak
	No assignment

Fig 3: House of Quality

The engineering requirement involves work with an engineering team who were tasked with developing the product and ensuring it matches the customer's needs. Based on the relation matrix, our design requirement should have lightweight material which can ease the installation, and smaller dimensions that can help save cost of the project. The same logic can also apply to installation time. The power use should also be minimized. The most important features that take as prioritized are compatibility and cost. We want our product to have high compatibility to support security, ease of use and installation, at the same time support affordability. Moreover, compatibility will greatly affect sensor range and response time since three features have up arrows and positive correlation.

4.0 Research

Following our brainstorming sessions we had a general overview of the tasks we wanted our product to accomplish. We also had multiple ideas on how to achieve many of these tasks. It was agreed upon that since there are more than one way to accomplish many parts of our product, we should research each part individually, in order to better meet the requirements and specifications laid out in section 3. The following research is presented to you categorized by hardware and software. We then list the possible solutions for each task, stating their pros and cons. Lastly we will compare them, in order to choose which will best fulfill our requirements and/or specifications.

4.1 Similar Project

The idea of our project is to enhance the technology for pet owners to support a busy lifestyle. Our goal for our product is to build a system that provides support for usability, security, adaptability, and affordability. We begin our research by reviewing previous Senior Design Project which can help with our creativity, address any issue, or concern, and enhance our design and implementation. We found two similar projects with Door Mechanism related products, both projects were conducted to target consumers which are Homeowners and their visitors or pets. Those two projects have the same functionality that can improve our implementation of the projects and its features.

4.1.1 Home Observable Monitoring Entry System (HOMES)

The HOMES project was built in Spring 2015. The senior group members are Colleen Caffey, Bruno Calabria, and Ricardo Georges. This project was sponsored by Boeing which budget for the system can be supportive for this product. The main purpose of HOMES is for the Homeowner to have the ability to see who comes to the door before users continue device operation. Facial recognition and fingerprint will grant Homes' access. This project also includes a pet door in the door panel which is the focus for our team in this research. Our product will include a system that can attach to the existing door to allow pets to enter. The difference is our system from HOMES is live video for users to view in real-time instead of camera captures. In addition, our lock system can be operated using Wi-Fi if there is internet connection.

4.1.2 A1 Security System

Another previous project that has similar functionalities and features is A1 Security System. The project was created in Spring 2017; Team members are Timothy Henry, Computer Engineering, Brandon James, Computer Engineering, Jonathan Chew, Electrical Engineering. Objective of this project to improve security on the door. The system included sensors for motion detection and Mobile application to operate lock/unlock the device. This project also includes facial recognition, video camera, lock system that can operate the device remotely. Our system will be different in terms of two steps verification from Ultrasonic Sensor and RFID tag before push notification to the users. Moreover, our Mobile application also will make it available on both android and iOS users.

4.1.3 AA_Good 1

A previous senior design project with the title and group members' name are unknown was used as a reference for the printed circuit board research section. The following document provided valuable insight on the design constraints and printed circuit board guidelines to adhere in order to ensure the best quality. The following project is a portable and affordable blind spot detector for large vehicles like a truck. Although the functionality of this project differs from our project, this documentation is a good reference on research material and reference document.

4.2 Hardware Research

Hardware is considered to be all the physical components involved in building our project design. This section will be filled out with the individual research for all hardware used in our design. Some of the main hardware components include single board computers, power input, printed circuit boards, the motor and door locking mechanism. Some other hardware components are the peripherals of the system. These peripherals are motion sensors, identification sensors, LED's, LCD's, and audio / visual input and output.

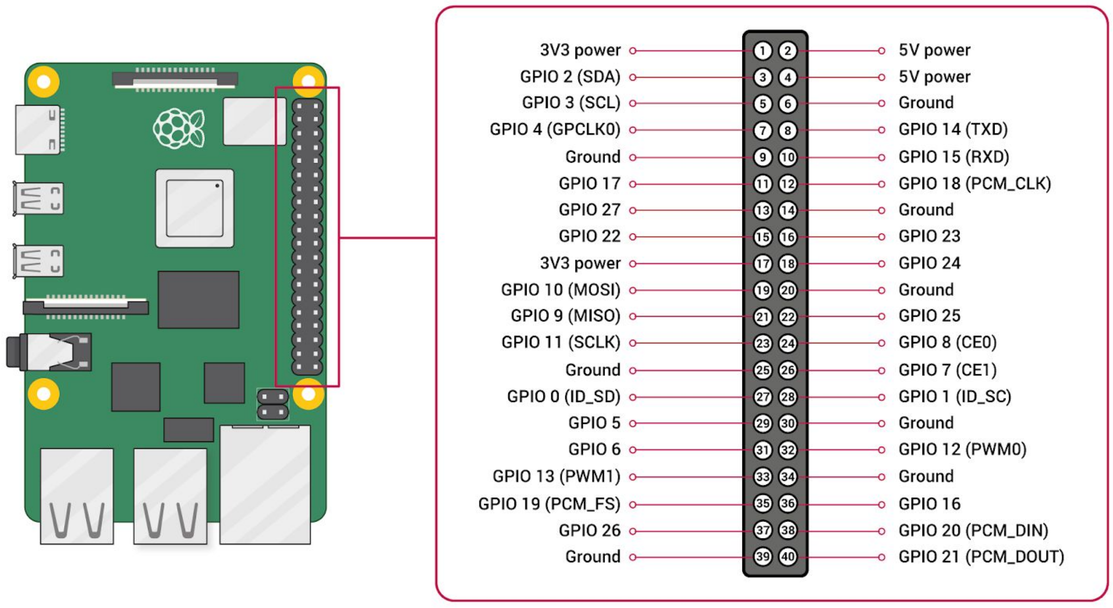
4.2.1 Single Board Computers

During our brainstorming process we originally thought that our design would only require a microcontroller to handle all system processing. With the need to include a security element that includes video streaming, it was quickly discovered that in order to handle the bandwidth and data needs for image processing on top of other functionality, microcontrollers were not sufficient. Our research helped us discover single board computing and the vast "do it yourself" community it built.

Single board computers are much more than just a microprocessor, housing a complete computer on a small board, including memory, input/output, power, and more. The single board computer can be seen as "the brain" of our design and will be used to control all operations of our system and communicate to its peripherals. The following is a list of single board computers considered for our design.

4.2.1.1 Raspberry Pi 4B

The Raspberry Pi 4B is the newest model in the series, replacing the Raspberry Pi 3 in June 2019. It offers better processor speed, multimedia performance, more memory, and more connectivity compared to the Raspberry Pi 3 boards. It also retains backwards compatibility with only slightly increasing power consumption. The Raspberry Pi 4B offers Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz and 2, 4, or 8GB LPDDR4-3200 SDRAM. For our project the 4GB RAM would be able to operate with ease. The Pi 4 also comes with built in 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE or wireless communication. The Raspberry Pi 4 would offer us multiple ways of peripheral communication via 2 USB 3.0 ports; 2 USB 2.0 ports a 40 pin GPIO header, 2-lane MIPI DSI display port, 2-lane MIPI CSI camera port, 4-pole stereo audio and composite video port and more. On top of general input and output devices, the GPIO pins can be used with PWM, SPI, I2C, and Serial functions. The Pi 4 power acceptance is 5V DC via USB-C connector (minimum 3A*) or 5V DC via GPIO header (minimum 3A*). The Raspberry Pi 4 starts at a price of $35. Figure 4 depicts the GPIO layout for the Raspberry Pi 4B.



3V3 power 1 2 5V power
GPIO 2 (SDA) 3 4 5V power
GPIO 3 (SCL) 5 6 Ground
GPIO 4 (GPCLK0) 7 8 GPIO 14 (TXD)
Ground 9 10 GPIO 15 (RXD)
GPIO 17 11 12 GPIO 18 (PCM_CLK)
GPIO 27 13 14 Ground
GPIO 22 15 16 GPIO 23
3V3 power 17 18 GPIO 24
GPIO 10 (MOSI) 19 20 Ground
GPIO 9 (MISO) 21 22 GPIO 25
GPIO 11 (SCLK) 23 24 GPIO 8 (CE0)
Ground 25 26 GPIO 7 (CE1)
GPIO 0 (ID_SD) 27 28 GPIO 1 (ID_SC)
GPIO 5 29 30 Ground
GPIO 6 31 32 GPIO 12 (PWM0)
GPIO 13 (PWM1) 33 34 Ground
GPIO 19 (PCM_FS) 35 36 GPIO 16
GPIO 26 37 38 GPIO 20 (PCM_DIN)
Ground 39 40 GPIO 21 (PCM_DOUT)

Figure 4: Raspberry Pi 4B GPIO Pinout with permission from RapsberryPi.org

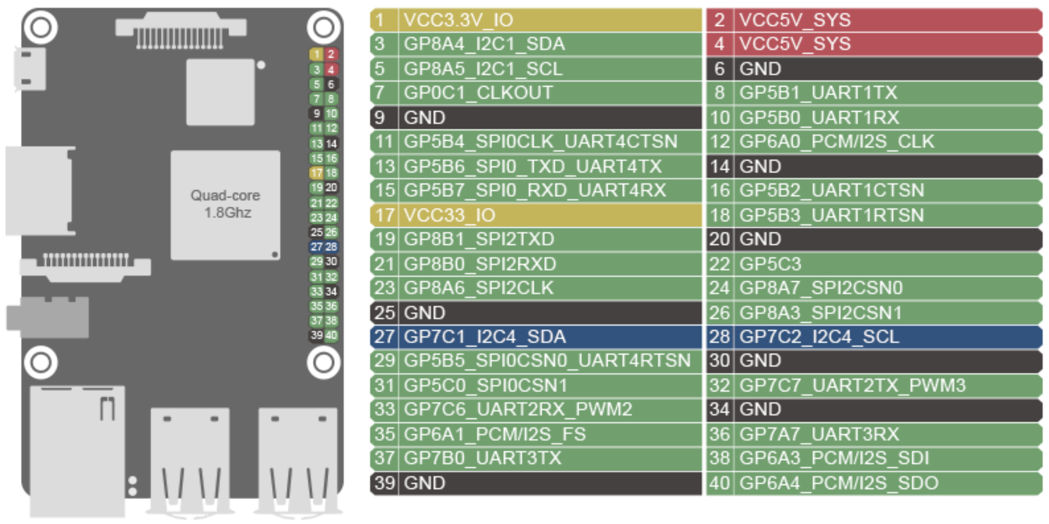
The Raspberry Pi 4 would be a great choice for our design because it has more than enough memory, processing speed, and peripheral support for our proposed system. The Raspberry Pi also has a huge 3rd party and community support. It is the top choice for DIY projects and is advertised as “Your tiny, dual-display, desktop computer…and robot brains, smart home hub, media centre, networked AI core, factory controller, and much more.” Although the Raspberry Pi

4B has everything we need for our design, it also has a lot of capabilities we don't need or will be using. This makes the purchase price more expensive than what is needed, but also allows room for more implementation and design expansion.

4.2.1.2 ASUS Tinker Board

The ASUS Tinker Board is another single board computer that boasts an ultra small form with class leading performance. The Tinker Board offers a quad-core ARM-based processor Rockchip RK3288 @ 1.8GHz and 2GB of LPDDR3 dual-channel memory. Tinker Board features numerous connectivity options, including a 40-pin GPIO, a DSI MIPI and CSI MIPI connection, GBit ethernet, Wi-Fi and Bluetooth controller, HDMI output, and four USB 2.0 ports. The Tinker Board comes with a heatsink included and pricing starts at around $70.

The Tinker Board offers a small design and fast CPU which can be useful since our design requirements include a physical size maximum. It also has enough peripheral support to cover all design aspects of our project. The initial price point for the Tinker Board is high compared to the others and our project does not require high processing speeds. The Tinker Board is also limited to 2GB RAM, and USB 2.0 which might be a problem with the security peripherals in our design. Figure 5 depicts the GPIO layout for the ASUS Tinker Board.



1	VCC3.3V_IO	2	VCC5V_SYS
3	GP8A4_I2C1_SDA	4	VCC5V_SYS
5	GP8A5_I2C1_SCL	6	GND
7	GP0C1_CLKOUT	8	GP5B1_UART1TX
9	GND	10	GP5B0_UART1RX
11	GP5B4_SPI0CLK_UART4CTSN	12	GP6A0_PCM/I2S_CLK
13	GP5B6_SPI0_TXD_UART4TX	14	GND
15	GP5B7_SPI0_RXD_UART4RX	16	GP5B2_UART1CTSN
17	VCC33_IO	18	GP5B3_UART1RTSN
19	GP8B1_SPI2TXD	20	GND
21	GP8B0_SPI2RXD	22	GP5C3
23	GP8A6_SPI2CLK	24	GP8A7_SPI2CSN0
25	GND	26	GP8A3_SPI2CSN1
27	GP7C1_I2C4_SDA	28	GP7C2_I2C4_SCL
29	GP5B5_SPI0CSN0_UART4RTSN	30	GND
31	GP5C0_SPI0CSN1	32	GP7C7_UART2TX_PWM3
33	GP7C6_UART2RX_PWM2	34	GND
35	GP6A1_PCM/I2S_FS	36	GP7A7_UART3RX
37	GP7B0_UART3TX	38	GP6A3_PCM/I2S_SDI
39	GND	40	GP6A4_PCM/I2S_SDO

Figure 5: ASUS Tinker Board Pinout (Pending permission from tinkerboarding.co.uk)

4.2.1.3 ODROID XU4

The ODROID XU4 was built on open source software and designed specifically to support android, but also supports Linux. It offers a Samsung Exynos5422 Cortex-A15 2Ghz and Cortex-A7 Octa core CPUs, Mali-T628 MP6(OpenGL ES 3.1/2.0/1.1 and OpenCL 1.2 Full profile), 2GB LPDDR3 RAM PoP stacked. The XU4 has connectivity through 2 x USB 3.0 Host, 1 x USB 2.0 Host, Gigabit Ethernet port. The ODROID XU4 also comes with a mounted cooling fan and has a 30 pin GPIO and starts at around $60.

The ODROID XU4 has the fastest CPU speed so far and is smaller than the Raspberry Pi 4B. The XU4 also has less connectivity than the other single board computers and has what seems to be smaller community support for DIY and IoT projects. The higher price also accounts for the included cooling fan, a feature we do not envision needed in our design. The board layout for the ODROID XU4 is shown below in figure 6.

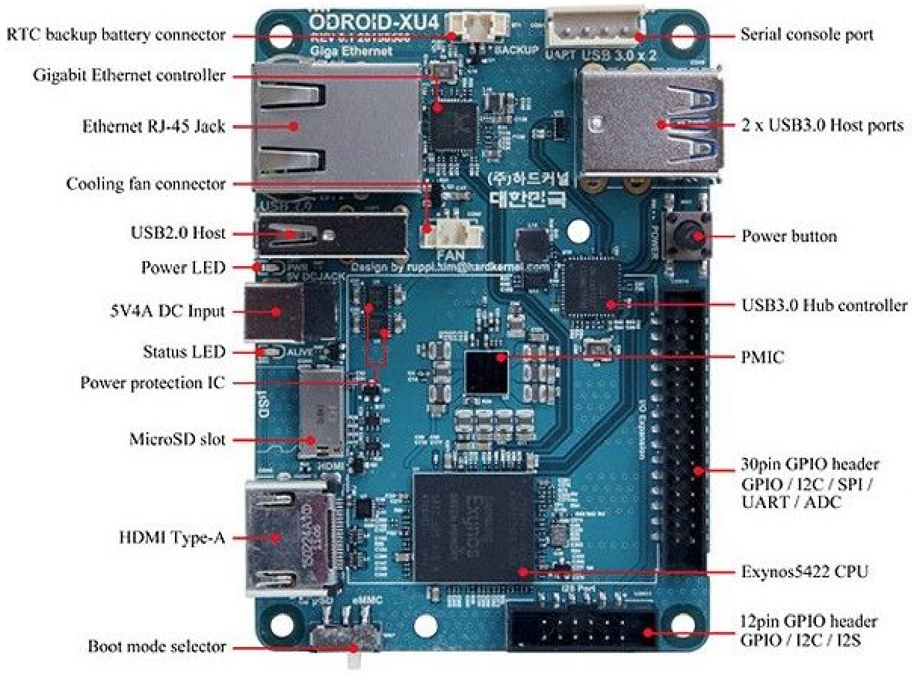


Figure 6: ODROID XU4 Board Layout with permission from hardkernal.com

4.2.1.4 NanoPi NEO4

The NanoPi NEO4 is a RK3399 SoC based ARM board with Dual-Core Cortex-A72(up to 2.0GHz) + Quad-Core Cortex-A53(up to 1.5GHz) and 1GB DDR3-1866. For connectivity the NEO4 has an onboard 2.4G wireless module, a single USB3.0, 2 x USB2.0, a 40 pin GPIO and more. The NanoPi NEO 4 starts at around $50.

Again this single board computer was considered due to its physical size and comparable computing and connectivity. Although it is faster and almost half the size of a Raspberry Pi 4B it does lack in other ways. It is limited to only 1 GB RAM and has fewer USB ports. It is also limited to 2.4G WiFi. While the connectivity and slower WiFi are worse than the others, the NEO4 still has enough to implement our design. The real drawback is the 1GB RAM, which we do not see being able to handle our security peripherals. The board layout for the NanoPi NEO 4 is shown below in figure 7.

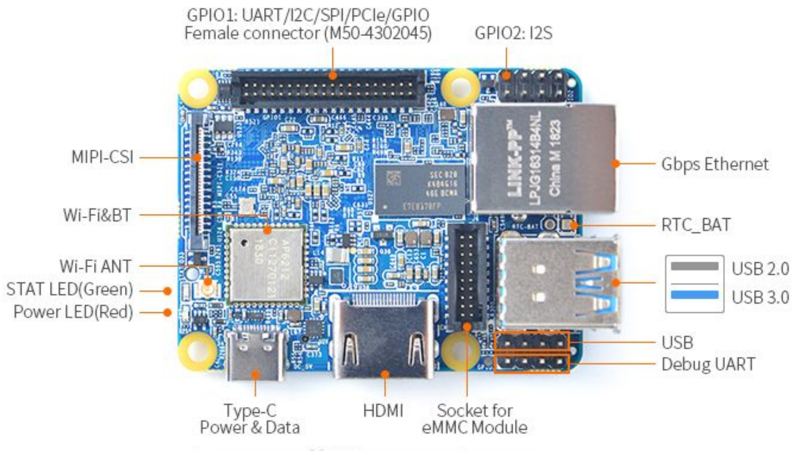


Figure 7: NanoPi NEO4 Board Layout (Permission pending from friendlyarm.com)

The major factors in choosing our single board computer come from our requirements and specifications. The cost of a single board computer is a huge factor, due to the direct connection to the overall product cost. We are also looking to design a compact product, so the board size is important. Our design includes many peripherals that will require USB connections and possibly a sizable GPIO structure. From our requirements we must have wireless communication to the internet of things, so WiFi connectivity and speed are important to us. Lastly processing speed and memory are going to be important to the system design, which will be directly affected by all our connected peripherals, such as audio and video. Table 3, shown below, is a comparison of the researched single board computers and their specifications related to our requirements.

	Raspberry Pi 4B	ASUS Tinker Bd	ODROID XU4	NanoPi NEO4
Cost	$35	$70	$60	$50
Size	88 x 56 x 20 mm	86 x 54 x 18 mm	83 x 59 x 18 mm	60 x 45 x 18 mm
IO Components	2 USB 3.0 2 USB 2.0 40 pin GPIO 2.4 GHz and 5.0 GHz 802.11ac WiFi	4 USB 2.0 40 pin GPIO 2.4 GHz 802.11 b/g/n Wi-Fi	2 USB 3.0 1 USB 2.0 30 pin GPIO 2.4 GHz 802.11 b/g/n Wi-Fi	1 USB 3.0 1 USB 2.0 40 pin GPIO 2.4GHz 802.11 b/g/n Wi-Fi
CPU	4 x 1.5GHz	4 x 1.8GHz	4 x 2GHz + 4 x 1.4GHz	2 x 2Ghz + 4 x 1.5GHz
RAM	2,4,8 GB	2GB	2GB	1GB

Table 3: Single Board Computer Comparison

4.2.2 Power Supply

Determining a suitable power system for this project stems from understanding the electrical components within the system and the requirements. The microcontroller and other electrical devices in our design will require DC voltage that can be obtained either through an AC-DC wall power supply or battery. The power supply designed will meet the American standards to ensure safe power transfer and to prevent electric hazards. Table 4 presents a clear summary of the voltage and current requirement of components on the system. We should consider the manufacturer's recommended value for the components to gain a good estimate to determine an appropriate power requirement.

Components	Voltage	Max Current
Microcontroller	5 V	< 300 mA
Raspberry Pi 4	5 V	300 mA
Motor	12 V	5 A

Table 4: Subsystem Power Estimation

4.2.2.1 Batteries

Battery technologies have been viewed as a solution for supplying power to electronic devices that requires movement and mobility. Batteries are made of a combination of earthly elements such as zinc, and potassium, and electrolyte. Due to the properties of the combination of these elements, they divided into two electrode classes: Anode and Cathode. The Anode is connected to the negative end while the Cathode is connected to the positive end of the battery. These two electrode classes and the electrolyte will chemically react to generate an electric current. Batteries can often characterize based on the metal used and packaging. There are four basic batteries technologies that will be explored: Alkaline, Nickel-Cadmium, Nickel-Metal Hydride, and Lithium-ion.

Alkaline batteries are the most popular and well-known type of batteries. The general usage of Alkaline batteries lies within low-drain applications such as flashlight and alarm clocks. Alkaline batteries are desired due to its high energy densities. Although alkaline batteries generally are disposed of after a single use, alkaline batteries have longer lifespans compared to all other disposable batteries. Another fact is that Alkaline batteries do not contain mercury and any heavy metals that would harm the environment. The drawback of alkaline batteries is its high internal resistances which restricts the current flow, drops voltage and creates a large heat dissipation.

Nickel-Cadmium batteries are rechargeable batteries generally used in laptops, and power drills. These batteries are best known for its rechargeability and its ability to maintain voltage very well and hold its charge while not in use. A major disadvantage of NiCd batteries are its relative low energy density meaning its short lifespan for a single charge. NiCd batteries are considered environmentally unfriendly and require a special location to be properly disposed of.

Nickel-Metal Hydride batteries are more expensive rechargeable batteries than its Nickel-Cadmium counterparts. Although Nickel Metal Hydride are cheaper than lithium ion batteries, they have a lower energy density resulting in newer technologies improving efficiently and a great understanding of the lithium ion. Another disadvantage of NiMH is its high self-discharge (roughly 50% greater than NiCd) and its degraded performance if stored at elevated temperature.

Lithium ion batteries technologies is considered one of the most promising battery technologies due to its rechargeability and Lithium-ion's unique properties. The reason why Lithium-ion batteries are gaining a lot popularity within the battery technology is due to its lightest in comparison to all other metals, and It has the greatest electrochemical potential and provides the largest energy density for its weight. Lithium ion batteries have almost twice the amount of energy density compared to standard nickel-cadmium. With no memory issue seen in many NiCd batteries, lithium-ion batteries are considered low-maintenance. A major drawback of lithium-ion batteries is the requirement to have a protection circuit to ensure the battery functions safely. Lithium-ion batteries require to be stored in cool places to prevent aging and are very expensive (roughly 40% more in cost in relation to NiCd batteries).

4.2.2.2 AC Power

In North America, AC power from wall outlets is usually 120V at 60 Hz which defines the alternating current (AC) will oscillate at 60 times per second. We will have to make the necessary calculations to determine the power requirements for this project in order to choose power supply cords and wall adapter systems. Given that the microprocessors presented in the previous section require around 5 V DC to operate, we would need to step down the voltage from wall power of 120 V to 5V and use a rectifier and voltage regulator circuit to create a constant 5 DC voltage supplying into the microcontroller.

It's also important to convert the AC current to DC current from the wall outlet. Based upon what we learned in electronics class, there are two types of rectifiers: half-wave and full-wave rectifier. The half wave rectifier is a type of AC to DC converter that rectifies only one-half cycle of an input AC waveform. The placement of the diode will determine either positive or negative cycle where the rectification will occur.

Half-wave rectifier is significantly less efficient compared to a full-wave rectifier because half of the input waveform is rectified. On the contrary, a full wave rectifier will convert the entire input waveform to one constant polarity. The advantages of implementing a full-wave rectifier are more efficient and maintain a continuous stream of power to the rest of the circuit compared to its half-wave counterpart. As a result, we will be implementing a full wave rectifier to obtain DC current from the wall outlet and supply constant power to the rest of the circuit.

A full-wave rectifier consists of four diodes in a certain arrangement to obtain the DC transformation of the AC wall power. One function of the diode is to restrict the polarity of the current, thus creating direct current. In four diode configuration, a pair of diodes will flow direct current for one half of the cycle and the other pair of diodes will flow direct current on the other half to ensure a constant flow of direct current.

Upon evaluation between these two sources, ac power from a wall outlet would best suit for this application as the product is intended to be stationary on the rail of the sliding door. A major advantage of ac power is the low maintenance to the product and its ability to supply constant ac voltage and current to the product.

4.2.3 Printed Circuit Board

Engineering design and create printed circuit boards as their primary method of assembling modern electronic circuits. PCB provides two important functions. Mechanically, a PCB provides housing for all components in a consolidated area. A PCB electrically connects electronic components through the usage of conductive pads, tracks, and other conductive methods to connect them to a substrate. A PCB can be viewed as a sandwich consisting of one or more insulating layer and one or more copper layer to support the signal traces, powers, grounds, and other electrical connections. Traditionally, components were placed on the top layer in a hole that goes straight through to the back layer. These components are known as through hole parts. Nowadays, surface mount components are the preferred choice due to its smaller footprint and its versatility of mounting either on the top or bottom layer.

4.2.3.1 Composition of a PCB

PCBs have carefully thought out layers of different materials laminated together with heat and adhesive to create one unit. These layers on a PCBs are often composed of different conductive and insulative materials to provide a specific function. A general scheme in designing for a multilayer PCB is to alternate conductive and insulative layers to electrically isolate each distinct layer. The composition of a PCB can be divided into four parts: copper, soldermask, substrate, and silkscreen.

Copper layer is an essential for a PCB as without it the PCB would not be able to conduct electricity or provide electrically connection. This layer is usually a thin copper foil applied onto the board with heat and adhesive. Common PCB practices for the copper layer is to apply a copper layer on both sides of the substrate layer. When characterizing a PCB board as a 2-layer board, electrical engineers are referring to the number of copper layers on the board. The thickness of the copper layer is determined on the usage. For instance, high power traces may require 2 or 3 ounce copper layers while data lines may use 1 ounce copper layers.

Soldermask is the layer on top of the copper foil which provides the PCB its green color. This is a layer of polymer applied to the copper traces of the PCB for protection against oxidation and external environment. The solder mask layer also provides a separation for closely spaced solder pad and prevents accidentally solder bridges. A solder bridge is an unwanted electrical connection between two conductor pads or plating by a blob of solder. The solder mask layer has an additional benefit for a professional machine. This layer helps the user and professional components placer to solder components in the correct place.

Substrate layer provides the printed circuit board its thickness and rigidness. The FR-4 notation is a NEMA grade designation to rate for glass-reinforced epoxy laminate materials like printed circuit boards. This grade designation is given for a good strength to weight ratios and almost no water absorption. FR-4 materials are known for their high mechanical values and electrical protection and insulating qualities in both dry and humid conditions.

Silkscreen layer is applied onto the top of the solder mask layer. This top is normally used to identify components, test points, and other marks needed on the board. The Silkscreen layer is essential for testability and allows for engineers reliable references. Indicators can be used to label exactly where each component will be placed on which pad. Typically silkscreen layer color is white, however silkscreen can be found in other colors like black, grey, red and yellow. The silkscreen layer is designed and declared on the board layout.

4.2.3.1.1 PCB Terminology

This section describes some common PCB terminology that is used in the Hardware Design Detail section.

- **Pad** - a portion of exposed copper on the surface of a PCB board that a surface mount component is soldered to.
- **Surface mount devices (SMD)** - electronic components that are design to mounted directly on the PCB
- **Though-hole technology (THT)** - leads of components inserted into a drilled hole through the PCB and is either soldered onto pads or other methods on top and bottom side of the board.
- **Keepout area** - an area on the PCB that should be free of components and/or traces . Often use a reminder for PCB designers to isolate an area of the PCB for various reasons such as thermal, grounding, and mounting constraints.
- **Footprint** - an arrangement of pads or through holes used to connect an electrical component to a PCB. Also known as land pattern.
- **Copper traces** - a signal trace on a PCB and is equivalent to a wire for conducting signals.
- **Via(s)** - provide a conductive path or an electrical connection for traces and/or pads on different layers of a PCB.

4.2.3.2 PCB Design Recommendation and Practices for Better Reliability

It’s important to design PCB with the expectation for reliability and long lasting. As a PCB is a critical electrical housing and as an entire system, reliability is important. This section will lay out some electrical general knowledge and basic design practices to improve performance and reliability of a system. These guidelines help PCB designers and electrical engineers by reducing the dimensions of the board thus making the overall electrical design fit in a more compact board.This section will lay out a brief summary of the recommended techniques. These techniques will be used in Section 6.0.

Guildline #1: Keep traces as short and direct as possible

Longer and excess traces means an unnecessary increase in resistance and inductance on the data and power lines on the PCB. Maintaining the traces short can ensure the maximum efficiency and performance of the PCB. Long traces increases the impedance and increases EMI risks particularly affects high-speed digital circuits and analog circuits.

Guildline #2: Group related components in the same circuitry and test points together

Typically electrical designs consist of multiple smaller circuitry. Keep all related components for one circuit together since they will be most likely to be connected to one and another. Grouping them will reinforce Guideline #1 by creating short paths and traces between each component. Bypass capacitors and resistors are important to group together and place them as close to your integrated circuits (ICs) to noise cancellation and reduction properties.

Guildline #3: Utilize decoupling capacitors when needed

Decoupling capacitors play a fundamental role in shielding ICs and components on the board from high frequency noise and electromagnetic interference (EMI). Due to nowaday technology, capacitors are cheap and robust. These capacitors should be utilized when needed. These capacitors also functions to provides other components from any unexpected

Guildline #4: Pick the right component footprint

Finding the grid spacing or component footprint is important in designing an electrically sound and reliable printed circuit board.Many of these component footprints may be found through the company website. In search of looking for acceptable electrical components, one needs to keep it mind of the component footprint for the board layout phase.

Guideline #5: Manage power and ground using a designate power plane

An individual power plane is used to ensure that power will be delivered to its destination on the PCB with minimal impedance and maximum efficiency (minimal or zero drop in voltage). A recommendation is to place a power plane to reduce noises that may propagate from the power supply from one circuit to another. Power planes will improve the noise isolation between each internal circuit within the PCB, improve electromagnetic compatibility (EMC) performance due to a shorter return path, and reduce the operating temperature of the board thus improving the thermal management of the board with a larger current capacity. [ADK]

Guildline #6: Have an even number of board layers on PCB

Odd number of layer stackup does not provide any economic saving and its asymmetry may lead to wasping, twisting, and other mechanical structural defects that is detrimental to printed circuit boards. It is more beneficial to have a ground plane instead of a power plane for a 2 layer board. One will be seeing a power plane on PCB board design with 4 or more layer stackups. [CAD]

Guildline #7: Use automatic signal routing tool on CAD software with extreme caution

Although this functionality is provided on many CAD software, it is highly not recommended to use this feature due to the multiple of design constraints that need to be taken when determining the routing situation. Signal routing on the printed circuit board should be done by the electrical engineer or by hand to avoid any undesired coupling and noise interference. A properly designed multilayer PCB can eliminate the EMI concerns and be immune to radio frequency fields. A function that is recommended is the Design Rule Check function or Design validation function to ensure that all functionality and necessary traces are found in the hand placed traces.

Guildline #8: Use the Silkscreen

The Silkscreen layer on the PCB can be used to identify components using reference designators and indicators to express on oriented sensitive parts like diodes should be placed. The silkscreen is essential to give engineers and humans a frame of reference to understand the board, Testing on the PCB will be easier will a well-thought out silkscreen notation.

4.2.4 Motor

A major component to this project is the mechanical mechanism in serving the sliding door. As this project is to improve upon the un-technological or un-smart version pet door, it is imperative to select an appropriate mechanism of the locking system to ensure safety of the household and functionality of this product. Servo motor is one of the common motors that consists of a DC motor, a potentiometer and a control circuit. Due to servo's circular rotation, A typical servo motor will need to pair with some type of cabling to open and close the sliding door. Figure 8 will illustrate how the servo motor can be used to operate a sliding door. This motor would be used on the railing of the sliding door.

Figure 8: Olide Automatic Sliding Door Opener

Compared to a typical servo motor, a linear actuator or linear servo motor differs from a typical servo motor applied linearly as opposed to rotationally and has a magnetic shaft that extends, and contacts depending on the current flow. Depending on the linear servo or actuator, the mechanics of the motor can vary. Some linear servo motors have some type of potentiometer that one may connect to a microcontroller to control speed of the shaft while other linear servo motors have fixed speed and length once current is applied. For this project, the linear servo or actuator motor seems to be the best choice because opening and closing a sliding door requires a linear movement. This linear servo actuator should also have an control input from a microcontroller to provide the team more flexibility and control on the motor's movement.

