Door**Sure**:
The Garage Door Status Detector

University of Central Florida
Department of Electrical and Computer Engineering
Advised by Dr. Chung Yong Chan

EEL 4914 Senior Design I
Spring - Summer 2025

Reviewers
Professor Suboh Suboh
Professor George Atia
Professor Mark Llewellyn

Group 22

Mina Ghandour	Electrical Engineering
Felix Liu	Computer Engineering
Khang Nguyen	Electrical Engineering
Andrea Rosa	Computer Engineering

Table of Contents

Tables

Figures

Chapter 1. Executive Summary

Chapter 2. Project Description

In this section of the report, our team will properly introduce our Senior Design project idea. We will discuss project motivations, goals and objectives, and proposed system descriptions. We will further analyze our proposal by discussing engineering and marketing requirements and specifications, as well as detailing the hardware and software aspects for our proposed device.

2.1 Motivations

Imagine this: you are rushing out of your home, running late for work. In the chaos of the morning— getting dressed, making breakfast, grabbing your keys— you pull out of the driveway without a second thought. But by the time you make the drive and arrive at your workplace, a nagging thought crosses your mind– did you remember to close the garage door? In today's fast-paced world, this scenario has more than likely been a reality for the average person. Preoccupied with more pressing tasks, it becomes easy for one to forget something as important as closing the garage door before leaving home.

Leaving your garage door open is not just an inconvenience, it poses a serious security risk. An open garage door provides easy access for burglars, putting your home, valuables, and personal safety at risk. According to the 2019 FBI UCR: Crime in the United States, approximately 6.9 million property crimes were reported. Of these property crimes, 1.1 million were burglaries, resulting in losses of an estimated $3 billion in stolen valuables and direct property damage. More recently, 2024 reports from Securiteam indicate that burglary remains a major concern, with approximately 2.5 million burglaries occurring in the United States, resulting in an estimated $3.4 billion in losses. However, advancements in smart security technology and neighborhood surveillance systems have contributed to an overall declining trend in burglary rates over the years. Enhanced security measures— such as smart home devices, motion-activated cameras, and automated alerts—are making it harder for intruders to go undetected, reducing the likelihood of break-ins.

Beyond theft, however, an open door exposes your garage to the elements - rain, wind, or even wandering animals that could cause property damage. If you're away for hours, or even the entire day, these risks only increase. In

response to this problem, modern technology can be used to create smart home solutions that provide not only property security but also peace of mind. As engineers, we can create solutions by combining integration of sensors and real-time monitoring systems with software technology that will allow homeowners to remotely check the status of and receive alerts about their garage door, as well as giving said users the ability to interact with their garage door, ensuring their home remains secure even when they are miles away.

This project will serve as our first real immersion into the world of engineering design, allowing us to take an idea from concept to complete prototype. Unlike structured classroom labs, where solutions are often known before experimentation begins, this project will require us to identify and design a device that will resolve a real-world issue, research and iterate on our designs to create a functional and reliable system, and integrate a software application that will directly interact with the hardware. We will gain hands-on experience with key engineering principles such as circuit design, sensor integration, and software development, all while addressing a real-world problem that affects millions of people. This project is not just about building a functioning product—it's about developing and applying the concepts and technical skills we have learned throughout our academic career here at University of Central Florida. By combining hardware and software components, we will explore the intersection of electrical and computer engineering as a team, preparing us for more complex projects and team dynamics in our future careers. Ultimately, this project is an opportunity to gain valuable hands-on design experience while working with teammates of differing levels of skill sets to achieve a common goal: to transform an everyday inconvenience into an innovative, user-friendly solution that enhances security and convenience for homeowners everywhere.

2.2 Existing Products

In our research, our team noticed existing garage door status detectors tended to be directly linked to their corresponding garage door brand. One of the brands found in our research, the Chamberlain MyQ Smart Garage Door Sensor, is cost-effective but is only compatible with Chamberlain and LiftMaster garage doors. Moreover, Chamberlain offers an indoor security camera with motion detection that connects to their corresponding application. However, these two devices are sold separately. Expanding from this idea, our device will combine both features of a garage door status detector and a security camera, while also offering open/close door features via a servo motor with an MCU and having a live streaming camera module that can be placed anywhere the client wants and

connect seamlessly together with the web application. Chamberlain works with most garage door brands that use photoelectric sensors that do not shut off and it is not intended for use with openers in which the photoelectric sensors are located near the bottom of the garage door, all these requirements for the device to work properly which is a major inconvenience to customers.

Our project aims to combine the garage door status detector with the camera while aiming for a lower cost compared to what Chamberlain offers; for approximately 60 dollars combining both devices into one for our project will see lower costs. In addition, our project will aim to offer full compatibility with garage doors regardless of the year of manufacture, offering a peace of mind when buying our device.

Another device that has similar properties to our proposed device is the Genie's Aladdin Connect. The Aladdin connect can enable the use of smartphones to open and close any garage door after 1993, but this is a costly item, being over $50 to purchase a set of 2 garage control keys. Although this product offers control keys, it does not offer the camera addon, which would be a safety concern as there wouldn't be a live feed of the garage. Our proposed device will offer the live feed of the camera viewable at any and all times on the accompanying application.

An honorable mention is SwitchBot Smart Switch Button Pusher, an automated device designed to press physical buttons and switches. It attaches to the surface of the button, and a small arm presses the button when activated. The device primarily operates via Bluetooth, and the mobile app is available as well. The remote operation is achieved by pairing with the SwitchBot Hub, which connects to Wi-Fi. The SwitchBot is a popular choice for light switches, coffee makers, and other appliances, and this application makes it an easy solution for smart home devices without modifying existing hardware.

Unlike the SwitchBot, our garage status detecting device is specifically designed for garage door control, featuring real-time monitoring of the door's open/close state, which SwitchBot lacks. Also, while SwitchBot relies on Bluetooth or hub control, our solution will operate on Wi-Fi and Bluetooth, accessing without the need of additional hardware. Another key advantage of our project is object awareness, preventing the garage door from closing if an object is detected in the path of the garage door while open. Moreover, our device uses a replaceable battery instead of a rechargeable battery, which reduces downtime with frequent recharging. DoorSure ensures excellent compatibility and reliability concerning garage doors and home security compared to the more generalized approach of SwitchBot.

Our application aims to expand on the existing functions of the Chamberlain's device and the Genie's Aladdin, as well as incorporating the SwitchBot's actions to create a universally compatible device with existing garage doors, camera integration showing a live feed within our application, object detection and identification, and real time notification updates. While recognizing that each product has their strengths and weaknesses, for example Chamberlain's product offer the garage door status detector with a camera but with a trade off on cost and compatibility with all garage doors, the Genie's Aladdin connect offers control keys with the mobile application connectivity but with a trade off of high cost and no camera integration, we aim to find a happy medium between cost and extra components such as camera integration. Using these devices as a baseline, our proposed device will expand on these existing ideas to create a new cost effective, user friendly, and efficient device that will combine different aspects of the existing products into one clean cut alternative specific to garage doors.

2.3 Goals and Objectives

Basic Goals:

1. The user will have constant and accurate access to their garage door's current status.
2. The user will be able to alter the status of the garage door regardless of their location.
3. The user will be notified if the garage door status is altered from its current state.
4. The user will be unable to request a status change if there is an object in the way of the garage door.

Advance Goals:

1. The user will be able to view live camera feed from the device on the web application.

Stretch Goals:

1. The device will be able to accurately identify objects that are causing any potential blockage to the garage sensors, therefore not allowing the garage door status to be changed.
2. The user will be notified of the object's presence, along with the object's identification.

Objectives:

To ensure constant and accurate access to the garage door's status, our device will utilize a microcontroller to process sensor inputs and manage communications. A reed switch will be used to detect whether the garage door is fully open or fully closed, while a sensor component will measure the garage door's position in real time. This will improve our device's accuracy rate concerning the real time status of the door. Furthermore, the microcontroller's wireless capabilities will transmit collected status data to a database, allowing the user access to the data through our secure application at any and all times.

Throughout this process, LED indicators on our device will provide visual feedback on its operation, such as the power and wifi connectivity status. The device will be powered by a replaceable battery. When the battery voltage on any one of the device MCUs has been measured to be below a specific voltage level, the user will receive a notification stating the battery needs replacement.

In order to enable remote control functionality, our application will send information to and from the user's device via a remote networking service. Depending on the results of latency testing while prototyping, MCU bluetooth modules can be applied for local communications. Security features such as encrypted data transmission and user authentication will be implemented as needed to prevent unauthorized access to the data of the garage door.

The system will notify users whenever the garage door status changes, according to preferred notification configurations. The reed switch and object detecting sensor will send updates to the status to the database, triggering real-time notifications whenever movement changes are detected. Notifications will be sent as push, SMS or email.

To ensure safety and property protection, the system will prevent the garage door from operating if an obstruction is detected. The object detection sensor will continuously scan for objects in the path of the garage door when the garage door position is open. If an object is found to be in the path of the door, a function call will disable the servo motor from operating and send an alert to the user of the found obstruction. LED indicators will visually indicate the blockage on the device itself. This detection system will be optimized for accuracy to minimize false positives and increase accuracy of the device during the testing phase.

The camera component will be used to capture and stream live footage at a press of a button. The camera will communicate with one of the microcontrollers, allowing the user access to this footage at all times through the

application. This will allow users to view a live video feed of their garage at any time through the web application.

As a stretch goal, the system will incorporate AI-based object recognition to identify specific obstructions preventing the garage door from closing. The camera module will capture images of the detected objects, and an AI model running on the microcontroller will classify them accordingly, such as persons, pets, bicycles, etc… For our smaller scale testing, we will focus on a smaller set of objects for classification. The system will then notify users of the identified obstruction, displaying relevant details in the application.

Although our device will connect directly to the user's home Wi-Fi network and utilize wireless communications, we may require further user authentication such as account creations to increase security of our device as needed. Although our device requires wireless connectivity to send any status alerts, it is important to add this level of security to not allow just anyone with access to the Wi-Fi control of the garage. This ensures that if an unauthorized user connects to the home Wi-Fi, they will not simply be able to enter the application and open the user's garage door. Depending on ongoing security concerns further into our project research, these choices can be further modified and built upon. User configurations such as customized door notifications and battery warning thresholds will be saved onto the microcontroller built-in non-volatile storage. These configurations can be altered by the user directly on our web application. As the project progresses, we will evaluate the need for additional external components, such as a microSD card, to expand storage capacity as necessary.

2.4 System Description

The hardware components for our device consist of the following bullet points. Please note that these components and possible alternatives based on potential costs and setbacks are further explored in Chapter 3 of this report.

- Microcontrollers
- Wireless Modules
- Camera Module
- LED Indicators
- Enclosure
- Sensor
    - Reed Switch
    - Ultrasonic/ IR sensor
- Power Source

    - Replaceable/ Rechargeable battery

The software aspect of this project will consist of an application, designed for direct user interactivity with the hardware. From the accompanying application, users will be able to:

- Receive garage status notifications:
    - The garage door has been opened
    - The garage door has been closed
    - Garage door left open after x amount of time elapsed
- Directly interact with the garage:
    - Fully close the garage door
    - Fully open garage door
- Receive low battery notification of device
- View live camera feed (advanced goal)
- Receive notifications on what object is causing an obstruction (stretch goal)

2.5.1 System Specifications

The system specifications indicate what the smart garage door system must achieve to meet customer needs and project goals. They ensure the system can detect, process, and communicate the user's garage door status in various environmental and operational conditions. We will be testing the highlighted requirements in our demo, and is how we will determine whether our proposed device is successful.