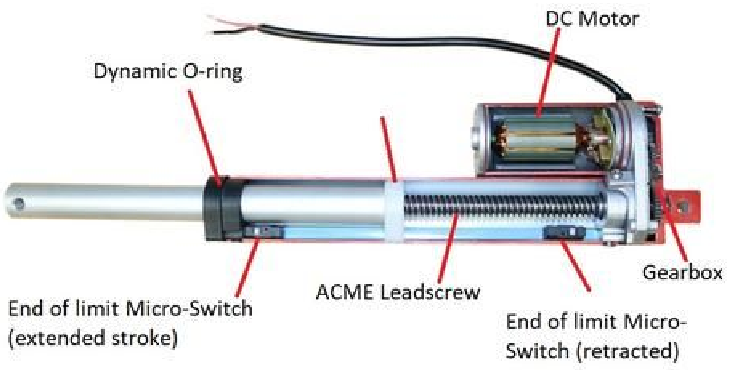

DC Motor
Dynamic O-ring
Gearbox
ACME Leadscrew
End of limit Micro-Switch (extended stroke)
End of limit Micro-Switch (retracted)

Figure 9: Linear Actuator

4.2.5 Lock

Although it is possible to increase the versatility of this product, we are designing this Smart Pet Door to sliding door. With the consideration and understanding that many sliding doors are made of see through material such as glass, we designed our design to be minimalist as possible without compromising the security of the household and the product's ability to have full control of the door. As many sliding doors may be used as patio doors, it's important to ensure the security of the home. Therefore, understanding how sliding doors are locked and its mechanics is crucial. The main function for this lock is to mechanically open and close the sliding door. Security has an important aspect of modern society. With the increasing home invasion and front porch package theft, choosing a lock is important to the safety and security of home. There are three lock mechanisms that will be explored: handle/hinge lock, and bar lock.

Many traditional sliding doors and patio doors will have a barn or hook lock to securely separate the indoor from the outdoor environment. The hook lock usually requires the user to either turn or switch the hook off or on the frame of the sliding door to secure it in place. With improvement in home invasion, hook locks for sliding or patio doors are tracked and susceptible. Our product will improve security and safety of the household utilizing the strength of the door material to protect home invaders from our product. Figure 10 and 11 below shows a modern hook and traditional barn hook lock for sliding door.

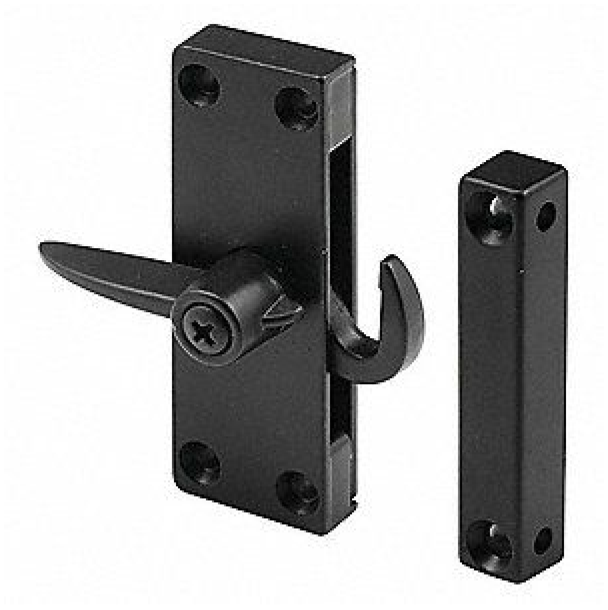


Figure 10: Modern Hook Lock for Sliding Door

Figure 11: Traditional Barn Hook Lock for Sliding Door

Another example of the lock mechanism is Autoslide - Sliding Glass Door Opener. Installing a railing to an existing sliding provides greater control of the door. Although there's a longer installment period, the installment of one's railing system increases the versatility of the product. Having control of the railing system allows for one to expand on the functionality of the product. With the railing locking system. The locking mechanism of the door is found within the product clamping onto its custom railing and the motor.

With our product, it is imperative to gain control of the sliding door. As our product does not intend to interact with the original locking system of the sliding door, we intend to create a startup sequence that serves to determine whether the original lock system is enabled by trying to open the sliding door. If our product is unable to unlock the door, the user will be notified through the mobile application to disable the locking system. Our product will be placed on the railing of the sliding door as opposed to handle, thus preventing intruders from unlocking the door despite turning off the power or breaking the motor housing. Our locking mechanism will reside in the effectiveness of the linear servo actuator to maintain the integrity of the door.

4.2.6 Sensors

During our research we determined that sensors were needed for our design from two of our requirements. Those requirements are that the system should detect any entity within one meter of the door via a sensor and entities within one meter of the door should be identified as an owned pet or not with greater than 90% precision. We plan on using multiple sensors in conjunction with each other in order to satisfy these requirements. The types of sensors needed to achieve this are a motion or occupancy sensor and an identification sensor.

4.2.6.1 Motion Sensor

A motion sensor or occupancy sensor is a device that detects the presence of an entity that is in the sensor's range, which is not a part of the baseline. The three main types of sensors used to detect motion that we researched are passive infrared (PIR), ultrasonic, and microwave. Each type is very different to the next and has their own ideal environments.

The ultrasonic sensor uses a SONAR technique by using sound waves to detect objects. The sensor detects objects by receiving reflected sound waves that were emitted by the device. The time elapsed by the sound waves round trip is used to calculate a distance. We can utilize this calculated distance to then determine if an object has moved into the desired activation range. A major benefit to using ultrasonic sensors is that they are not affected by external sources such as heat, light, dust, etc. that can often cause false positives for other sensors.

A passive infrared sensor works similarly to an ultrasonic sensor in that it uses reflected waves (light instead of sound) to detect an object's presence. The passive infrared sensor detects and compares the heat of an object to the heat of its background. The distance a PIR sensor can detect within is usually adjustable by a component on the sensor itself. Since these sensors measure heat signatures they are mostly applied in indoor, temperature regulated applications. Because of this fact, a passive infrared sensor may not be the best choice for an exterior glass door, where heat and light may be directly cast upon it.

4.2.6.1.1 HC-SR04 Ultrasonic Sensor

The HC-SR04 has a theoretical measuring distance of up to around 450cm, with a practical measuring distance around 100cm, and an accuracy close to 3mm. The HC-SR04 also measures an area within a 15° angle from sensor facing. It operates at 5 volts using less than 15mA and at a frequency of 40Hz. Below is the pinout for a HC-SR04 ultrasonic sensor. Figure 12 shows the pinot of the HC-SR04, which will be used to communicate with our PCB.

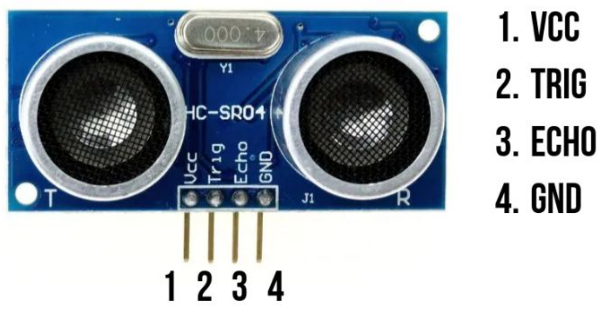


Figure 12: HC-SR04 Pinout (Pending permission from Arduino.cc)

The specifications of the HC-SR04 fit well within our requirement needs as we only want to detect motion within 100cm and will be able to provide the power needed. Accuracy does not matter to us due to the functionality used is not actually measuring a distance. The measuring angle is the only slight drawback for this sensor as it restricts where we can physically place it in our design.

4.6.1.1.2 HC-SR501 PIR Motion Detector

The HC-SR501 has a detection coverage range of up to 7 feet from the sensor and within 120° from the sensor facing. It also has a variable input voltage range of 4 to 12 volts, using around 65mA of power. It can distinguish object movement from human and animal movement and can operate in a temperature range of -20 to +80 Celcius. The HC-SR501 will output a high (3.3v) voltage when detection occurs and it has two operating modes, repeatable and non-repeatable. The HC-SR05 board layout and pinout is shown in figure 13.

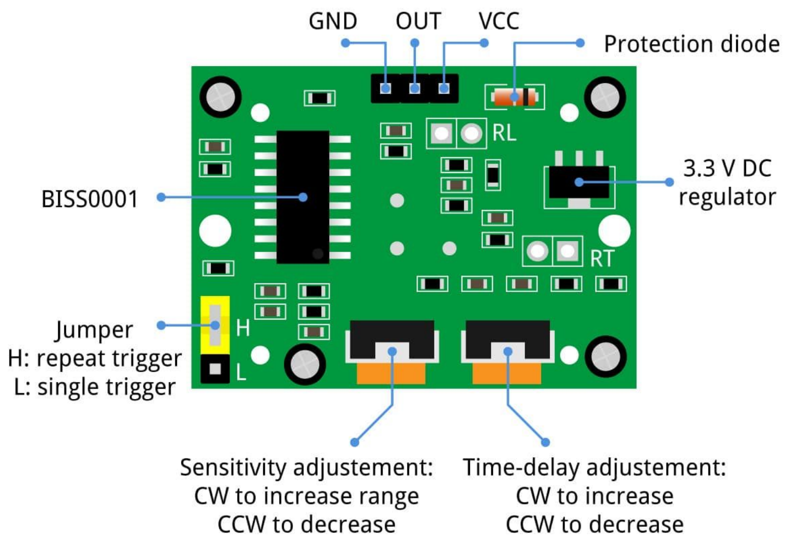


Figure 13: HC-SR501 Pinout with permission from Makerguides.com

This sensor satisfies our requirements for the product design and offers more flexibility in where to physically implement the sensor. Furthermore, the fact that this sensor can recognize human and animal movement separately from object movement could be useful in our design. Unfortunately, the fact that variable external heat can cause false positives will greatly impact our decision to use this sensor. Below is a comparison between the two motion sensors talked about.

The major factors in choosing our motion sensor come from our requirements and specifications. The cost of a motion sensor is important due to the direct impact to the overall product cost. We are also looking to design a compact product, but both sensors are planned to be external, and already have a small footprint. The motion sensor for our design, and the two we have researched, will be connected to our system via the GPIO pins. The input voltage of the sensor now becomes an important parameter due to our single board computers GPIO voltage output. From our requirements we listed, detection range is important to our design, and the sensor must be able to operate efficiently at a distance of one meter. Lastly our design needs to avoid any hindrances a sensor might incur when used in a light and heat sensitive area. Below in table 5 is a comparison of the researched motion sensors and their specifications related to our requirements.

	HC-SR04 Ultrasonic	HC-SR501 PIR
Cost	Owned	appx. $5
Detection Range	480cm, within 15°	213c, within 120°
Input Voltage	5v	4-12v
Hindrances	vibrations	heat and light,

Table 5: Motion Detection Sensor Comparison

4.2.6.2 Identification Sensors

There are a couple different ways to identify an entity using sensors. The most straightforward and simple way is to scan a barcode or QR code. This obviously will not work with the design we have planned. Some other ways are using real time locating systems such as WiFi RTLS and Infrared RTLS. Both of these implementations are too complex of a process to use for the simple design we have. The most commonly used identification sensing available is RFID or radio frequency identification. RFID sensors use radio waves to emit and receive signals from their associated RFID tags. When a tag receives a signal from the sensor it transmits a signal back consisting of its data, usually an ID number.

4.2.6.2.1 RC522 RFID Module

The RC522 is a 13.56MHz RFID module that supports I2C, SPI and UART. It has an operating voltage of 3.3v at about 20mA (10uA when in powered down mode) and a max data transfer rate of 10MBps. The RC522 has a read range of 5cm. This module comes with a single RFID card that has 1kB of memory. The RC522 can read and write into the card memory and costs around $8. Figure 14 shows the pinout for the RC522 RFID module.

Incorporating this module into our design would satisfy the requirements previously stated. The input power and SPI/UART support align with what our possible choices for single board computers can provide. The low power mode is also a positive for the RC522. The read range, however, might be a problem. 5cm might not be a large enough range to read a ID card on a pet's collar.

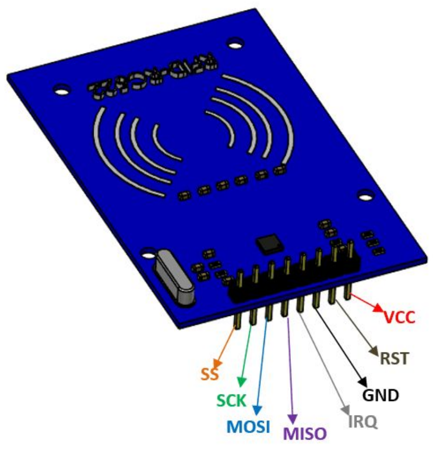


Figure 14: RC522 RFID Module Pinout (Permission pending from Components101.com)

4.2.6.2.2 PN532 NFC RFID Module

The PN532 is a slight step up from the RC522. This module is compatible with multiple RF protocols including Mifare 1K, 4K, Ultralight, DesFire cards, and more. The PN532 also supports I2C, SPI, and high speed UART. The read range is barely larger than the RC522 with 5 to 7cm. It requires 5v input power, and acts as both a reader and writer. The PN532 also supports NFC with Android phones. The price for the PN532 starts at $12.

Again this RFID choice covers all the bases when looking at the requirements for our design. The PN532 has more flexibility with the added support for I2C, which would simplify implementation of the sensor. The read range of only 7cm is still not what we want for our design.

4.2.6.2.3 PN5180 NFC RF Module

The PN5180 NFC RF Module is an exceptionally integrated high power output NFC utilizing contactless communication at 13.56 MHz. The required input voltage is variable at up to 250mA to allow for more flexible applications. The Pn5180 also includes a Dynamic Power Control that controls current through the antenna and RF power to maximize performance. The module also supports 8 different RFID protocols, including ISO/IEC 14443-A up to 848 kBit/s, MIFARE, ISO/IEC 14443-B up to 848 kBit/s, and JIS X 6319-4. The PN5180 communicates through SPI only and has a read range up to 15cm. The price for the PN5180 starts at around $12. Figure 15 shows the PN5180 block diagram and pinout.

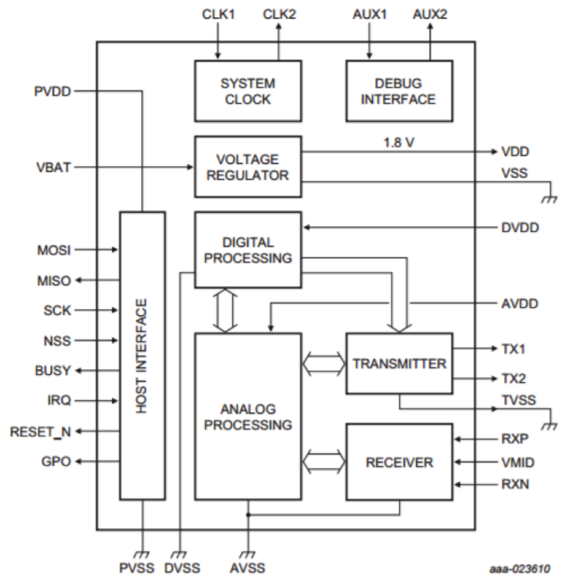


Figure 15: PN5180 Block Diagram with permission from Futureelectronics.com

The PN5180 is well equipped for satisfying the design requirements related to our entity identification and verification. It supports most, if not all, RFID protocols and does not require host controller interaction. It can also be optimized for power using the Dynamic Power control. The biggest positive for this module is the read range, allowing for our design to function as desired. Below is a table comparing the PN5180 to the other RFID modules. Table 6 shows the comparison of the different RFID modules researched.

	RC522	PN532	PN5180
Cost	$8	$12	$12
RFID protocols	MIFARE and NTAG	Mifare 1K, 4K, Ultralight, DesFire, +	Most
Communication protocols	I2C, SPI, UART	I2C, SPI, High Speed UART	SPI
Read range	5cm	5 to 7 cm	Up to 15cm

Table 6: Radio Frequency Identification Module Comparison

4.2.7 Security Peripherals

The security element of our design includes having video coverage and two way audio. We will require a camera that can provide a live video feed of the immediate area around the door, preferably both inside and outside. We would like to connect the camera to the rest of the system via wire to avoid having to power it separately and to simplify the data transfer. The design is also based on the user being able to listen to the surroundings as well as communicate with the pet if desired, which will require a speaker and microphone. There are many ways to implement these three peripherals, either individually or combined.

4.2.7.1 Raspberry Pi Camera Board v2

This camera option comes directly from the same makers of the Raspberry Pi. It is a 8 megapixel Sony IMX219 image sensor with a fixed focus lens. The Raspberry Pi Camera Board v2 connects via CSi interface and offers the ability to take images and video, providing 3280 x 2464 pixel for static images, and supporting 1080p30, 720p60, and 640x480p90 video. The software is directly supported by the Raspbian OS. The optic size is ¼ inch and pricing is $30.

We like that the camera uses the CSi interface, because it is a great way to transmit pixel data, and is much faster than other data transfer options.The Raspberry Pi Camera Board v2 also has great video support, offering higher quality than what is needed for our design. Without a CSi ribbon extension, this camera can only be placed within inches of the single board computer. Overall this is a good option for our video needs.

4.2.7.2 Logitech C270

The C270 is a USB Webcam that comes with an internal microphone. The widescreen HD camera has a max resolution of 720p at 30 fps, has a fixed focus lens, automatic light correction, and has a field of view at around 60°. The built in microphone is noise reducing and can pick up sound up to 3m. The C270 also comes with a 5 foot USB 2.0 cable. The price of the Logitech C270 starts at $40.

This option gives us more flexibility with less quality. The camera resolution should be enough for what we require from it. It has a decent field of view and a built in microphone which would eliminate the need for a separate audio input peripheral. The 5 foot cable included should be enough for an installation that can view the door the way we want. This camera is priced near the middle of the field for similar products. Table 7 shows the comparison of the different cameras researched.

	Raspberry Pi Camera Board v2	Logitech C270
Cost	$30	$40
Resolution	1080p30, 720p60, and 640x480p90	720p30
Interface	CSi	USB 2.0
Microphone	No	Yes

Table 7: Camera Peripheral Comparison

4.2.7.3 Plug and Play Home Studio Adjustable USB Microphone

This USB microphone can sense audio in a 360° omnidirectional space up to and beyond 2 meters. It also provides anti-noise control at a signal to noise ratio of more than -67 dB. This microphone has a working voltage of 4.5v and a cable length of 13.5cm. The price for the USB microphone is $8.

We would only need to include this if we chose a camera that doesn't already come with a built in speaker. The working voltage of 4.5v is a little worrisome due to the 5v power requirement for the single board computers themselves. If we add too many peripherals that need more power, we would then need to add a hub. Despite these concerns the microphone would meet our audio design and requirements. Table 8 shows the comparison of the different microphones researched.

	Logitech C270	Plug and Play Home Studio Adjustable USB Microphone
Cost	$40	$8
Reception length	3m	2m
Operating Voltage	4.5-5v	4.5v

Table 8: Microphone Peripheral Comparison

4.2.7.4 HONKYOB USB Mini Speaker

The HONKYOB USB Mini Speaker is connected via USB 2.0 and consists of two 0.5W stereo speakers. It also comes with a 47 inch cable. The HONKYOB USB Mini Speaker has an operating voltage of 5v. The price for this speaker is $14.

If we end up having an externally powered USB hub then this speaker should be a viable option. The length of the cable gives us a lot of flexibility for implementation. Without a powered USB hub this speaker might draw too much power and force us into other peripheral options.

4.2.7.5 FIYAPOO Mini Portable Speaker

This speaker is a portable device with a single 3W speaker. It is connected via a 3.5mm audio jack, has an internal 3.7v rechargeable battery, and comes with a charging cable. This speaker provides a signal to noise ratio over 80dB. The price of the FIYAPOO Mini Portable Speaker is $18.

This speaker would allow us to operate USB peripherals without having to compete for power, but also has the draw back of needing to be recharged. It also does not come with a cable, so an extender will be needed for physical design purposes. Table 9 shows the comparison of the different speakers researched.

	HONKYOB USB Mini Speaker	FIYAPOO Mini Portable Speaker
Cost	$14	$18
Interface	USB	3.5mm audio jack
Operating Power	5v via USB	3.7 via external
Output	2 x 0.5W	3W

Table 9: Speaker Peripheral Comparison

The major factors in choosing our security peripherals come from our requirements and specifications. We need to account for the total cost of all the security peripherals and expect this cost to be a decent portion of the total design budget. Therefore, we must choose wisely between the performances provided and the cost that is associated with them. These security peripherals can be internal or external to the main product design, so keeping their sizes in mind should be important to our decision making process. How each security peripheral connects to our single board computer is also an important factor to consider, making sure we have enough of a certain connection available. From our requirements we must have one way visual and two way audio communication, so we must choose our peripherals in order to meet these needs. Lastly we need to compare how each peripheral will affect the overall system power requirements. Table 10, seen below, is a comparison of the researched security peripherals and their specifications related to our requirements.

	Option 1	Option 2	Option 3	Option 4
Video	RPi Camera Board v2	RPi Camera Board v2	Logitech C270	Logitech C270
Audio In	USB Microphone	USB Microphone	Logitech C270	Logitech C270
Audio Out	USB Mini Speaker	Mini Portable Speaker	USB Mini Speaker	Mini Portable Speaker
Total Cost	$52	$56	$54	$58
Connectivity	CSi + 2 x USB 2.0	CSi + USB 2. + 3.5mm jack	2 x USB 2.0	USB 2.0 + 3.5mm jack
Power Usage	3 x 5v	2 x 5v + 3.7v	2 x 5v	5v + 3.7v
Complexity	High	High	Medium	Medium

Table 10: Comparison of Security Peripheral Choices

4.2.9 LCD Display

Our design also features a display that can be placed next to the sliding glass door. This will be a small screen that aims to not crowd the door with wires too much. This will be used to display some basic information such as the current mode the user is in, if the door is locked or unlocked, and if the pet is inside or outside. We take a look at the possible options for the design team in this section.

4.2.9.1 Sunfounder LCD 1602

The Sunfounder LCD 1602 [SUNF] module consists of a blue background with white display text. The option that we are considering has 16x2 possible characters available for us to show any message that we might need. This runs on a working voltage of 5 V and we have the option to adjust the brightness with a 50k potentiometer. There are other options from the same manufacturer such as 20x4 characters. With a size of 80x36x12 mm this would be a good option that would not take up a lot of space next to the sliding glass door.

Another flexible part of this option is the I2C module. Sunfounder also offers the same 16x2 display with the I2C module already soldered on to the back with header pins. This could make the connection to the Raspberry Pi easier as we could use I2C communications to write the message instead of having to solder and wire up each of the 16 header pins ourselves. Then again this could also be a drawback as we would be required to use the I2C module and take away some of the design freedom as just getting the display screen by itself.

This option is appealing because of the features mentioned above and all of the design team has experience using this product. The past schematics, board diagrams, and experience could decrease the time it takes to implement the display screen if this option is chosen. Figure 16 shows Sunfounder LCD 1602 Display.

Figure 16 : Sunfounder LCD 1602 Display

4.2.9.2 Sparkfun SerLCD

The Sparkfun SerLCD [SPAR] is another similar sized LCD as the Sunfounder option. With an 81 x 38 mm device the requirement of having a somewhat small device is met. This also has 16x2 characters available for displaying the messages or statuses. The feature that stands out about this option is the RGB backlight and the black text. This means that we would be able to make the background of the text red, green, blue, or any color combination of the three. This feature may not be needed but it would allow us to assign different colors to the different operation modes that the system is in. The people at home would then easily be able to see what mode they are in or in another example, if the pet is outside.

Another feature with the Sparkfun SerLCD is that it has an Arduino compatible bootloader in the AVR ATMega328p on the back. This makes it easy for an Arduino to communicate with the LCD screen which is not needed for our purposes. Other communications involve UART, SPI, and I2C. However unlike the Sunfounder LCD, there is not an option to have an I2C module soldered on with the part purchase. The screen has a required 3.3 working voltage but is also equipped with a 3.3 voltage regulator that will handle voltages up to 9 volts.

While this LCD has many great features it also has a higher price. The RGB backlight screen could be a great addition to our design, but simple LEDs alongside the screen could also accomplish the same thing. This option will increase our project cost due to having additional features which may not be worth it, especially because we would not be using all of them. Figure 17 below shows the Sparkfun SerLCD.

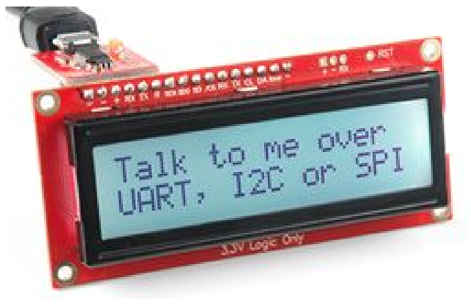


Figure 17 : Sparkfun SerLCD

4.2.9.3 Uctronics OLED

This display option is an organic light-emitting diode screen rather than a liquid crystal display. It is made up of 128x64 pixels separated into 2 yellow rows and 6 blue rows as demonstrated in the example picture. The shape is a smaller square compared to the other two rectangular options, measuring in at 27 x 27 x 4 millimeters. Although it is smaller, the 8 rows would give us more than enough screen space to convey the necessary information. That small size could also bring up a downside of having the letters too small for some users to read.

The Uctronics OLED [UCTR] needs to use I2C to communicate with the Raspberry Pi and it is advertised as being compatible with the Arduino and the Raspberry Pi. This shouldn't be a problem for our design but it does take away from the possible design options with this display board. As with the other two options, the Uctronics OLED board has a support voltage between 3.3V and 5V.

While having a screen that is not too large is important to us, we also need to think about if we are going to need the 8 possible rows of OLEDs. Along with considering if the users are going to have a hard time seeing the information displayed on this screen compared to the two other larger font LCDs. Figure 18 belows shows the Uctronics OLED Module.

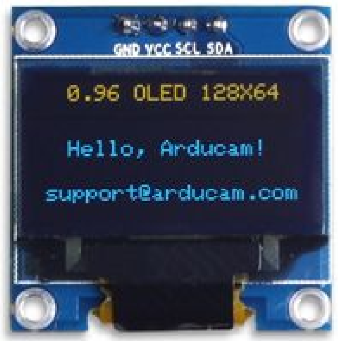


Figure 18 : Uctronics OLED Module (pending permission from Uctronics)

4.2.9.4 LCD Comparison

Table 11 below shows a comparison of the three LCDs that were mentioned in this section. We will use this table to decide which product is the best option for us to implement in our project. The Sunfounder LCD is familiar to all of the members on the team as it was used in a previous college course at the University of Central Florida. This aspect is not listed in the table but adds an advantage to the Sunfounder option due to a low learning curve. This would mean we know what to expect when it comes to wiring up and programming the LCD to have the output that we want. Given that, the Sunfounder LCD is at the top of the list and the choice will be explained more in the Hardware Design section.

	Sunfounder LCD 1602	Sparkfun SerLCD	Uctronics OLED
Price	\$6.49, (\$8.99 I2C)	\$19.95	\$6.99
Size	80 x 36 x 12 mm	81 x 38 mm	27 x 27 x 4 mm
Characters	16 columns x 2 rows (32 total)	16 columns x 2 rows (32 total)	8 total rows
Pixels	N/A	N/A	128x64
Working Voltage	5 V	3.3 V	3.3V – 5 V

Table 11 : LCD Comparison

4.2.10 LEDs

Along with the display screen that was talked about in the previous section, we are also considering LEDs to indicate to the user what mode the system is in. This would be placed along one of the sides of the LCD screen so they can easily see the mode or something along the lines of the pet being outside or the door being locked. If the Sparkfun SerLCD was selected then the need for the extra LEDs may not be necessary but we still look into the different options in this section.

4.2.10.1 Single Color LEDs

The first option is the single color LED that is common in most beginner electronic kits or classes. These could be used with three different colors to represent the three modes of operation of ‘Away’, ‘At Home’, and ‘Always Closed’. We could also implement a fourth color to signify an open door or a pet outside. A pack of 100 LEDs is priced at $5.55 from eboot [EBOT]. This would include 10 different colors from red, orange, yellow, green, emerald green, blue, purple, warm white, white and color flash. Each of the LEDs has a max current of 20 mA and 1.8 – 2.0 forward voltage for the red, yellow, green, and orange colors. While the blue, white, warm white, emerald green, and purple colors have a forward voltage of 3.0 – 3.2 V. This would give us more than enough options and flexibility for our design.

4.2.10.2 RGB LEDs

The alternative to single color LEDs are RGB LEDs which have the ability to change colors based on different voltage levels. They are common cathode but made with 4 pins instead of 2. The different voltage levels on each pin is what produces the multiple color options. This would change our design from needing three or four single color LEDs to just one RGB LEDs. We could just change the color on the one RGB LED to still represent each operation mode. These LEDs also come in a pack of 100, which also includes a 300 pack of resistors and is sold for $8.99 from EDGELEC [EDGE]. Voltage and current values are about the same as the single color LEDs with current at 20 mA and red voltages at 2.0-2.2 V and blue and green voltages being at 3.0-3.2 V. This option could reduce the amount of electronics mounted next to the door while also still allowing us to use the visual feature.

4.2.11 Wifi Module

A major part of our design for Pet Connect is that the pet owner can use the mobile app to control the door opener and get notifications based on the sensors. In order to do this the single board computer needs to connect to the internet by the way of a Wifi module. While the Raspberry Pi 4 does have a built in wireless chip, we also research other options for the case that we end up not using the Raspberry Pi 4 and to make sure it is the right choice for the project design.

4.2.11.1 Raspberry Pi 4B Built-In Wireless

One of the Raspberry Pi 4B’s main features is that it comes with built in wireless connectivity. This is a dual-band 2.4/5.0 GHz IEEE 802.11ac wireless that allows us to connect the Raspberry Pi to the internet and the mobile application. We could also do this through the Gigabit Ethernet port but this is not ideal as the Pi might not be close to the home router. The wireless feature also allows multiple options with remote access. This would through the internet with port forwarding, and using a wenclude using the Pi as an Access Point, accessing the Pi b server on the Pi. We would most likely use the wireless to connect to the mobile app or to connect to the cloud hosted database, but the flexibility of the options is an added bonus.

This would be a great option for our project design as the wireless would already be built in and would not require any solder or configurations that would be needed if we bought another wifi module or single board computer. The built in wireless would also be integrated with the Raspberry Pi 4B and would make it easier to set up. As mentioned in the single board computers section the Raspberry Pi has great third party and community support that would help us if we ran into any problems when connecting to the Raspberry Pi 4B.

4.2.11.2 Sparkfun ESP8266

The ESP8266 from Sparkfun is another option that would allow us to connect our mobile app with the smart board computer. It works with any microcontroller or single board computer with the need of some soldering or jumper wires. It has an integrated TCP/IP stack to support 2.4/5.0 GHz 802.11 b/g/n IEEE standard. Like the Raspberry Pi, there is also great community support that could aid us with the configurations and usage.

Priced at $6.95, the ESP8266 would add to our project cost which is a drawback. Another drawback would be the added design to implement this wifi module through soldering and wire connections. This option does not make sense if we are going to use the Raspberry Pi 4B that is already built in. This wifi module would also not have the same simplified integration that comes with the Raspberry Pi.

4.2.11.3 Adafruit Mini USB

The Adafruit Mini USB can add wifi capabilities to any microcontroller or single board computer that has a USB port. For this option, all we would need to do is plug it in and do some simple configurations and the board would be connected. It offers 2.4 GHz on the IEEE 802.11 b/g/n standard but does not support the 5.0 GHz option that the other two modules have. This product is advertised as a way to add wifi to older versions of the Raspberry Pi that did not have the wireless built in. With an $11.95 price, this would be a good option for older models of single board computers.

For our purposes, the Mini USB would allow us to use an older version of the Raspberry Pi and still have wireless capabilities. The plug and use set-up would be much simpler than the Sparkfun ESP8266. However, this would take up a USB slot that we might need to connect certain peripherals. This option would seem more justifiable if we needed to use the older versions of the Raspberry Pi or something that does not already have wireless already built in.

4.2.11.4 Wifi Module Selection

After looking at the comparison table below, we have decided to use the built-in wireless option that comes with the Raspberry Pi 4B. This is the obvious choice as it does not increase our project cost, it has great community support and the convenience could not be matched by any other option. The Sparkfun ESP8266 and the Adafruit Mini USB will act as backup options if we run into major problems with the Raspberry Pi 4B. Table 12 comparison of wifi module selection.

	Raspberry Pi 4B Built-In Wireless	Sparkfun ESP8266	Adafruit Mini USB
Price	Free($35, with Raspberry Pi 4)	$ 6.95	$11.95
Connection Method	Built-In	Soldering/jumper wires	USB Port
Compatibility	Raspberry Pi 4B (Built-in)	Any microcontroller (Arduino, MSP etc.)	USB Port needed as a part of microcontroller
Standard	2.4/5.0 GHz IEEE 802.11ac	2.4/5.0 GHz IEEE 802.11b/g/n	2.4 GHz IEEE 802.11 b/g/n
Community Support	Great	Great	Good

Table 12 : Wifi Module Comparison

4.3 Software Research

In this section, we conduct various software research as part of Software Development to gain an understanding of the problems that we can face during the development. This process provides development organization, determine resourcing needs for the project as well as decide which technologies to use and how to implement them. Our software research consisted of determining the type of mobile application, development environment, programming languages, database selection, and wireless communication.

4.3.1 Mobile Application

One of the major goals of developing this project is to create the Internet of Things product that is easy for pet's parents. We want our product to become the most promising technologies that will change and accommodate pet owners' daily life and their beloved pet. We conclude that mobile applications will be the great support to implement hardware and software. We want our product application as the central control system dashboard that allows users to manage devices via mobile with internet-enabled things. This Dashboard also should collect information from sensors and show it to the users. Our objectives for creating mobile applications is to provide user experience, protect security, and support real-time access.

4.3.1.1 Native Mobile Application

Native mobile application is software that is designed for iOS, Android, or Windows Mobile. Users require these apps through an online store or marketplace such as the app store or android app on google play so these native apps do not run in the browser. They are designed and developed for one or specific platform so it can take full advantage of all device features for example camera, list of contact, voice assistance or so on.

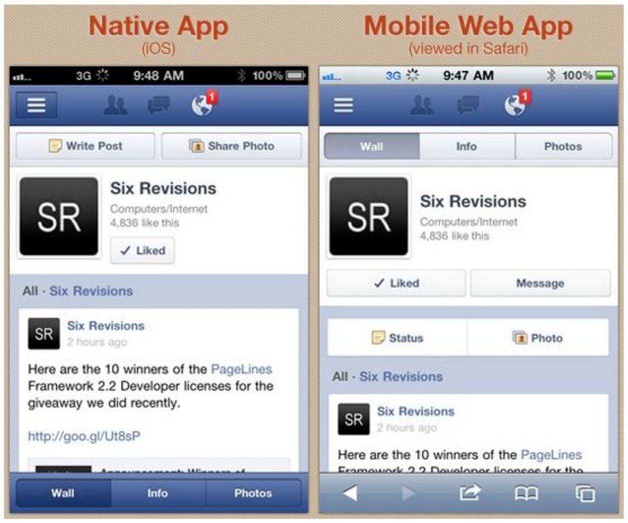
Native applications offer offline support without internet activity based on data caching like device's notification system that can work off-line. Information message can alert the user for occurrences events within the system weather is actioned-required notification or passive notification. In addition, native application development has inherent support for all devices and offers SDK level to support lower end devices and it directly interacts with native APIs without depending on middleware such as plugins. Here are the benefits and capabilities that Native Mobile Application support developments: Immediate access to new features, ability to add AI and Voice Assistant, easy publish from app to app stores, increase usability/ User experience.

Referred to Smart Phone app in section 3 which required strong response time and cost in our mobile app. Native development mobile applications consider having very good response time and are limited only by device specification. The cost of building native app on the other hand will be very high due to minimal code reuse across platforms, however native mobile development is likely to meet our requirement since the system can access built-in features of the smartphone such as microphone and camera by default which is the most important part to operate the device in this project.

4.3.1.2 Mobile Web Application

Mobile Web applications are not real applications; they are websites that look and feel like native applications. They can be accessed through the mobile device's web browser and they do not need to be downloaded and installed on the device. This application is written in HTML, CSS and JavaScript which can leverage a wide range of frameworks and libraries such as angular, PHP, Python or react. There are no standard software development kits (SDKs) that are required to make mobile web apps. One of the biggest challenges for developing mobile web applications are common application UI control, tab button, camera access and push notification which play a major part in our project. Those components can be recreated however it might lead to our application not working as expected.

In 2017, Google introduced Progressive Web Applications (PWA) that allow Mobile Web Application to adopt more app-like features such as push notifications, offline capabilities, app icon and more. Microsoft also announced their plan and solutions to bring Progress Web Application (PWA) become “first class citizens” on Windows 10. This solution makes it easier for developers to create applications from a single code base which can then be used on various platforms and quickly updated with new features or bug fixes by simply deploying to servers. Figure 19 Compare UI develop from Native and Mobile Web App.



Here are the 10 winners of the PageLines Framework 2.2 Developer licenses for the giveaway we did recently.

Figure 19: Compare UI develop from Native and Mobile Web App

4.3.1.3 Hybrid Mobile Application

Just like native apps, hybrid apps run on the device, throughout the use of plugin, these applications can access to mobile device’s features. Hybrid apps are written with HTML, CSS, and JavaScript and run from native container and its own embedded browser, which is invisible to users. Hybrid apps depend on WebViews which are in-app browsers to allow mobile applications to render user interface and web contents. This is how Android and iOS devices can run hybrid apps as native mobile applications. Even though development costs from building the hybrid app met our requirement in section 3 since the code can be reused across platforms, device support for low-end devices will be a challenge. Table 13 shows key features for Native, Web & Hybrid.

Feature	Native	Web-Only	Hybrid
Device Access	Full	Limited	Full (with plugins)

Feature	Native	Web-Only	Hybrid
Performance	High	Medium to High	Medium to High
Development Language	Platform Specific	HTML, CSS, Javascript	HTML, CSS, Javascript
Cross-Platform Support	No	Yes	Yes
User Experience	High	Medium to High	Medium to High
Code Reuse	No	Yes	Yes

Table 13: Comparison for Native, Web, and Hybrid Mobile Application

4.3.1.4 Development Environment

Development platforms help designers and developers build the software and application they need. Difference platforms have different features to support our design project. We want development tools that are easy to use and have enough online resources available for developers to learn. Below are the lists of IDE that have potential for our developer to use during iOS/Android development.

4.3.1.4.1 Android Studio

Android Studio is Google's supported IDE for developing android apps. During the process of software research, our main focus is target audience and android users are also one the main users in our product. Android studio is a native mobile development tool that provides shortcuts for coding and designing; its layout design makes it very easy to use which can reduce time of coding. Android studio supports all the same programming languages like Java or C++.The best part of android studio is providing drag and drop features to design the layout for our project. This is an open source tool which meets our requirement for smart phone app cost. The developer can use Android's built-in developer kit for free as well.

System requirements:

- Microsoft Windows 7/8/10 (64-bit)
- Mac OS X10.10 Mac OS X 10.10 (Yosemite) or higher, up to 10.13 (High Sierra)

- Linux OS Tested on Ubuntu 14.04 LTS, (64-bit distribution capable of running 32-bit application
- 4 GB RAM minimum, 8 GB RAM recommended.
- 2 GB of available disk space minimum, 4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

4.3.1.4.2 React Native

React native is cross-platform development tools that can develop apps for other platforms or operating systems. We also want our mobile application to be published and used on both an android phone and iPhone therefore considering cross-platform tools can be beneficial for both smartphone users. React Native was developed by Facebook community in which their framework is based on JavaScript Technology. This is also an open source tool which also allows developers to develop apps that feel like native one. Expo is a framework of tools that service this platform without touching Android studio or XCode. Vs code is recommended IDE for code editor.

System requirement:

- JDK for Windows or SDK for Mac (Version 8 or newer)
- Visual Studio Code
- Android studio for Android app or XCode for iOS app
- Build-emulator for Android Studio or built-in iOS Simulator
- Node Package Manager (NPM)
- Node.js (Version 8 or Newer)
- React Native Command Line Interface (React Native CLI)

4.3.1.4.3 Flutter

Flutter is also a cross-platform mobile application developed by Google. It was released in 2018. Flutter framework is an open-source UI software development kit, it is considered high quality native interfaces on iOS, Android, Mac, Linux, and Windows. Flutter uses language Dart which is similar to Java or JavaScript. Flutter is single code based that can be run in multiple platforms therefore it saves time for the developer to code since the application will work the same on different platforms. Other benefits for single based code are Low-Cost App development.

Windows

- 7 SP1 or later (64-bit)
- Disk Space: 400 MB (does not include disk space for IDE/tools).
- Tools: Windows PowerShell 5.0 or newer (this is pre-installed with Windows 10)

Mac:

- 64-bit macOS Disk Space: 2.8 GB (does not include disk space for IDE/tools)
- Tools: Access to bash, curl, git 2.x, mkdir, rm, unzip, which.

Linux:

- Operating Systems: Linux (64-bit)
- Disk Space: 600 MB (does not include disk space for IDE/tools).
- Tools: curl, git 2.x, mkdir, rm, unzip, which, xz-utils
- Shared libraries: libGLU.so.1 - provided by mesa packages such as libglu1-mesa on Ubuntu/Debian

4.3.1.4.4 Development Environment Selection

Based on Mobile Operating System Market Share Worldwide, data shows that the percentage of Android users as of June 2020 is 74.14 %. As initial development, we want to target as many users as we can, the feedback and comments from the user experience can help with our further development and implementation, as a result we narrow our selection development to Android native mobile applications. Android development benefits easy for integration, Open Source Platform, and easy for adoption. Android studio became our top choice for IDE because not only Android Studio now is Android's official IDE but also Android Studio has the Android Emulator that can be used to run the test faster than a real device. Our system features require fast and real-time testing on data transformation therefore testing tools and framework can help testing Android apps. Since we want the IDE that allows us to write native Android apps so react native and flutter are not the options because they are the framework for developing cross-platform apps.

4.3.1.5 iOS Exclusive Development

iOS is a mobile operating system created and developed by Apple which runs the iPhone, iPad, and iPod Touch device. iOS is the second-largest operating system after android. It supports supports Objective-C, C, C++, and swift programing language or build cross platform native application using React Native (JavaScript) or Xamarin (C# & F#). Unlike Android, iOS is a source-closed platform that was created for Apple's hardware therefore mac devices are required for iOS platform to run the latest version of Xcode for IOS simulator. Xcode is APPLE's IDE for both iOS apps and Mac. Windows developers can solve this issue by installing VM and Virtual Box to install Xcode. Testing an iOS app on a real device is critical and very important for our application since we want our product to remotely operate the device. Testing can determine performance; Beta testing can provide feedback such as testing Push Notification, data storage, and third-party API. Cloud testing also tests applications on real devices to ensure quality of our application. In addition, we need to consider the cost to deploy the application to the app store therefore there is a member fee associated with iOS development as well.

System Requirement

- A Mac with macOS Catalina
- Intel i5 or i7 equivalent CPU 2.0 GHz
- At least 8 GB of RAM
- At least 265 GB disk Storage

4.3.1.6 Android Exclusive Development

Android is a mobile operating system developed by Google. It is known as the world's most popular operating system that overtake windows. Android is released as an open-source format to help advance standard mobile devices. During the discussion of our research and to meet our software requirement, our team agreed that we want to mainly have Android applications for a supposed number of Android users. To understand its operating system is critical for mobile development. Android in developer perspective is a Linux-based operating system for smart phones and tablets. Android supports various applications through Google Play Store. The android platform allows end users to develop, install their own application on top of its framework. Android SDK tools is a component for Android SDK necessary for the developers to build, test, and Debug apps for Android.

System Requirement

Window/ Mac

- Microsoft® Windows® 7/8/10 (32- or 64-bit)
- Mac® OS X® 10.10 (Yosemite) or higher, up to 10.12 (macOS Sierra)
- 3 GB RAM minimum, 8 GB RAM recommended; plus 1 GB for the Android Emulator
- 2 GB of available disk space minimum,
  4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution
- For accelerated emulator: Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality

Linux

- GNOME or KDE desktopTested on Ubuntu® 14.04 LTS, Trusty Tahr (64-bit distribution capable of running 32-bit applications)
- 64-bit distribution capable of running 32-bit applications
- GNU C Library (glibc) 2.19 or later
- 3 GB RAM minimum, 8 GB RAM recommended; plus 1 GB for the Android Emulator.

4.3.2 Development language

To consider the language to use in our mobile application, the first thing we must consider is the developer's development background and skills. We need to ensure that our project meets the deadline so choosing the language that our development team is comfortable with would be easier for integration, adaptation, and implementation between the software and the device. Development environment also takes a major factor for choosing programming languages as well. Below are programming language options for software development.

4.3.2.1 Java

Java is the most popular choice for developing Android mobile applications and it supports Android Studio. Java was created for the game developer and launched by Sun Microsystem in 1995. Java is strongly typed, compiled language [14] which means error and exceptions are more likely to occur because each type of data is predefined, and constants or variables defined must be described. Java also relies on virtual machines to execute the program [15] Google classified JAVA as the official language to use in developing Android apps and most chosen for developers with games in play store. On the downside Java is among most different languages to learn but since Java language has been around so many years there are many tutorials online for starters available. In this project, most of our developers have an Object-oriented Java background and we can bring visual designers along with features that are advanced and powerful. However, our main requirement for creating an Android and iOS app by using a single codebase is concerning. Time is of essence to make sure we meet our deadline therefore Java language will not be a good option.

4.3.2.2 Dart

Dart is a new language launched in 2011. It is developed by Google and is used to build mobile, desktop, server, and web applications. It is open-source, object-oriented, client-optimized language for applications on multiple platforms on both native and Cross-platform mobile development. Dart uses C-style syntax and optionally transcomplies into JavaScript. Dart can be very flexible because it can be efficiently compiled AOT or JIT or into other languages. In 2013 Dart team announced an update that Dart-to-JavaScript compilers now run faster than handwritten JavaScript with the dart2js compiler. Dart language support flutter for native mobile development on Both Android and iOS [16]. Advantage of Dart is flexible, fast, and by having hot reload can benefit our implementation and testing during our development.

Dart languages seem to meet our criteria for mobile development because single codebase can be re-used across platforms and hot reload that can support function mode in our device. Our developer still needs to take some time to learn this new language before starting building the applications.

4.3.2.3 JavaScript

JavaScript or JS is lightweight, Object-oriented, and best known as the scripting language for web pages. JavaScript runs on the client side of the web which can be used for UI design and how events occur in the program. The basic syntax of JavaScript is like both Java and C++. Javascript is a non general programming language however JavaScript with HTML5 making a viable way to deliver browser-based applications to mobile devices. There are various JavaScript framework developer's choices to use during development applications. Here is some popular JavaScript framework.

NodeJS is an open-source, cross platform, Java-script runtime environment that executes JavaScript code outside web browsers. The operating support Window, Linux, MacOS, SmartOS, Free-OpenBSD, and IBM. It is a server-side platform built by Google Chrome's JavaScript Engine. This JavaScript is part of MEAN stack.

ReactJS considers more JavaScript libraries. It is open-source JavaScript released in 2013 introduced by Facebook to build dynamic user interfaces. React uses the Virtual DOM which makes it easier for developers to update changes performed by users without other parts of the interfaces.

AngularJS is a client/server fully featured framework developed by Google. Google introduced NativeScript with angular framework for Android, iOS, and Windows Platform. This framework comes with JavaScript virtual Machine to bridge modules and runtimes.

4.3.2.4 Development Language Selection

Based on section 4.3.1.4 for Development Environment, the team concluded that we want to choose Android studio for our development platform. Therefore Java language seem to be a right options for our project development based on following reason; Our team member has more experience in Java development so it would be easier to learn and develop the code without new learning, Second Java is known as the official language for android app and it is one of the most support language by google and Play Store. Moreover, there are many Android's API's available for the Java language. We tend to use some API in Camera, audio, and push notification which are the main system of our mobile application so using Java language can support our development.

Dart does not fall in our selection even though dart is a client-optimized language for fast apps on any platform. To use Dart, we need to install flutter and dart plugins to Android Studio. There are a few downsides of Dart is that Dart class code cannot be written. We want a classes-based object-oriented programming language that can be used in place for all classes in our package. Another reason is learning time: our developers need to learn a new language which can be time consuming and a big factor that can affect development time. Last, since we want to build native app, therefore Javascript can not be support by choosing platform like Android studio

4.3.3 Databases

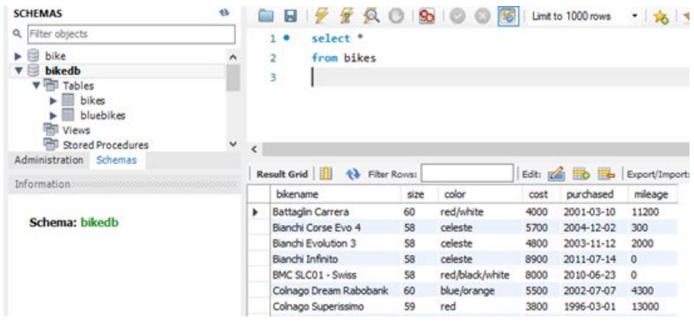
While our project is not the most data driven application there is still some data being passed around that we need to be keeping track of. To do this we are going to need to implement a database management system. This will need to hold things like the user's username and password, a pet ID, a pet name, and possibly a door ID. In this section we will lay out the options that we considered when selecting which database system to use for our project. We are using price, compatibility, connectivity, and ease of use to determine which database to go with.

4.3.3.1 MySQL

MySQL is a relational database management system that was first released in 1995 [MYSQ]. It is now a popular choice for web app developers as it is a part of the commonly used LAMP stack of Linux, Apache, MySQL, and PHP. It is open source and free to use which already make this an appealing option. Security is something that MySQL is known for as some major web applications use this such as Facebook and Twitter. As we could possibly be storing sensitive information this is definitely something to consider with MySQL. The minimum hardware requirements have a RAM of 4 GB and 6 GB recommended. This means that MySQL would need to be set up on a hosting computer for our app. For our purposes this is a downside to using MySQL instead of another database that does not have to be installed as a server database.

As far as connectivity, MySQL offers many options and is not that limited. It even has Connectors for the possible programming languages that we might use to develop the mobile app for our project such as, Java, Python and JavaScript. If we do not choose to use a connector we also have the option of writing the backend API's with the PHP language as it was mentioned above as part of the common LAMP stack.

MySQL also comes with a Workbench that is easy to use and install. This tool would be useful for initially creating the database tables as well as for testing sample data. It should be noted that the developers of the mobile application have some experience with MySQL and MySQL Workbench as this will decrease the time spent on learning a new database management system. Figure 20 is a visual representation of the MySQL database in use.



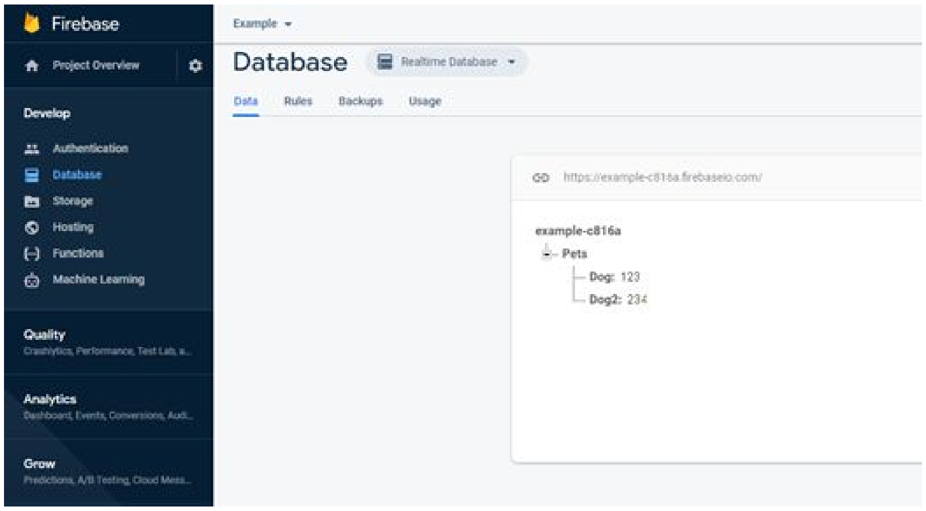
```
select *
from bikes
```

bikename	size	color	cost	purchased	mileage
Battaglin Carrera	60	red/white	4000	2001-03-10	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-02	300
Bianchi Evolution 3	58	celeste	4800	2003-11-12	2000
Bianchi Infinito	58	celeste	8900	2011-07-14	0
BMC SLC01 - Swiss	58	red/black/white	8000	2010-06-23	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-07	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000

Figure 20: MySQL Workbench Example (pending permission from MySQL)

4.3.3.2 Firebase

Firebase is Google's database management system. It aims to make app development easier for developers by having a NoSQL database that is stored on the cloud rather than an application server. It also provides the features of syncing all data across all clients in seconds and is still accessible when the app is not connected to the internet. Another major benefit to using Firebase is that it is available with documented support for iOS, Android, JavaScript and other languages. As with other databases, Firebase secures the data we want to store and provides a better way to authenticate users. It gives the option to have our users sign in with their Google, Facebook, or Twitter accounts while also using phone number or email and password authentication. Firebase also offers some tools to help us manage our project such as Performance Monitoring, Crashlytics, and Test Lab. All of the features are laid out together in an easy to use web console shown in the picture below. Companies like Venmo and Lyft trusting their data with Firebase, are two good examples of why we can also trust our data with it. Figure 21 shows an example of the Firebase database console.

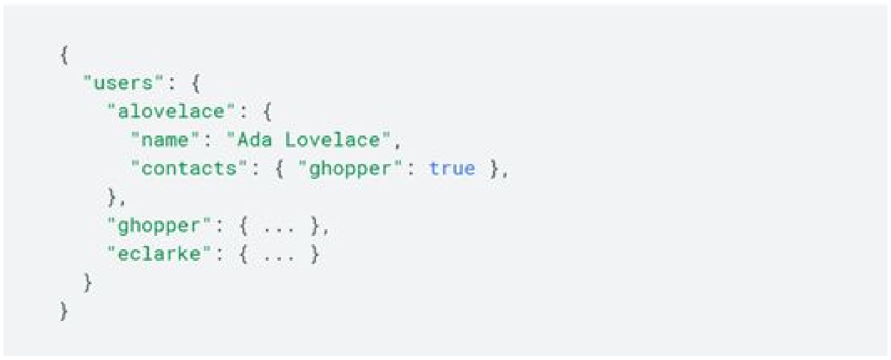


```
example-c816a
  Pets
    Dog: 123
    Dog2: 234
```

Figure 21: Firebase Console Example

For our design we like the free option that comes with Firebase. It gives us free app distribution, authentication, 1 GB of storage data, 5 GB of file storage (such as photos or videos), and the possibility of 100 simultaneous connections. All of those should be more than enough for our project design. If in the future these limitations become too much of a problem we can always upgrade to the Blaze Plan. Another pro is that with whatever platform we choose to go with Firebase is easily connectable and there are a multitude of guides to help us get started. We also have the option of choosing between their two types of databases, the Realtime Database and the Cloud Firestore. These options are very similar but the Cloud Firestore is for larger scale applications while the Realtime Database [FIRE1] is more suited for our project design at this time. However, we are able to use both of these types of databases in our project. With either option we have access to Google Firebase's Storage which is where we could save files for video or photo uploads. This could be useful to us if we decide to store any video recordings from the camera or if we wanted to save preset audio clips from the user.

One drawback of Firebase is its use of the NoSQL database in both of the database options. This is a drawback only because of non-familiarity of this from the development team. NoSQL in Firebase stores data differently than relational databases as instead of tables the data is stored in JSON trees. New data entered creates a new node in the JSON tree. An example of this JSON structure can be seen in the code snippet below from the Google Firebase documents.



```
{
  "users": {
    "alovelace": {
      "name": "Ada Lovelace",
      "contacts": { "ghopper": true },
    },
    "ghopper": { ... },
    "eclarke": { ... }
  }
}
```

Figure 22: JSON Tree Structure Example Code Snippet

While there might be somewhat of a learning curve with Firebase it offers a free, reliable, and secure option for this project design. Not to mention the benefit of having a cloud database server along with the added features that come with Firebase which all makes it an intriguing choice.

4.3.3.3 MongoDB

Our third database option is MongoDB version 4.2. This version was released on August 13 of 2019 and like Firebase this is also a serverless database option. It is free to use starting at 512 MB of data that can then be increased to 2 GB for $9 a month and 5 GB for $25 a month. While our project design would most likely be okay with the 512 MB, the charge for the higher storage values is something to think about in the case of growth of the product to more users. For security MongoDB offers the authentication options through Facebook, Google, and Apple ID. Its security is trusted by companies like Verizon and Squarespace and it has the security certifications with Kerberos, x.509 PKI, and LDAP to back it up.

MongoDB is also open source with good documentation and guides to get us started. This is important in helping with the learning curve as it uses a NoSQL database. The JSON tree-like structure is used that is similar to Firebase. We are able to use this database management system with any application platform that we choose with guides and documentation for each of them.

The fact that it acts as a serverless database, the possible platform options with community support, and the security makes this option appealing. However the cost of growth in the storage amounts is a drawback.

4.3.3.4 Database Selection

Using Table 14 below we compared the pros and cons of each of the database options that we have researched. For our design, we selected Google Firebase as our database management system. Looking at MySQL, it was beat out by the other two options due to the fact that both Firebase and MongoDB are cloud hosted. Although Firebase and MongoDB both provide good community support and Google and Facebook authentication, Firebase is a better option for the amount of free storage that comes with it. MongoDB only offers 512 MB free while Firebase offers 1 GB. This along with the fact that Firebase comes with some extra features such as Crashlytics, Performance Monitoring and the Test Lab made Firebase the clear choice. With Firebase, we also have the option to use the Realtime Database or the Cloud Firestore. These will be discussed further in the design section.

	SQL	NoSQL	
DBMS	MySQL	Firebase	MongoDB
Pros	• Development team has some familiarity • Relational database • Open source	• Cloud hosted • Open source with good community support • Scalable storage options • Google/Facebook authentication • Easier implementation for the platform options	• Cloud hosted • Google/Facebook authentication • Popular in the community
Cons	• Must be hosted as a server database	• Learning curve of the new management system and NoSQL	• Learning curve of new management system and NoSQL • Only 512 MB free storage

Table 14: Database Pros/Cons Comparison

4.3.4 Wireless Communication

Given that our project design is a smart home system that interacts with a mobile application it is important for the design team to think about wireless communication. For the software side, that means figuring out how the different programs are going to interact and communicate the data that they receive from each of their devices. We need to take into account the possible connections of the mobile app and the database, the mobile app and the Raspberry Pi, the Raspberry Pi and some of the peripherals, and the Raspberry Pi and the database. In this section we will talk about the different methods that will be used to accomplish this.

4.3.4.1 Raspberry Pi OS (Raspbian)

Raspbian is the official Raspberry Pi OS. This OS is the recommended OS for the Raspberry Pi 4, 4B which is our first choice in our design. The Operating System is available for free download and use. Raspbian is based on Debian Linux and Debian is very light weight therefore it makes it very useful for the Raspberry Pi. The Raspbian includes tools for browsing, python programming, and a GUI Desktop. The Raspbian desktop environment is known as “Lightweight X11 Desktop environment” or LXDE. Users required to download Raspbian images to SD card. There are two different types of image; First "Raspbian Stretch with Desktop” which comes with a full Desktop Environment including pre-installed software. Second “Raspbian Stretch lite” which has no tools pre-installed and no GUI installed. There are two ways to configure Wi-Fi connection in Raspbian. First using the Raspbian configuration tool “raspi-config”, WIFI password will be stored in configuration text or setting the connection by command line for more security. Raspbian is our first choice for OS because it is simple and User-friendly. The command also makes it very easy for new users whenever we need to install software. The repository already provides an updated version of new update software. In addition, this OS is open-source so it is free of use which meets our requirement that we need to save costs during our application development.

Software Requirements:

- Raspbian OS image
- EdgeX package for Raspberry P
- IP scanning software
- WebNMS IoT platform

Window : Win 32 disk imager software is required to burn the RPI image onto the SD card.

Linux : Disk software is required to burn the RPI image onto the SD card

MAC : Apple Bake off or Burn it software is required to burn the RPI image onto the SD card.

4.3.4.2 Socket Programming

In networking, a socket is a combination of an IP address and a port number. A more formal definition for a socket is one endpoint of a two-way communication link between two programs running on a network [SOCK]. This works by having one of the devices acting as a server and the other acting as a client. The server program creates the socket with IP and TCP protocols and then binds its IP address and the port number that it is using. An example of this can be seen in the diagram below. The server can then listen and accept client sockets in order to communicate with other devices, typically with utf-8 bytes. On the client side, the program also creates a socket with the IP and TCP protocols but it is looking to connect to the server socket. It does this by using the same IP address and port number that the server socket was created with. At this point the two programs can send data back and forth from each other. It should be noted that this data would then need to be decoded from utf-8 bytes in order to understand a string for example. It should also be noted that in order for this to work when the server and client are not on the same network port forwarding would be required. Port forwarding is when you map any communication request to a specific port on a router. This would make that port public and give access to clients that are not on the same network as the server. While this is good for the purposes of our project design, it also opens up the possibility for malicious people to attack and get into the user's home network and data.

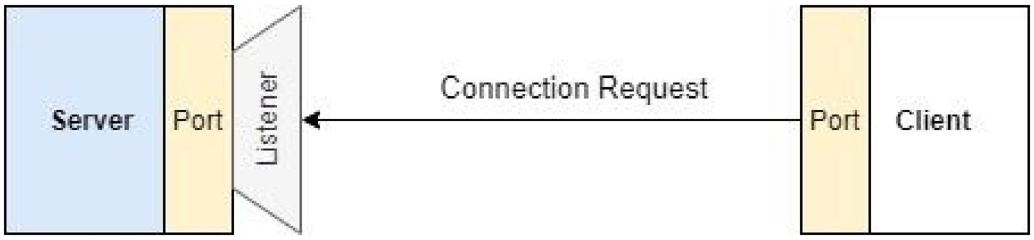


Figure 23: Socket Breakdown Example

This is an option for the connection between our mobile app and the Raspberry Pi. Sockets are a part of most programming languages basic libraries. For our purposes, this is the case for Python, used for the Raspberry Pi and Java, used for the mobile application. The Raspberry Pi would need to be acting as the server while the mobile app would act as the client. Then the app would be able to send data, in our case commands for the door mechanism and the Raspberry Pi could receive and handle that command. In the other situation the Raspberry Pi could send data to the mobile app such as the system detected the pet wants to go outside. The Java program could then handle that data and send a notification to the user. Another option would be to connect our Raspberry Pi to the database directly. We still would need the Socket Programming to send the live video and messages to and from the app and home system. This would take up a lot of space in the Realtime Database and would not be the most efficient way of streaming that data. Even connecting the Pi to the database would possibly provide a more secure option for the data transfers and both options will be considered in the design and implementation phases.

4.3.4.3 Google Services Pugin and Firebase SDK

Since we have selected Google's Firebase to act as our database management system, we need a way to connect it to our application. This would be done by adding the Google Services Plugin to the build.gradle file that will be created when the Android project is created. Our Firebase Realtime Database would then be able to be added to our Android Project as well.

The Google Services Plugin provides us with many methods that allow us to work and interact with our database. We are able to write to the database by first getting an instance of it with .getInstance() and then getting a reference to the node we want to access with .getReference(). From there we can use .setValue() to write the data to the database. As far as reading from the database, we need to implement an event listener on the reference to the data of the node that could be updating in real time. An onDataChange() method would then need to be used to hold the .getValue() call on a DataSnapshot object in order to capture the data that had just been updated. There are other methods and processes that are included in the Firebase SDK that will be explored during the implementation and design phases.

This method of connecting to the database is what we will use to allow our mobile app to communicate with our database. This will allow the mobile app to stay up to date with all of the saved and updated data. As for connecting the database to our Raspberry Pi, Google Firebase currently does not support Python. However, there are multiple third part helper libraries on the Firebase guides that could allow us to make the connection, such as Pyrebase and python-firebase. As mentioned in the socket programming section, these methods of connecting the Raspberry Pi to the Firebase Realtime Database will be further assessed in the design and implementation phases.

4.3.4.4 Dataplicity

Dataplicity is a tool that allows its users to access their Raspberry Pi from anywhere [DATA]. More specifically they can access the Raspberry Pi's command line. This would give people the ability to access and control their Raspberry Pi without the need for a long configuration process. With Dataplicity, Port Forwarding, VPNs and Static IP addresses would no longer be needed. The only thing that would be needed is a connection to the internet. This software is ideal for Internet of Things projects. The way it works is after the configuration, a secure HTTPS connection is made to the Dataplicity IoT router. You can then access the Pi terminal from this router and do any of the actions that are applicable from the terminal. There are three pricing options: free, standard, and pro. Free provides the user with 512 MB per day, HTTPS transmissions and the remote terminal. The standard is only 3 dollars a month and bumps it up to 1GB per day. The final pro plan would give us 2 GB per day at 4 dollars a month. We would most likely be fine with the free version for our design but it is good to know that the other two options are out there.

For the purposes of this project, Dataplicity would be used to stream the live video from the Raspberry Pi to the mobile application. This would provide the software developers an easier way to display the video feed into the mobile app. Instead of going through all of the configuration and port forwarding, a simple link would be provided that can be used by the wifi capabilities of the user's phone to see the video. This would be done using Dataplicity Wormhole which is what makes the connection to the Raspberry Pi and Hawkeye which is a lightweight video streamer. Dataplicity could also serve as a way to transmit the user's message as they are talking through their phone to be played out loud on the system's speaker. This option is much better for our project design as we should be able to implement this in a better way than by trying to steam the video with Socket Programming.

4.3.5 Version Control

Version control is very important with the development of any software project. It is used to keep track of all the updates you are making to the code, who is making them, and provides a way of backing up previous work. Our project will need to implement a version control system in order to easily share files between the development team and to keep our work organized. We are prioritizing familiarity and ease of use to decide on which option to choose in this section.

4.3.5.1 Git

Git is a one of the most popular version control systems that are out there. It's latest version of 2.27.0 which was released on June 1, 2020. It is mainly for organizing source code versions but it also allows project teams and companies to keep track of who is making changes to a file. Git is free and open source and due to its major popularity, has great community support. One major advantage of using Git is the branching and merging system. This allows developers to branch off and work on the same task while trying different ideas. They can then compare which idea worked better, delete the other branch and merge the better implementation to the master branch of the project. This makes software development much more efficient and easier to share files for the integration phase of development. Another advantage is that Git has many plugins and integrations into IDEs and text editors. This makes it easier to commit and push things to Git right in the development environment. We also have the option of using the git commands through the command line. Git is a very powerful tool and is in high consideration for our project.

4.3.5.2 Github

Github is a web-based application that provides version control with all of the same great features of Git. Github has an easier to use web interface rather than doing everything through the command line. It allows for the same file sharing, branching, merging, and tracking of work that Git offers. For the use of this project, the free version is sufficient but there are also other plans that can be upgraded. Github is also easily integrated into most IDEs and text editors to take one less step of committing, pushing and pulling from the repository. There is also a desktop application, Github Desktop that was released in 2015. This does all of the same things as Github and Git, it is just in the form of a desktop app. Github takes the many benefits of Git and adds more accessibility of the web interface. It should also be noted that the development team is familiar with Github and have used it with past projects.

4.3.5.3 Beanstalk

Beanstalk is another version control option that was built to take out the hassle of hosting code and managing deployments. Like most version control systems they want the software developers to focus on writing the software instead of having to put time into the management of the development. Beanstalk uses Git or SVN to act as the main version control component with the web application to manage the changes and provide extra features. These features include notifying the entire team when a push was made, assigning different permissions to each team member, and a code review system that allows the entire team to get involved. There are five price options available starting at $15 a month for 3 GB of storage and going up to $200 a month for 60 GB of storage. While the features of Beanstalk seem beneficial the price would just drive up our cost for our project, while we have two other free options.

4.3.5.4 Version Control Selection

After weighing each of the options the design team selected Github to use for version control. It was clear that Beanstalk would not be used simply because of the price. With the other two options both being sufficient and free, it did not make sense to add to the project cost when we did not need to. Now between Git and Github, we are able to use all of the same version control features with Github as we can with Git. The difference is that with Github we also have access to the web interface which makes commits, pushes and pulls easier to understand. This and the fact that we are familiar with Github is what made it the clear choice.