Table 2.5.1: System Requirements

Parameter	Target Value	Description
Garage Door Status Detection	≥ 95% Accuracy	Ensures sensor precision for detecting door position.
Notification Delay	≤ 2 seconds	Provides near-instant security alerts.
Obstacle Detection accuracy	≥ 90% Accuracy	Will not allow garage door to be closed if an object is obstructing the path

Parameter	Target Value	Description
Obstacle object identification accuracy	≥ 75% Accuracy	Will identify the object in the path of the garage door and notify the user of the obstruction (Stretch goal)
Door Opening time duration	≤ 15 seconds	Ensures the garage door opens within a reasonable timeframe for user convenience while maintaining smooth and safe operation.
Door Closure time duration	≤ 15 seconds	Ensures the garage door closes efficiently without unnecessary delays, enhancing security and usability.
Operating Voltage	3.3V - 5V	Ensures compatibility with embedded systems.
Overall Power Consumption	~2W - 6W	Total energy usage of the system
Unit Cost	$40-$60	Affordable, competitive cost
Battery Life	≥ 3 months	Extends operation without frequent replacements.
Installation Time	≤ 20 minutes	Ensures a user-friendly setup.
Operating Temp	-20°C to 60°C	Ensures reliable operation in various environments.
Humidity Tolerance	10% - 90% RH	Prevents failures due to humidity exposure.

2.5.2 Component Specifications

The component specifications define the specific hardware modules and capabilities, explaining the role of each component in the overall system. These

specifications ensure the smart garage system is compatible, reliable, and meets the technical demands of the project.

Table 2.5.2: Component Specifications

Component	Parameter	Value
Microcontroller	Clock rate	80 MHZ - 240 MHZ
Camera Module	Resolution	At least 720P
Power Source	Battery life	Last 3 month
Wi-Fi Bands	Bandwidth	2.4GHz, 5GHz
Wi-Fi Range	Coverage	≥ 30 meters indoors
Bluetooth Range	Coverage	≥ 10 meters indoors

2.5.3 House of Quality

The House of Quality diagram showcases our component and system specifications in terms of engineering and marketing requirements. Through the ABET lectures, our team selected specifications from the previous tables 2.5.1 and 2.5.2 to detail the most important specifications that will measure how successful our project is at the end of our Senior Design.

The House of Quality is constructed by first getting and prioritizing customer requirements such as the compatibility with all garage doors and then we would translate those requirements to corresponding engineering requirements which would be more technical related to the project, it can be seen as the way to meet those customer requirements, additionally, we can set product target related to the engineering requirement so that we can actually have a numerical target to measure how successful our project is.

The customer requirements and engineering requirements are the sets of data that will define the house of quality and our project in general, we will define the relationship between each customer requirement and each engineering requirement based on their correlation and whether it is positive or negative using the legend in the upper left corner based on the direction of desirability of each of the requirements. Finally the roof shape would define the correlation between the engineering requirements themselves.

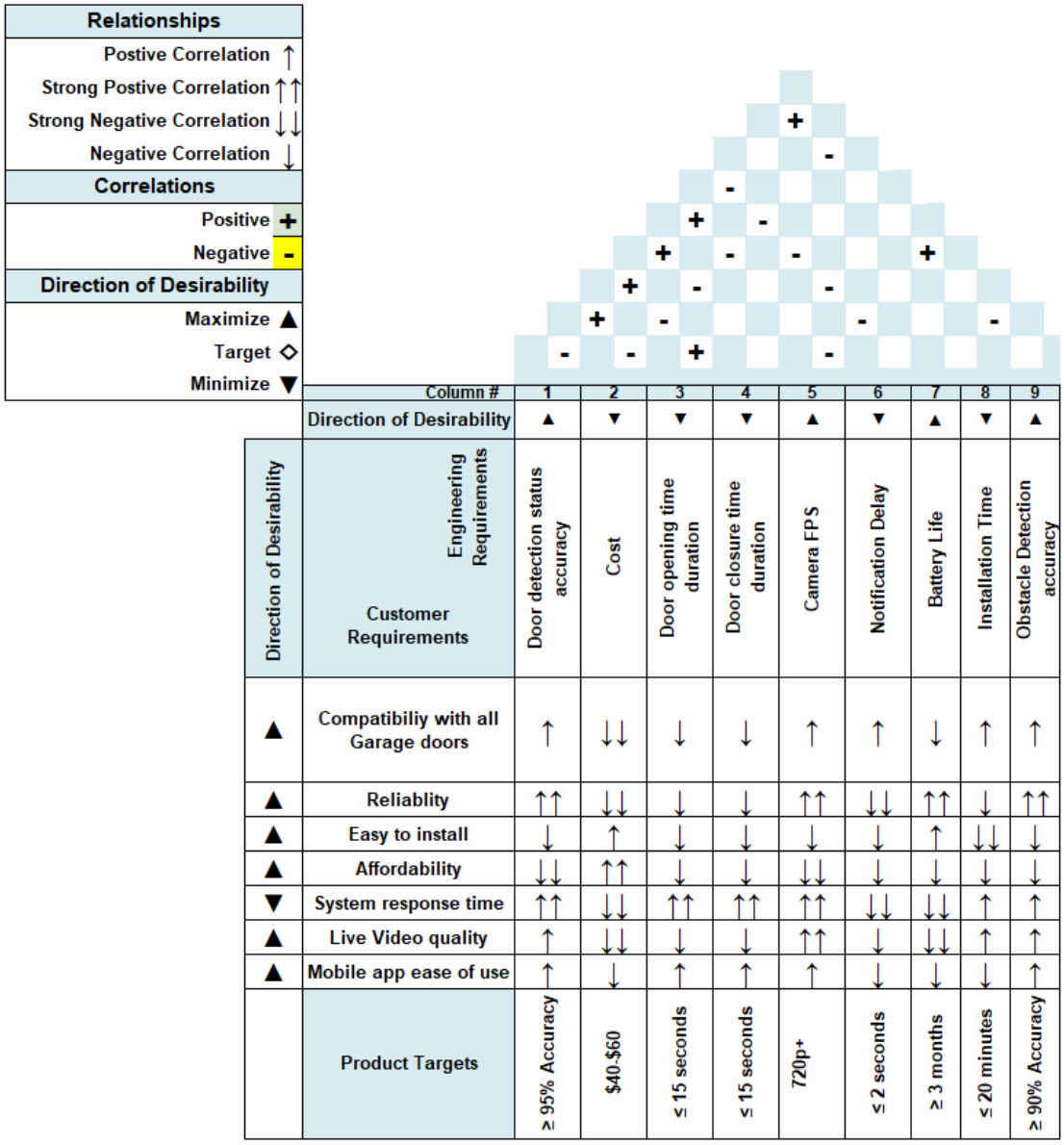


Figure 2.5.3: House of Quality

2.6.1 Hardware Block Diagram

Our device hardware requires having several peripherals to meet the goals of a successful PCB fabrication in the future of our course. Following our proposed system specifications table, in terms of operating temperature and humidity, our hardware will have a built in temperature sensor. This will help monitor the device in the garage, especially based here in Florida where humidity and temperature are constantly fluctuating and in high concentrations. Our device

will also feature a buzzer system, which will couple with LEDs to alert when the device has received an indication of data.

Note that our proposed hardware is split into two devices, one with the sensors for garage status and object detection, and another with the servo housing. The specifics are detailed below accordingly.