5.0 Related Standards and Design Constraints

Engineers need to take account for various constraints when designing and meeting design requirements and specifications. Constraints are conditions where the engineers or designers need to be worked with or around in addition to desired requirements and goals for the project. Constraints often will set boundaries for the engineers to work with. For instance, big companies like Google and Apple will not have the same economic constraints as a start-up company. Time constraints limit the creativity and the aspiration in which the project could be. For a senior design project, there are multiple constraints such as time, budget, and social that greatly impact on the project. Understanding the constraints and the scope of the project will provide team members a better feel of the project and generally will perform better.

5.1 Related Standards

Standards are technical documents that specify industry normals or requirements. They state certain details and characteristics that must be met by all products and systems covered under that standard. These standards are set in place to guarantee safety, performance, consistency, and connectivity of their products and systems. The table below (Table 15) shows what standards we have used for our project design.

Technology	Standard	Description
Soldering Specification	IPC J-STD-001	Describes materials, methods, and verification criteria for a high quality solder connections
Printed Circuit Board Specification	IPC-A-610	Describe acceptable methods for hardware installation on PCB assemblies
RoHs- part components	2002/95/EC	Limit the use of lead, mercury, cadmium, chromium (VI), PBBs, and PBDEs in electrical and electronic components. Effective as of July 1, 2006

Technology	Standard	Description
RFID	ISO/IEC 20248	Automatic Identification and Data Capture Techniques – Data Structures – Digital Signature Meta Structure
WiFi	IEEE 802.11	Computer communication in various frequencies, including 2.4 GHz, 5 GHz, 6 GHz, and 60 GHz frequency bands.
Data Transfer	HTTPS	Describes how data being transferred between the browser and website will be encrypted and secured.

Table 15: Related Standards

5.2 Environmental and Economic Constraints

Environmental constraints comprise all external factors that may be interfered with by the design, such as pollution. There are many types of pollution such as air, water, soil, light, and noise. Our design doesn't directly affect any of the types mentioned previously. Since the entire system is designed for inside use, the immediate environment is the inside of the users house. With the limited operation time and usage, there should be little to no pollution created. Indirectly the system can contribute to overall pollution by its power consumption. The system will be designed to minimize the power it consumes during operation and during standby.

Economic constraints deal with all aspects in the financial realm of a project. For our design, the budget we set for all expenses is considered an economic constraint. Since there are no sponsors for senior design this semester, and we are not working with any sponsors, all project funding will come directly from the group members. The amount we are comfortable spending on the project will ultimately constrain or define the project's budget. This constraint limits the spending on certain design aspects of our design. We will have to use the research process well in order to best choose hardware that can perform the desired functions and that are cost efficient. This means we will also have to keep a detailed list of parts to keep us from overspending.

The market price of the end product is also an economic constraint. Our research needs to involve similar product pricing as well as potential customer surveying. We must be able to design and produce our product at a competitive price compared to the research findings. Again this affects the choice of hardware used in the project design. All higher performance hardware parts will directly contribute to the overall cost of our design. This pricing would include manufacturing costs as well, but since we are focused on a single working prototype, manufacturing costs will not apply.

5.3 Social and Political Constraints

Social constraints comprise the social behaviors and characteristics that can impact or determine the sustainability of any project designs or systems within a certain community. Social constraints can directly affect who does and does not purchase your products. One social constraint that applies to our project is the type of people who would wish to purchase our design. This group of consumers is mostly limited to pet owners of indoor and outdoor pets. Another social constraint would be the location of where the consumer lives. Our design has security risks involved including replacing the sliding door lock with a door jamb type lock, and the security risk of having the door open when no one is home. If a potential consumer lives in an area where crime is high or they just don't feel safe with the possible risk factors involved, this would affect their decision to purchase the design or not.

Political constraints are all constraints or restrictions of a project imposed by governing officials, by law, including individual rights. Our project is designed to be privately owned and operated in a private residence. From the research performed, we could not find any laws or regulations on the implementation of our design for private use.

5.4 Ethical and Health & Safety Constraints

Ethical constraints on technology and engineering production can include many issues. These can be safety, privacy, sustainability, and autonomy. Sustainability is a relevant constraint for our product design. We want to be able to produce a reliable product that will last the consumer a long time. We also want the product to be able to prevent itself from stopping during operation. This will involve designing our hardware in a way it will maximize its lifetime and software to have redundancy checks. For privacy, the audio and visual security measures that come with the system have its data sent over WiFi. To prevent privacy violations, possible data encryption techniques can be used, as well as private password protected user accounts for the mobile application. Since safety is a major concern for all designs, we will talk about it separately.

Health and safety constraints encapsulate all constraints dealing with a product's user's physical health or the safety of using the product. We have to design around our users and their pets' health. Nothing interior to our design poses a health threat to either the user or the pet. There are however safety concerns that can arise from our design. There are two main safety constraints that affect our design. The safety of the pet or users by a malfunctioning door. We must design our door to have fail safes in order to prevent the door from closing on a user or their pet. The second is the safety of the user's house. Operating an exterior door to someone's residence from a remote location is a risk factor. Because of this risk, we have built security measures into our design to help mitigate this security risk. Other health risks could arise from an electrical fire. We must design our product using the proper standards for all aspects of the system and make sure that possible faults that could cause fires are minimized.

The safety of our users are also under risk from having their personally identifiable information stolen from the use of our mobile application or from their home WiFi network. Their data will be stored in a Google Firebase Realtime Database. This data is secured by having customized rules that are written by the software development team. These rules would prevent users from accessing other users data and other aspects that would not be permitted from the users standpoint. Another security aspect to the Realtime Database is authentication, which is the process of recognizing the user's identity. When a user first signs up, authentication would take place to ensure that they are who they say they are and then continuously as they log into the application. This would work together with the rules to ensure that the data being passed and stored is in the correct place for the correct authorized users.

Authentication does not only apply to the people using the mobile app but we would need it for the Raspberry Pi as well. Since the Raspberry Pi would be reading and writing to the Realtime Database we must ensure that it is only accessing the data it needs just like we would for a person. Instead of a username and password that a person would have, the Raspberry Pi would have some type of key that identifies it and makes it distinguishable to the database and the mobile app.

5.5 Manufacturability and Sustainability Constraints

Manufacturability constraints deal with the concerns of a product's reliability and efficiency of being produced, all at a cost that is sufficient to the manufacturer. Our prototype will be mostly made up of pre-existing parts; the Raspberry Pi, webcam, linear actuator, and sensor modules are all existing products we are reusing to construct our prototype. We do have to design and implement a printed circuit board for our power solutions, and construct a housing for everything to fit into. For the manufacturing of our design, the Raspberry Pi and other sensors can be substituted by a custom computer board, designed for the sensors and peripherals only needed for the design. The power can also be incorporated into a single circuit board or the same computer board. This would cost more upfront to design the boards, but then mass producing them would be more cost efficient, and also reduce the product size. This reasoning also applies to making a single peripheral that handles audio input and output and video. The product design also could be adjusted to increase efficiency by making custom motion and identification sensors. If we could get the sensors to be a part of the main system housing, the time and ease of installation of our system would decrease dramatically to only placing the housing in the sliding door track and attaching the camera with a few screws.

Constraints for product sustainability cover the environmental effects over a product's life as well as any lingering effects after product use has stopped. Since our product is meant for indoor use only the sustainability of our design is quite high. Ors product will not be affected by or affect its immediate surroundings. Our design is also intended to avoid any environmental effects outside of the power usage required to operate the system. We have designed our product to minimize power usage during idle operation as well as reduce the active operating power usage draw.

6.0 Hardware Design Details

This section will lay out all information related to each hardware piece or portion of our project. An overview of our hardware design can be seen below in the hardware overview flowchart (Figure 24).

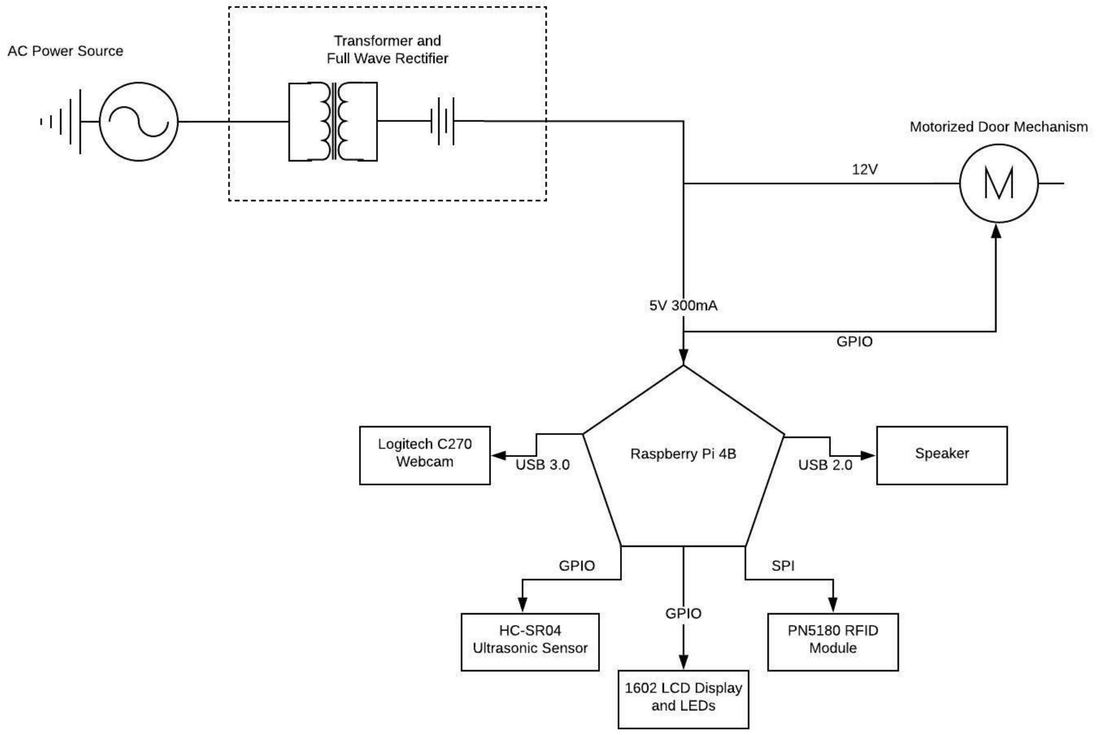


Figure 24: Hardware Overview Flowchart

The information given will include the decisions we made for each part of our hardware research and how we intend to implement the hardware in our design. Each decision will include a reasoning for why the hardware was chosen and each implementation will be described in detail. The project design includes a housing that sits inside of a sliding glass door, in the track where the movable door would slide. The housing will have an extendable and retractable arm that will connect to the moveable glass door. It will also hold the single board computer and PCBs. The outside of the housing will have an LCD screen as well as LEDs. The housing itself will assist in acting as a door jam or lock for the sliding door, when the housing arm is fully extended. Externally connected from the housing will be a webcam and speaker for audio and visual security measures that will be connected to the housing via USB cabling. The webcam is planned to be installed above the housing angled towards the sliding door opening. There will also be external sensors, RFID and an ultrasonic sensor, that will extend from the housing using copper wires. These sensors are planned to be attached to the sliding door opening. The last piece of hardware is a wearable, the RFID tag, which will be attached to the pets collar.

6.1 Single Board Computers

Our design incorporated a single board computer to be the backbone of the systems operations. The single board computer had to be strong enough to process audio and visual data as well as control all external sensors and peripherals. Although the Raspberry Pi 4B doesn't have the fastest processing speed among the single board computers researched, it has more positive ways to be incorporated into our design than the others. We chose the Raspberry Pi 4B for two main reasons. The first is that the Pi offers multiple options for internal RAM. Where the other single board computers are limited to a max of 2GB RAM, the Raspberry Pi 4B has 1, 2, 4, and 8 GB options. Our research led us to believe that 4GB RAM is what will be needed to achieve our design goals. The second reason we chose the Raspberry Pi over the others is the vast DIY and hobbyist community that supports it. The community offers guides and tutorials that will be helpful to implement our design, and save us time by eliminating trial and error. The Raspberry Pi 4 also has more connectivity than the others, which offers us more flexibility in our choice of peripherals.

The Pi is intended to be used primarily for collecting our data and input from peripherals and communicating with a mobile application over WiFi. We plan on powering the computer internally, after converting a normal home electrical outlet AC power source. Of the four USB ports the Raspberry Pi has we plan on using two of them. The first will utilize USB 3.0 and will connect to a webcam with a built in microphone. The second will be a speaker using USB 2.0. All other sensors and peripherals will be connected through the Raspberry Pi's GPIO pins. The RFID alone requires specific communication protocol (SPI) pins. We need to allocate six total pins for the RFID sensor, seven for the LCD display, two for the motion sensor, and six for LEDs. We also need to send signals to our linear actuator, so we have set aside one pin for that. The design for how the Pi will connect to each peripheral or sensor is shown in Table 16 below.

Connection	Description	PIN	PIN	Description	Connection
🟥	3V3 power	1	2	5V Power	🟥
	GPIO 2 - SDA	3	4	5V Power	🟥
	GPIO 3 - SCL	5	6	Ground	▨
LCD	GPIO 4 - GPCLK0	7	8	GPIO 14 - TXD	LCD
▨	Ground	9	10	GPIO 15 - RXD	LCD
LCD	GPIO 17	11	12	GPIO 18 - PCM_CLK	LCD
LCD	GPIO 27	13	14	Ground	▨
LCD	GPIO 22	15	16	GPIO 23	Ultrasonic
🟥	3V3 power	17	18	GPIO 24	Ultrasonic
RFID	GPIO 10 - MOSI	19	20	Ground	▨
RFID	GPIO 9 - MISO	21	22	GPIO 25	RFID

Connection	Description	PIN	PIN	Description	Connection
RFID	GPIO 11 SCLK	23	24	GPIO 8 - CE0	RFID
	Ground	25	26	GPIO 7 - CE1	RFID
N/A	GPIO 0 - ID_SD	27	28	GPIO 1 - ID_SC	N/A
Blue LED	GPIO 5	29	30	Ground	
Red Led	GPIO 6	31	32	GPIO 12 - PWM0	Motor
Green LED 1	GPIO 13 - PWM1	33	34	Ground	
Yellow LED	GPIO 19 - PCM_FS	35	36	GPIO 16	RGB LED
Green LED 2	GPIO 26	37	38	GPIO 20 - PCM_DIN	
	Ground	39	40	GPIO 21 - PCM_DOUT	

Table 16: Raspberry Pi GPIO Connections

6.2 Power

A power supply must fulfill the requirement to power all electrical components on the PCB and other parts such as the linear actuator. The research in previous sections has led us to using AC wall power 120 V at 60 Hz due to its low maintenance and reliability. In this project, all components have different voltage and current rating in order to function and be operational. Below in Table 17, is a list of all components needed for this project and calls out their voltage and current rating.

Component Name	Voltage Rating	Current Rating
Linear Actuator (Motor)	12 V	5 A
Raspberry Pi 4	5 V	300 mA
LCD Display	3.3V - 5 V	< 200 mA
LED	2V	20 mA
Ultrasonic Sensor	5V	< 15 mA
Camera/Speaker	4.5V - 5 V	300 mA
RFID Sensor	5 V	200 mA

Table 17: Power Requirement

The 120V 60 Hz power outlet requires dedicated resources to be a viable option for power. This method will require a step down transformer to obtain the low voltage required for this project. The 120V wall power is an AC signal which requires rectification, conversion, and regulation in order to be compatible and supply power to all of the components of this project. A step down transformer is needed to create electrical isolation from the power grid and reduce the voltage level from 120 V from the wall outlet to circuit.

Diodes are needed in a full wave bridge rectifier configuration to provide a consistent positive voltage along with capacitors charging and discharging to smoothen the AC signal from the wall outlet. The full wave rectifier circuit requires four standard diodes in an unique configuration to rectify the AC signal during each positive and negative cycle to uniform the direction of the current. A downside to this full wave rectifier circuit is a loss of voltage and power from the four diodes.

A capacitor in parallel to the load will smoothen out the rippling resultant current from the full wave rectifier by charging when the voltage increases beyond a certain threshold and discharging when the voltage drops beyond a certain threshold. Energy from the capacitor will be released and charge the load due to the collapse of the electric field around the capacitor. The capacitor will be used to stabilize the voltage across the load and will oppose any instantaneous change to the voltage. To protect the diodes for unexpected surge current, we may use a current-limiting resistor or an inductor in series with the wall wave bridge rectifier to reduce the current to the diodes.

There's also the option to purchase high current capable diodes to account for this issue, however these diodes intend to be expensive. A downside to placing a current-limiting resistor in series to the rectifier is the reduction of voltage to the load as this resistor will be creating a voltage divider circuit. An inductor in series with the full-wave rectifier will deal with the surge and drop in current through the expansion or collapse of the magnetic field from the inductor respectively. An additional benefit of the inductor is to maintain a constant current flow and will oppose any change in current. The inductor will offer no impedance if a constant current is applied from the full wave rectifier, thus no additional loss in voltage in relation to a resistor in series. The LC filter will produce a stable output..

The addition of a zener diode in parallel with the capacitor and load is beneficial as it will eliminate all ripple voltage if any from the LC circuit. One can set the voltage across the load by selecting the desired voltage rated zener diode. If the voltage between the zener cathode and zener anode is greater than the voltage rating, then the zener diode will be active and will flow current through the diode in reverse bias. Figure 25 below depicts a schematic of bridge full-wave rectifier with LC and zener diode circuit that will convert 120 V AC power to a desired voltage level and DC current.

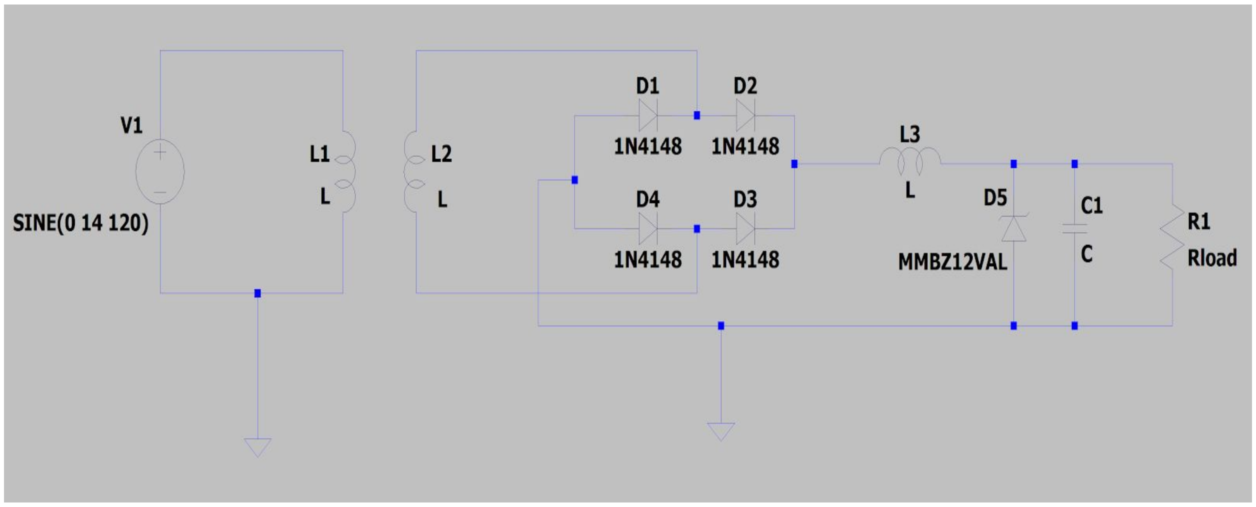


Figure 25: Full Wave Rectifier Circuit

In comparison to batteries, the power outlet provides practically constant supply of power as long as the system is connected to the outlet. Since our product housing will be stationary, the best option is to go with the power grid as it's more dependable and eliminate the cost of replacing and disposing batteries. The power requirements for this project, shown in Table 6.2.1, will dictate the part selection and values.

The characteristics of a step-down transformer that must be taken into consideration are the turn ratio of the primary and secondary coil, efficiency and the size or the footprint. These considerations will limit the choices for the step-down transformer. With the lack of experience of soldering among the team, it's beneficial to obtain a larger size package or footprint to make it easier to solder.

The characteristics of standard diodes that need to be considered when selection is the forward voltage, peak forward surge current, and its suitable for full-wave rectifier. The full-wave rectifier requires four standard silicon diodes. These standard diodes are made of silicon and generally have 0.7 V forward voltage. This configuration for these four diodes creates an efficient AC filter to rectify power signal from the grid.

6.3 PCB

The printed circuit board will be designed using Autodesk Eagle software that allows for a development of a schematic and a smooth transition into a board layout. One obstacle from Eagle will be downloading or importing all the necessary libraries needed. A solution to this obstacle is to solidify the build of materials (BOM) for this PCB and as a team search for these libraries to reduce the time spent. Using the LTspice, we will simulate the behavior of the schematic to validate our design. Following the schematic design, we will move forward with a board layout that consists of determining the routing of all of the connections. The Autodesk Eagle software greatly eases the transition from schematic capture to board layout.

The process to create and design a PCB using the Autodesk Eagle software requires a significant amount of time. This process is critical as the schematic will be converting AC power to regulated DC power to supply power to the linear actuator, microprocessor, and the Raspberry Pi. As the main function of the PCB is to regulate and supply power to the linear actuator motor and Raspberry Pi, it's important to add reference designators and possible test points on the PCB for testability of the board. An important consideration is creating schematic is to adhere to good engineering practices like making sure all grounds are pointing down and spread out the wire and signals to make it easier user friendly.

Another tool introduced in junior design is Webench from Texas Instrument. Webench is an online part selector tool that narrows down the selection and provides the footprint file of the component which can be easily integrated to Eagle. In addition, we will be searching for surface mount components for all parts for this PCB design.

Once the schematic design is finalized, we move onto the board layout. Eagle will ensure all connections made in the schematic will be accounted for. The next challenge is determining the trace width and part placement. In addition to determining the routing, it's important to take consideration of the components emitting electric and magnetic fields and its effect on other components. Trace widths will be determined by the amount of current and voltage supplied through the connection. In this project, maximum voltage and current rating is 12 V and 5 A.

There are different options for fabrication companies vary by price and industry reputation. This fabrication process can range from a few days to a few weeks. To ensure that we don't waste resources and time, we will need to lockdown the design in advance in hopes of a few errors. With peer review amongst the team, we should be able to reduce the amount of errors with the design and be more time efficient.

The final step of this process is to solder our surface mount component onto the designed printed circuit board. Through our testing procedure and verification, we will be able to determine if the fabrication process went success and true to our design. With reference designators and large package footprint, the placement and soldering process will be easier. The lack of experience with using solder components will incentivize us to fabricate multiple copies of the board and additional components for the team's learning curve.

6.4 Motor

The mechanism design of this project is the motor and lock system of this project. After extensive research, the team decided on a linear actuator as the mechanism lever to open and close the sliding door. A linear actuator is a motor that generates thrust and motion in a straight line, in construct to many traditional motors that creates circular movement. A linear actuator motor fits the project needs with some linear movement with a certain amount of force to open and close the sliding door.The classic rod linear actuator fits the bill because of its affordability, and its simplicity. The select linear actuator shown below in Figure 26 provides around 150 lbs of force and rod extension of 18 inches at 0.5 inches per sec.

Figure 26 Classic Rod Linear Actuator

6.5 Sensors

Like many autonomous systems, our design needs a reliable way to know when to operate. Our product is designed to activate either remotely by mobile application or by the pet at home. In order for the pet to activate the system we need a motion sensor. Our research led to two different motion sensors, the ultrasonic and the PIR. Both of these sensors are previously owned by the group, so cost was not a factor in making our final decision. Our design has the motion sensor attached to the sliding glass door, where it is exposed to outside heat and light. Due to the nature that the PIR sensor can be negatively affected by light and temperature, we chose to use the ultrasonic sensor to detect the pets movement near the door.

This motion sensor is planned on being external to the system housing, with a wired connection to the Raspberry Pi for communication and power. The HC-SR04 uses 4 lines; VCC, Trigger, Echo, and Ground. VCC for the sensor is 5 volts and will be connected to the Raspberry Pi's 5 volt out pin (Pin 2). The Trigger receives a signal that tells the sensor to start detecting. The trigger will be connected to the Raspberry Pi's GPIO23 pin (Pin 16). The Echo sends a return signal to the source when an entity has been detected. The Echo will be connected to the GPIO24 pin on the Raspberry Pi (Pin 18). The ground will connect to ground on the Pi (Pin 14). Below (Figure 27) illustrates the HC-SR04 motion sensor wiring schematic.

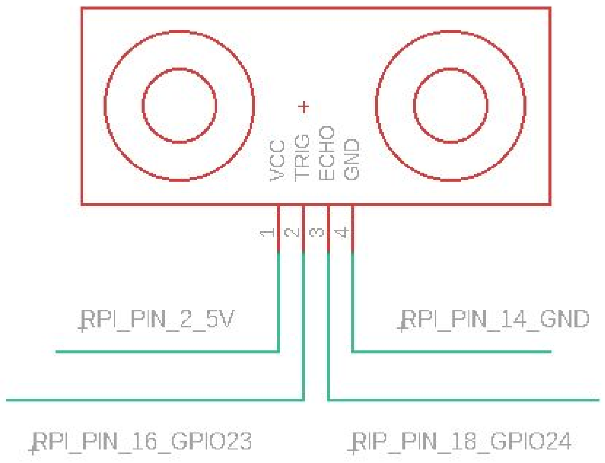


Figure 27: HC-SR04 Motion Sensor Schematic

To run alongside the motion sensor, we have designed our system to use an identification sensor to verify if any detected entity should activate the door opening procedure. We researched multiple types of RFID sensors to satisfy this design requirement. Cost wasn't a factor in our choice for this sensor either since all RFID sensors researched were in the same price range. The main factors of our decision were based on the protocols supported by the sensor, the sensing range, and the communication protocol it used to relay data. The RFID protocols supported by the individual sensors varied greatly, from two protocols, to a handful, to one sensor supporting most of them. The PN5180 was the sensor that supported most RFID protocols actively used. The PN5180 also had the largest sensing range of up to 15cm, which was more than double the range the other sensors offered. All of the sensors communicated via SPI, but the RC522 and PN532 also had UART communication options. Ultimately, the vast RFID protocol support and the large sensing range of the PN5180 made it the obvious choice for our design.

The RFID sensor needs to be in a position close enough to where the pet would be standing in order to detect and verify the ID, so this sensor will also be externally wired and attached to the sliding glass door. The PN5180 has 13 pins, which are all explained in the Table 18 below.

PN5180 Pin	Description	Raspberry Pi Connection
+5V	5 volt DC supply	Pin 2
+3.3V	3.3 volt DC supply	Pin 17
RST	Reset	Pin 22
NSS	Non Slave Select	Pin 24
MOSI	Master Out Slave In	Pin 19

PN5180 Pin	Description	Raspberry Pi Connection
MISO	Master In Slave Out	Pin 21
SCK	System Clock	Pin 23
BUSY	Busy State	Pin 26
GND	Ground	Pin 6
GPIO, IRQ, AUX, REQ	Not used	-

Table 18: PN5180 to Raspberry Pi Connections

6.6 Security Peripherals

Since our product will be in control of an access point to the users home, we have made specific requirements to ensure their home security. We have planned our design to incorporate audio and visual components. From our research we have found multiple different hardware combinations to satisfy these security needs.

We have chosen the Logitech webcam over other hardware considerations because it provides the system with a visual input that has a high enough resolution without requiring too much bandwidth as well as coming with a built in microphone that will satisfy audio input. The last security requirement, audio output, will be fulfilled by the USB Mini Speaker. We chose this speaker over the 3.5mm audio jack speaker because it has an internal battery that is charged via the USB connection, whereas the 3.5mm audio jack speaker needs to be charged externally.

The web camera is the last peripheral planned to be located externally from the main housing. The attached five foot USB cable gives the user some flexibility on where they can install the camera without purchasing an USB cable extension. The recommended installation point is five feet above the main housing, inside the sliding door jam opposite the moveable sliding door. The webcam will be directly connected to the Raspberry Pi through the top USB 3.0 connector.

The USB speaker is planned to be a part of the main housing. The speaker comes with a long enough USB cable to reach outside our main housing, so we can attach it to any available housing space. We plan to have it attached or slightly protrude from the main housing's normal (upwards) facing. The speaker will connect to the Raspberry Pi through either the bottom USB 3.0 connector or the top USB 2.0 connector. All security peripheral connections are shown in Table 19 below.

Peripheral	Description	Raspberry Pi Connection
Logitech Webcam	USB 3.0	USB 3.0 (Top)
Mini USB Speaker	USB 2.0	USB 2.0 (Top)

Table 19: Security Peripheral Connections

6.7 LEDs

Our product design is mostly an autonomous system, which means every operation is done by the machine itself. Outside of the mobile application controls, the user has no operational control of our system. This means that there is no indication as to what is happening in the system at any given time. In order to help the user know what the system is doing or not doing, we plan to incorporate LEDs to our main housing that will help visually display the system operations and modes. The group already owns a number of single color LEDs as well as a RGB LED and an LED matrix. The LEDs will be built into the systems main housing on the interior facing. The chart below, Table 20, displays each LED function and representation.

LED	Operation	Connection
Blue	System On	Pin 29
Red	System Mode	Pin 31
Green	Entity Detected in Range	Pin 33
Yellow	Scanning ID	Pin 35
Green	ID Accepted	Pin 37
RGB	Door Opening/Closing	Pin 36

Table 20: LED to Raspberry Pi Connections

6.8 LCD

To go beyond what the LEDs provide the system and its users, we have integrated an LCD screen as a second visual indicator of system operations. Although the serLCD and OLED have more functionality built into the hardware, we have chosen the LCD 1602. We made our decision based on the fact that the group already owns one and that we all have previous experience in using this model. The LCD 1602 offers sixteen characters by two lines, which is enough to display all intended messages properly.

The LCD screen itself will also be attached to the main system housing, on the interior facing. The LCD 1602 will be directly connected to the Raspberry Pi by GPIO pins and without using any communication protocols. It requires five volts for operating power, a ground, and seven data lines. The LCD to Raspberry Pi pin connections are shown below in Table 21.

LCD Pin	Description	Raspberry Pi Pin
VSS	Ground	Pin 14
VDD	5V DC Supply	Pin 4
VO	Potentiometer	Pin 7
RS	Register Select	Pin 8
RW	Read / Write	-
E	Clock Enabled	Pin 10
D0 - D3	Bits	-
D4	Bit 4	Pin 12
D5	Bit 5	Pin 11
D6	Bit 6	Pin 13
D7	Bit 7	Pin 15
A	LED Anode +	Pin 4
K	LED Cathode -	Pin 14

Table 21: LCD 1602 to Raspberry Pi Connections

7.0 Software Design Details

In this section we will discuss software design and development to implement our project. Software plays an important part that's basically a dashboard for managing and controlling devices connected to it. This dashboard will collect information from sensors and send signal to users to operate activities. Our software design is divided into six main parts which include software prototype, mobile application, use case diagram, user interface, class diagram, database design, and wireless communication. Figure 28 below shows our general flowchart for the project.

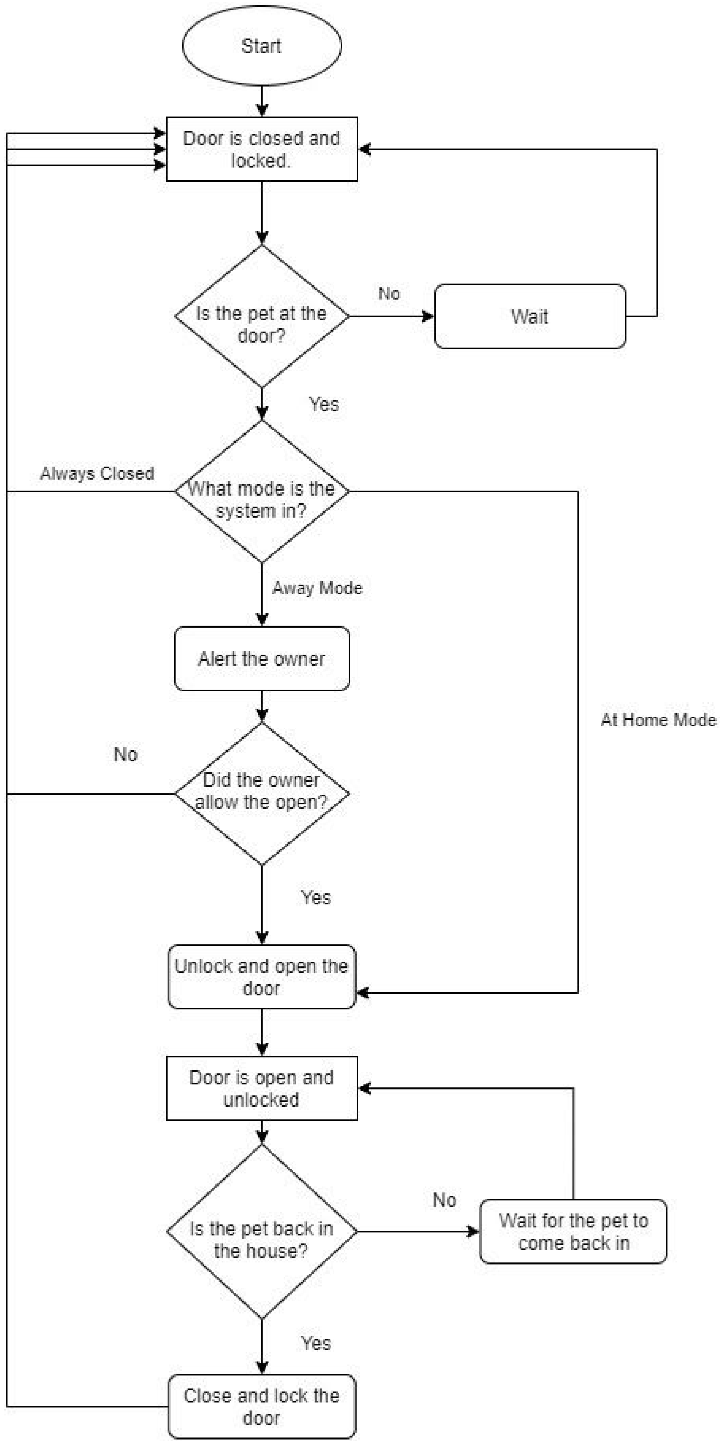

Start
Door is closed and locked.
Is the pet at the door?
No
Wait
Yes
Always Closed
What mode is the system in?
Away Mode
At Home Mode
Alert the owner
No
Did the owner allow the open?
Yes
Unlock and open the door
Door is open and unlocked
No
Is the pet back in the house?
Wait for the pet to come back in
Yes
Close and lock the door

Figure 28 : General Project Flowchart

The flowchart begins with the door being closed and locked and then looks for the pet to be at the door. Depending on the mode that the user has the system in, it will then take the appropriate action. This would include alerting and asking the owner if it's in Away Mode, opening the door if it's in At Home Mode, or simply staying closed if it's in Always Closed Mode. This process then repeats as the owner can change the setting in the mobile app. Understanding this diagram is important when designing the mobile app that will allow the user to navigate through the flow of our system.

7.1 Android Mobile Application

The mobile app is used as our control panel which users will interact to send commands to the system. We decided to build Native Android development application to run on an android device. Native Android apps can take full advantage of software and the OS's features, especially direct access to the hardware of the device such as camera and microphone which are the main components used in operating the system for this project. Moreover, Native apps are faster in execution since we prefer to use real-time databases which ultimately result in better user experience. We choose Android studio as the platform to develop the mobile app. Java is the language that commonly uses and supports Android Studio. Android app contains multiple app components including activities, fragments, service, content providers, and broadcast receiver. In addition, android studio has built-in support for integrating Google Cloud service and Firebase which is the database that we chose to implement in our software development.

We start our mobile application design by selecting the right architecture. We want mainly use of the internet, Mobile app, and Android SmartPhone to be interconnected. If the smartPhone has no internet connection the users will not be able to use the application or receive and update any fetch data. This limitation for offline mobile applications can be further implemented in the future but for the scope of our design, our mobile application will rely on the internet connection. Moreover, our main functionality relies on Push notifications as well as real- time updates and ability to access built in features on the smartphone. In Figure 29 below is Android Mobile App Architecture to support our Software design.

The architecture in Figure 29 shows how each layer interacts with one another. We tend to logic more in view because they are responsible for handling notifications from the model. Activities and fragments only depend on the view model. Activity and Fragment are classes that represent contracts between Smartphones and the app. These classes should only contain logic that handles UI. Models are the components that handle data from the app and we want our design to be based on drive UI from persistence model so they're unaffected by the app's lifecycle. ViewModel provides data for specific UI which can call other components to load data and can forward user requests to modify any data. In addition we want our UI to be able to access LiveData state from the camera. This is to ensure that LiveData can only update app components that are active states. LiveData classes will be included in the viewmodel.

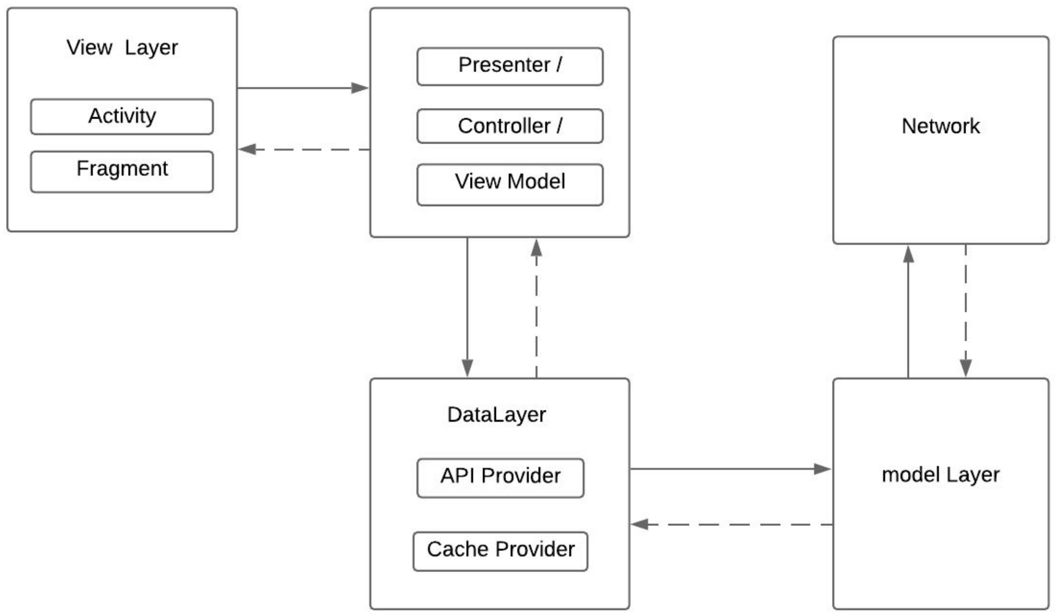


Figure 29: Android Mobile App Design Architecture.

7.1.1 Android Studio

After we conducted our research and concluded that we will implement an Android Mobile app, we chose the Android Studio platform as our IDE. The best features of Android Studio use in our software system helps to give a real-time experience with IOT based project development. We want to build–high quality application and this is also the reason we use Firebase connectivity to support scalability. Android studio has a feature called "New Module Wizard", this is advanced technology that delivers instant results without taking time to rebuild the apk and installation make. This is very important in our app testing because we want our app to be able to change the state if there is trigger event or data fetch, moreover with camera, audio, and device control that may change state instantly require the user to see the change in the app immediately Another factor that takes in consideration is that Android has a fast emulator. This is a great feature of the Emulator which is exactly like an android smartphone. It's given real-time experience to the android applications and helps the development life cycle be shorter and efficient.

7.1.2 Java Language

Android studio supports Java language to use in Android development. Based on our development team coding experience we decide to use Java as our preferred language in writing code in our app design. Java is the most popular choice for developing Android mobile applications and it supports Android Studio. To create the app that connects and interacts with peripheral and support use experience in our application we need to list available peripherals to connect hardware directly to the app. Java has many frameworks and classes for features in our project such as I/O operations that require different peripherals in the function, threading and internet network that developers can leverage these qualities features of Java to develop IOT app development. In addition, an Android application uses Java Virtual Machine (JVM) to generate and run Android executable (APK). And JVM reads Java byte so no need to recompile it for every hardware. To write the data to create the app we will need MainActively class to list all available Port. We can use a system service called PeripheralManager to manage Peripherals connection. Beside user interface design that we can use to benefit from Android studio drag and drop features we also can write our own Java Code to cooperate with the design. To handle the Button events state transition will generate call back events with setEdgeTriggerType () method and register GpioCallback to receive trigger events.

7.1.3 Mobile App Flowchart

In any software development it is common to use a flowchart to represent all of the different possibilities that could arise during the use of that application. Figure 30 below is the flowchart for the Android app that we are going to develop. Unlike the general project flowchart in the Software Design introductory section, this flowchart covers only the mobile app and provides more detail with each action. It should be noted on the legend to the left of the diagram showing what each symbol represents.

The chart starts off how the user would and first determines if the user has already made an account. If they did not they go through the signup process and the data is added to the database. If they do then they can login or have already been logged in and come to another option in the chart. The 'Notification Received?' option represents the user interacting with the app by just opening it with the 'No' option or getting the alert that their pet is at the door with the 'Yes' option. With the user opening the app they first see the control screen and have the option to navigate to any of the other two screens. All of three require initial reads from the database to display the appropriate data. The Profile screen also gives the users the ability to edit their information which is then updated in the database. From the control screen the user can interact with the speaker and camera peripherals, change their current mode, open or close the door directly, or simply just view the data. The way each of these options are handled are shown in the diagram as an interaction with the peripherals requires connecting directly with the Raspberry Pi but changing a mode would just update the database.

The other path for interacting with the mobile app would be when the notification is received. When opened, the user would be shown the live video from the camera and be asked if they want to allow the door to open, letting their pet outside. If they do not allow it then the flow just goes back to the 'Notification Received?' option. If they do allow it then the database is updated to reflect the decision and the door will open.

This diagram will help the software development team when it comes to programming all of the necessary functionalities of the app. Understanding this and the other aspects of our software design mentioned throughout this section are crucial to developing the correct and functional mobile app.

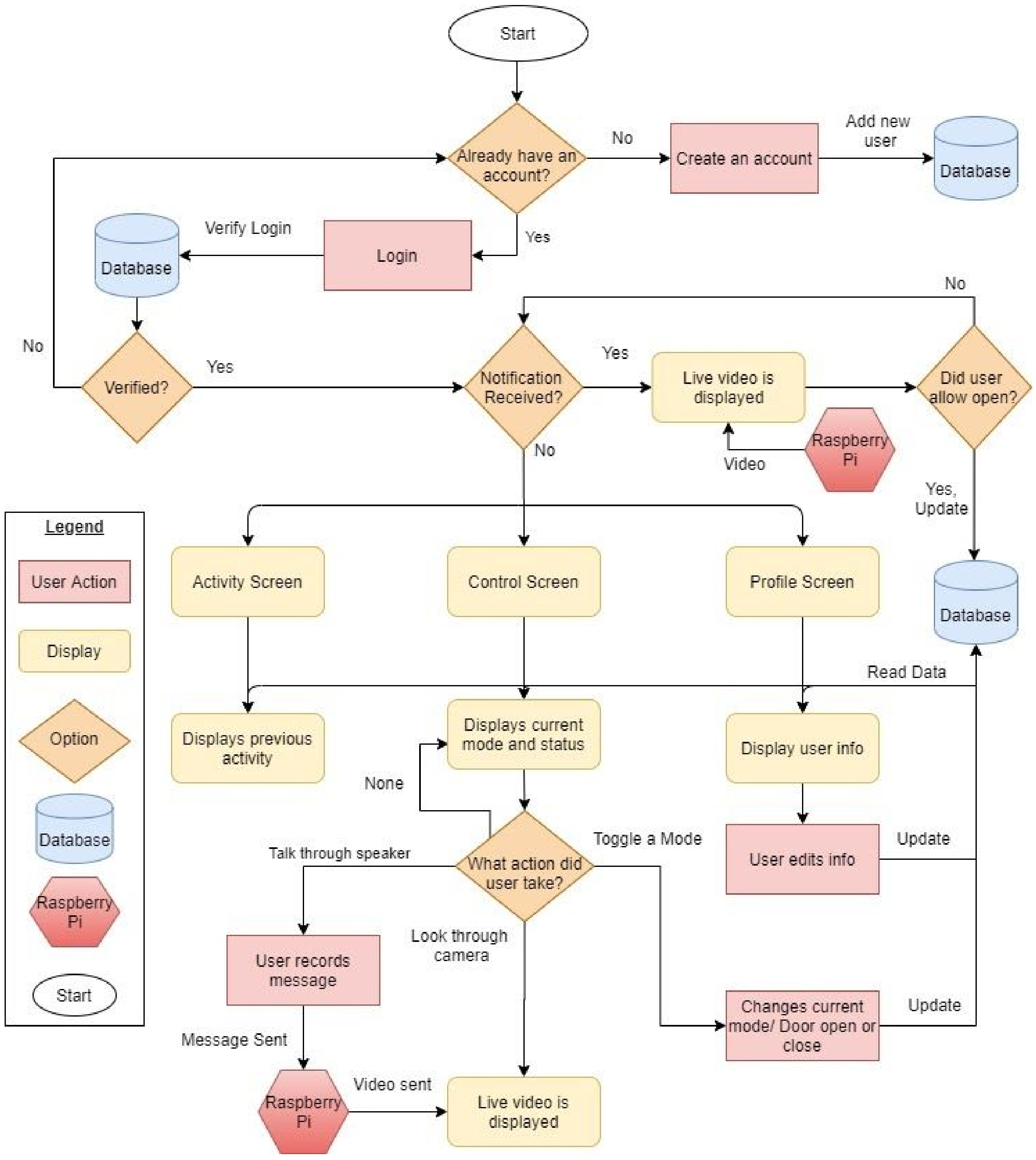


Figure 30 : Mobile App Flowchart

7.2 Use Case Diagram

In our software design, we want to collect all steps done by users so building a use case will help capture the interactions between users and the system which is associated with it. Use case diagrams are also called behavioral diagrams that describe the business system and record how the user interface responds to the system. Figure 31 is the use case diagram for Software Interface of Android Application.It demonstrates the functionality of the system and describes the behavior sequence of steps in which each step specifies actions. Our use case diagram is designed based on section 2 for software Goal.

When users first launch the applications, users have the ability to create an account to login to use the application. We design to have an email address as username and password as well as name of users and phone number which we can use further for step of verification as needed. If users already have an account created, they can just directly start using the application. Inside the application, users now can store their pet information and address location. Users have the ability to manage device setting to set up the Home Mode to Disarmed, Home, and away mode. The main function is that users can receive notification which trigger further actions to operate the device which are access to camera view, using audio and speaker phone, operate open or close the door. Finally, users should have the ability to view the status of doors for security enhancement.

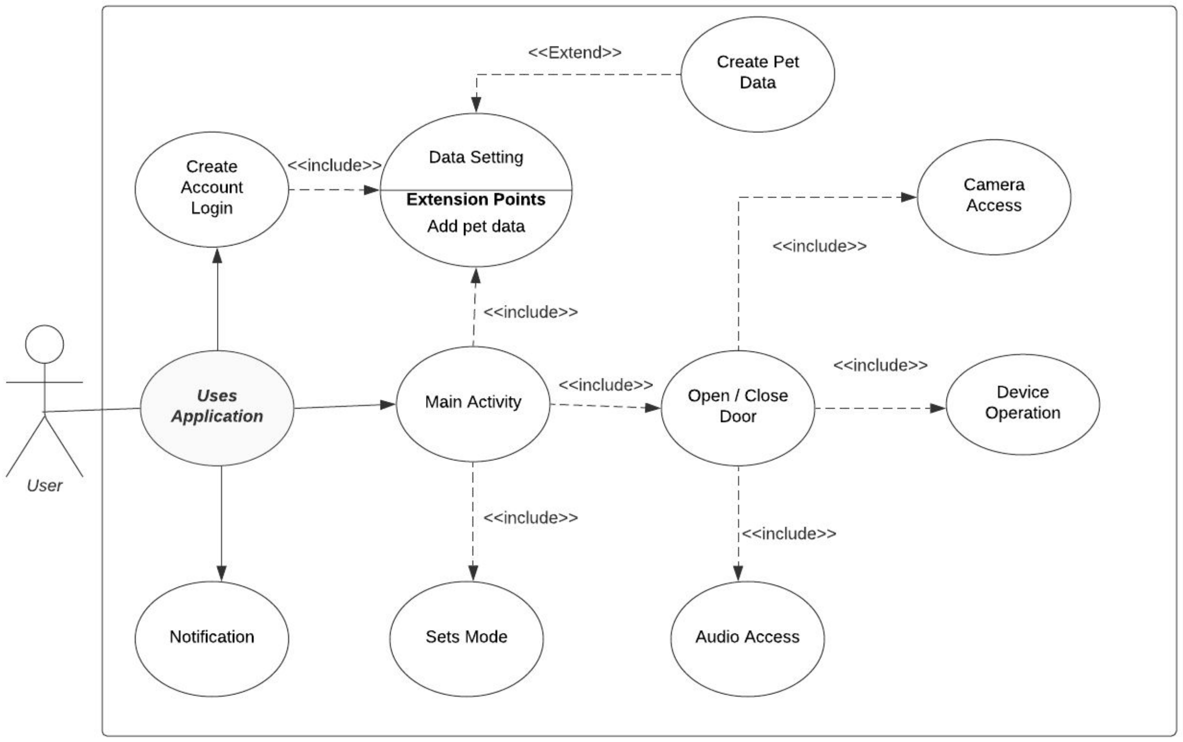


Figure 31: Basic Use Case for Software

7.3 User Interface Design (UI) - WireFrame

As we mentioned, we want mobile apps to be a dashboard that is the main control of the device. The user interface therefore should support user experience and develop with ease of use and optimization. Figure 32 shows user interface design for our mobile app.

Our mobile app user interface is divided into four different views. First screen is the User Login page. This page consists of a username and password input box. New users screen is applicable to create their account by input new username, email, and password. Second screen is the user profile screen, this is a data screen that collects necessary information from account owner and pet information including petID for. The page has an option for users to edit/delete information and logout from the app. Third, Control page which is a main dashboard for the app. This user interface includes important functions that allow users to view the camera, access audio and speaker phone, control door, set mode, view door status and receive notification . Last screen is the activity log screen which extends information to set mode and status mode. The screen indicates specific date and time on the screen to track activities from the door and pet.

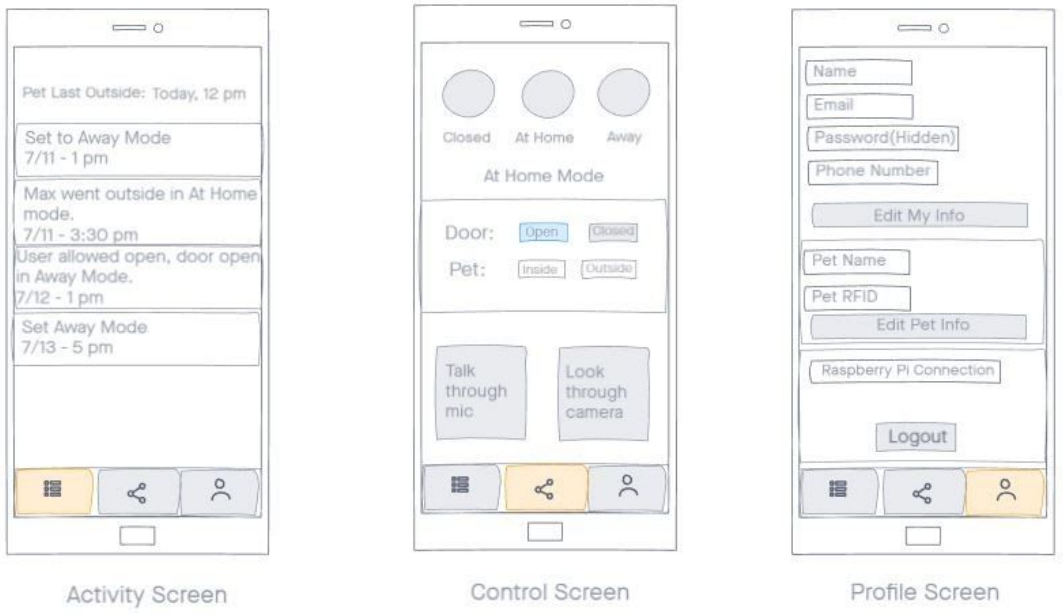


Figure 32: User Interface Design for Mobile Application

7.4 Class Diagram

We demonstrate a high level of software design by creating Unified Modeling Language Class diagrams to better determine modeling and documenting the flow of the software. Figure 33 shows class diagrams illustrate work flows of our design. The workflow started by

Login/Registration class which requires further validation before moving to the next steps. For security purposes we implement the user's access by validating their login information. That information is then stored in the database to use for future authentication every time users login. Next, the User class holds various methods that are used to implement the software functions. Users class will retrieve information from the database to display, verify, and operate necessary actions from users. AddPetData is the class that indicates pet information and pet status if the pet is inside or outside the house. Last, Activity Log class will get current date and time and output necessary messages related to mode and status of the applications.

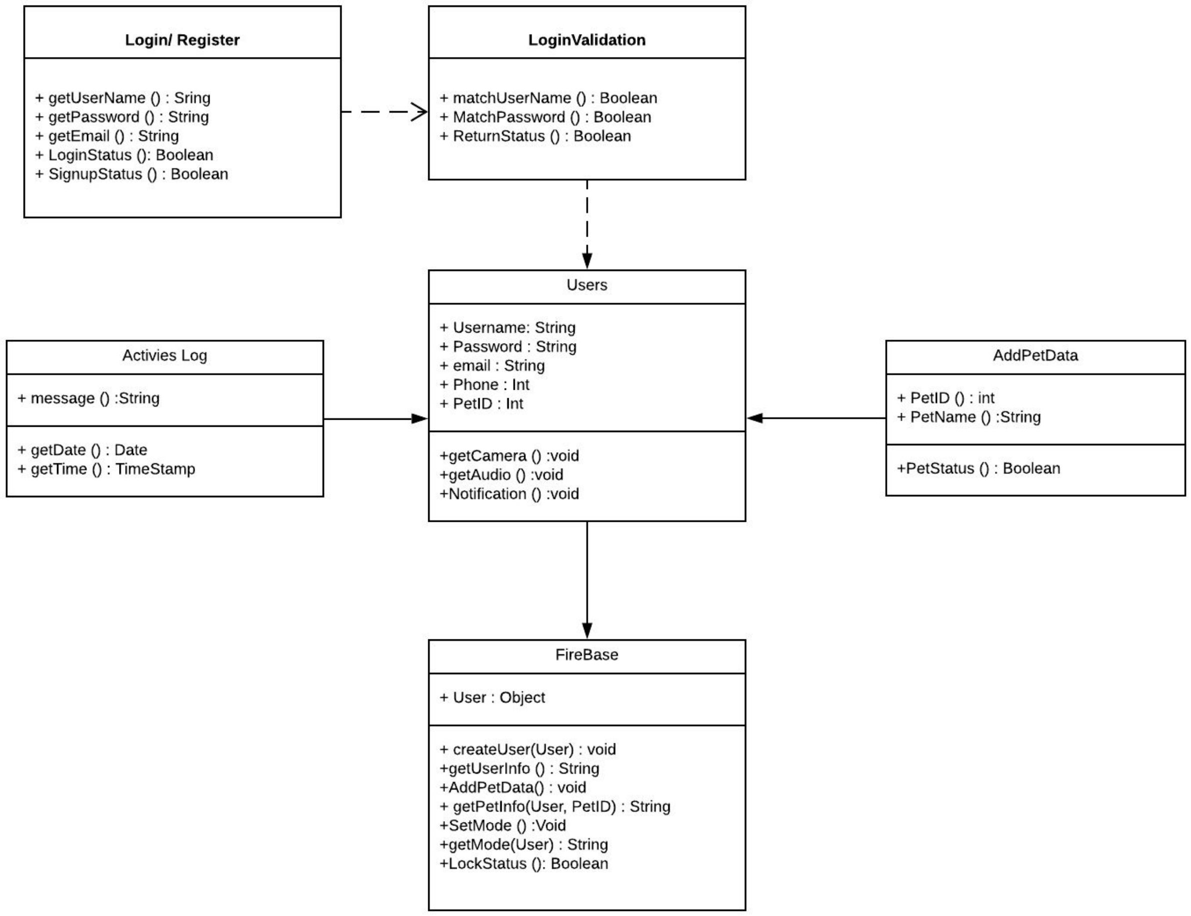


Figure 33 Class Diagram for Software Design

7.5 Database

As previously mentioned, we will be implementing Google Firebase as our database system. With this comes two options the Cloud Firestore or the Realtime Database. Both of these options offer realtime updates that do not require a server to deploy or maintain. They are both also free to use for the purposes of our project but also both have the options to be billed based on the usage. Realtime Database is Firebase's first database and uses JSON trees to structure its data. The original purpose was to ease the development of mobile applications that require synchronized data across all of the clients. Cloud Firestore is the newer database option and was built based off of the Realtime Database. Its data is organized as a collection of documents and aims to provide faster and more complex queries. The Cloud Firestore is made for applications that are data heavy and need advanced querying on large amounts of data. Considering we need our database to mainly store some basic information and to communicate the status of the pets and doors, the Realtime Database is the more sensible option and what we will be using.

In Figure 34, you will see the layout of how we will be organizing the Realtime Database. Our initial database design has us organizing the data into three different trees. The first would be the Users which holds the basic user information like the username, password and email where username is the key that will be used to access the other data. All of these fields would be strings except for the pet ID which would be an Int. This PetID would be used as the key to look up the user's pet in the Pets tree.

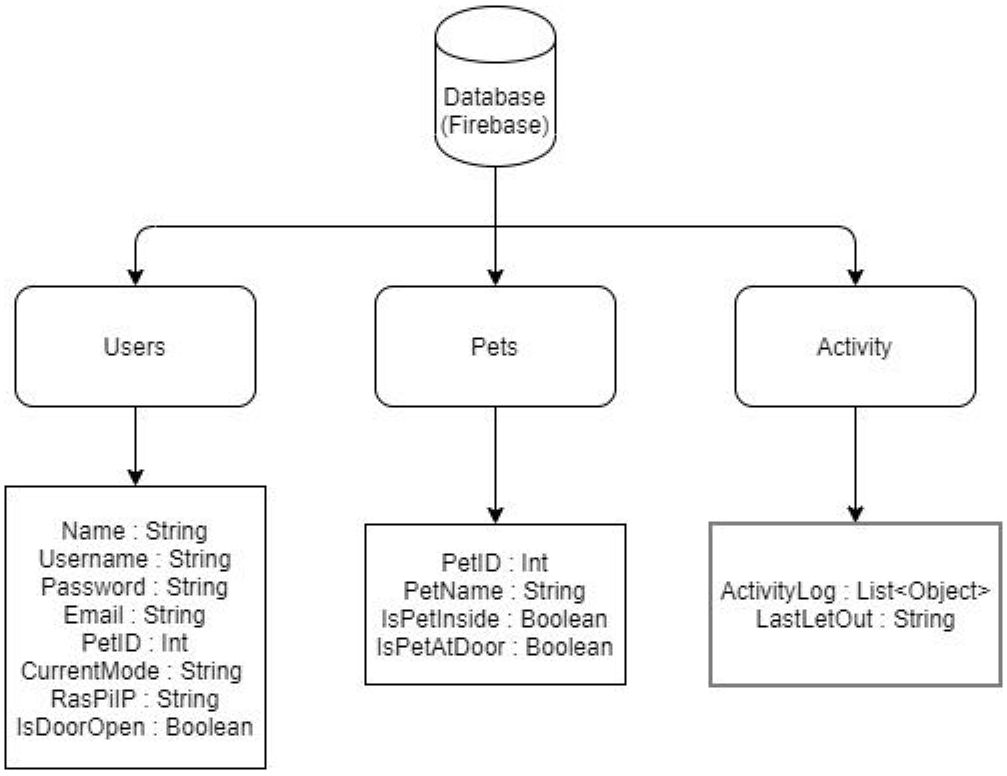


Figure 34: Database Design Tree

In that we store the pet name and some boolean values in IsPetInside and IsPetAtDoor to signal where the pet is in relation to our system. The Activity tree is the more flexible tree and would hold things such as if the door is open, possible sensor statuses, and possibly an activity log. Again the username is the primary key in this tree. These things are more of stretch goals unlike the other two trees which are needed for our system to work properly. The Activity tree will most likely be altered more throughout the design process. This information is spread out to another tree instead of including it in the Users tree because it makes the data structure flatter. If we did lump the two trees together this could cause longer reading times as the entire child is loaded to the client when a node is accessed. The possibility of the data log would add a lot of unneeded data to be loaded when the user is just trying to do something like entering their password. The separation of these branches is also a way to organize the data in a more readable way.

This data needs to be stored so the developers can display the necessary information to the user and to also send information back and forth between the Raspberry Pi and the Java app. During the setup process the user will enter the required information such as, username, password, email, their pet name, and the pet's RFID. New nodes will then be created in the Realtime Database in each of the three trees. If the data needs to be accessed, the developers can access it by calling the correct path to the needed node. For example, when trying to get a pet's name, the path would be the Firebase root, then the Pets node, the PetID node, and finally the PetName node. For data that is essential to the system's flowchart such as CurrentMode, IsPetInside, IsPetAtDoor, and IsDoorOpen another approach to reading the data needs to be taken. These data fields need to have listeners that perform certain actions any time a change is made by either the Raspberry Pi or the mobile app. An example of this would be when the pet is at the door and the owner is away, the Raspberry Pi would update the Firebase IsPetAtDoor field to true. The listener for this item in the Java application would then be triggered and send the notification to the user. A similar process would take place for the other important information in the system and happen from both the mobile app and the Raspberry Pi.

7.6 Security and Authentication

Authentication and authorization are very crucial when it comes to security vulnerabilities. Our application stores important data relevant to home access which can be dangerous to the users in case the device phone has been lost or stolen. Mobile application authentication is the verification of a user's identity to secure access when users use a mobile device. Our app provides users with access to remote service, so we design app authentication using username/password authentication to be performed at the remote end point. Firebase provides a backend service that allows users to sign up and authenticate against multiple providers. Authentication SDK can easily integrate with drop-in UI flows. In our login authentication, we prefer to use a custom authentication system and option of Sign in with a pre-built UI. FirebaseAuth object will initialize Firebase Authentication. When users sign in for the first time, new accounts which include username and password, and phone number are stored in the Firebase project to use across the application. This action is created linked to the credential. In the event of users forgetting their username or password, Email Link Authentication is the option for users to authenticate their owner by using their email address. Figure 35 below shows sequence authentication diagram communication between application, backend server and database.

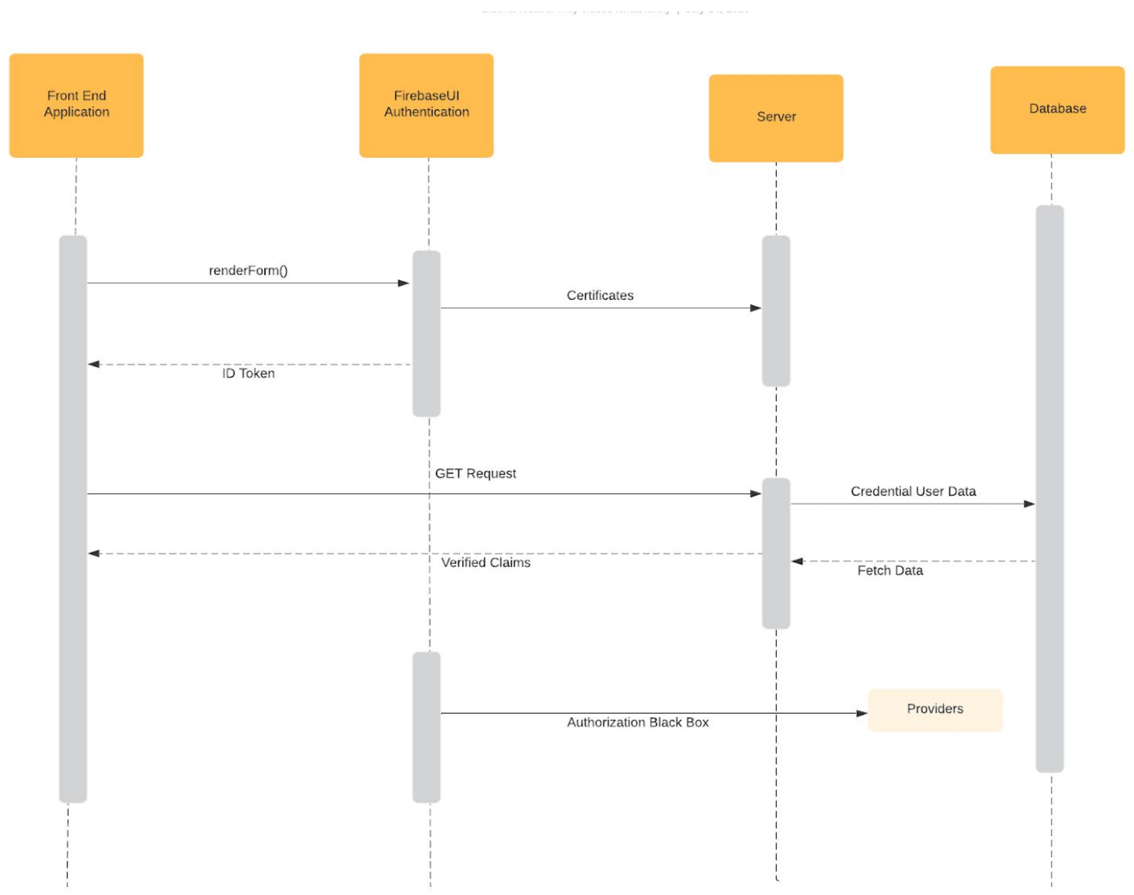


Figure 35 Sequence Diagram for Application Authentication

7.7 Wireless Communications

Since our project consists of automating the process of letting a pet go outside, communication is key to the project's success. From a software standpoint we have three items that need to be in sync with one another, which is the Android mobile application, the Raspberry Pi, and the Firebase Realtime Database. The pet owner will use the mobile app to get alerts about the system in their home and communicate with it to do things like allowing the door to open. Both the app and the Raspberry Pi will be updating the Realtime Database to make sure that everything has the most up to date information. Figure 36 depicts the software technologies that we are going to use to make this happen. We see that there is a path from the mobile application directly to the Raspberry Pi. This is due to the fact that not all of the data that is being passed around should go through the Realtime Database. The live video feature gives us the need to have a communication path directly between the two points of the system. The two arrows coming from below the Raspberry Pi in the diagram represent any of the hardware connections that will be made in order for the Pi to communicate with the peripherals. Given that this section is focused on software, these connections did not need to be depicted for this diagram. The following subsections will describe each technology, the data they are responsible for, and the components of the system that they are interacting with.