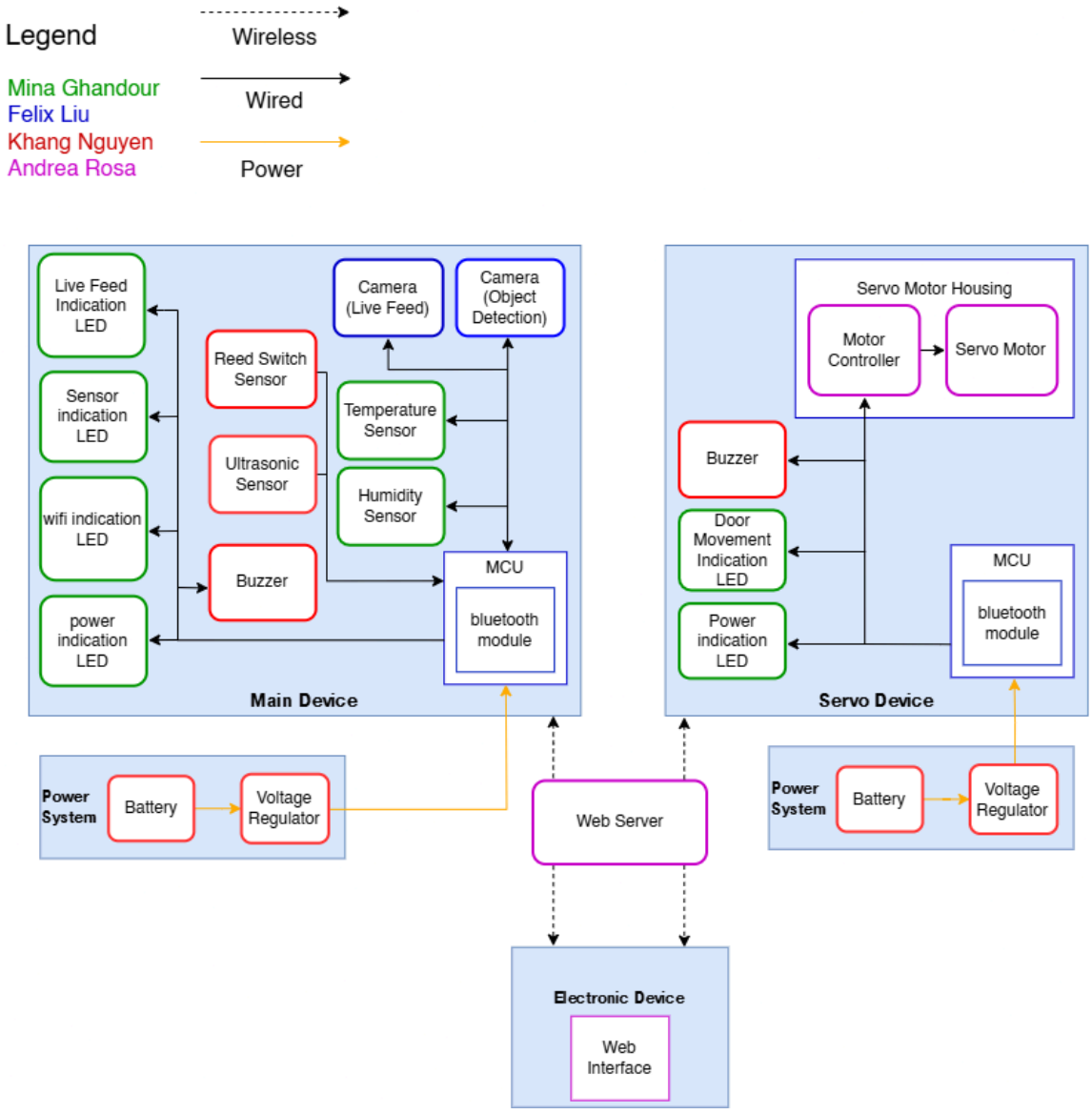


Figure 2.6.1: Hardware Block Diagram

2.6.2 System Flowchart Diagram

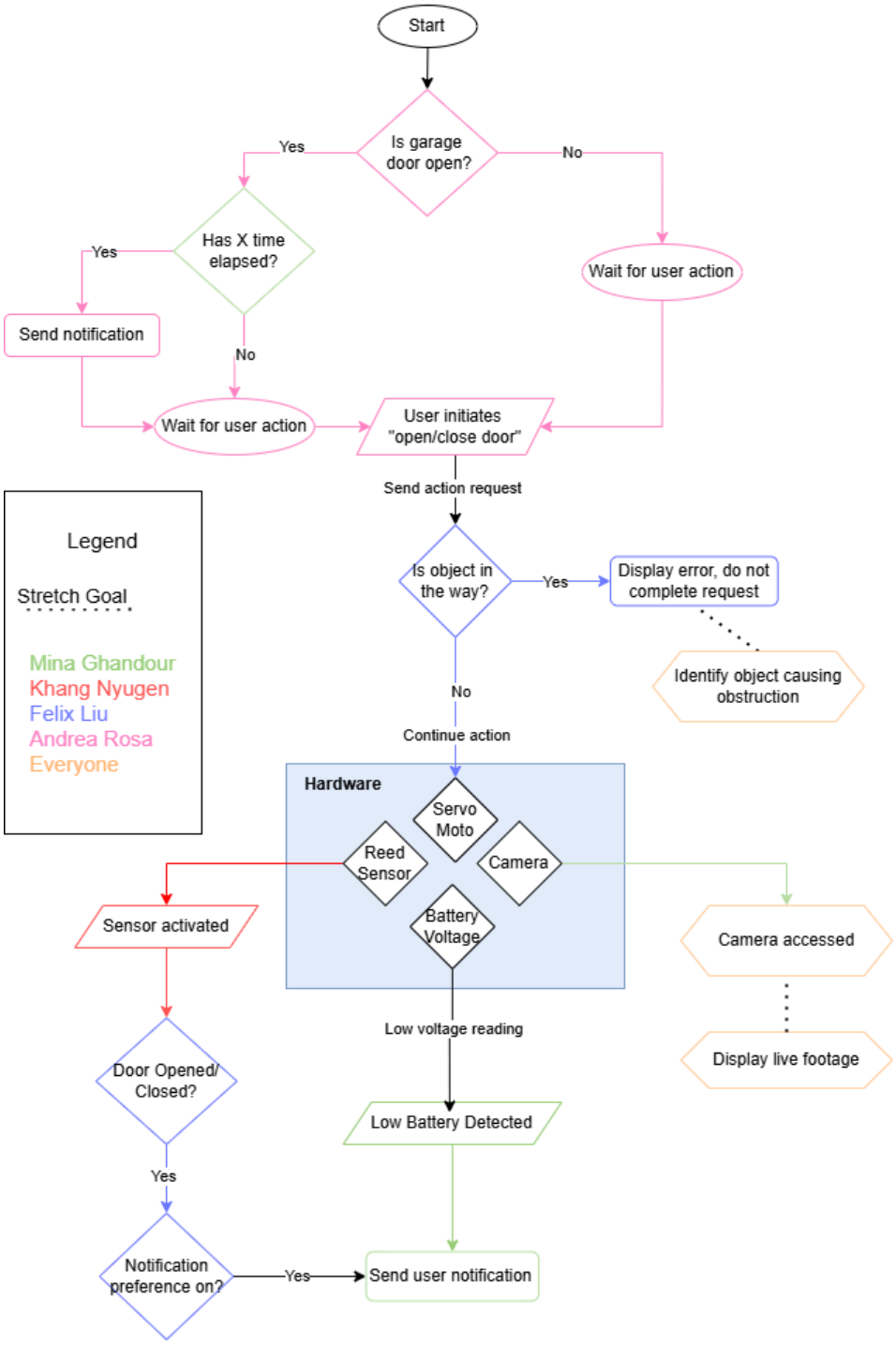


Figure 2.6.2: System Flowchart Diagram

Chapter 3. Research and Investigation

Extensive research is one of the most important aspects of project design. As discussed in Senior Design I ABET lectures, it is important for a project to be relevantly researched to avoid failure. In this section of the report, our team will be delving into different technological aspects of our project proposal, including both hardware and software comparisons. These comparisons are absolutely essential to our designing process as it not only allows the team to make the most informed decisions on hardware parts and software technology that will minimize overall costs and communication, but also maximize efficiency, performance, and reliability. By thoroughly evaluating different hardware and software options, our team aims to select the most optimal solutions.

For simplicity's sake, our selections will be highlighted in green on the table of comparisons throughout this chapter.

Hardware Comparison

In this section of the report, we will be comparing differing hardware parts and technological components that will be needed to build our device. We will be exploring our options with microcontrollers, batteries, voltage regulators, sensors, cameras, motor controllers, and servo motors. We will take into account their cost, power consumptions, and overall capabilities to ensure our device is as cost and software effective as possible to achieve our outlined engineering requirements.

3.1.1 MCU Technology Comparison

For our garage detector we will require the use of a microcontroller unit (MCU) as the devices themselves do not require any high complexity compared to using a field programmable grid array. As our devices will be battery powered, using the microcontrollers would be advantageous to use due to their low power consumption. Programming for mcu’s utilize simple software development languages like C++, python and other low and high level programming languages, allowing the time to configure each device will be of ease to do. (IBM) For the choices to compare which MCU we can potentially use we decided to pick components that have a development kit that has the respective MCU integrated already. This allows the process of prototyping the devices easy to configure and troubleshoot.

Table 3.1.1: Comparison between MCU and FPGA (IBM, 2024)

	MCU	FPGA
Hardware Structure	Highly Modular	Fixed
Processing	Parallel	Sequential
Power Consumption	Optimized for low power	High power consumption
Programming	Software Language (C++, Python, Java)	Hardware Language (HDL, Verilog)
Cost	Expensive	Cheap
Versatility	Very flexible; customization on hardware level	Applicable to broad range of applications; not as flexible compared to FPGA

3.1.2 MCU Part Selection

ESP8266

The ESP8266 is a low-cost microcontroller that integrates Wi-Fi capabilities and a large peripheral interface that will provide many external connections for our devices. The MCU contains a 32-bit CPU processor that can run on 80 to 120 MHz depending on clock speed needed.

ESP32

The successor to the ESP8266, the ESP32 is a dual core MCU containing Wi-Fi and Bluetooth features, unlike the ESP8266. All of the features of the 8266 are the same in the ESP32, but contain upgraded specifications such as having a maximum of 240 MHz clock speed and increased memory of 520 KB SRAM. These improved features still make this chip low-cost to purchase.

BCM2835

The BCM2385 processors can be found in the Raspberry Pi Zero series of microcontrollers. Due to its configuration as an ARM processor, the BCM2835 achieves good performance at a low cost.

RP2040

The RP2040 is an ARM processor that is mainly used in the Raspberry Pi Pico series of microcontrollers. The Pico W microcontroller allows usability of Wifi and Bluetooth features and performance is comparable to the ESP32 and 8266 processor chips.

RP2350

The RP2350 is an dual-core ARM processor that is mainly used in the Raspberry Pi Pico 2 series of microcontrollers. This is an improved version of the RP2040, giving higher performance due to the increased memory limitations.

Table 3.1.2: MCU Part Selection

MCU and Devkit	ESP8266 (NodeMCU)	ESP32 (ESP32 WROOM)	BCM2835 (Raspberry Pi Zero W)	RP2040 (Raspberry Pi Pico W)	RP2350 (Raspberry Pi Pico 2 W)
Price	$8	$9	$15	$4	$5
Wi-Fi	Yes	Yes	Yes	Yes	Yes
Bluetooth	No	Yes	Yes	Yes	Yes
Wireless connectivity built in MCU?	Yes	Yes	No	No	No
Clock Speed	160 MHz	240 MHz	1 GHz	133 MHz	150 MHz
Flash Memory	4 MB	4 MB	None	2 MB	4 MB
RAM	64 KB	520 KB	512 KB	264 KB	520 KB
Operating Voltage (V)	3.6V	3.6V	5V	5V	5V

3.2.1 Door Position Detection Technology

Door position detection provides critical information about the status of doors in different environments. In several applications, these sensors detect whether a door is open, closed, or partially open. Our focus is on excellent accuracy, low power consumption, and strong reliability. The goal is to build a door detection system that works efficiently and dependably in a home setting.

Reed Switch

Reed switches are a special kind of electrical switch, actuated (turned on and off or changed over) by magnetism. The most common type features two thin, flexible, ferromagnetic metal wires or blades - the reeds - positioned slightly apart in a hermetically sealed glass bubble. These function as the reed switch contacts. It should also be noted that the change-over type has three reeds instead of two. (RS-online, 2023)
These reed switches serve as electrical contacts that open or close when exposed to a magnetic field. In some cases, a third reed is present to create a changeover switch, which alternates between two circuits.

Unlike traditional mechanical switches that require direct physical movement, reed switches operate using magnetic forces. This means they can be in the device without direct access. Reed switches come in different types, each designed for specific uses. A normally open (NO) switch stays open until a magnet comes close, closing the circuit and allowing electricity to flow. A normally closed (NC) switch does the opposite—it remains closed, letting current pass until a magnetic field forces it open.

Reed switches have a lot of advantages over mechanical and semiconductor switches, making them a solid choice for many applications. First, they don't need physical contact to work—they use a magnetic field instead. That means there's way less wear and tear, so they last much longer. Second, they're also super energy-efficient because they barely use any power. Third, they're tough. Since they're sealed inside glass tubes, they can handle dust, moisture, and even corrosive environments without a problem, making them perfect for harsh conditions. Fourth, they don't create sparks when they switch on and off, making them a much safer choice in places where explosions or fires could be risky. Fifth, if you need something specific, no problem—reed switches come in all sorts of sizes and sensitivity levels so that they can be tailored to different applications. Best of all, they're built to last. Some reed switches can be used millions or even billions of times without failing, making them one of the most reliable options out there.

Reed switches also have disadvantages, such as the need for precise alignment between the magnet and sensor, which requires calibration. Also, it is limited to detecting only open/closed states(it cannot determine partial open). Last, it is affected by strong external magnetic fields, so the working environment is also a consideration.

Reed switches are used in various industries because they're versatile and reliable. One of the most common places you'll find them is in consumer electronics. For example, laptops and smartphones work as lid sensors—automatically turning off the screen when the lid is closed. They're also a key part of security systems. Many door and window alarms use reed switches as proximity sensors to detect when someone tries to enter without permission. They're a go-to choice for these kinds of applications because they're simple, durable, and highly responsive.

The automotive industry has made great use of reed switches, integrating them into fuel level sensors, anti-lock braking systems, and speedometers. Since they operate without physical contact, they're perfect for the harsh conditions inside a vehicle. Reed switches can be found in home appliances like refrigerators, washing machines, and dishwashers. They play a key role in safety and efficiency—ensuring these devices won't run if the door isn't closed correctly. This simple yet effective function helps prevent accidents and wasted energy.

In industrial and medical applications, reed switches play a key role in flow sensors, anemometers, and proximity detection systems. Its rugged construction makes it suitable for hazardous environments; dust, moisture, or chemicals may affect other types of switches. Reed switches are a go-to choice in lab equipment because they're precise and reliable. Since they don't need direct electrical contact to work, there's no risk of sparks, which makes them perfect for environments dealing with flammable or volatile materials. They're built to handle strict conditions while staying accurate, so they're trusted in many lab setups.

A reed switch is a magnetic switch that works without physical contact and activates when a magnet comes close. Its sealed glass housing keeps out dust, moisture, and other environmental factors. Because it operates without mechanical contact, it won't suffer from wear and tear like a mechanical switch, making it a reliable choice for detecting whether a garage door is fully open or closed. Also, it is energy efficient and requires no power supply when not in use. Reed switches need precise alignment with the magnet to work correctly. We think The Reed switch is a good choice for our project.

Hall Effect

Hall effect sensors are magnetic sensors that detect a magnetic field's strength, direction, and polarity. They work based on the Hall effect, a principle of electromagnetism discovered by Edwin Hall in 1879. (JU) The Hall effect describes how a charge carrier in a conductor or semiconductor behaves when exposed to a vertical magnetic field, resulting in an electric potential, often referred to as a Hall voltage. Hall effect sensors are becoming increasingly important in modern applications due to their non-contact operation, reliability, and ability to operate in harsh environments. These sensors are widely used in the automotive, industrial, consumer electronics, and energy sectors for position sensing, speed measurement, and current detection tasks. (Monolithic Power Systems, Inc.)