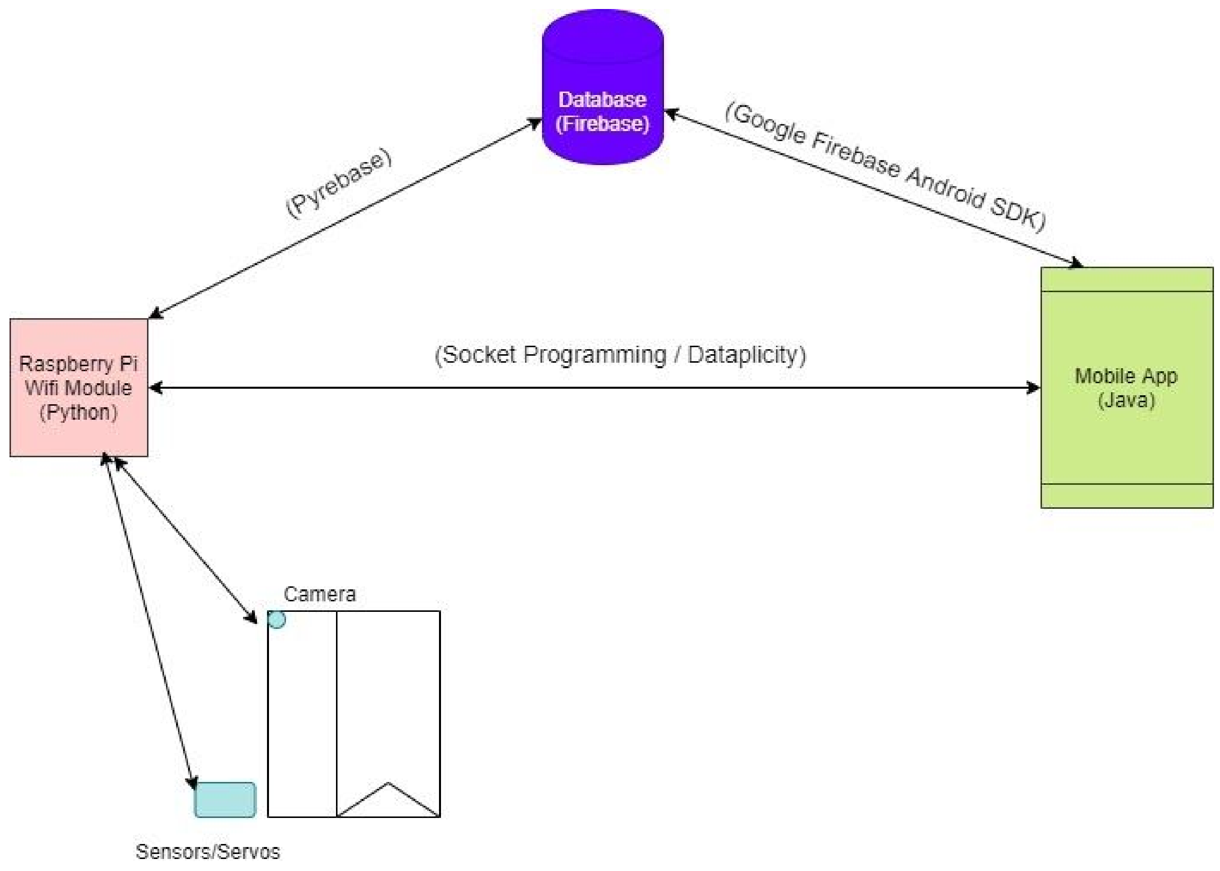

Database (Firebase)
(Pyrebase)
(Google Firebase Android SDK)
Raspberry Pi Wifi Module (Python)
(Socket Programming / Dataplicity)
Mobile App (Java)
Camera
Sensors/Servos

Figure 36 : Wireless Communications Diagram

7.7.1 Google Firebase Android SDK

For the purposes of connecting the database to the mobile app we will be using Google Firebase's Android SDK [FIRE2]. This has all of the methods that we would need to read and write data to the database. Table 22 lists out some of the methods from the Android SDK that are relevant to our project. There are a few classes that should be noted for this implementation. The FirebaseDatabase class allows us to first get an instance of the database that we will be working with. The DatabaseReference class then allows us to reference specific nodes and build paths throughout that database instance. These will both be essential in implementing basic read and write functionalities from the Java programs. Writing data simply is handled by the DatabaseReference class with the setValue method. Reading data is more complicated as we need to add a ValueEventListener object to a DatabaseReference object first. Then when the data is changed on that node the onDataChange method is triggered and performs any of the actions that we put in it. At that point we can then call the getValue method on a DataSnapshot object to read the data. This is what our program will need in order to alert the user that their pet is at the door. The Raspberry Pi will change a value in the database which will trigger the onDataChange method. This method would then use the Android notification system to alert the user.

Name	**Parameters**	**Return Type**	**Description**
getInstance	None	FirebaseDatabase	Returns an instance of the configured database
getReference	None or (String path)	DatabaseReference	Returns a reference to the root node of database or the specified node in database
child	(String pathString)	DatabaseReference	Returns the reference to the parent's child node
setValue	(Object value)	void	Sets the value at the current reference node to the object value passed
getValue	None	Object	Returns the object that is at the reference node that triggered the DataSnapshot
onDataChange	DataSnapshot	void	Called every time that data is changed at a specific database reference.
addValueEventListener	ValueEventListener	void	Adds a ValueEventListener to a DatabaseReference

Table 22 : Table of Android SDK Methods [FIRE2]

7.7.2 Pyrebase

Unlike the Firebase Android SDK, Python and the Raspberry Pi are not natively supported by Google Firebase. However, there are third party libraries that allow Python programs to connect and interact with Firebase. One of those is Pyrebase which is what we will implement on our Raspberry Pi to allow it to communicate with our Realtime Database. Again, the functions from Pyrebase that are related to our program design and implementation are listed in table 23. The process is overall the same as with the Android SDK. We first configure a dictionary that is used to connect to the database with the initialize_app and database functions. From there we can create different paths to different nodes in our database with the child method as needed throughout the programs. To write data, there is a set method which is implemented in the same way as its Java counterpart. For reading data, the get and val methods are used together to first return a PyreResponse object on the node and then return the value that the node is holding. We also have the ability to have the python program continuously listen for data changes by using the stream and a stream_handler methods. The stream method first needs to be attached to a node in the database then when that node's data changes the stream_handler method takes the actions that we program it to take. This would be important in our implementation of having the Raspberry Pi react based on what the user is controlling through the app.

Name	Parameters	Return Type	Description
initialize_app	dict : config	Void	Initializes the firebase reference with the contents of config
database	None	Database	Returns a database object for our Realtime Database
child	str : path	Database	Returns the new reference to the parent's child node
set	Object : data	Void	Sets the value at that node reference to data
get	None or Object : token	PyreResponse	Returns the data from the current path

Name	Parameters	Return Type	Description
val	None	Object	Returns the data from a PyreResponse
stream	Stream_handler	Stream	Attaches a stream handler to a specific node in the database.
stream_handler	str : message	Void	Listens to the live changes from Firebase

Table 23 : List of Pyrebase Relevant Functions [PYRE]

7.7.3 Socket Programming and Dataplicity

Most of the data being passed around is going to be stored, read from and written to the Realtime Database as mentioned in the previous two sections. There are however certain features of our system that involve data being transferred directly to and from the mobile app and the Raspberry Pi. That would be when the user is using the app to look at the live video from the webcam or when the user is sending a message to the system to be played on the speaker to the pet.

One way to make this happen is to configure the Raspberry Pi as an Access Point and then connect to it by running a server program on the Raspberry Pi and a client program from the mobile application. This would involve using socket programming to connect the client to the server by linking up the same IP address and port number on both programs. This would then allow the two endpoints to send the data back and forth, for the video or audio feeds. This is a good solution but to make it work when the client is on a different network, for example away from the house, port forwarding would be required to have that IP address and port number public so the mobile app can connect to it. Port forwarding would route anyone trying to access that socket to the correct IP address and port combination which would be the Raspberry Pi. This would require the user to do some configurations on their home router and is not a good idea for security reasons as it could give anyone access to the user's home network. We could have the Raspberry Pi port forwarded and this would take away the need for the user to do the extra configurations but would not solve the security issue.

Instead of using sockets to transfer the video, we could use Motion, which is a tool that is commonly used in camera surveillance projects. This would connect to the camera and stream the video and take away the need for the client and server programs in sharing the live video. Instead the mobile app would look up the socket that is set up through motion and then it would have access to the video. Again, the port forwarding would be needed and we run into the same issues with configuration and security as before.

One alternative is to just upload the video to the database from the Raspberry Pi and then continuously read from the mobile app that way. This could work but would be costly in terms of taking up storage in the database and would have a delay from the real video and the video being displayed through the mobile app.

The solution we found is to use Dataplicity and Hawkeye. Dataplicity is a tool that allows us to access the command line of the Raspberry Pi remotely. An account would need to be created and linked to the specific device and then we would just need internet access and a web browser to open up a terminal remotely. For the video streaming aspect, Hawkeye is used with Dataplicity to stream the video remotely. After all the necessary configurations, a terminal command would start the video on the Raspberry Pi and a link would give us remote access to the video feed. We would then use this link to embed the video into the mobile android app. This would solve both issues of having the user to configure their home router a specific way and also not have their home network open to attacks.

Dataplicity and Hawkeye will be the first option when it comes to implementing the video feature and all of the other options mentioned in the section will serve as backups. For the purposes of transferring the message through a speaker a similar process will be used through Dataplicity. With this we could also send the message as a string through the database and use a text to speech library to play it out loud through the speaker. As ease of use and security our two of the most important items with our project design it makes the most sense to use Dataplicity for these data transfers and only use Socket Programming for anything that would need to directly transferred when the user is on the same network as the Raspberry Pi.

8.0 Prototyping

Prototyping can be a very beneficial step in any design process as it gives the design team a visual example of to work off of. Once they have a prototype certain aspects of the design can become clearer and the appropriate changes can be made. Prototyping is also a way for the design team to present an unfinished version of the project to the client or customer. Since a lot of times this person is not always technologically advanced the prototype is a good indication of what the product is going to end up being. From this they can make the changes and suggestions that they feel is necessary. In this section we will discuss the bill of materials, our printed circuit board designs, and our software prototype based on our initial designs.

8.1 Part Acquisition and Bill of Materials

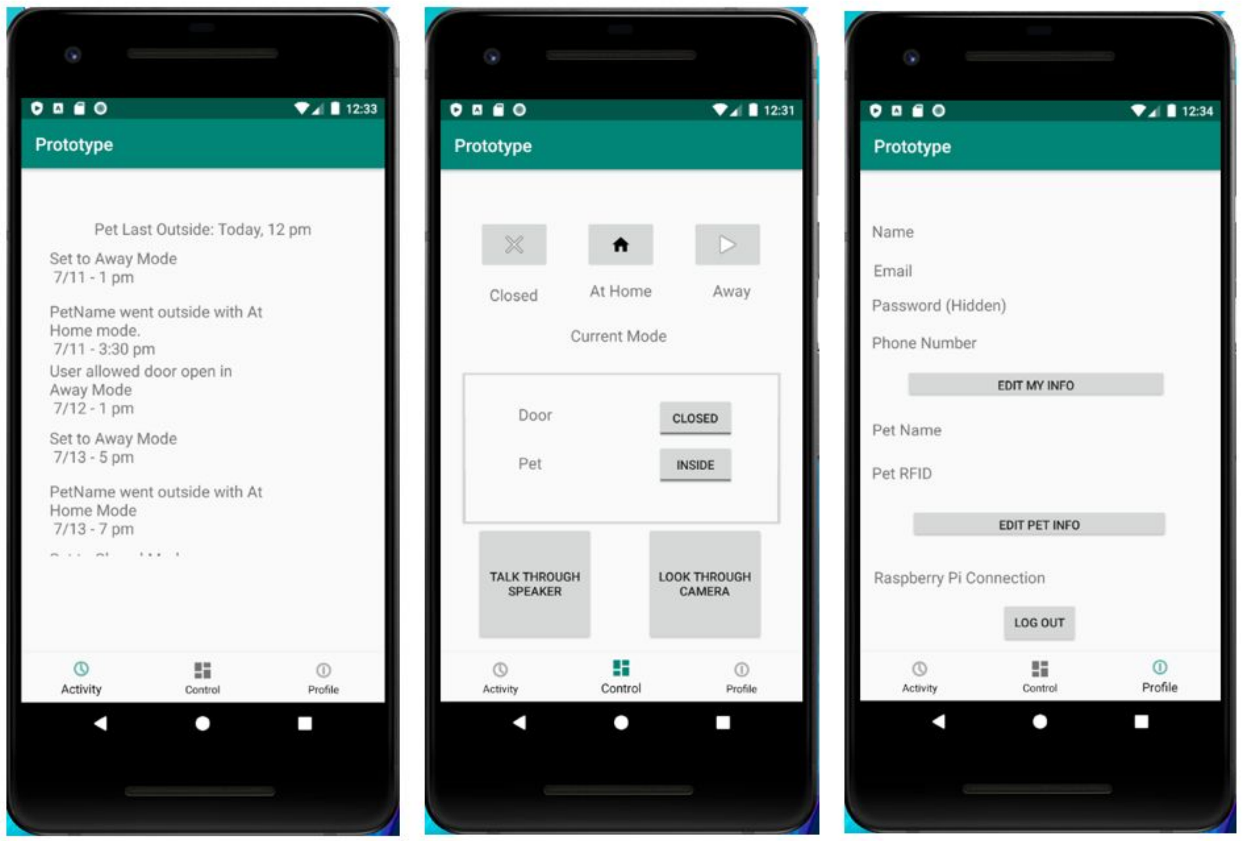
Engineers design and develop a schematic drawing and outline functions of a project. The build of materials (BOM) is beneficial to people outside the developing team and the team themselves. A bill of material is a comprehensive list of raw materials, parts, components needed in addition to the quantities of each to manufacture one product. In early stages, engineers stay to finalize the schematic drawing as well as the bill of materials as the preparation and the time to obtain all of the materials and parts may take time. Many engineering projects have deadlines and parts' lead time is another unexpected time constraint needed to take into account.

Many electrical components can be found in large distributors like Digikey and Mouser. Due to the complexity and scope of the project, we will not be receiving a discount on components since we only need a small quantity. It's a good engineering practice to purchase a couple extra inexpensive components like resistors, capacitors, and inductors in case of any complications that may lay ahead in testing. Digikey and Mouser make it easy for engineers to search and select components through their search engine.

8.2 Software Prototype

The software prototyping consisted of taking the wireframe design from the User Interface Design section and creating the user interface in Android Studio. While the functionality will be implemented in the building phase of our project, the screenshots below serve as a second draft of the user interface design after the wireframes. User interface is very important to software development because if it is not a good design the user will have trouble accessing all the features that we will include in the mobile application. In figure 37 below shows the three main screens of the mobile app. From left to right they are the Activity, the Control, and the Profile screens. While there are other screens that we will need to make these three have a slightly higher priority because they deal with the everyday usage of the app. This is not the final product as is stated in the title it is just the first prototype. We will use this to make more design decisions on what the user interface should look like and how it should work.

When thinking of design we always want the user to know the current status of their home system. The Control screen is where they will be able to see these statuses and also make changes to the current mode and open or close the door. In the prototype we have three buttons at the top of the page for the three modes. The working design will display the current mode where the 'Current Mode' text is shown. The button of the mode the system is in will also stay highlighted as another indication. Below that the door toggle button will be used to both indicate the current state of the door and allow the user to change the status by tapping on it. The pet status is also set up the same way right now but the software design team is considering changing it to only show the status of the pet and differentiate it from the door toggle. While we like the layout of the prototype, the style of the screens needs to be adjusted. This will make it easier for the user to see the current screen they are on as well as providing a better overall visual experience while user the application. These are all things that the software team is going to take into consideration as we move forward with the design and implementation stages.


12:33

Prototype

Pet Last Outside: Today, 12 pm

Set to Away Mode
7/11 - 1 pm

PetName went outside with At Home mode.
7/11 - 3:30 pm

User allowed door open in Away Mode
7/12 - 1 pm

Set to Away Mode
7/13 - 5 pm

PetName went outside with At Home Mode
7/13 - 7 pm

Activity
Control
Profile

12:31

Prototype

Closed
At Home
Away

Current Mode

Door
CLOSED

Pet
INSIDE

TALK THROUGH SPEAKER
LOOK THROUGH CAMERA

Activity
Control
Profile

12:34

Prototype

Name
Email
Password (Hidden)
Phone Number

EDIT MY INFO

Pet Name
Pet RFID

EDIT PET INFO

Raspberry Pi Connection

LOG OUT

Activity
Control
Profile

Figure 37: Software Prototype Images

9.0 Testing

Troubleshooting and testing are crucial steps in the design verification. To reduce all errors and unexpected failures, one needs hands-on verification testing of every component in the system to ensure the product will meet standards and expectations. Test Engineering is a growing professional field that aims toward determining a procedure or method to best test a product. Test engineers also contribute during early stages and prototype stages to ensure the testability of the product. Electrical test equipment, such as oscilloscopes and multimeters, are used to aid test engineers. Oftentimes, test engineers will work with the developing team which consists of engineers of different disciplines to create a design verification testing (DVT) document.

9.1 Hardware Testing

Hardware testing is vital in the design process. It's important to validate individual components and ensure that each component works on its own. Problems can be isolated through this process. Once the validation for all individual components is successful, integration of systems such as populating the PCB.

9.1.1 Printed Circuit Board (Bare board Test)

Purpose:

The purpose of bare board testing of the PCB is to ensure the bare PCB meets the electrical design expectation. By performing an isolation testing and continuity testing, one can ensure that the PCB meets its expected electrical design expectation. The following bare board test will be divided into two tests: Isolation and Continuity Test. Please perform the preparation stated below individually at the start of each test (Isolation and Continuity Test)

Supplies:

- Bare board PCB
- Schematic drawing
- Multimeter
- Test leads
- Electrostatic Discharge (ESD) wrist strap
- ESD mat

Preparation:

- Properly ground tester with a ESD wrist strap to ESD mat
- Connect positive and negative leads to its corresponding location on the multimeter
- Turn off and on multimeter and turn it to a ohmmeter
- Test the multimeter by touching the positive and negative lead together (the multimeter should be reading roughly zero ohms)

Isolation Test Procedure:

- Using the ohmmeter, measure the resistance between the following locations on the board presented on Table 24
- With the following locations express in Table 24, ohmmeter should be measuring a large resistance 100 Megaohms

Location 1	Location 2	Examiner	Resistance Measured	> 100 Megaohms
TBD	TBD	Michael	-	N/A
TBD	TBD	Michael	-	N/A
TBD	TBD	Michael	-	N/A

Table 24: Isolation Test

Continuity Test Procedure:

- Using the ohmmeter, measure the resistance between the following location on the board on Table 25.
- With the following locations, ohmmeter should be measuring a zero or negligible resistance

Location 1	Location 2	Examiner	Resistance Measured	< 0.5 ohms
TBD	TBD	Michael	-	N/A
TBD	TBD	Michael	-	N/A
TBD	TBD	Michael	-	N/A

Table 25: Continuity Test

9.1.2 Power

Purpose:

The purpose of testing the AC to DC power conversion circuit on the PCB as it is a system critical circuit. The following critical circuitry will provide stable DC power to the microcontroller and all other peripherals on this product. It is important to ensure that this circuitry meets the design requirements.

Supplies:

- Populated PCB
- Power Source (wall plug)
- Schematic drawing
- Multimeter
- Test leads
- Electrostatic Discharge (ESD) wrist strap
- ESD mat

Preparation:

- Properly ground tester with a ESD wrist strap to ESD mat
- Connect positive and negative leads to its corresponding location on the multimeter
- Turn off and on multimeter and turn it to a ohmmeter
- Test the multimeter by touching the positive and negative lead together (the multimeter should be reading roughly zero ohms)

Procedure:

- Use multimeter to verify AC Wall Power
- Verify AC-DC power conversion circuit connectivity with the schematic drawing. Make sure all diodes are facing the right direction or the way expressed in the drawing.
- Plug in wall power to PCB.
- Use the multimeter to check the stable DC voltage output from circuitry.

9.1.3 Motor

Purpose:

The purpose of testing the motor is to ensure that the linear actuator is able to provide the amount of thrust or force needed to operate the sliding door mechanism. A test to check the motor's capability to resist linear external force to check the linear actuator's ability to keep the sliding door lock from external forces.

Supplies:

- PCB
- Linear Actuator (motor)
- Sliding door
- Power Source (wall plug)
- Multimeter
- Test leads
- Code

Preparation:

- Ensure the PCB has pass the PCB bare board test shown 9.1.1 and all components have placed on the PCB
- Ensure PCB has pass the power test shown in Section 9.1.2
- Connect positive and negative leads to its corresponding location on the multimeter
- Turn off and on multimeter and turn it to a ohmmeter
- Test the multimeter by touching the positive and negative lead together (the multimeter should be reading roughly zero ohms)
- Create and upload the code for the motor to the Raspberry Pi

Procedure:

1. Attach linear actuator based on the following setup below (Figure 38)
2. Power on the PCB
3. Check on Raspberry Pi is on
4. Download code to Raspberry Pi
5. Turn on Raspberry Pi to close door or extend shaft
6. Observe the movement of the linear actuator shaft extend

7. Turn on Raspberry Pi to open sliding door or contract shaft
8. Observe the movement of the linear actuator's shaft contract
9. Fill out the Table 26 below with the results.

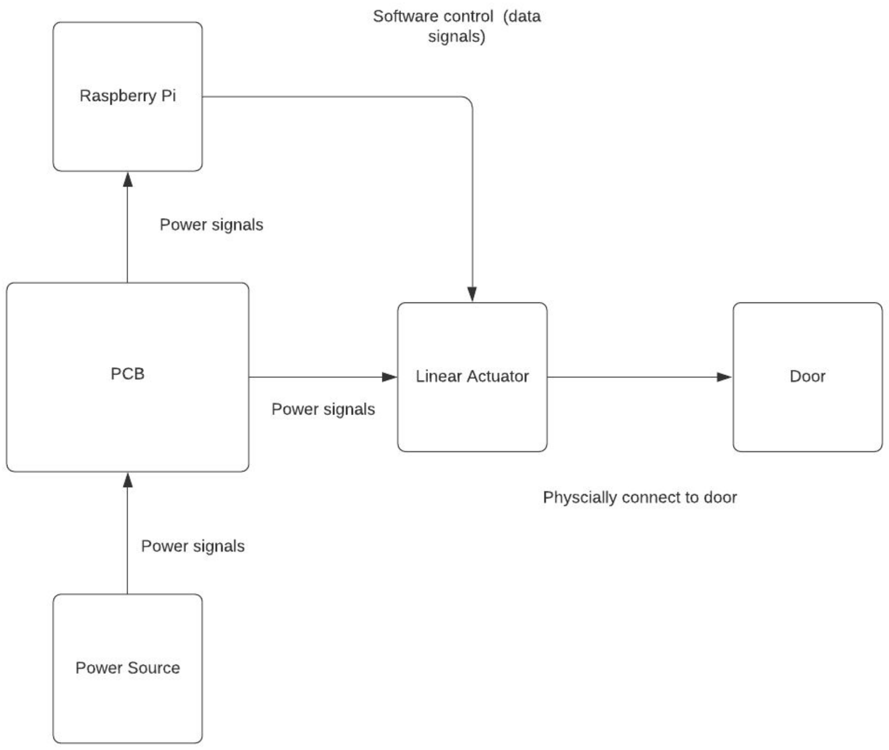


Figure 38: Linear Actuator Test Setup

	Description of linear actuator	Pass/Fail
Step 6	Shaft extends or sliding door closing	-
Step 8	Shaft contract or sliding door open	-

Table 26: Linear Actuator Test Summary

9.1.4 RFID Sensor

Purpose:

The purpose of testing the RFID sensor is to observe if the sensor can read RFID tags and notify the system when a positive ID is read.

Supplies:

- Raspberry Pi
- PN5180 RFID Sensor
- RFID Tag
- Code
- Wires

Preparation:

- Create and upload the code for the RFID sensor to the Raspberry Pi.
- Connect the sensor to the Raspberry Pi using the wiring table shown below (Table 27)
- Place the sensor in an environment similar to the project's design placement.

PN5180 Pin	Description	Raspberry Pi Connection
+5V	5 volt DC supply	Pin 2
+3.3V	3.3 volt DC supply	Pin 17
RST	Reset	Pin 22
NSS	Non Slave Select	Pin 24
MOSI	Master Out Slave In	Pin 19
MISO	Master In Slave Out	Pin 21
SCK	System Clock	Pin 23
BUSY	Busy State	Pin 26
GND	Ground	Pin 6
GPIO, IRQ, AUX, REQ	Not used	-

Table 27: PN5180 to Raspberry Pi Wiring Table

Procedure:

1. Power on the Raspberry Pi
2. Observe that the sensor is actively taking readings.
3. Observe that the initial sensor output is set to 0.
4. Place an unregistered ID tag within sensor range.
5. Observe that the RFID sensor reads the tag by seeing it display the tag ID to the console.
6. Observe that the sensor output remains at 0.

7. Place a registered ID tag within sensor range.
8. Observe that the RFID sensor reads the tag by seeing it display the tag ID to the console.
9. Observe that the sensor output changes to 1.

9.1.5 Ultrasonic Sensor

Purpose:

The purpose of testing the HC-SR04 ultrasonic sensor is to observe if the sensor properly recognizes movement, and only notifies the system when that movement is within one meter from the door.

Supplies:

- HC-SR04 Ultrasonic Sensor
- Raspberry Pi
- Code
- Wires
- A pet-sized object

Preparation:

- Create and upload the code for the ultrasonic sensor to the Raspberry Pi.
- Connect the sensor to the Raspberry Pi using the wiring diagram shown below (Figure 39)
- Place the sensor in an environment similar to the project's design placement.

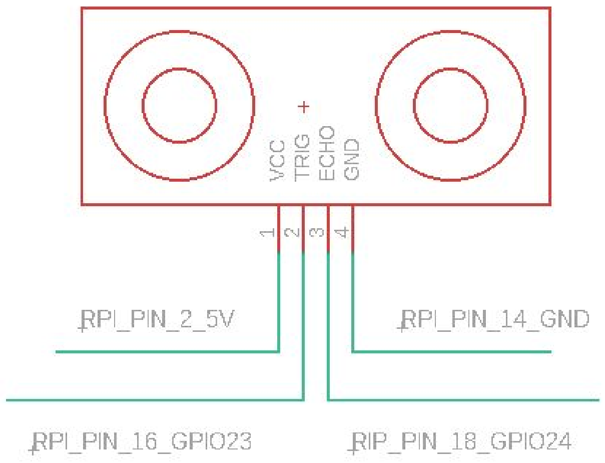


Figure 39: Wiring Diagram for HC-SR04 to Raspberry Pi

Procedure:

1. Power on the Raspberry Pi

2. Observe that the sensor is actively taking readings.
3. Observe that the initial sensor output is set to 0.
4. Place the pet-sized object in sight of the sensor but out of the one meter signaling range.
5. Observe that the sensor recognizes the object.
6. Observe that the sensor's output signal still reads 0.
7. Move the pet-sized object within the one meter signaling range.
8. Observe that the sensor recognizes the object.
9. Observe the sensors output signal changes to 1, to notify the system that motion has been detected within the systems signaling range.

9.1.6 Audio and Visual

Purpose:

The purpose of testing our audio and visual peripherals is to observe if they turn on and off when they are prompted and if they appropriately transmit their media.

Supplies:

- Logitech Webcam
- USB speaker
- Raspberry Pi
- Code
- Audio file
- External web browser

Preparation:

- Connect the Logitech webcam to the top Raspberry Pi's USB 3.0 port.
- Connect the USB Speaker to the top USB 2.0 port.
- Configure Raspberry Pi to use both USB devices.
- Place both peripherals in an environment similar to the project's design placement.

Procedure:

1. Connect power and turn on Raspberry Pi.
2. Use shell commands to turn on the webcam.
3. Observe that the webcam has been powered on.
4. Enter the associated IP address for the Raspberry Pi's camera output in a web browser.
5. Observe that the web browser now shows a live feed from the webcam.
6. Make noise in the area of the camera's microphone.
7. Observe that the noise is output to your browser.
8. Use shell commands to turn off the webcam and turn on the speaker.
9. Play audio file.

10. Observe that the audio file is transmitted from the USB speaker.

9.1.7 LCD

Purpose:

The purpose of testing the LCD screen is to see that it displays the correct messages according to what operation the system is performing.

Supplies:

- 1602 LCD screen
- Raspberry Pi
- Code
- Wires

Preparation:

- Create and upload the code for the LCD display to the Raspberry Pi.
- Connect the LCD to the Raspberry Pi using the wiring diagram shown below (Table 28)
- Place the LCD screen in an environment similar to the project's design placement.

LCD Pin	Description	Raspberry Pi Pin
VSS	Ground	Pin 14
VDD	5V DC Supply	Pin 4
VO	Potentiometer	Pin 7
RS	Register Select	Pin 8
RW	Read / Write	-
E	Clock Enabled	Pin 10
D0 - D3	Bits	-
D4	Bit 4	Pin 12
D5	Bit 5	Pin 11
D6	Bit 6	Pin 13
D7	Bit 7	Pin 15

LCD Pin	Description	Raspberry Pi Pin
A	LED Anode +	Pin 4
K	LED Cathode -	Pin 14

Table 28: LCD 1602 to Raspberry Pi Wiring Table

Procedure:

1. Power on the Raspberry Pi
2. Observe that the LCD screen displays the idle message according to the message table. (Table 29)
3. Wait for 30 seconds.
4. Observe that the LCD screen lowers its brightness.
5. Use code to simulate entity detection inside of activation range.
6. Observe the LCD screen displays the proper detection message according to the message table.
7. Use code to simulate the entity verification process is running.
8. Observe the LCD screen displays the proper verification message according to the message table.
9. Use code to simulate that the entity verification process has identified an invalid tag.
10. Observe the LCD screen displays the proper identification message according to the message table.
11. Use code to simulate that the entity verification process has identified a valid tag.
12. Observe the LCD screen displays the proper identification message according to the message table.
13. Use code to simulate that the door opening process has been allowed.
14. Observe the LCD screen is displaying the proper door operation message according to the message table.
15. Use code to simulate that the door opening process has begun.
16. Observe the LCD screen is displaying the proper door operation message according to the message table.
17. Use code to simulate that the door closing has begun.
18. Observe the LCD screen is displaying the proper door operation message according to the message table.
19. Use code to simulate that the system is transitioning back to the idle state.
20. Observe that the LCD screen displays the idle message according to the message table.

System Operation	LCD Display Message
Idle between activations	“System Idle”
Motion detection	“Entity Detected”

System Operation	LCD Display Message
RFID Entity Verification	“Verifying…”
RFID Tag is invalid/unrecognized	“Entity invalid, Return to idle”
RFID Tag is valid	“ID accepted, Notifying User”
Door Opening Allowed	“Permission Granted”
Door is opening	“Opening...”
Door is closing	“Closing...”
Door cycle is completed	“Operation Complete” “Returning to idle”

Table 28: LCD Message Display per System Operation

9.1.8 LEDs

Purpose:

The purpose for testing the LEDs is to make sure that the system can visually represent each operation and that the associated LED to that operation functions properly.

Supplies:

- Raspberry Pi
- LEDs (1 RGB, 1 Blue, 1 Red, 1 Yellow, 2 Green)
- Code
- Wires

Preparation:

- Create and upload the code for the LEDs to the Raspberry Pi.
- Connect the LEDs to the Raspberry Pi using the wiring diagram shown below (Table 30)
- Place the LEDs and resistors on a breadboard.

LED	+ Connection	- Connection
Blue	Pin 29 via 470Ω Resistor	Pin 39Ground
Red	Pin 31 via 470Ω Resistor	Pin 39Ground

LED	+ Connection	- Connection
Green	Pin 33 via 470Ω Resistor	Pin 39Ground
Yellow	Pin 35 via 470Ω Resistor	Pin 39Ground
Green	Pin 37 via 470Ω Resistor	Pin 39Ground
RGB	Pin 36 via 470Ω Resistor	Pin 39Ground

Table 30 LED to Raspberry Pi Wiring Table

Procedure:

1. Power on the Raspberry Pi. Default system mode is "Remote."
2. Observe that the blue and red LEDs function according to the LED behavior table below (Table 31)
3. Use code to change the system mode to "Home."
4. Observe that the red LED functions according to the LED behavior table.
5. Use code to simulate that an entity is within detection range.
6. Observe that the green 1 LED functions according to the LED behavior table.
7. Use code to simulate that the system is scanning for and identifying the RFID tag.
8. Observe that the yellow LED functions according to the LED behavior table.
9. Use code to simulate that the system accepted the RFID tag.
10. Observe that the green 2 LED functions according to the LED behavior table.
11. Use code to simulate that the system is opening the door.
12. Observe that the RGB LED functions according to the LED behavior table.
13. Use code to simulate that the system is closing the door.
14. Observe that the RGB LED functions according to the LED behavior table.

LED	Operation	Behavior
Blue	System On	LED on
Red	System Mode "Remote"	LED on
Red	System Mode "Home"	LED off
Green 1	Entity Detected in Range	LED on
Yellow	Scanning ID	LED Flashes
Green 2	ID Accepted	LED on
RGB	Door Opening/Closing	LED Flashes/Changes Color

Table 31: LED Operation and Behavior Table

9.1.9 Hardware Testing Checklist

The checklist below in table 32 will serve to help us complete the testing phase in the next course. It will be used to keep track of the status of each test case that was designed in the previous subsections. By separating out the individual hardware components, we can isolate any possible problems that might arise. This will stop major issues when it comes to bringing the other components of the project together.

Test	Examiner	Test Date	Pass/Fail
Bare Board - Isolation	Michael	TBD	Pending
Bare Board - Continiuty	Michael	TBD	Pending
Power	Michael	TBD	Pending
Motor	Michael	TBD	Pending
RFID Sensor	Graham & Michael	TBD	Pending
Ultrasonic Sensor	Graham & Michael	TBD	Pending
Audio and Visual	Graham & Michael	TBD	Pending
LCD	Ryan & Michael	TBD	Pending
LEDs	Ryan & Michael	TBD	Pending

Table 32: Hardware Testing Checklist Summary

9.2 Software Testing

We need to ensure that our software application works properly and each component has been tested to determine if there are any issues. Unit Testing is the process of checking a small piece of code to ensure that the functionality of the application can be delivered correctly. We also conduct integration testing where independent development units get tested together. This is very common in software development as the unit tests would need to be passed before the integration tests can begin. This is because one bug that could be caught in a unit test could cause multiple errors in an integration test. The debugging process would be more complicated this way and less efficient. In this section we conduct various test performances based on observation throughout the database console, mobile interface, and raspberry Pi terminal.

9.2.1 Registration

Purpose:

The purpose of this test is to ensure that the registration process is working correctly. When the user goes into the application without an account the app needs to take them through all of the necessary steps and then add that new user to the Realtime Database.

Supplies:

- Android Studio Emulator or Android Device
- Google Firebase Realtime Database
- RFID Tag

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Configure the Realtime Database with the appropriate security rules.
- Create the user interface android screens that will be displayed to the user and capture all of the user's input.
- Create the necessary back end code that deals with adding new users to the database.

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Select 'Create Account'
3. Enter in test user information
4. Enter in test pet information
5. Tap the 'Submit' button
6. Check on the Firebase Console for the new user to be shown.

9.2.2 Login

Purpose:
The login sequence must be tested in order to make sure that the user's who already have an account can come back to use the app with all of their correct data.

Supplies:

- Android Studio Emulator or Android Device
- Google Firebase Console

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Configure the Realtime Database with the appropriate security rules.
- Create the user interface android screens that will be displayed to the user and capture all of the user's input.
- Create the necessary back end code that deals with adding new users to the database.

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Select 'Log In'
3. Enter a test user's username and password.
4. Check the profile screen for the correct user information
5. Check the control screen for the correct modes and status that the test user last left the system in.

9.2.3 Add /Edit /Delete

Purpose:

The purpose of this test is to ensure that data is getting updated in the database and shown on the mobile application screen correctly. When the users perform add, edit or delete any data in the application, those data should get updated in realtime and deliver updated data to the application.

Supplies:

- Android Studio Emulator or Android Device
- Google Firebase Realtime Database

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Configure the Realtime Database with the appropriate security rules.
- Create the user interface android screens that will be displayed to the user and capture all of the user's input.
- Create the necessary back end code for add, edit, delete in the database.

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Enter sample data like address information to test 'Add' data, click 'Save'
3. Check the Firebase console to see if address information has been added
4. Go back to the mobile application and view address information on screen.
5. Edit address information and click 'Save'
6. Check the Firebase console to see if address information has been changed.
7. Go back to the mobile application and observe if the address has been updated.
8. Delete address information from the application.
9. Check on the Firebase Console if data has been deleted.
10. Go back to the application and observe for deleted data no longer available.

9.2.4 Push Notification

Purpose:

The purpose of this test is to ensure that the server successfully sends push notification to users as mobile alter that pop up on the mobile device when there is any update or read new data from the Database.

Supplies:

- Android Studio Emulator or Android Device
- Firebase Cloud Message (FCM) SDK.

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Configure the Realtime Database with the appropriate security rules.
- Retrieve and access device's registration token from Firebase Cloud Messaging SDK.
- By calling FirebaseInstanceId.getInstance().getInstanceId()
- Send and store token in the app server using Firebase Instance ID to authenticate and authorize actions.

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Ensure the app is in the background on Android device
3. Open the notification composer and select New notification.
4. Enter text message and select Send test message
5. In the field labeled Add an FCM registration token and enter the registration token
6. Click Test
7. Users tap on notification to see if notification is delivered to device's system tray

9.2.5 Mobile App Camera Access

Purpose:

The purpose of camera testing is to ensure that the application can access the device camera hardware like an android smartphone. Users should be able to send requests to an existing camera app to capture a photo or video and return data back to the users to view or display.

Supplies:

- Android Studio Emulator or Android Device
- Raspberry Pi
- Camera

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Enable camera software in raspberry Pi and connect the camera to the CSI port.
- Connect network by enabled ssh port and enter IP address

- Create preview class to display live data coming from camera so tester can capture a picture or video

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Connect Camera with Raspberry Pi.
3. Tap on the access camera on the application to start camera configuration and activate the device.
4. Observe on the display screen if there is any picture or live data has been captured on Android Device.

9.2.6 Mobile App Audio Access

Purpose:

The purpose of the testing is to ensure that the audio function works properly. Users should be able to speak via microphone in the smart phone and voice return to the device.

Supplies:

- Android Device

Preparation:

- Properly configure and implement on the Android Studio Project.
- Launch Media Controller Test
- Create Media Browser service

Procedure:

1. Open the Mobile App and make sure the app runs in the background
2. Start MCT and allow connection to the app's browser service.
3. Pass the configuration to audio record
4. Run ‘Test’ and monitor MCT control page for ‘Supported’

9.2.7 User Interface and Gui

Purpose:

UI testing is conducted to ensure that UI elements like buttons, input box, selection fields, images are verified and have proper appearance on the screen. Color and style font should be invisible and have a consistent look to build users friendly between application and users.

Supplies:

- Android Studio Emulator or Android Device

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Ensure that the emulator is set for Android phone configuration to get a proper size screen and resolution when the application is run on a real smartphone device.

Procedure:

- Test screen orientation, observe if screen can rotate in both portrait and landscape mode
- Test if loading in progress is visible when page loading when activating device from smartphone app
- Test if Home mode button for on - off if can be clicked with any kind of touch or slide.
- Check and test under the activity menu if it can display different log activities.
- Check and test under the user profile menu if users' information list displays properly.
- Test for camera and audio button can turn on and off.
- Test for each page if it can navigate properly.

9.2.8 Mobile App Database Connection

Purpose:

This test case will handle testing of reading and writing to the database from the Android App. This focuses on the aspects of updating the necessary fields for the everyday use of the system.

Supplies:

- Android Studio Emulator or Android Device

- Google Firebase Realtime Database
- Google Firebase Console

Preparation:

- Properly configure and implement the Google Services Plugin into the Android Studio Project.
- Link the correct Realtime Database with the Android Studio Project.
- Configure the Realtime Database with the appropriate security rules.
- Create the user interface android screens that will be displayed to the user and capture all of the user's input.
- Create the necessary back end code that deals with getting and updating information in the database.

Procedure:

1. Open the Mobile App on an emulator or connected Android device.
2. Make sure that a test user is logged in.
3. Enter a test user's username and password.
4. Open the Google Firebase Console in another window.
5. Test the updating to the database by changing the current mode and closing or opening the door.
6. To test the app reading from the database, change the current mode manually in the Google Firebase Console. Also change the pet name manually in the console.

9.2.9 Raspberry Pi Database Connection

Purpose:

This test case will handle testing of reading and writing to the database from the Raspberry Pi. This again focuses on the everyday use of the system such as the current status of the door.

Supplies:

- Google Firebase Console
- Raspberry Pi

Preparation:

- Properly configure and implement the Google Services Plugin and the Realtime Database keys in the python file.
- Configure the Realtime Database with the appropriate security rules.
- Create the necessary back end python code that deals with getting and updating information in the database.
- Create a test program that makes some simple updates to the database.
- Make another test program that has stream listeners on the CurrentMode string, and the isDoorOpen and isPetAtDoor booleans

Procedure:

1. Open the Google Firebase Console.
2. Open a terminal on the Raspberry Pi with a connect screen.
3. Call a test script to test that the code is updating values in the Realtime Database such as the CurrentMode string, the isDoorOpen boolean or the isPetAtDoor boolean.
4. Run the second test code and manually change one of those values in the Firebase Console. This test code should ensure that the stream listener is set up and print a message to the terminal. This message would simulate the Raspberry Pi controlling an aspect of the home system.

9.2.10 Control System With Raspberry Pi

Purpose:

The purpose of this test case is to make sure that the Raspberry Pi software is properly controlling the peripherals of the home system.

Supplies:

- Raspberry Pi
- LCD Screen
- LEDs
- Motor
- Speaker
- Camera

Preparation:

- Configure all of the hardware components with the necessary connections, circuitry and software settings.
- Configure the Realtime Database with the appropriate security rules.
- Write the necessary programs that interact with each of the hardware components listed in the above supplies section.
- Write simple test programs to invoke each of the hardware components.

Procedure:

1. Open a terminal on the Raspberry Pi and with a connected screen.
2. Run the test programs to ensure that the following tasks are completed successfully:
    a. Display a message to the LCD screen
    b. Turn on/off the LEDs
    c. Turn the motor on and off
    d. Play a test sound on the speaker
    e. Display a video from the webcam.

9.2.11 Read Sensor Information From Raspberry Pi

Purpose:

While the previous test case aims to look at the Raspberry Pi writing and updating the hardware components this test case focuses on the Raspberry Pi software reading from the necessary sensors.

Supplies:

- Raspberry Pi
- RFID Sensor
- RFID Tag
- Ultrasonic Sensor

Preparation:

- Configure all of the hardware components with the necessary connections, circuitry and software settings.
- Configure the Realtime Database with the appropriate security rules.
- Write the necessary programs that interact with each of the hardware components listed in the above supplies section.

- Write simple test programs to invoke each of the hardware components.

Procedure:

1. Open a terminal on the Raspberry Pi and with a connected screen.
2. Run test programs or simple print statements to complete each of the following tasks
    a. Print the distance of an object in front of the Ultrasonic Sensor.
    b. Use the RFID to check that the RFID sensor is recognizing the RFID tag.
    c. Test the Ultrasonic and RFID sensors together by placing an object in front of the Ultrasonic sensor and display that the tag was not recognized.
    d. Repeat task c, with an object that has the RFID tag attached to show that the Ultrasonic and RFID sensors are working together to recognize the RFID tag.

9.2.12 Integration

Purpose:

This serves as a test of the overall system. While the other test cases isolate certain aspects of the software system this will look at the different components working together as a whole.

Supplies:

- Android Emulator or connected device.
- Raspberry Pi
- Motor
- RFID Sensor and Tag
- LEDs
- LCD
- Audio and Visual Peripherals

Preparation:

- Configure all of the hardware components with the necessary connections, circuitry and software settings.
- Configure the database for both the Raspberry Pi and the Android app along with the necessary security rules for each.
- Complete the necessary code for the mobile app and the Raspberry Pi.

Procedure:

1. Power on the Raspberry Pi.
2. Open the mobile app on the emulator or connected device.
3. Test that door opens and closes by tapping on the door toggle button on the mobile app.
4. Make sure the system is in the Away Mode.
5. Use the RFID on a petsized object to trigger a notification to the phone.
    a. Make sure that the live video is displayed when this notification is opened.
    b. Observe that the door opens or closes based on the user's input.
6. Change the current mode to At Home in the Android app.
7. Use the RFID tag and the pet sized object to make sure that the door opens automatically.
8. Change the mode to Closed.
9. Use the RFID tag to make sure that the door does not open.
10. With each step make sure that the mobile app, the LEDs and the LCD screen are all updating correctly.

9.2.13 Software Testing Checklist

When it comes time to start testing the software components of Pet Connect we will use the following checklist in table 33. This will allow us to plan out when each test will take place as well as give us a good understanding of which test cases will need to be passed before moving on to other components to test. This will give the software development team a chance to go back and fix any bugs or issues that will arise before moving on. With the test cases defined in the above subsections we are able to see exactly what needs to be done on that specific aspect of the software.

Test	Examiner	Test Date	Pass/Fail
Registration	Joy	TBD	Pending
Login	Joy	TBD	Pending
Add/Edit/Delete	Joy	TBD	Pending
Push Notification	Joy	TBD	Pending
Camera App	Joy	TBD	Pending
Audio App	Joy	TBD	Pending

Test	Examiner	Test Date	Pass/Fail
UI and Gui	Joy	TBD	Pending
App Database Connection	Ryan	TBD	Pending
Raspberry Pi Database Connection	Ryan	TBD	Pending
Control System With Raspberry Pi	Ryan	TBD	Pending
Read Sensor from Raspberry Pi	Ryan	TBD	Pending
Integration	Ryan and Joy	TBD	Pending

Table 33: Software Testing Checklist Summary

10.0 Administrative Content

This section is dedicated to the project management that comes along with any major group project. This is important as we need to be as organized as possible to ensure that everyone is on the same page. The milestones allow everyone to know where we are at in the project and the deadlines that we have set both for ourselves and for on time submission. The budget and finances section deals with the price of any of the hardware parts or software tools that would need to be purchased before we begin the implementation phase. Finally the responsibilities lays out what each team member contributed to the overall design process. Following these outlines will help us to deliver a functioning design on time.

10.1 Milestones

In order to keep our team on schedule for project completion we created and used milestones. Instead of focusing on overall project completion, we used these milestones to center our attention on smaller tasks that would build up to the overall project completion. We first had to decide on what the smaller tasks should be. We used previous projects and the divide and conquer assignment to help guide our choices. Each task or “milestone” is assigned a date range for completion, followed with the current status of that milestone. The responsibilities of each milestone are individual, multiple individuals, or the whole group. What follows are two tables that represent our groups milestones for both semesters of Senior Design, with Senior Design 2 to be updated when applicable.

Reference #	Milestone	Dates	Status	Responsibility
Senior Design 1				
1	Project Ideas	5/11 - 5/15	Completed	Individual
2	Divide and Conquer	5/18 - 5/29	Completed	Group
3	Group meeting with Dr. Richie	6/2	Completed	Group
4	Research requirements and preliminary components	6/3 - 6/28	Completed	Group
5	Hardware Research (LED, LCD)	6/3 - 6/28	Completed	Ryan
6	Hardware Research (Single Board Computer, Sensors, Peripherals)	6/3 - 6/28	Completed	Graham
7	Software Research (Mobile application, Development Environment, Development Languages)	6/3 - 6/28	Completed	Joy
8	WiFi Communication Research	6/3 - 6/28	Completed	Joy, Ryan
8	Software Research (Databases, Version Control)	6/3 - 6/28	Completed	Ryan

Reference #	Milestone	Dates	Status	Responsibility
9	Hardware Research (Power Supply and Distribution, Motors, Locking Mechanism)	6/3 - 6/28	Completed	Michael
10	Submit 60 page draft	6/3 - 7/3	Completed	Group
11	Meeting with Dr. Richie	7/6	Completed	Group
12	Begin writing draft paper for SD1 100 Page Report	6/30 - 7/17	Completed	Group
13	Software Design (Databases, Wireless Communications)	6/30 - 7/17	Completed	Ryan
14	Software Design ((Mobile application, Development Environment, Development Languages)	6/30 - 7/17	Completed	Joy
15	Hardware Design (Single Board Computer, Sensors, Security Peripherals)	6/30 - 7/17	Completed	Graham
16	Hardware Design (Power Supply and Distribution, Motors, Locking Mechanism)	6/30 - 7/17	Completed	Michael
17	100 Page Report Due	7/17	Completed	Group
18	PCB Layout	7/20	Pending	Michael
19	Final Documentation	7/28	Completed	Group
20	Part Order	7/28	Pending	Group

Table 34: Senior Design 1 Milestones

Reference #	Milestone	Dates	Status	Responsibility
Senior Design 2				
1	Troubleshooting/Testing	TBD		Group
2	Finalize prototype	TBD		Group
3	CDR presentation	TBD		Group
4	Peer review	TBD		Group
5	Conference Paper	TBD		Group
6	Mid-Term Demonstration	TBD		Group
7	Final report	TBD		Group
8	Final Presentation	TBD		Group
9	Exit Interview	TBD		Group

Table 35: Senior Design 2 Milestones

10.2 Budget and Finance

When we started this project we estimated a budget of $500. We have agreed to split all costs evenly between all group members. The following parts will be sourced from Amazon, Adafruit, Texas Instruments, as well as some are already owned by a group member. The below table, Bill of Materials, is tentative and subject to change. All software design for front and back end will be done using free license software products. The prices are listed as of June 30 2020 and we plan on ordering the parts at the end of the semester or early August, and will update accordingly.

Part	Unit Cost	Quantity	Total Cost
Raspberry Pi 4B with USB-C power supply	$63	1	$63
HC-SR04 Ultrasonic Sensor	Owned	1	$0
PN5180 NFC RFID Module with tags	$12	1	$12

Part	Unit Cost	Quantity	Total Cost
Logitech Webcam	$40	1	$40
USB Speaker	$14	1	$14
LEDs	Owned	10	$0
1602 LCD Display	Owned	1	$0
PCB	$30	1	$30
Motor/Linear Actuator	$110	1	$110
Misc design parts	$100	-	$100
Totals	$369		$369

Table 36: Bill of Material

10.3 Responsibilities

The responsibilities section provides a detailed breakdown of how each group member contributed to the project's hardware or software design. This individual breakdown has sub categories where the members have contributed to. They are research, design, testing, and other. The list that follows shows what each member of our group contributed (either cooperatively or individually) to the project.

Graham Goerg

Research:

- Single Board Computers
- Motion Sensors
- Identification Sensors
- Security Peripherals

Design:

- Single Board Computer
- Motion Sensor
- RFID Sensor
- Security Peripherals
- LEDs
- LCD

Testing:

- RFID Sensor
- Ultrasonic Sensor
- Camera
- Speaker
- LCD
- LEDs

Other:

- Hardware/Software Goals
- Requirements and Specifications
- House Of Quality
- Related Standards
- Design Constraints
- Milestones
- Budget and Finance

Joy Weaver

Research:

- Mobile Application
- Development Environment
- Programing Languages
- iOS Exclusive Development
- Android Exclusive Development
- Raspberry Pi OS

Design:

- Mobile App Architecture
- Android Studio
- Java Language
- Uses Case Diagram
- User Interface
- Class Diagram
- Security and Authentication

Testing:

- Add/ Edit/ Delete
- Push Notification
- Mobile App Camera Access
- Mobile App Audio Access
- User Interface and Gui

Other:
- Function
- Project Operation Manual
- House of Quality
- Similar Projects

Ryan Flynn

Research:
- LCDs
- LEDs
- Wifi Modules
- Databases
- Wireless Communication
- Version Control

Design:
- Class Diagram
- User Interface
- Mobile App Flowchart
- Database
- Wireless Communication

Testing:
- Register
- Login
- Mobile App Database Connection
- Raspberry Pi Database Connection
- Control System With Raspberry Pi
- Read Sensor Information
- Integration

Other:
- Project Description
- Motivation
- Goals and Objectives
- Software Prototype
- Conclusion

Michael Choi

Research:
- Power Supply

- Batteries
- AC Power
- PCB
- Motor
- Lock

Design:
- Power
- PCB
- Motor

Testing:
- Prototype
- PCB Bare Board Test
- Power
- Motor

Other:
- Executive Summary
- Requirement and Specifications
- Hardware Design Details

10.4 Project Conclusions

With the completion of this design paper, our group is set to start building and implementing the Pet Connect system. We started off by simply thinking of ideas for a project and ended up giving a solution to the problem pet owners face of letting their pet outside when they are away from the house. From there we were able to come up with the goals for this project and needed requirements that came along with them. This would propel us into the research section where the team could start learning more about what we need to use to make our project goals a reality.

We have researched many different aspects and options for the parts and tools we want to use. From a hardware standpoint we looked at motors, power, single board computers, locks, senors, LCDs, LEDs and wifi modules. After looking at numerous options for each of these aspects we were able to make well informed design decisions with each of them. The same process was done on the software side with the research of mobile platforms, development environments, development languages, databases, wireless communications and version control systems. Doing this research and taking into account the skills and experience of the development team made designing the software for the system a simpler task. These steps were necessary for both the hardware and software teams to help us start brainstorming for the design and get a better sense of the technologies that we will need to complete the intended functionality of our project.

The design process consisted of taking the parts, technology, and tools that we looked at in the research phase, and determining which are the best ones to use. We were then able to use these decisions to come up with plans of how we are going to implement them. This ranged from making schematics, flowcharts, and tables for the hardware components while we made more flowcharts, tables, class diagrams, wireframes, and prototypes for the software. While we may have wanted to jump right into making the product the design phase made us slowdown and think about the possibilities with each design decision. This is more efficient as we have the whole project planned out instead of starting to write code in the very beginning and running into roadblocks.

The remaining tasks included developing a plan for testing to ensure that our project will be working correctly. As seen in that section numerous test cases were developed for both the hardware, software and the integration of the two. The goals and requirements that we initially created were a major influence on these test cases as they will determine if our project is working correctly.

In the end, this has been a great experience for our group. We have been able to use the previous lessons from our respective programs to complete this design paper for our Senior Design project which will set us up for the second part of Senior Design. Coming from our different experiences we all had to learn how to work together, which is a great skill to have as we enter the industry. Some of the group members have not worked on a project of this scale before. This gave us a learning experience unlike any other class in our previous coursework. This was beneficial as the Senior Design class aims to create a work-like environment for students. The skills we gain from completing this class and this project will carry on to our careers after we graduate.

11.0 Appendices

In this appendices section we include a references section which consists of a list of the references used throughout this design paper. We then include a section for all of the usage permissions that we have obtained or are in the process of obtaining for the images used in our paper.

11.1 References

[AAA] *AA_Good 1 (Senior Design Paper)* Retrieved by July 25,2020. PDF:

[ADK] *Autodesk - PCB Basics for Electronics* Retrieved by July 27, 2020. URL: https://www.autodesk.com/products/eagle/blog/printed-circuit-boards-10000-feet-introduction-electronics-beginners/

[CAD] *Power Plane PCB: Best Practices for Power Planes in Multi-board Design* Retrieved by July 24, 2020 URL: Power Plane PCB: Best Practices for Power Planes in Multi-board Design

[DATA] dataplicity. *Stream live video from your Pi.* URL: https://docs.dataplicity.com/docs/stream-live-video-from-your-pi

[DEVL] Developers, *Android Studio*, Retrieved by June 13, 2020 URL: https://developer.android.com/studio

[EBOT] EBOOT. *100 Pieces Clear LED Light Emitting Diodes Bulb LED Lamp, 5 mm (Multicolor).*URL: https://www.amazon.com/Pieces-Clear-Light-Emitting-Diodes/dp/B06XPV4CSH?th=1

[EDGE] EDGELEC. *EDGELEC 200pcs 10 Colors x 20pcs 5mm LED Light Emitting Diode Assorted Kit 29mm Lead Clear Round Lamp White Red Blue Green Yellow UV Bright LEDs Bulb +300pcs Resistors for DC 6-12V Lights.* URL: https://www.amazon.com/EDGELEC-200pcs-Emitting-Assorted-Resistors/dp/B077X8P33G?th=1

[FIRE1] Firebase Realtime Database. *Structure Your Database.* URL: https://firebase.google.com/docs/database/android/structure-data

[FIRE2] Firebase Realtime Database. *Installation & Setup on Android.* URL: https://firebase.google.com/docs/database/android/start

[MYSQ] MySQL. *MySQL Workbench.* URL: https://www.mysql.com/products/workbench/

[PYRE] thisbejim. *Pyrebase.* URL: https://github.com/thisbejim/Pyrebase

[REAC] React Native, Environment Set up, Retrieved by June 16, 2020 URL: https://reactnative.dev/docs/environment-setup

[SOCK] Lucas PenzeyMoog. Understanding Socket Connections in Computer Networking. May 24, 2019. URL: https://medium.com/swlh/understanding-socket-connections-in-computer-networking-bac304812b5c

[SPAR] SparkFun. *SparkFun 16x2 SerLCD - Black on RGB 3.3V.* Retrieved by June, 16, 2020. URL: https://www.sparkfun.com/products/14072

[STAT] StatCounter Global Stats, Browser Market Share Worldwide, Retrieved by June 14,2020 URL: https://gs.statcounter.com/os-market-share/mobile/worldwide

[SUNF] Sunfounder. *SunFounder LCD1602 Module with 3.3V Backlight.* Retrieved by June 15, 2020 URL:https://www.sunfounder.com/lcd1602-module.html

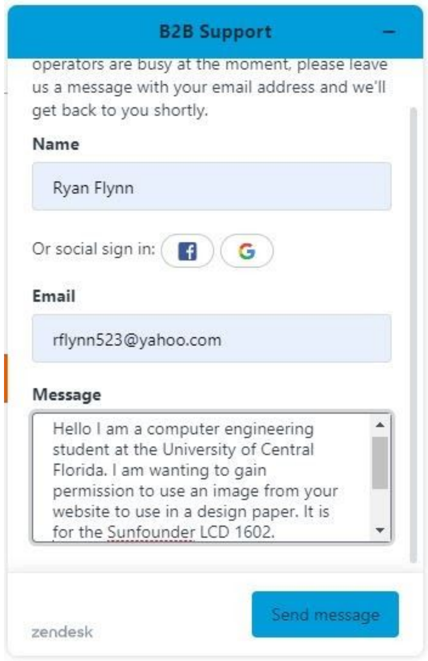
[UCTR] Uctronics. *UCTRONICS 0.96 Inch OLED Module 12864 128x64 Yellow Blue SSD1306 Driver I2C Serial Self-Luminous Display Board for Arduino Raspberry PI.*URL:https://www.uctronics.com/index.php/uctronics-0-96-inch-oled-module-1

2864-128x64-yellow-blue-ssd1306-driver-i2c-serial-self-luminous-display-board-for-arduino-raspberry-pi.html

[WEBF] WebFX, *Native App vs Mobile Web App: A quick comparison,* Retrieved by Jun 14, 2020 URL: https://www.webfx.com/blog/web-design/native-app-vs-mobile-web-app-coon/

11.2 Permissions

Figure 16: Sunfounder LCD 1602 Display
Status: Approved



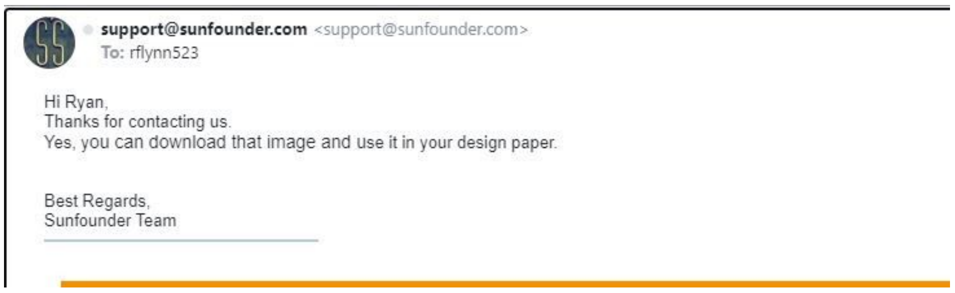
operators are busy at the moment, please leave us a message with your email address and we'll get back to you shortly.

Name: Ryan Flynn

Or social sign in:

Email: rflynn523@yahoo.com

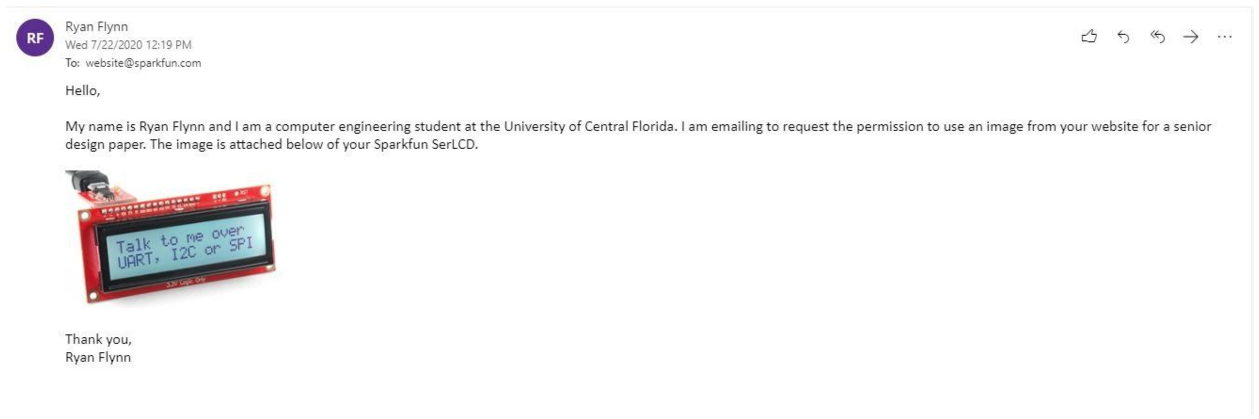
Message: Hello I am a computer engineering student at the University of Central Florida. I am wanting to gain permission to use an image from your website to use in a design paper. It is for the Sunfounder LCD 1602.


support@sunfounder.com <support@sunfounder.com>
To: rflynn523

Hi Ryan,
Thanks for contacting us.
Yes, you can download that image and use it in your design paper.

Best Regards,
Sunfounder Team

Figure 17: Sparkfun SerLCD
Status: Approved

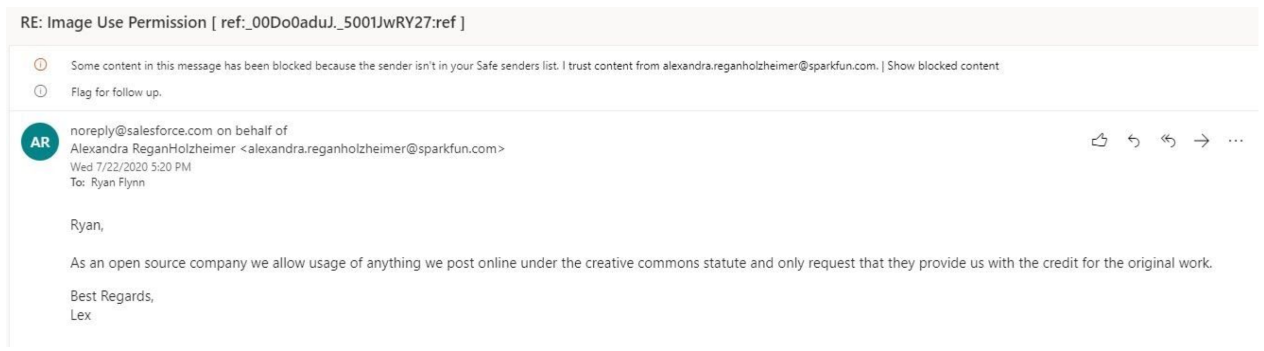


Ryan Flynn
Wed 7/22/2020 12:19 PM
To: website@sparkfun.com

Hello,

My name is Ryan Flynn and I am a computer engineering student at the University of Central Florida. I am emailing to request the permission to use an image from your website for a senior design paper. The image is attached below of your Sparkfun SerLCD.

Thank you,
Ryan Flynn



RE: Image Use Permission [ref:_00Do0aduJ._5001JwRY27:ref]

Some content in this message has been blocked because the sender isn't in your Safe senders list. I trust content from alexandra.reganholzheimer@sparkfun.com. | Show blocked content

Flag for follow up.

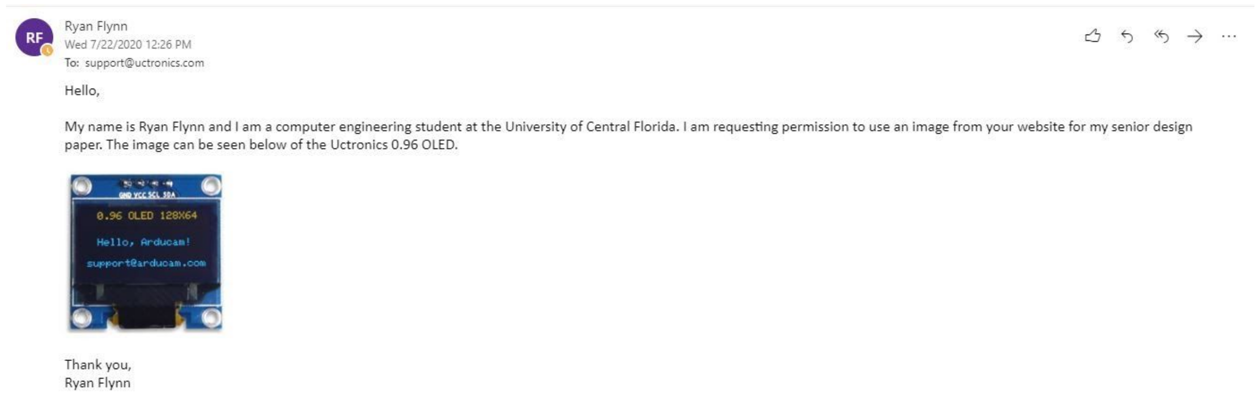
noreply@salesforce.com on behalf of
Alexandra ReganHolzheimer <alexandra.reganholzheimer@sparkfun.com>
Wed 7/22/2020 5:20 PM
To: Ryan Flynn

Ryan,

As an open source company we allow usage of anything we post online under the creative commons statute and only request that they provide us with the credit for the original work.

Best Regards,
Lex

Figure 18: Uctronics OLED Module
Status: Pending



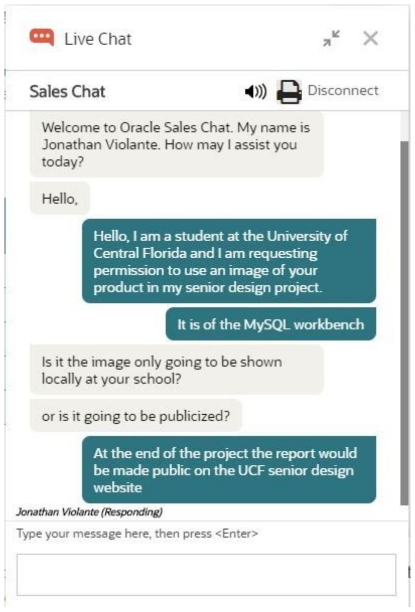
Ryan Flynn
Wed 7/22/2020 12:26 PM
To: support@uctronics.com

Hello,

My name is Ryan Flynn and I am a computer engineering student at the University of Central Florida. I am requesting permission to use an image from your website for my senior design paper. The image can be seen below of the Uctronics 0.96 OLED.

Thank you,
Ryan Flynn

Figure 20: MySQL Workbench Image
Status: Pending



Welcome to Oracle Sales Chat. My name is Jonathan Violante. How may I assist you today?

Hello,

Hello, I am a student at the University of Central Florida and I am requesting permission to use an image of your product in my senior design project.

It is of the MySQL workbench

Is it the image only going to be shown locally at your school?