Hall Effect sensors provide several advantages over mechanical and optical sensors. First, they operate without physical contact, making them highly reliable over time. Second, these sensors allow real-time monitoring and control in high-speed applications. Third, their durability is another key benefit since they have stable performance even in harsh environments.

Though Hall effect sensors have many advantages, some drawbacks can affect their performance in some applications. First, the temperature change leads to less accuracy. To solve this problem, many Hall sensors require additional temperature compensation circuits, which increases complexity and cost. Second, magnetic interference could affect sensor reading as well. To minimize this interference, proper shielding and sensor placement are usually required. Third, Hall effect sensors require signal amplification for high-precision applications. They are also limited in detecting tiny magnetic fields, compared to fluxgate magnetometers or superconducting sensors, which makes them less suitable for high-sensitivity applications. Fourth, Hall sensors are excellent for non-contact sensing, but they are not ideally suited for applications that require direct contact force measurement because they can only detect magnetic field changes, not mechanical pressure or force.

Hall effect sensors are widely used in various industries due to their reliability,accuracy, and versatility in detecting magnetic fields. In the automotive industry, they play a vital role in engine and safety systems, such as crankshaft and camshaft position sensing for precise ignition timing, wheel speed detection in anti-lock braking systems (ABS), and steering position sensing in electric power steering (EPS) systems. In industrial Automation, Hell sensors are used in motion control, conveyor belt monitoring, and robot automation to ensure accurate and non-contact detection of mechanical motion. They are also common in consumer electronics, enabling features like laptop cover detection

to trigger sleep mode and smartphone clamps to lock or unlock the screen automatically. In the field of renewable energy, Hall sensors are used for current monitoring and energy optimization in solar and wind energy systems. In addition, the medical device integrates Hall-effect sensors for applications such as infusion pumps that monitor level and MRI scanners that detect position motion. In the field of power electronics, Hall effect sensors provide highly efficient non-contact current sensing for motor controllers, power inverters, and energy management systems. Hall-effect sensors can operate in harsh environments with high accuracy and durability, making them indispensable in modern electronics and industrial and automotive applications.

Hall effect sensors also operate based on magnets but provide continuous position tracking rather than simple on/off detection. These sensors provide real-time feedback on door position, which is helpful for applications that require step-by-step motion tracking or automatic adjustment based on the door position. They are low maintenance and durable. However, the Hall effect sensor requires an external power source. Also, they are more sensitive to electromagnetic interference (EMI), which can lead to unreliable readings in some environments. Hall effect sensors are ideal for advanced automation systems but not for basic garage door detection systems. Since our project only detects whether the garage door is open or closed, the additional complexity of the Hall effect sensor is not necessary.

Limit Switch

A limit switch is an electromechanical device that converts physical interaction into an electrical signal. They provide feedback to control machine operations, which are widely used. They change its contacts by either opening/closing state when external force is activated. They are commonly used in all kinds of applications.

The limit switch relies on mechanical actuation, where the physical displacement of the actuator triggers a change in the electrical connection. There are three main electrical configurations: normally open (NO), normally closed (NC), and double-pole double-throw (DPDT). A normally open switch remains open until activated, while a normally closed switch remains closed until triggered. The DPDT configuration combines NO and NC contacts to allow for multi-circuit control. Most limit switches contain a spring mechanism that returns the actuator to its original position after the force is eliminated to ensure accurate and repeatable operation.

There are many advantages of the limit switch. First, its stable performance over long periods of time. Second, they are highly resistant to environmental factors.

Third, they are cheap and easy to integrate into programmable logic controllers (PLCS), making them ideal for automation and control systems. Fourth, they consume no power in the passive state and are very energy efficient. Limit switches are available in many configurations for applications that require precision and control.

Limit switches are helpful, but they do have some downsides. Since they rely on physical contact to work, they wear out over time and may need maintenance or replacement. They also have a relatively slow response time, so they're not the best choice for detecting fast-moving objects. Environmental factors like dust, moisture, and temperature changes can affect how well they perform. Plus, they have fixed activation points, which might not work for projects that need more flexibility.

Limit switches are popular in automation, safety systems, and everyday devices because they're simple, reliable, and affordable. They're great for machine control and safety applications since they can detect movement and turn it into an electrical signal. While they have some limitations, improvements in materials and design have made them more durable and efficient over time.

Limit switches are mechanical sensors that are different than both Reed switches and Hall effect sensors, which require physical contact to activate. On the upper side, they are incredibly durable and an excellent choice for industrial applications. The limit switch is a good choice for garage door projects because it provides suitable door position sensing. However, unlike Reed switches and Hall effect sensors, the Limit switches rely on direct mechanical contact. They wear out over time and may require maintenance. Also, misalignment between the door and actuator can result in inconsistencies triggering. Given the project's reliability, limit switches are not the best solution for garage door systems.

Table 3.2.1: Door Position Detection Sensors

Sensor Type	Description	Pros	Cons
Reed Switch	Use a magnet to detect open/closed states	-Low cost -Low power draw -Reliable	-Need precise alignment -Only detect open/close
Hall Effect Sensor	Detects magnetic field changes	-No wear and tear -More precise	- High power draw

			-Requires calibration
Limit Switch	Mechanical switch that detects physical contact	-Simple and reliable -Low cost	-Wear over time

3.2.2 Door Detection Sensor Selection

As the device will be required to detect the status of the garage door and if there are any interferences, sensors will be needed to have the system to be fully operational. The devices will require many sensors to be implemented so it is important to see which sensors would be feasible for the project. We will be analyzing sensors that have proximity capabilities and sensors that activate when objects interact with them.

Reed Switch Sensor

Reed Switches are electromechanical devices that contain two ferromagnetic plates that interact with each other when a magnet is applied in their area. When a magnetic field is present the plates close on each other, inducing electricity as a result. Depending on the situation applied, the behavior of reed switches can trigger certain events. In our case, when the magnet attached to the garage door goes into the range of the reed switch, the device will communicate to the electronic device saying that the door is closed; the door status would be open if the magnet is out of the device's range. Due to there being no physical contact between the magnet and the reed switch, the lifespan of the switch is very long.

Hall Effect Sensor

Like reed switches, hall effect sensors use electromagnetism to provide electrical signals to a circuit. Hall effects rely on the existence of magnetic fields to induce an output, meaning that depending on the polarity of the magnet the behavior will vary on the hall effect switch. They can either be unipolar, which only uses one polarity or bipolar, which uses both polarities of the magnetic field. Hall effects have a long lifespan like reed switches due to having no physical contact to produce their output. In order for the hall effects to detect the magnetic field there must be a current applied to the system, which can be detrimental for systems that require low power usage.

Limit Switch Sensor

Limit switches are electromechanical switches that activate when a physical force is applied to an actuator. In the case of using limit switches as an application for our garage door status, when the garage door closes a physical part of the door comes into contact with the actuator, allowing the circuit to activate and notify the user that the garage door is closed. Although this is simple and easy to install, due to the physical contact the switch will gradually wear down, compared to devices that use only light or magnets to activate, which will virtually not degrade at all.

Ultrasonic Sensor

Ultrasonic sensors are a proximity sensor that uses ultrasonic vibrations to detect the presence of objects and calculate the distance from them. Because ultrasonic waves are a high frequency sound, they do not operate with color in mind. Their independence of color allows their measurement accuracy to be unaffected by transparent and translucent materials. Furthermore, they can be adapted to detect objects from a wide range. Performance depends on conditions of the location as temperature and humidity can affect sound; if the interior of the garage is either too cold or hot, sound might travel differently compared to normal conditions.

Infrared Sensor

Infrared (IR) sensors are proximity sensors that use infrared light to detect infrared radiation. Their high frequency does not allow IR to travel far distances unlike ultrasonic and they are very accurate when applied on a straight line. They are effectively the best if there is no visible light in the area, meaning that they should be applied in dark areas. Even if the main detecting device is improperly installed by a few degrees this can heavily affect the performance of the ir sensor.

LiDAR Sensor

Standing for Light Detection And Ranging, LiDAR uses laser beams to detect the distance and position of an object. The use of lasers allows greater accuracy and precision compared to using infrared and sound waves. The light that is being reflected off of surfaces returns to the lidar sensor for it to make proper time of flight measurements. These exact measurements require internal electronics to be implemented to calculate the real time systems. As such

implementing a LiDAR sensor will be very costly to install due to how advanced the technology is.

Motion Sensor

Motion sensors, as mentioned in their name, detect whether or not some kind of motion is in range to activate a response. There are three kinds of motion sensors that are used: Passive Infrared Sensors (PIR) sense a change in temperature from an oncoming object compared to the background temperature. Microwave sensors send out frequency waves and detect if there is a shift in frequency depending on the object that is moving. The third kind is combining both passive IR and microwave sensors to accommodate both strengths and also reduce the chance for any false alarms.

Camera Object Detection Sensor

Alternatively, instead of using external sensors to detect the presence of an object, the camera can be used to detect objects via the use of computer vision. There are different methods that can be applied to allow the camera lens to analyze and process the existence of physical objects, but they all do require an instance of deep learning to keep a database of what can be classified as an object.

Table 3.2.2: Comparison of Door Detection Sensor

Sensor	Type	Cost	Accuracy	Distance Range	Detection Activation type
Reed Switch	Status Detection	Cheap	High	N/A	Magnet
Hall Effect	Status Detection	Expensive	High	N/A	Magnetic Field
Limit Switch	Status Detection	Cheap	High	N/A	Mechanical
Ultrasonic	Object Presence	Cheap	High	Up to 10m	N/A
IR	Object	Cheap	Low	Up to 5m	N/A

	Presence				
LiDAR	Object Presence	Expensive	High	Up to 200m	N/A
Motion	Object Presence	Cheap	Medium	Up to 20m (PIR)	N/A
Computer Vision	Object Presence	Expensive (time consuming)	Medium	Determinant	N/A

3.3.1 Obstacle Detection Technology

The obstacle detection sensor is typical in automation, robotics, and safety systems. It helps detect objects in the path and measures distances using technologies like ultrasound, infrared red, laser, or camera AI. Each type has its own advantages and disadvantages when applied to different environments and scenes.

One of our goals is to feature the ability to monitor the garage door's clearance of the environment. Obstacle detection sensors warn the user when an object is detected within the garage door's path, letting them know there's something in the way. We need something that detects in real time, has good accuracy, and has low power consumption. When selecting an obstacle detection sensor, several parameters must be considered. One key requirement is reliability under different environmental conditions, such as light changes or environmental noise. Additionally, the sensor should have a detection range that is good enough for our needs while maintaining a balance between accuracy and power consumption. Low power consumption is important since the device is running on batteries, so there is no need to replace it frequently. Here are the sensor options we selected for this project

Ultrasonic Sensor

An ultrasonic sensor measures distance using sound wave technology. It uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect across boundaries to produce distinct echo patterns (MaxBotix Inc.).

Ultrasonic sensors work by echolocation, similar to how bats and dolphins navigate their surroundings. They have two main components: First, the transmitter emits ultrasonic waves, which travel through the air until they hit an object. Second, the receiver picks up when the waves bounce off the object. Then, the sensor calculates the distance based on how long it takes for the wave to return. This process keeps happening in real time, which means it is constantly measuring all the time.

Ultrasonic sensors have several advantages. One of the most significant is that they can operate under almost any lighting condition. This is due to their reliance on sound waves, unlike optical sensors, which are affected by bright or dark lights. Additionally, they do not require physical contact with the object. This reduces any wear and tear over time, which makes them reliable. Another advantage is that they can detect different materials and surfaces. Last, they have good accuracy, making them a reliable choice for this project.

Despite their advantages, ultrasonic sensors have some limitations. First, environmental factors like temperature, humidity, and air pressure can affect the accuracy of distance measurements. Second, they are limited to short or medium distances and are not suitable for long-distance detection. Third, multiple sensors may cause inaccurate readings. Fourth, they have a minimum detection distance. The result may not be accurate when the object is too close.

Ultrasonic sensors are widely used in modern life. They are often used in parking sensors to help drivers detect obstacles while backing up in automotive applications. They also help robots navigate their surroundings and avoid obstacles, such as vacuum robots at home.

Infrared Sensor

An IR sensor is an electronic device that detects IR radiation falling on it. Proximity sensors (used in touchscreen phones and edge-avoiding robots), contrast sensors (used in line following robots), and obstruction counters/sensors (used for counting goods and in burglar alarms) are some applications involving IR sensors. (CD-Team)

Infrared sensors work by emitting or detecting infrared radiation. There are two main types: active and passive infrared sensors. Active infrared sensors emit infrared light and measure an object's reflection to determine its presence or distance. Passive infrared sensors detect infrared radiation emitted by objects and are used for motion detection. The sensor triggers a response when an object moves in front of the sensor due to its infrared level changes.

There are many advantages of infrared sensors. First, they are low cost and easy to implement. They are inexpensive compared to other types of detection sensors, making them a popular choice for simple applications. Second, they have low power consumption, which is ideal for battery-powered devices. Third, they can provide fast object detection in short-range applications. Fourth, passive infrared sensors can detect motion without physical contact.

Despite their advantages, infrared sensors have some limitations. First, they are sensitive to environmental conditions. The result can be inaccurate when the environment is not ideal. Second, they can only detect short ranges, which limits their capability when it comes to further distances. Third, they can be affected by the heat source. High temperature can reduce their accuracy and may cause a false trigger.

Infrared sensors are widely used in modern life. They measure body temperature by infrared radiation from the skin in the healthcare system. Passive infrared sensors trigger an alarm when movement is detected in security systems. They are commonly found in remote controls, where the remote transmits signals to TVs.

Lidar Sensor

Light Detection And Ranging (LiDAR) is a laser-based remote sensing technology. The idea behind LiDAR is quite simple: point a small laser at a surface and measure the time it takes the laser to return to its source. (BEAUD)

Lidar sensors use laser beams to scan areas and measure distances. They emit rapid laser pulses, reflecting the surface and returning to the sensor. The system calculates how long each pulse can return into distance measurement. The lidar can build a detailed 3D map of its surroundings. Some are 360-degree fields of view, and others are fixed to specific areas.

There are several advantages to lidar sensors. They are highly accurate and work in both bright and dark environments. They provide precise distance measurements, such as detailed mapping and object detection. In addition, they detect objects at a distance, which is helpful for self-driving cars and drones. They can scan large areas quickly.

Although Lider sensors have many advantages, they also have some limitations. First, they cost much more than other sensors. Second, they require more power, making battery-powered devices suffer. Third, detecting certain surfaces,

such as reflective or transparent materials, can be difficult. Fourth, some weather conditions can interfere with laser pulses.

Lidar sensors are used in many industries due to their high accuracy and 3D mapping. They help detect objects and measure distances in self-driving cars. They create high-precision maps for a number of things. Robots can use LiDAR sensors for obstacle detection and navigation.

Camera with AI

The camera with AI is an advanced vision system that combines image capture with artificial intelligence to analyze and interpret visual data. Unlike traditional cameras, which only record images, it can recognize objects, track movement, and make real-time decisions based on what it sees.

AI cameras work by using machine learning algorithms to process the images or videos they capture. They are equipped with advanced software that can detect objects, recognize faces, recognize patterns, and even predict movements. Some AI cameras use deep learning models to improve their accuracy, learning from past data to improve their detection and classification capabilities. Depending on the application, these cameras can operate independently or in conjunction with cloud-based AI systems for greater processing power.

AI cameras have many advantages. One of the most significant is that they can analyze detailed visual data in real time and recognize different objects. They can provide more than just distance measurement. Also, they can be updated and trained through software, which can be improved over time.

Although AI cameras have many advantages, they also have some limitations. First, they require processing power, which can lead to higher power consumption and may require additional hardware such as GPUs. Second, they require images to analyze and train. Privacy could be a concern. Third, they may struggle in low or bright lighting conditions in which the image clarity is reduced.

AI cameras have been widely used in recent years. They have been used in security access, facial recognition, and so on. Also, they can help vehicles recognize traffic lights, obstacles and more. Even in healthcare, they assist in diagnosing medical treatments through image analysis.

Table 3.3.1 Obstacle Detection Technology

Sensor Type	Description	Pros	Cons

Ultrasonic Sensor	Measuring distance using sound waves	-Good accuracy -Works in many lighting conditions	-Affected by environmental noise -Power consuming
Infrared Sensor	Detects based on infrared reflection	-Low cost -Simple to implement	-Poor performance in bright light -Short range
Lidar Sensor	Laser to measure the distance	-High accuracy -Works in all lighting conditions	-Cost –Higher power consumption
Camera with AI	Processes images to detect object	-Versatile -More visual information	-Processing power -Privacy concerns

3.3.2 Obstacle Detection Selection

Since we chose the camera with artificial intelligence capability as our object detection mechanism after carefully considering other options making the camera is a central component in our garage door project since it would actively detect objects in the way of the garage door possibly a human or an animal so fast response would be crucial to ensure the safety of users and prevent any accidents or damage to property, additionally it would allow the user to view live feed to confirm that the garage door is closed which would be an additional step beside the reed switch, therefore, the camera will serve as object detection mechanism stopping the door if it detects an object in the way, ensuring safety and peace of mind of the user.

As safety is a key priority in this project, we will ensure to implement all standards and safety regulations applicable to our project to ensure reliability and security, since the camera will trigger an interrupt upon detecting an obstacle in the way of the garage door, we intended to do intensive testing to ensure no false triggers, balancing between reliability and cost will be key in choosing our camera development board with the correct camera module.

Camera development boards

First of all, we need to explain the difference between the two tables below, comparing camera equipped ESP32 development boards which would contain

an ESP32 microcontroller, PSRAM which would be significantly important for image processing ,USB to UART bridge (if needed) and most importantly the pre installed camera module such as the OV264 and individually camera modules which are either pre-installed or compatible with the development board.

Making that development board able to work independently as a minimum system, we recognize that each of the following choices have applications for them but we would looking for the development board and camera module that would be available to suit our project's needs while having a low cost so that we it would applicable to our case, moreover, if we chose to offer our product on the market, having a low cost that would help us lower the overall cost, moreover choosing a board that would be available to purchase more often would help us as well if we are trying to start producing our project in large numbers

Next, we would consider different ESP32 Cam development boards with different camera modules balancing between multiple parameters such as the cost and quality, and explaining the reason we chose the Development board we will be using in our project, which will be explained below:

Hiletgo ESP32-S

The Hiltego ESP32-S option would stand out as perfect choice for our project having the perfect balance between cost and capabilities that we need, since we would be needing more than one ESP32-S board for testing purposes and implementation, we would resort to the cheapest option, moreover it would meet all our requirements for the project such as 4MB of PSRAM and OV2640 camera module that would be enough to handle our requirements, moreover it is capable of supporting OV7670 but we chose the OV2640 so that we would lower the cost as the OV2640 already getting shipped with the development board which would give us an extra opportunity to save more money, moreover, this board would support multiple low power modes which is highly desirable since we would be operating on a battery and low frequency maintenance would a one of our goals, although the Hiltego ESP32-S would require a micro USB to Serial port CH340 Board but since it is already shipped with the development board and still maintaining being the cheapest option, therefore, we chose the Hiletgo ESP32-S with the OV2640 camera module as it is the cheapest, most available and library supported camera module.

ESP-EYE V2.1

The ESP-EYE would be a miniature ESP32-based development board that has an ESP32 dual core Tensilica LX6 processor with 8 MB PSRAM, coming

equipped with 2 LEDs, a MEMS Microphone and an OV2640 camera module which would be 2MB camera, the ESP-EYE generally is optimized for face and speech recognition application making it easy and simple for new electronics users to experiment with these features and possibly configure it, although this version would meet our criteria for the project but due to it's high price and unavailability, we didn't choose the ESP-EYE V2.1.

ESP-S3-EYE

The ESP-S3-EYE represents the next generation in the ESP32 camera boards, this would make this version a miniature AI development board optimized for image recognition and audio processing, this version would be based on the ESP32-S3 and ESP-WHO framework, ESP-WHO is an image processing development platform based on Espressif chips with many examples online human detection, face recognition and many other applications, The ESP-S3-Eye is perfect for heavy image processing application with plenty of memory of 8 MB octal PSRAM, 8 MB flash and a 2 MB camera to develop a wide range of AIOT applications, this board comes pre-equipped with an LCD display and a microphone which would be needed for image processing and audio processing, it also comes with default firmware which would be automatically loaded upon connecting the board to a power supply via a micro USB cable, then the LCD display will show the live video stream recognizing the human face of the user, due to the high cost and lower availability compared to the Hiltego ESP32-S, moreover, we won't need an LCD display showing the live stream therefore the Hiltego ESP32-S will be preferred.

T-Camera plus

The T-Camera plus based on the ESP-32-S3 microcontroller with Tensilica Xtensa dual core lx7 microprocessor with OV2640 camera module and also supporting OV5640, one key advantage this board has is the 1.3 LCD capacitive touch screen and a microphone, this would allow immediate visual feedback allowing users to monitor live camera feed without the need to integrate with a web application or an external screen, this feature would be incredibly useful in a variety of applications such as smart surveillance doorbells, although this development has more capabilities and offer more advanced features than the Hiltego ESP32-S such as the touch screen and the microphone, it would also bring some trade off in the software as it adds more complexity, moreover, it significantly increase power consumption, also, it would reflect in the elevated price point of $51.09, since we are integrating our project with a web app therefore we wouldn't be needing a screen, for testing purposes we may use an LCD screen which is significantly cheaper than the T-Camera Plus, therefore the Hiltego ESP32-S will be preferred.

AI Vision WonderCam

The AI vision camera module would be an unconventional choice boasting 8 built in functions including color, face, number and road sign recognition,vision line following, image classification and feature learning, this module can be used for people without extensive programming background since you can implement multiple AI features with one click training using a built in display and control key, moreover WonderCam's I2C connector makes it compatible with multiple MCUs including Arduino, Raspberry Pi and ESP32, also, the AI vision WonderCam supports programming using Arduino IDE while providing Arduino routines and source code so users can easily develop projects by themselves.

Despite it's highly sophisticated capabilities, LewanSoul (the brand behind WonderCam) focused on making their device as user friendly as possible by incorporating two lego compatible holes, this board would be perfect to empower hobbyists to integrate AI Vision functionabilities to various projects including lego based projects, although this board is for beginners but it would work for our project, but since it's cost of nearly $60, we chose the Hiltego ESP32-S as it has a much lower price.

XIAO ESP32 S3 Sense

The Xiao ESP32 S3 known for it's thumb sized compact design of 21 x 17.5 mm with light weight which makes it perfect for space limited projects such as a smart watch, equipped with a built in USB for ease of programming and debugging, moreover, it is provided with a detachable OV2640 and a digital microphone enhancing it's small size design, another key advantage of the Xiao ESP32 S3 board is the outstanding RF performance supporting 100 meters or more of remote communication when connected to U.FL antenna, the Xiao ESP32 S3 leverages a dual core ESP32 S3 chip supporting bluetooth and wifi connectivity, it also contains an external SD card slot for an external 32 GB of memory which make it suitable for AI driven applications by supplying massive amount of data storage and processing capabilities of 8 GB PSRAM, although, this board has numerous advantages, the increased capabilities while maintaining the small size results in high cost compared to the Hiltego ESP-S 32Cam.