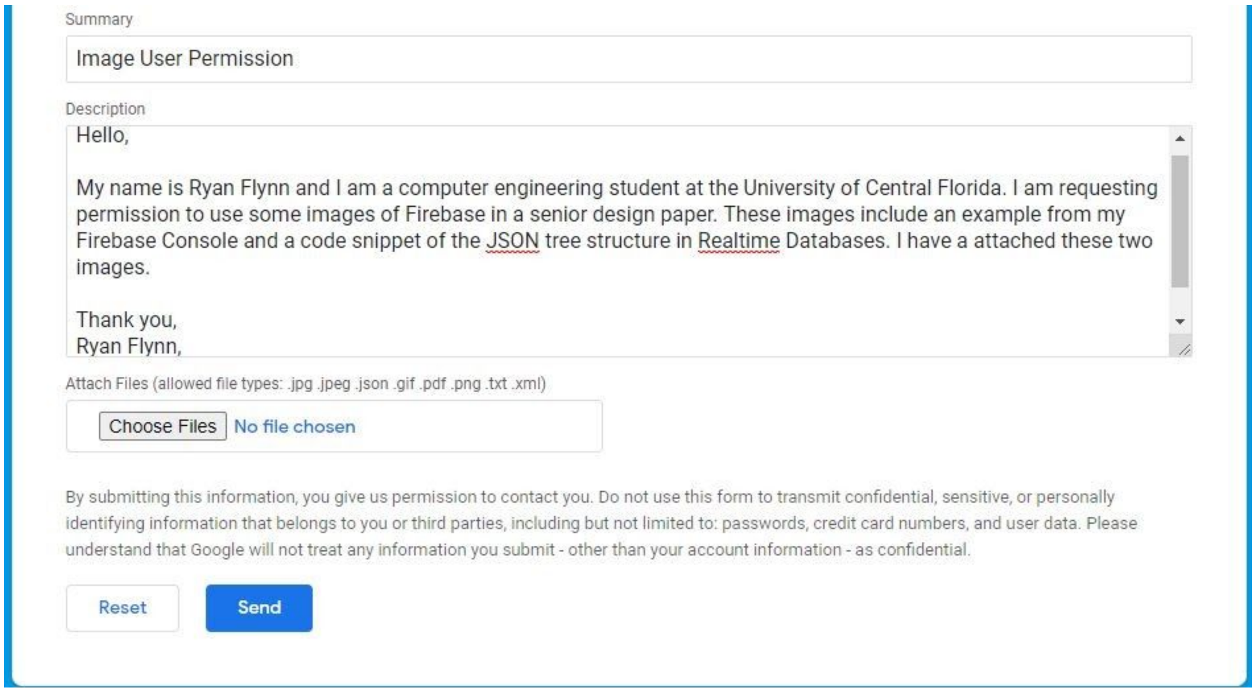
or is it going to be publicized?

At the end of the project the report would be made public on the UCF senior design website

Jonathan Violante (Responding)

Type your message here, then press <Enter>

Figure 21/22: Google Firebase Console Image and Code Snippet
Status: Approved



Summary

Image User Permission

Description

Hello,

My name is Ryan Flynn and I am a computer engineering student at the University of Central Florida. I am requesting permission to use some images of Firebase in a senior design paper. These images include an example from my Firebase Console and a code snippet of the JSON tree structure in Realtime Databases. I have a attached these two images.

Thank you,
Ryan Flynn,

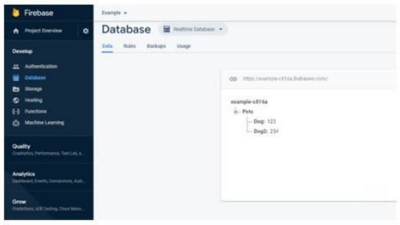
Attach Files (allowed file types: .jpg .jpeg .json .gif .pdf .png .txt .xml)

By submitting this information, you give us permission to contact you. Do not use this form to transmit confidential, sensitive, or personally identifying information that belongs to you or third parties, including but not limited to: passwords, credit card numbers, and user data. Please understand that Google will not treat any information you submit - other than your account information - as confidential.

Ryan Flynn <rflynn0523@gmail.com> Fri, Jul 24, 11:25 AM (2 days ago)
to Firebase

Thanks for getting back to me Jonathon,

I saw in that link people do need permission to use the Google logo so I just wanted to make sure that I had that permission to use these two images in my senior design paper.



```
{
  "users": {
    "alovelace": {
      "name": "Ada Lovelace",
      "contacts": { "ghopper": true },
    },
    "ghopper": { ... },
    "eclarke": { ... }
  }
}
```

Please let me know if these would be a problem

Thanks for the help,
Ryan Flynn

Firebase Support Jul 24, 2020, 8:51 PM (2 days ago)
to me

Hi Ryan.

No problem, feel free to use the images that you share with us in your paper.

Cheers
Jonathan

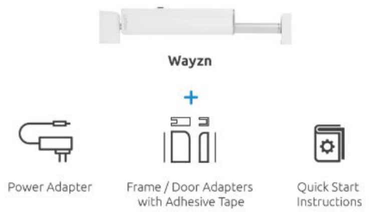
ref:_00D1Ux0Jq._5003p2QDUFE:ref

WAYZN Product Image
Figure 1: Similar product Wayzn
Status: Pending

From: vjoy01@yahoo.com
To: support@wayzn.com; Cc & Bcc

Permission to Use Picture

My name is Joy Weaver, I am computer Engineering student at the university of central Florida. I'm wanting to gain permission to use image from your website to use in a design paper. It is for image attach



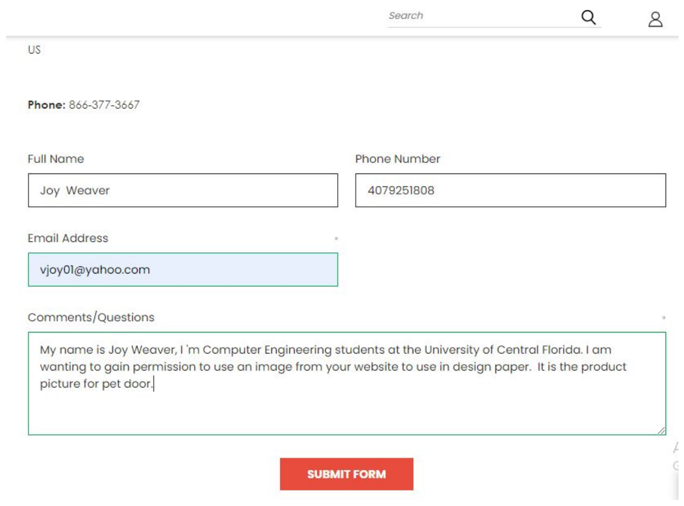
below.

Thank you,

Joy Weaver

Sent from Mail for Windows 10

Auto Slide Product Image
Figure 2: Similar product Autoslide
Status: Pending

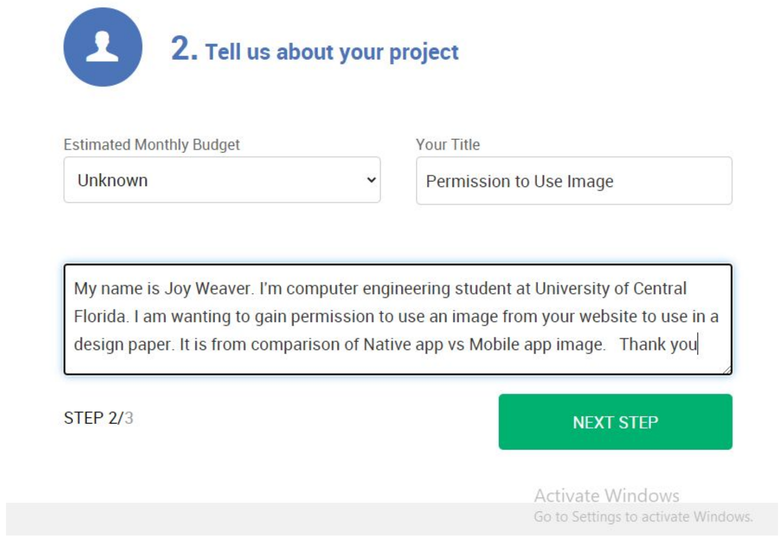

Search

US

Phone: 866-377-3667

Full Name: Joy Weaver
Phone Number: 4079251808
Email Address: vjoy01@yahoo.com
Comments/Questions: My name is Joy Weaver, I 'm Computer Engineering students at the University of Central Florida. I am wanting to gain permission to use an image from your website to use in design paper. It is the product picture for pet door.

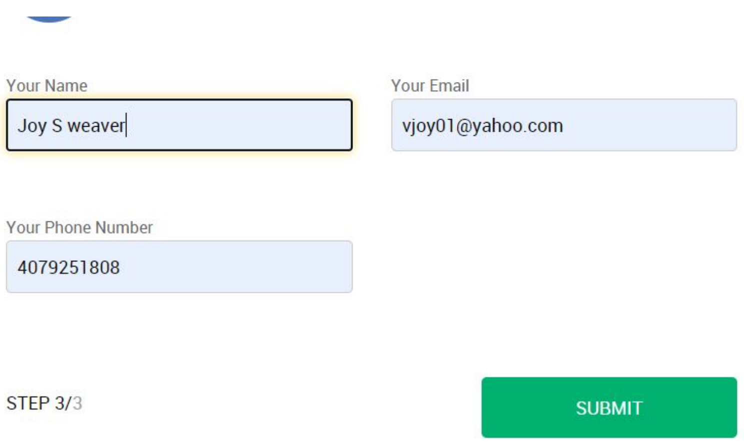
UI of Native and Mobile Web App Image
Figure 19: Compare UI develop from Native and Mobile Web App
Status: Pending


2. Tell us about your project

Estimated Monthly Budget: Unknown
Your Title: Permission to Use Image

My name is Joy Weaver. I'm computer engineering student at University of Central Florida. I am wanting to gain permission to use an image from your website to use in a design paper. It is from comparison of Native app vs Mobile app image. Thank you

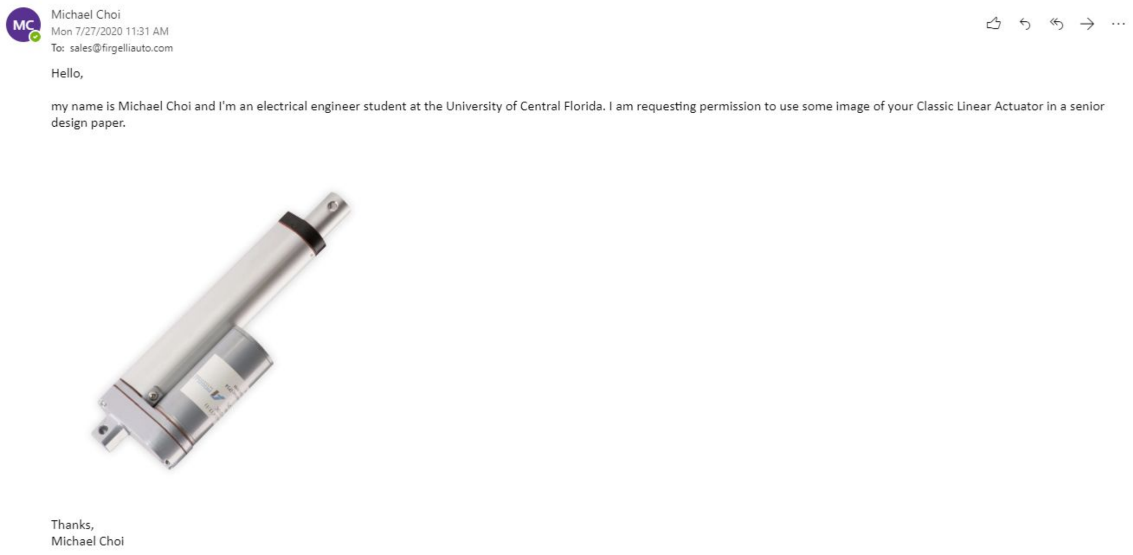
STEP 2/3



Your Name	Your Email
Joy S weaver	vjoy01@yahoo.com
Your Phone Number	
4079251808	

STEP 3/3

Firgelli Product Image
Figure 26: Classic Linear Actuator Permission
Status: Pending



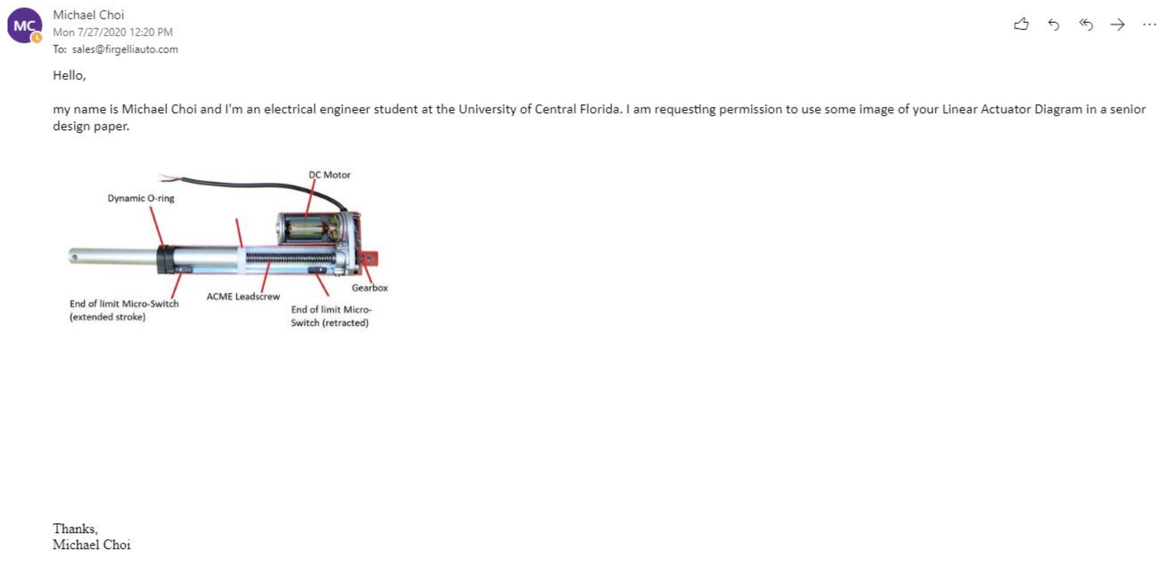
Michael Choi
Mon 7/27/2020 11:31 AM
To: sales@firgelliauto.com

Hello,

my name is Michael Choi and I'm an electrical engineer student at the University of Central Florida. I am requesting permission to use some image of your Classic Linear Actuator in a senior design paper.

Thanks,
Michael Choi

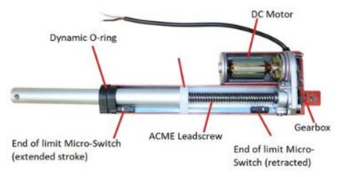
Firgelli Product Image
Figure 9: Linear Actuator
Status: Pending



MC Michael Choi
Mon 7/27/2020 12:20 PM
To: sales@firgelliauto.com

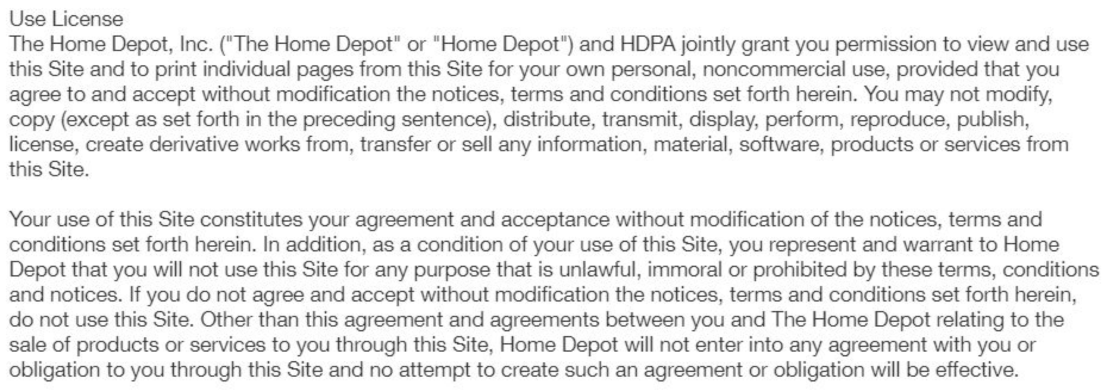
Hello,

my name is Michael Choi and I'm an electrical engineer student at the University of Central Florida. I am requesting permission to use some image of your Linear Actuator Diagram in a senior design paper.


DC Motor
Dynamic O-ring
Gearbox
End of limit Micro-Switch (extended stroke)
ACME Leadscrew
End of limit Micro-Switch (retracted)

Thanks,
Michael Choi

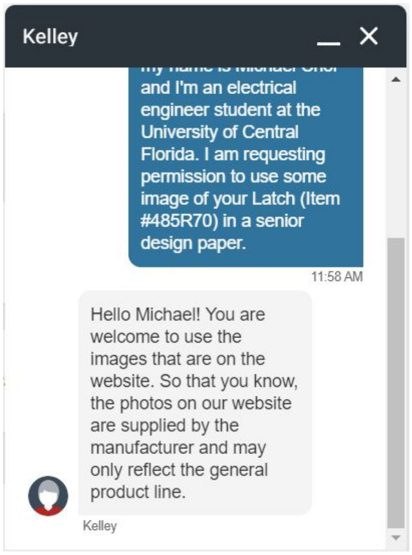
Home Depot - User License
Figure 11: Traditional Barn Hook Lock for Sliding Door
Status: Pending



Use License
The Home Depot, Inc. ("The Home Depot" or "Home Depot") and HDPA jointly grant you permission to view and use this Site and to print individual pages from this Site for your own personal, noncommercial use, provided that you agree to and accept without modification the notices, terms and conditions set forth herein. You may not modify, copy (except as set forth in the preceding sentence), distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer or sell any information, material, software, products or services from this Site.

Your use of this Site constitutes your agreement and acceptance without modification of the notices, terms and conditions set forth herein. In addition, as a condition of your use of this Site, you represent and warrant to Home Depot that you will not use this Site for any purpose that is unlawful, immoral or prohibited by these terms, conditions and notices. If you do not agree and accept without modification the notices, terms and conditions set forth herein, do not use this Site. Other than this agreement and agreements between you and The Home Depot relating to the sale of products or services to you through this Site, Home Depot will not enter into any agreement with you or obligation to you through this Site and no attempt to create such an agreement or obligation will be effective.

Grainger Product Image
Figure 10: Modern Hook Lock for Sliding Door
Status: Approved



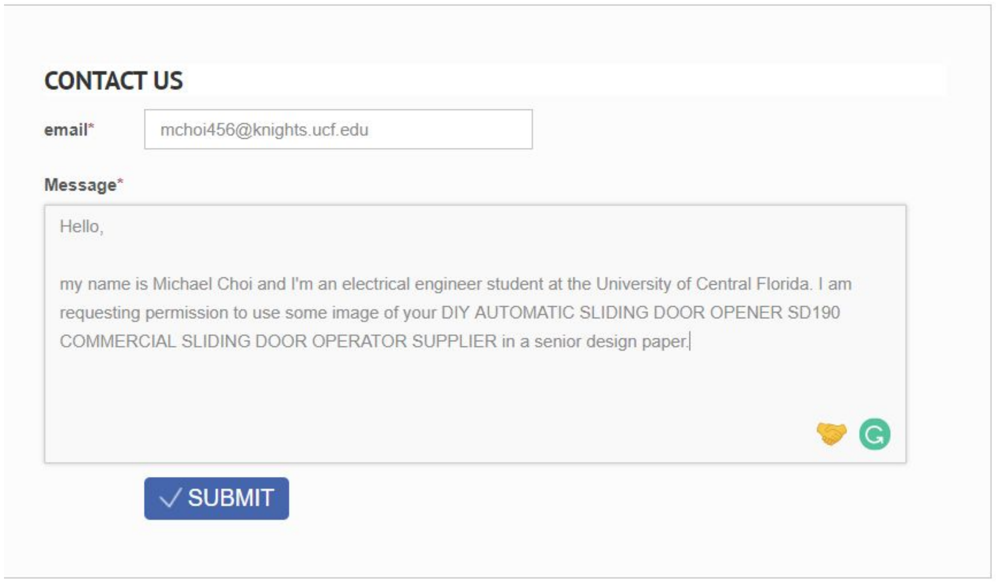
my name is Michael Choi and I'm an electrical engineer student at the University of Central Florida. I am requesting permission to use some image of your Latch (Item #485R70) in a senior design paper.

11:58 AM

Hello Michael! You are welcome to use the images that are on the website. So that you know, the photos on our website are supplied by the manufacturer and may only reflect the general product line.

Kelley

Olide Chinen Tech Product Image
Figure 8: Slide Automatic Sliding Door Opener
Status: Pending



CONTACT US

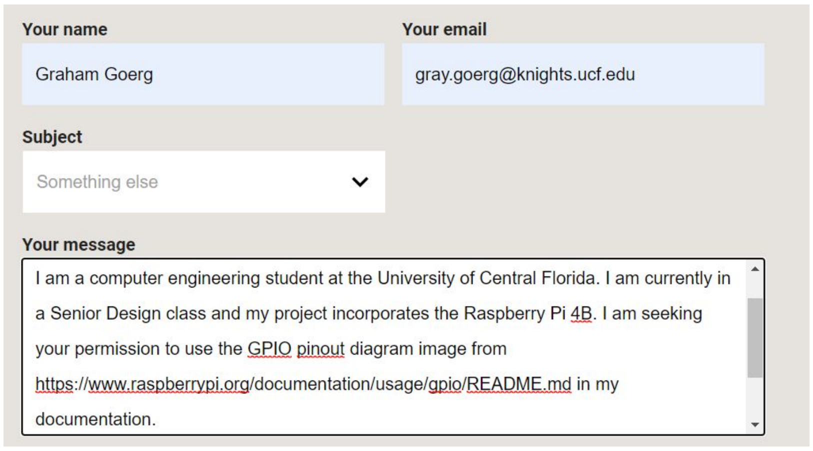
email* mchoi456@knights.ucf.edu

Message*

Hello,

my name is Michael Choi and I'm an electrical engineer student at the University of Central Florida. I am requesting permission to use some image of your DIY AUTOMATIC SLIDING DOOR OPENER SD190 COMMERCIAL SLIDING DOOR OPERATOR SUPPLIER in a senior design paper.

Raspberry Pi GPIO Pinout
Figure 4: Raspberry Pi 4B GPIO Pinout
Status: Approved

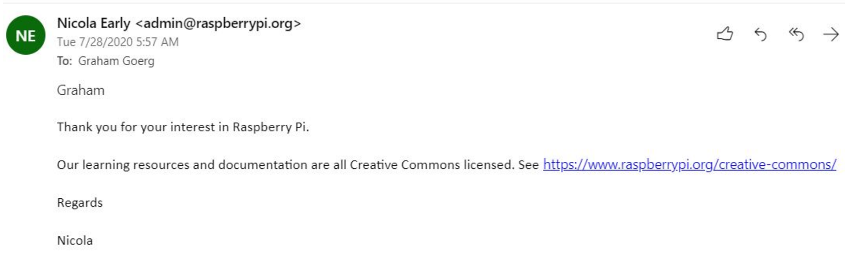


Your name
Graham Goerg

Your email
gray.goerg@knights.ucf.edu

Subject
Something else

Your message
I am a computer engineering student at the University of Central Florida. I am currently in a Senior Design class and my project incorporates the Raspberry Pi 4B. I am seeking your permission to use the GPIO pinout diagram image from https://www.raspberrypi.org/documentation/usage/gpio/README.md in my documentation.



Nicola Early <admin@raspberrypi.org>
Tue 7/28/2020 5:57 AM
To: Graham Goerg

Graham

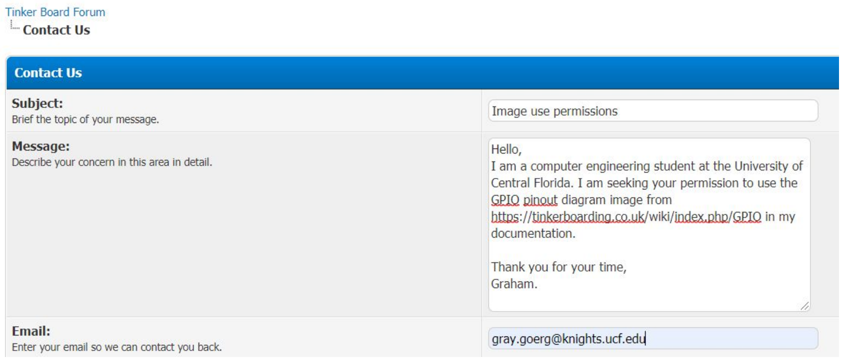
Thank you for your interest in Raspberry Pi.

Our learning resources and documentation are all Creative Commons licensed. See https://www.raspberrypi.org/creative-commons/

Regards

Nicola

ASUS Tinker Board GPIO
Figure 5: ASUS Tinker Board Pinout
Status: Pending

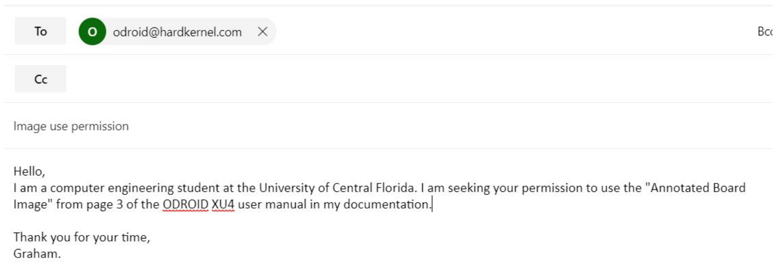


Tinker Board Forum
Contact Us

Contact Us

Subject: Brief the topic of your message.	Image use permissions
Message: Describe your concern in this area in detail.	Hello, I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the GPIO pinout diagram image from https://tinkerboarding.co.uk/wiki/index.php/GPIO in my documentation. Thank you for your time, Graham.
Email: Enter your email so we can contact you back.	gray.goerg@knights.ucf.edu

ODROID XU4 Board image
Figure 6: ODROID XU4 Board Layout
Status: Approved

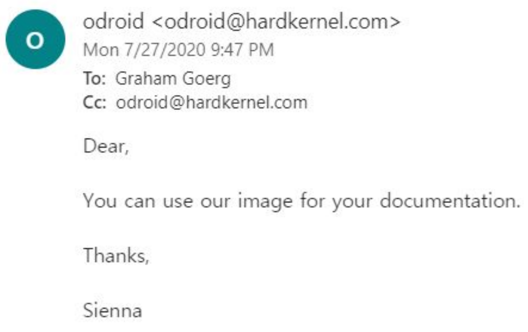


To odroid@hardkernel.com Bcc
Cc

Image use permission

Hello,
I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the "Annotated Board Image" from page 3 of the ODROID XU4 user manual in my documentation.

Thank you for your time,
Graham.



odroid <odroid@hardkernel.com>
Mon 7/27/2020 9:47 PM
To: Graham Goerg
Cc: odroid@hardkernel.com

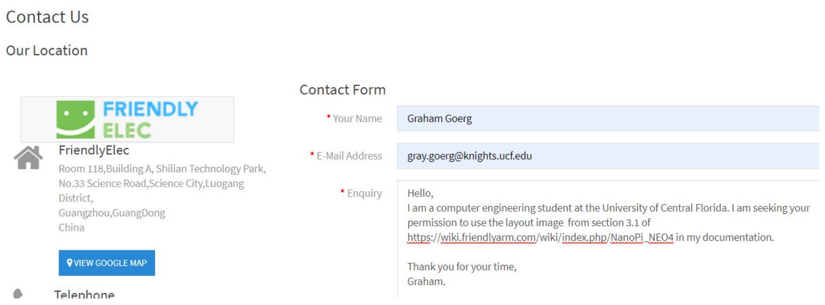
Dear,

You can use our image for your documentation.

Thanks,

Sienna

NanoPi NEO4
Figure 7: NanoPi NEO4 Board Layout
Status: Pending



Contact Us

Our Location

FriendlyElec
Room 118,Building A, Shilian Technology Park,
No.33 Science Road,Science City,Luogang
District,
Guangzhou,GuangDong
China

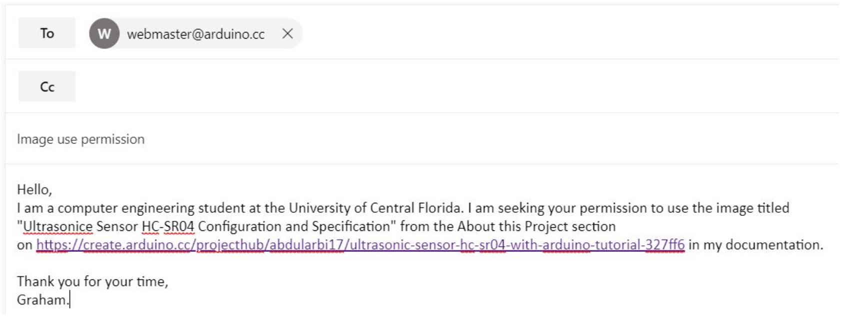
VIEW GOOGLE MAP

Telephone

Contact Form

* Your Name	Graham Goerg
* E-Mail Address	gray.goerg@knights.ucf.edu
* Enquiry	Hello, I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the layout image from section 3.1 of https://wiki.friendlyarm.com/wiki/index.php/NanoPi_NEO4 in my documentation. Thank you for your time, Graham.

HC-SR04 sensor
Figure 12: HC-SR04 Pinout
Status: Pending



To webmaster@arduino.cc
Cc

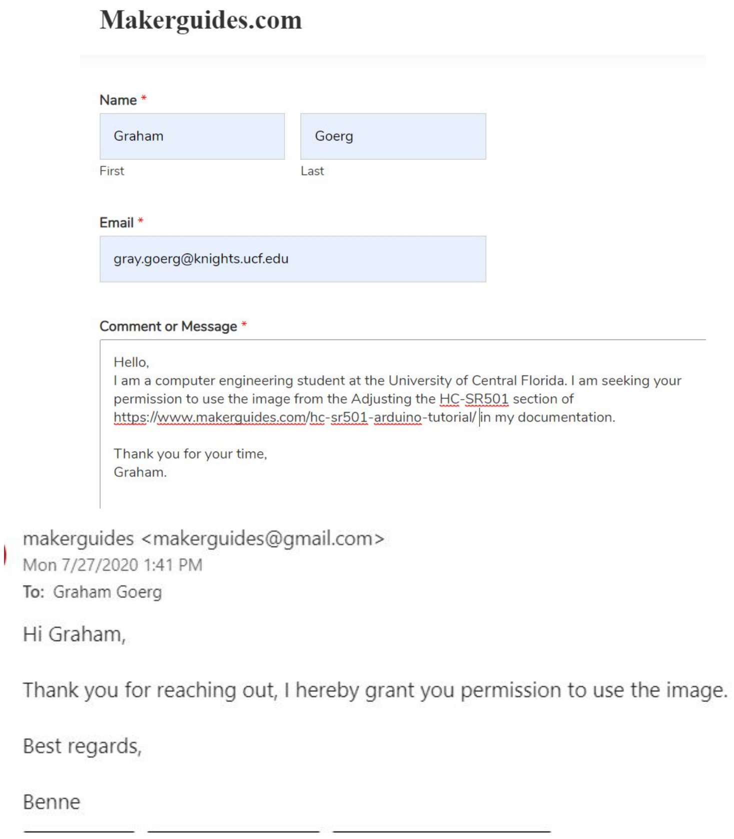
Image use permission

Hello,
I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the image titled "Ultrasonice Sensor HC-SR04 Configuration and Specification" from the About this Project section on https://create.arduino.cc/projecthub/abdularbi17/ultrasonic-sensor-hc-sr04-with-arduino-tutorial-327ff6 in my documentation.

Thank you for your time,
Graham.

HC-SR501 sensor

Figure 13: HC-SR501 Pinout
Status: Approved



Makerguides.com

Name *

Graham	Goerg
First	Last

Email *

gray.goerg@knights.ucf.edu

Comment or Message *

Hello,
I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the image from the Adjusting the HC-SR501 section of https://www.makerguides.com/hc-sr501-arduino-tutorial/ in my documentation.

Thank you for your time,
Graham.

makerguides <makerguides@gmail.com>
Mon 7/27/2020 1:41 PM
To: Graham Goerg

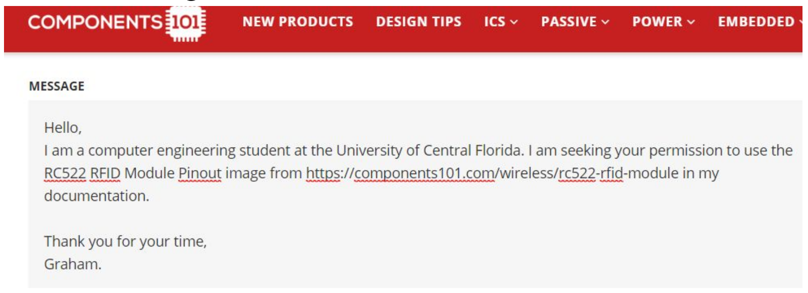
Hi Graham,

Thank you for reaching out, I hereby grant you permission to use the image.

Best regards,

Benne

RC522 module
Figure 14: RC522 RFID Module Pinout
Status: Pending

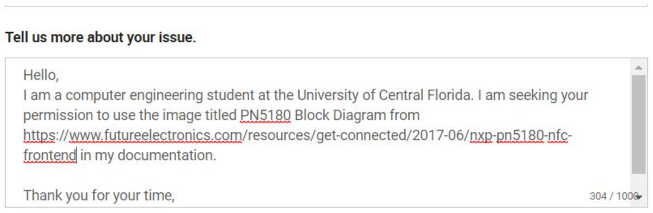


MESSAGE

Hello,
I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the RC522 RFID Module Pinout image from https://components101.com/wireless/rc522-rfid-module in my documentation.

Thank you for your time,
Graham.

PN5180 Block Diagram
Figure 15: PN5180 Block Diagram
Status: Approved


Tell us more about your issue.

Hello,
I am a computer engineering student at the University of Central Florida. I am seeking your permission to use the image titled PN5180 Block Diagram from https://www.futureelectronics.com/resources/get-connected/2017-06/nxp-pn5180-nfc-frontend in my documentation.

Thank you for your time,

Hi Graham,

Please see the response from my Content Manager:

The block diagram is owned by NXP and future is placing a visual representation on our website with their permission. Permission to use our image and place a citation of source is not an issue but Future would not be liable for the accuracy of the information, this would be NXP.

Thanks and have a nice day,
Brandon Hodge

Support Web / Web Support
Future Electronics
237, boul. Hymus
Pointe-Claire QC H9R 5C7
T: 514-694-7710 ext: 5036
Fax: 514-695-3707
www.futureelectronics.com