Therefore, we will be choosing the Hiltego ESP-S 32-Cam development board as it achieves the perfect balance between low cost and capabilities needed for the project which are a high quality, highly library supported camera to ensure reliability.

Table 3.3.2: Camera Development Board Comparison

Camera development boards	Short description	PSRAM Size (MB)	Pre-installed module and compatible modules	USB-to-UART bridge	Cost
Hiletgo ESP-S32-CAM	WIFI+Bluetooth dual-mode development board uses PCB on-board antennas and cores based on ESP 32 chip	4	OV2640 (OV7670)	Needs a USB to Serial port adapter CH340	$ 9.25
ESP-Eye V2.1	WISOC ESP32 dual core Tensilica LX6 processor with WIFI and Bluetooth	8	OV2640	CP2102	$24.95
ESP-S3-EYE	Miniature AI development board with an LCD display	8	OV2640	CH340	$45.00
T-Camera Plus S3	T-Camera-Plus-S3 ESP32 with 1.3 inch touch LCD	8	OV2640 (OV5640)	CP2104	$51.09
AI vision WonderCam	AI vision boasting 8 built in functions including image	8	OV2640	CP2102	$59.99

	classification				
XIAO ESP 32 S3 Sense	ESP32 S3 Sense dual core Xtensa processor chip operating up to 240 MHZ	8	OV2640 (OV5640)	Built in USB	$23.99

3.3.3 Camera modules

Next, we will be comparing different camera modules and focus on the sensor specification and capabilities alone comparing different resolutions, Lens size, library support, power consumption and cost, different camera development boards are compatible with different camera modules but most camera development boards are compatible with OV2640 and prefer to use OV2640 than other camera modules.

OV2640

The OV2640 is the most used and widely supported camera module since all the development boards above have the OV2640 pre-installed, as it achieves the perfect balance between image quality, power consumption and cost. Having high image quality of 2 MP that it can be used for many AI application such as image recognition and in the context of our project object detection, moreover, having a balanced image resolution reflect in the power consumption, since we will be operating using a battery, power consumption is significantly important when choosing every component in our project to lower overall power consumption and as a result of that reduce maintenance needs, which would enhance system reliability, the OV2640 popularity would reflect in strong library support and resources that would help us integrate it into our project and a low cost compared to other camera modules, due to all the above reasons we chose the OV2640 over the other camera modules to implement in our project.

OV3640

The OV3640 is another camera module we considered, offering a higher resolution of 3 MP compared to the OV2640 which would result in better live feed and increased capabilities in terms of object detection, it also would lead to increased power consumption, therefore lowering the overall battery life, additionally, the OV3640 has a moderate library support which would impact

community support for this camera module, all these reasons will contribute for us choosing the OV2640 as it provides the optimal choice balancing between camera resolution and battery life.

OV5640

The OV5650 will be the second most used camera module as two of the above development boards mentioned explicitly that they are compatible with the OV5640, even though OV2640 remains their pre-installed option. Since the OV5640 have significantly higher resolution of 5 MP, that would result in higher capabilities and more AI applications, that would also result in increased power consumption lowering the batteries life, since the OV2640 is sufficient for most applications including our project, therefore it is more preferred compared to other modules, therefore lower library support and community support, the higher resolution will affect the cost as well costing moret than three times the OV2640, which will enforce our choice of choosing the OV2640.

OV7670

Another camera module that is used would be the OV7670 that would be supported by Hiletgo ESP-32-S Cam development board which is our choice, but since we implementing an AI based object detection application, the OV7670's limited resolution of 0.3 MP would negatively impact our object detection accuracy causing decreased reliability of our project, moreover the OV7670 offers lower library support and consequently lower community support causing difficulty when testing and implementing the camera module in our project.

Although the OV7670 is compatible with our choice of development board and lower cost of $2.10, nearly quarter the price of the OV2640, the OV2640 would be the superior and more suitable for our project.

OV7725

OV7725 would be an alternative camera module, providing a resolution of 0.3 MP with a camera lens of ¼" which would result in a slightly better low-light performance compared to the OV7670 with smaller lens of ⅛" , the limited resolution would heavily impact the reliability of the camera impacting our whole project, moreover, the OV7725 lack library and community support which would enforce our choice of choosing the OV2640 despite having a lower cost compared to the OV2640, since we would completely rely on the camera module object detection, a high resolution is needed.

NT99141

Lastly, The NT99141 would another potential choice for our camera module having higher resolution than OV7670 and OV7725 but lower than the OV2640, one significant disadvantage of this camera module is having an incredibly high price of $21.58 due to its lower popularity and reduced availability, additionally, limited library support will make the development and integration with the whole system more challenging.

Ultimately, we have decided to use the OV2640 since it would be the optimal choice and achieve the perfect balance between the library support and cost, and between image quality and power consumption. Additionally, our choice of the development board (Hiltego ESP32-S) includes an OV2640 even lowering the overall cost more.

Table 3.3.3: Camera module

Camera Module	Resolution (MP)	Lens size	Library support	Power requirement	Cost
OV2640	2	¼”	High (Most used and supported)	Moderate	$8.99
OV3640	3	¼”	Moderate	Moderate-High	$13.63
OV5640	5	¼”	Moderate	High	$30.09
OV7670	0.3	⅙”	Low	Low	$2.10
OV7725	0.3	¼”	Moderate	Low	$4.72
NT99141	1	¼”	Low	Moderate	$21.58

3.4.1 Temperature and Humidity Selection

In situations where the garage room reaches temperatures that will cause the DoorSure devices to be inoperable, the devices must make sure to alert the user that they are going to power down due to operating in conditions that will potentially damage the system. Both main and servo devices will require the use

of a sensor that can detect the humidity and temperature of the garage room. Although temperature makes sense to include, humidity is also part of the system in order to accommodate areas where the weather is tropical and humid. To reduce complexity in building our devices we decided to find components where they can detect both temperature and humidity in one package.

DHT11

The DHT11 is a ultra low cost sensor that can sense both temperature and humidity. Many devices that require low power usage can utilize this component. The ranges for temperature and humidity are suitable for most situations, unless they are used in very extreme conditions, but for home applications they are sufficient to be used.

DHT22

The DHT22, like the DHT11 can also sense temperature and humidity. The main difference is the upgraded specifications in the ranges as the DHT22 can capture temperature and humidity ranges beyond the values of the DHT11. Due to its prioritization on high precision the component is more accurate than the DHT11. These specifications will be listed on the table below for more details.

SHT7

Created by Sensiron, the SHT7 is a humidity and temperature sensor that uses internal signal processing to provide a fully calibrated and fair measurement. The high quality build of the SHT7 and advanced innovations used to record the values to sense the environment does make this component significantly expensive and over-capable in a household environment.

Table 3.4.1: Temperature and Humidity sensors

Sensor	DHT11	DHT22	SHT7
Temperature range (°C)	-20 - 60	-40 - 80	-40 - 123.8
Temperature Deviation (°C)	+-5	+-0.5	+-0.5
Humidity Range (RH)	5-95	0-100	0-100
Humidity Deviation	+-5%	+-2%	+-3%

(RH)			
Power Requirement (V)	Min 3.3V, Max 5V	5V	Min 2.4V, Max 5V
Cost	$5	$6.25	$30

3.5.1 Motor Controller Technology

With the implementation of our servo motor, this will require the use of a motor controller. Motor controllers are responsible for regulating operation of motors, making sure the right amount of speed, torque, direction and power is applied. Without the usage of a motor controller, the motor will have no control over its output, causing major inefficiencies with the system and potentially damaging the power systems of a device. To properly handle the operation of motors, motor controllers use voltage control to adjust speed, pulse width modulation to apply precise control over the motor's speed, changing polarity in order to control the direction of the motor, and regulating current to control the torque applied. There are four main types of motor controllers that can be implemented for our project.

DC Controllers

DC motor controllers mainly operate and control motors using direct current. Because of their ease of control with current they have great precision for controlling the amount of speed a motor should exert. They are best used when electronics require precise speed control over motor performance.

AC Controllers

AC on the other hand uses alternating current. The use of alternating current allows the motors to operate efficiently with minimal power consumption. AC controllers are usually equipped with variable frequency drives to control the speed and torque of a motor. They are usually found in large motors or contraptions that would consume energy if a DC motor controller was used.

Servo Controllers

Servo controllers are used in situations where accurate positioning is a priority, like in manufacturing. By sending pulse width modulation signals to the motor, the motor shaft can apply a specified angle, based on the information the user gives to the controller.

Stepper Controllers

Using incremental movements, stepper controllers can provide precise positioning. They are similar to servo controllers, but use a specific sequence of electrical signals to make incremental positioning, which is useful for applications that require accurate positioning.

Table 3.5.1: Comparison of Motor Controllers

Controller Type	DC	AC	Servo	Stepper
Cost	Expensive	Cheap	Expensive	Expensive
Efficiency	Poor	Accurate	Accurate	Accurate
Power delivery	Current	Current	Pulse Width Modulation	Pulse Width Modulation (?)

3.5.2 Motor Controller Servo Selection

In our system, the servo motor is responsible for the movement that will physically press the user's garage button to change the state of the door, either from open to closed or vice versa. Because of this, the primary factors we need to consider when selecting a servo motor are torque, weight, size. The servo must be lightweight and compact enough to attach securely to the user's garage button without adding unnecessary strain or obstructing other components. At the same time, it needs sufficient torque to reliably press the button without stalling, resulting in delays or unsuccessful completion of action, especially if the button requires significant force to activate. A servo in the 5-10 kg/cm torque range is likely ideal, depending on the resistance of the button. Based on how resistant the button is, we can increase voltage to the servo using a voltage regulator. Additionally, the mounting system must ensure stability, preventing the servo from shifting or misaligning after repeated use. Given that the servo will be used frequently, power efficiency and durability are also key factors -- metal gears improve longevity, while a low-power design would be beneficial for systems running on limited energy sources. We also want to factor in the speed of the servo motor, as we want to minimize delays as much as possible. Ultimately, we need a high-torque, low-profile servo that balances power, efficiency, and cost.

SG90

The SG90 Servo Motor by Beffkkip has an operating voltage range of 4.8 to 6.0V with an idle speed of .11 seconds per 60 degrees at 4.8V and .08 seconds at 6.0V. It rotates at an angle of 180 degrees which is more than ample for our action. Although being a super cost-effective option -- with a 4-pack costing $7.99 – this servo motor uses a mix of plastic and copper metal gears, which is not highly reliable. Depending on how much torque is needed to activate the button, having a plastic and metal gear mix increases the chances of a stall or delay in the action due to too much pressure. Furthermore, with frequent movements as expected from opening and closing the door might cause fast wear and tear.he SG90 Servo Motor by Beffkkip has an operating voltage range of 4.8 to 6.0V with an idle speed of .11 seconds per 60 degrees at 4.8V and .08 seconds at 6.0V. It rotates at an angle of 180 degrees which is more than ample for our action. Although being a super cost-effective option -- with a 4-pack costing $7.99 – this servo motor uses a mix of plastic and copper metal gears, which is not highly reliable. Depending on how much torque is needed to activate the button, having a plastic and metal gear mix increases the chances of a stall or delay in the action due to too much pressure. Furthermore, with frequent movements as expected from opening and closing the door might cause fast wear and tear.

MG90S

The MG90S Servo Motor by Miuzei has an operating voltage range of 4.8 to 6.0V with an idle speed of .1 seconds per 60 degrees at 4.8V and .08 seconds at 6.0V. According to the internal structure diagram provided on their product page, the servo also includes a CNC-process for high precision, making sure of no angular deviation. This is crucial for making sure the motor is able to hit the garage button precisely. Additionally, its full-metal gear construction enhances durability, reducing the risk of gear stripping over time. However, with a stall torque of only 2.2 kg/cm at 6.0V, it may struggle if the garage button requires more force to activate.

MG90D

The MG90D Servo Motor by HOOYIJ is an upgraded version of the previous MG90s. It has an operating voltage range of 4.8V to 6.0V, with a speed of .1 seconds per 60 degrees and .08 seconds per 60 degrees, respectively. The stall torque at 4.8V is 2.1kg/cm and at 6.0V is 2.4 kg/cm. It also has metal gears similar to its predecessor, providing increased durability and resistance to wear. Compared to the MG90S, the MG90D offers slightly higher torque, which can improve reliability when pressing the garage button. Its compact size and

precise movement make it a viable option if the required button force is low.

MG996R

The MG996R Servo Motor by Deegoo has an operating voltage range of 4.8V to 6.0V. It features a stall torque of 9.4 kg/cm at 4.8V and 11 kg/cm at 6.0V, making it significantly stronger than smaller servos like the MG90S or SG90. This increased torque ensures that even stiff or resistant garage door buttons can be pressed reliably without the risk of stalling. However, this servo is noticeably larger and heavier than its predecessors, weighing 55g with dimensions of 40.7 x 19.7 x 42.9 mm. While the added weight may pose mounting challenges, its full metal gears contribute to long-term durability, reducing wear over time from the expected operation frequencies. The operating speed is 0.17 seconds per 60 degrees at 4.8V and 0.14 seconds at 6.0V, which, while slower than smaller servos, is still fast enough for practical use for our garage door button system.

RDS3225

The RDS3225 Servo Motor by ANNIMOS has an operating voltage range of 5V to 6.8V with an operating speed of .16 seconds per 60 degrees at 5V and .14 seconds per 60 degrees at 6.8V. This servo motor has a lot of tradeoffs when compared to the previously mentioned servo motors. It requires higher voltage, is heavier and slightly slower, but exerts a significantly higher torque, not stalling until almost three times as much kg/cm as other motors. The metal gears place it on par with the MG90s in terms of durability but its heavy weight due to being fully metal gears might cause the device to be more difficult to place over the garage door button. The higher torque does, however, significantly decrease risk of not being able to exert enough pressure to press the button, therefore increasing input accuracy.

Table 3.4.2: Servo Motor Comparison

	SG90	MG90s	MG90D	MG996R	RDS3225
Stall Torque (kg/cm)	1.4 at 4.8V 1.9 at 6.0V	1.8 at 4.8V 2.2 at 6.0V	2.1 at 4.8V 2.4 at 6.0V	9.4 at 4.8V 11 at 6.0V	24 at 5.0V 28 at 6.8V
Speed (seconds/60 degrees)	0.11 at 4.8V	0.10 at 4.8V	0.10 at 4.8V	0.17 at 4.8V 0.14 at 6.0V	0.19 at 5V

	0.08 at 6.0V	0.08 at 6.0V	0.08 at 6.0V		0.14 at 6.8V
Gear Material	Plastic	Metal	Metal	Metal	Full metal
Weight (g)	9	13.4	13.5	55	60
Dimensions (mm)	22.2 x 11.8 x 31	22.4 x 12.1 x 28.5	22.8 x 12.2 x 28.5	40.7 x 19.7 x 42.9	40 x 20 x 40.5

3.6.1 Voltage Regulation Technology

These devices will require a voltage regulator to be implemented to provide the proper voltage to the components that require power and prevent any potential short circuit overloading. There are mainly two types of regulators, linear and switching regulators. Linear regulators maintain a desired voltage by stepping down the input voltage to the target voltage. Despite their lower price point and simple construction these regulators have low efficiency and produce a lot of heat dissipation, making them inefficient to use in high level applications. Switching regulators have a complex design but allows them to be efficient in converting voltages. They can not only be step-down, but also boost low voltage sources into the targeted voltage.

Linear Regulators

Because of their simplistic design linear regulators only have the capability to step down voltages. Due to their low efficiency in terms of power dispersion, they can only be used for circuits that do not produce a significant amount of power to minimize the loss of power through heat. Two main types of linear regulators exist: Series regulators have the input load and transistor connected to series whereas shunt regulators have the two components connected in parallel. The difference comes from how each circuit manages the load. Series regulators work varying the transistor element from the output voltage to make sure they match the requirements of the reference voltage. Shunt transistors compare the feedback and reference voltage to control the specific output. Series regulators are more efficient out of the two as the amount of current drawn is effectively used by the load, therefore not using much power to regulate voltage.

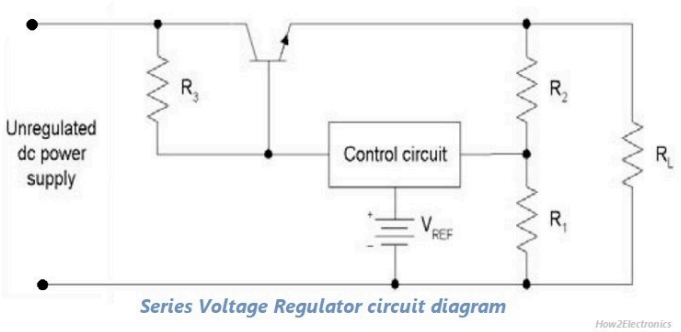


Figure 3.6.1.1: Series Linear Voltage Regulator (Subedi, 2024)

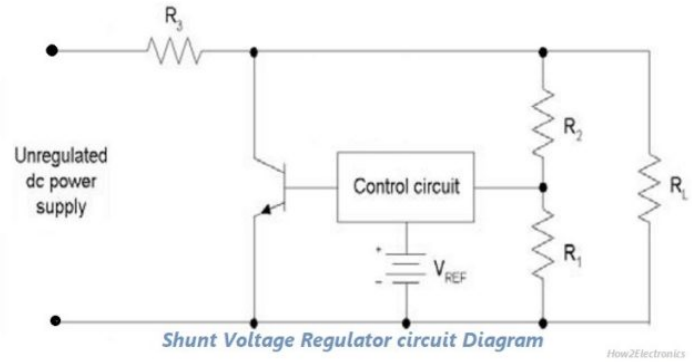


Figure 3.6.1.2: Shunt Linear Voltage Regulator (Subedi, 2024)

Switching Regulators

Switching regulators control the period of when the input voltage is active in supplying to the device by switching the element on and off via pulse width modulation to get the desired output voltage. The switching effect allows very minimal heat to be dissipated, which is why switching regulators are way more effective than linear regulators. Due to the nature of switching regulators, they can not only be used to step down voltages (buck converters) but also step up voltages (boost converters).

Table 3.6.1: Linear vs Switching Regulators

	Linear	Switching

Efficiency	Low	High
Heat Loss	Large	Minimal
Voltage Conversion	Step-Down	Step-Up, Step-Down
Design	Simple	Complex
Cost	Cheap	Expensive

3.7.1 Batteries Technology

Choosing the appropriate battery for the garage door status detector ensures long-term operation and reliable performance. For our garage door detector system, we needed to consider the type of battery that would be implemented for the devices. Before choosing which battery to use, we will need to look at the types of battery technologies that we will implement for our devices. Primary batteries cannot be recharged, hence they are labeled as single use batteries. Secondary batteries on the other hand can be rechargeable. (OpenSTAX) For our device, we concluded that using primary batteries would fit better in the long run, as we want the devices to keep running as much as possible; having one of the systems be offline due to the amount of time that a device needs to be recharged would be inefficient. Using primary batteries would also relieve users that are not experienced with handiwork installation as all they need to do is replace the batteries if they get low without doing any reassembly with the detector devices.

Table 3.7.1: Battery Technology Comparison

Battery Type	Description	Pros	Cons
Alkaline (AA/AAA)	Common disposable batteries	-Low cost -Availability	-Lower capacity -Less lifespan
Lithium	Primary Batteries	-Long lifespan	-higher cost
Carbon-Zinc	Primary Batteries	-good for low drain systems	-Low shelf life -Less capacity
Lithium-ion	High energy density	-Longer lifespan -Lightweight	-Higher cost

		-rechargable	

Although any primary battery would technically work with the system, we decided to process through which type of primary battery would be most effective and long lasting to install in the devices. Alkaline batteries are generally used everywhere in household products and are easy to find and cheap to acquire in large quantities. Lithium batteries have a higher capacity compared to alkaline batteries and can be recharged but are slightly expensive to buy compared to alkaline. Zinc Carbon batteries are best used for devices that do not consume energy at the high level due to their short shelf life. Based on these comparisons if the user does not want to constantly change their batteries on a frequent basis lithium batteries are the best choice, despite the slightly higher price point if left constantly running they will not experience the same properties as a leaking alkaline battery.

Along with the primary battery type of choice, the size of the battery also needs to be considered. Because we plan on making the devices small and compact, we cannot use batteries that are larger than a AA battery, in this case D batteries would not be sufficient.

3.6.2 Batteries Selection

AA Batteries

AA batteries are one of the most common sizes of batteries the general consumer uses and while they would be a good use for the devices, they would be insufficient to use in the long run. As AA batteries only hold a voltage of 1.5 V, this would require two AA batteries connected in series to properly function, which would contradict the design language in the small form factor. Since we are also using lithium batteries in our project buying a pack of 4 would be expensive to purchase. For these reasons the AA batteries are not a good choice to use.

AAA Batteries

AAA batteries are similar to AA batteries but only with a slightly smaller length and depth. Despite the slightly smaller size they still have the same issues as mentioned in AA batteries; only 1.5V of voltage and requiring at least 2 AAA batteries for each of the devices making the overall enclosure of the design slightly bulky. The AAA batteries would also not be good to use for the project.

9V Batteries

9V batteries are the only kind of battery with a square shape, and 1 battery can be used per device. While 9 volts is enough to supply the ESP32, the large value of the voltage can cause problems even if a step down regulator is used. (need to find sources other than bs and chatgpt) In addition, because we are using lithium batteries a lithium 9V battery would be expensive, on average around 5-7 dollars per 9V battery is not ideal for the project.

123 Batteries

Compared to AA batteries, 123 batteries have a small and compact size that carry 3V of voltage. They have been applied to many electronic systems (like smart door locks) and because they are mainly lithium batteries their battery life is long. The disadvantage of using 123 batteries is the cost of each battery due to the advanced technology in lithium batteries.

CR2 Batteries

CR2 batteries have a similar size to 123 batteries while having the same voltage of 3V. As one battery can suffice (with a voltage regulator), the only disadvantage like the 123 batteries are the high price point for a pack of 2.

N Batteries

N Batteries are sized like 123 and CR2 batteries, however they are only alkaline and like AA and AAA batteries they only have a voltage of 1.5V. These batteries would not be suitable for the project by this result.

CR2025 Batteries

CR2025 Batteries are small coin sized lithium batteries that have a voltage of 3V, meaning that one cell battery can be sufficient to use (with a regulator). Having a small height and diameter CR2025 batteries match the form factor idea of the garage detecting device. The only disadvantage is the limited capacity of 170 mAH, which could lead to a limited battery life compared to using 123 batteries.

CR2032 Batteries

CR2032 batteries also have a similar design to the CR2025 batteries, though they are slightly thicker. The difference of thickness is very negligible as the 3.2mm height does not affect any physical change to the housing for the battery. The extra thickness allows an increased capacity to 235 mAH, though compared to the capacity of 123 batteries the power life of the device might not last longer than what we desire.

Table 3.7.2: Battery Size Comparison (Renogy, 2024)

Battery Name	Shape	Available Battery Types	Dimensions	Voltage	Capacity
AA	Cylindrical	Alkaline, Lithium	L 50 mm x D 14.2 mm	1.5V	2500 mAH (Alkaline), 3000 mAH (Lithium)
AAA	Cylindrical	Alkaline, Lithium	L 44.5 mm x D 10.5 mm	1.5V	850–1200 mAh
9V	Square	Alkaline, Lithium	H 48.5 mm x L 26.5 mm x W 17.5mm	9V	500–600 mAh
123	Cylindrical	Lithium	L 34.5 mm x D 17 mm	3V	1500 mAH
CR2	Cylindrical	Lithium	L 27.5 mm x D 16 mm	3V	800 mAh
N	Cylindrical	Alkaline	L 30.2 mm x D 12 mm	1.5V	800–1000 mAh
CR2025	Cell	Lithium	H 2.5mm, D 20mm	3V	170 mAH
CR2032	Cell	Lithium	H 3.2mm, D 20mm	3V	235 mAH

Communication Protocols

Each component connected to the microcontroller, hardware and software wise, needs to follow certain protocols to properly transmit and exchange the necessary data for these parts to be able to safely operate.

3.8.1 Wired Communication Protocols

UART

Standing for universal asynchronous receiver/transmitter, this protocol allows serial data to be exchanged between two different devices. This is initiated through the RX and TX pins of the devices. Communication can be transmitted either one-side (simplex), one side at a time (half-duplex), or both simultaneously (full-simplex). Due to their asynchronous communication both devices need to transmit under the same timing speed, or baud rates. UART is implemented in systems where applications do not require higher speed transfer rates and high throughput due to their low-cost and simple implementation.

I2C

Inter-Integrated Circuits allows synchronous communication between peripherals without any complex wiring. Devices that use I2C communicate using SCL (a clock pin for the main microcontroller to send clock signals to) and SDA (serial data being between the devices).

SPI

Serial peripheral interfaces are a type of synchronous communication that sends data between devices with no interruption. Despite using 4 wires for MISO, MOSI, SS/CS, and SCLK, because of the setup data can be sent and received at the same time and can transfer data at a higher rate compared to I2C.

USB

Universal Serial Bus devices are an interface that allows communication between peripheral devices and host devices. This allows peripherals to be physically swapped to other hosts without the host restarting. The ESP-WROOM-32 device contains a micro-USB port to allow data to be transferred in order to flash new commands to the microcontroller.

Ethernet

Used in LAN (local area network) environments, ethernet allows data to be sent at fast rates through a local network. This is defined under the IEEE 802.3 and compared to wireless communications, wired networking offers more security and is less susceptible for network interference.

Table 3.8.1: Wired Communication Protocols

Protocol	Transmission Type	Communication type
UART	Serial	Asynchronous
I2C	Serial	Synchronous
SPI	Serial	Synchronous
USB	Serial	Synchronous
Ethernet	Ethernet (not serial or parallel)	Asynchronous

3.8.2 Wireless Communication Protocols

Wi-Fi

This wireless protocol is based on the IEEE 802.11 standards, which enables devices to connect to the internet wirelessly through the use of a router and a modem. Modern Wi-Fi standards have further range for connections and can handle situations where network traffic is congested.

Bluetooth

The bluetooth protocol can communicate with devices wirelessly using 2.4 GHz. Compared to Wi-Fi, the range that devices can connect through bluetooth is very short and consumes very less power. As such this is best used for devices that do not need a long term connection with each other.

Table 3.8.2: Wired Communication Protocols

Protocol	Range	Bandwidth	Frequency (GHz)	Power Consumption
Wi-Fi	Far	High	2.4, 5	High (if transferring large amounts of data
Bluetooth	Short	Low	2.4	Low

Software Comparison

Hardware alone is ineffective without the right software to support it. Software and hardware work in tandem to form a functional system, making it essential to dedicate as much thought to software selection as to hardware components. Given the vast array of software resources available today, it is crucial to conduct thorough research and carefully evaluate our options to determine the most practical and efficient technology stack. This section explores our team's software choices, from backend development to frontend implementation, ensuring a well-integrated and robust system.

3.9.1 Remote Networking Technology

One of the most important aspects of our project is the remote networking communication aspect associated with our device. The user must be able to alter the status of the garage door remotely without being connected to the local Wifi network. Besides initial physical setup with the sensors, camera, and main body device placements within the user’s garage, the majority of user interaction with said device will be completed through our web application. As such, it is crucial to research and decide on proper software technology that will allow the user to change the door status regardless of their location. When first beginning our research, it was important to understand how our software will interact with the hardware and vice versa. Our physical device will connect to the user’s home WiFi network and use bluetooth capabilities for local control. This means that, whenever our user is not home, they will need a way to alter the state of the garage.

To ensure communication between the device and the user while the latter is away from the home Wifi network, we can rely on either port forwarding, cloud servers, or even using a built-in mini-server from the microcontroller.. Port forwarding, however, is not the most user-friendly choice. Assuming the user does not already know how to port forward, the user will have to do their own research which increases installation time. Furthermore, if the user port forwards their router without ensuring a firewall is running beforehand, their security is at risk of being compromised. In addition, many people may not know how to port forward which will increase set-up time. One of our marketable aspects is to have an easy install, so this would not be a great choice. As designers, it’s important to take into account our desired consumer base and make decisions based on usability.

A cloud server provides many benefits to not just the user but also the developer. In addition to being a more user-friendly experience upon setup,

cloud servers allow for easier scalability, security, and low costs. If the team was interested in bringing this project into further fruition, cloud servers would definitely be the better choice . Purchasing a cloud server from a provider is relatively low but, seeing as port forwarding is free, in this case would be a disadvantage. Cloud servers have significantly higher security protocols in place which provides more protection for our users and takes away the need for the user to research and understand port forwarding beforehand, creating a more user friendly experience.

Table 3.9.1: Remote Networking Comparison

	Advantages	Disadvantages
Port Forwarding	- Fast communication times - No cost	- Not user-friendly - Increased security risk - Increases installation time
Cloud Server	- Scalability - Decreased security risk - High storage capacity - User friendly	- Higher cost than port forwarding - Slower communication times

3.10.1 Cloud Server Service Providers

Cloud servers are an essential portion of our system, necessary to ensure remote control and remote access to the garage door and its status while away from the home network in which the system will be configured to. As discussed in the previous section, cloud servers have a significantly better scalability, lower security risk, and overall better user friendly experience for set up and use of our system when compared to port forwarding. As we research and discuss the potential cloud service providers available to us, we will focus on the inclusion of services, such as database and authentication – to name a few, the pricing of the service based on data usage per month, and integration with our ESP32 MCUs

Amazon Web Services (AWS)

One of the most well-known and popular cloud platforms is AWS. Its extensive service offers a range of adaptability and makes developing significantly easier

to handle, an important aspect with a limited time constraint such as for our project. Starting with their Elastic Compute Cloud, or EC2, service offers scalable virtual servers for a range of tasks, such as hosting basic applications.

Additionally, AWS provides Simple Storage Service, S3, an object storage solution suitable for storing backup files, logs, and photos as it supports data redundancy. For databases, their DynamoDB serves as a fully managed NoSQL option that delivers high-speed and low-latency, particularly for IoT. Moreover, AWS provides Lambda, a serverless compute service that executes code in response to events, allowing developers the ability to build applications without the difficulty of managing the servers themselves. AWS also provides AWS IoT Core, a service tailored for managing and securing IoT devices, making it suitable for smart home applications similar to our garage door status system for its ease of access of data between connected devices.

In terms of integration with our microcontrollers, we can utilize AWS IoT core to manage the devices such as allowing the ESP32 to securely connect, send, and receive data from the database. The core supports MQTT, HTTP, and WebSockets. We can also utilize AWS Lambda for serverless processing in response to events such as the esp32 sending an update on the garage status, like changing the current status to open for example, causing the IoT core to trigger a Lambda function that would update the DynamoDB or notify the user of the change. DynamoDB is designed for fast read and write with low latency, optimal for receiving and displaying real time door status on our web application for user viewability. In terms of accessing the camera component of our device, we can use AWS S3 for storage to store logs, images, or video clips. For example, if the ESP32 captures a photo of an object in the garage door's path, it can upload it to the S3 storage, where the URL can then be saved within DynamoDB. We can implement built in S3 lifecycle policies to automatically delete older images to reduce storage costs, which can be beneficial for our stretch goal dealing with object identification using AI.

Security is one of AWS's strong points, offering their own multi-layered security infrastructure to protect cloud applications, IoT devices, and sensitive data. They follow a shared responsibility model in which AWS secures the cloud infrastructure with developers needing to configure security settings to protect the applications. The automatic AWS Shield for DDoS protection service would be highly beneficial for our IoT system, preventing disruptions caused by flooding AWS with fake requests. If an attacker tries to overload our system with false sensor updates, AWS Shield can mitigate the attack. Additionally, AWS Web Application Firewall (WAF) can be used to block malicious traffic before it reaches our web application. Another useful security measure provided by AWS

is the Key Management Service for encrypted data storage. KMS provides automatic encryption for stored data, ensuring that even if an attacker gains access to the database, they cannot read or modify the information without the proper encryption keys. This service can also encrypt data at rest like the camera capture logs in the S3 storage or sensor data stored in the DynamoDB. Additionally, AWS automatically encrypts all data in transit between ESP32 devices and AWS IoT Core using TLS encryption, adding another extra padding of security to ensure user security and also ensuring compliance with security standards.

Storing API keys and authentication credentials in firmware and source code is a major security risk. AWS Secrets Manager provides secure storage for credentials, allowing the ESP32 devices to retrieve access tokens securely without exposing them in source code, as is needed. This helps to prevent unauthorized users from intercepting credentials and also allows the source code to be more manageable. Furthermore, Secrets Manager enables automatic credential rotation, ensuring that even if a security key is compromised, it can be replaced without requiring firmware updates on the MCU, which is an added user friendly perk while also ensuring security from attackers.

AWS offers a variety of pricing models, each designed to accommodate different workloads and usage patterns. The main pricing models include On-Demand, Spot, and Free Tier, just to name a few. Given the nature of our garage door status system, if we select AWS for our cloud hosting, it is important to also select the right model to optimize performance and cost. With AWS On-Demand Instances, we may pay for storage and computation resources as needed with no long-term obligations. This paradigm is flexible and could be optimal for the testing and development stages of our system. However, we would have to ensure we keep track of our usage, as the data can eventually become pricey if the system needs constant data logging from the microcontrollers and users updating and polling data from the database. With Spot Instances, costs are significantly lower than on-demand instances, but AWS is allowed to terminate these instances at any time if demand increases. This pricing model is not ideal since our system requires real-time availability for monitoring and control. If the instance hosting our microcontrollers was terminated, the system would experience failures. In terms of the Free Tier, AWS provides limited services to support initial development and prototype such as AWS Lambda free for 1M requests/month, IoT core free for 2.25M messages/month, and S3 storage 5GB free for logging MCU events. Of course, once these free thresholds are met, costs would increase accordingly.

Google Cloud Platform

Google Cloud Google Cloud is another major cloud provider known for its high-performance infrastructure, developer-friendly services, and integrations with AI tools like Gemini and Microsoft Copilot. Google Cloud provides scalable solutions for various applications including IoT and web hosting. One of the core services in Google Cloud allows developers to execute code in response to events, making them perfect for automating tasks such as updating Firestore when an ESP32 device sends a status update. Another key service, Google Cloud IoT Core, is designed for managing IoT devices, providing a secure way to connect, manage, and collect data from our ESP32 microcontrollers. IoT Core supports MQTT and HTTP, making it compatible with ESP32's built-in networking capabilities.

Security is managed through Firebase Authentication and database rules, which define access control policies to prevent unauthorized modifications. Additionally, Firebase provides offline synchronization, allowing the system to function even if the device temporarily loses internet access, which was previously one major perk solely attributed to local communications such as Bluetooth or ESP NOW. While RTDB is cost-effective for smaller projects, pricing is based on bandwidth usage and data transfer, which can increase with a high number of parallel running connections.

Google Cloud has security and access controls to ensure that IoT devices and cloud applications are safe. Identity and Access Management (IAM) enables developers to specify fine-grained access rights, guaranteeing that only approved ESP32 devices and users can make changes. Google Cloud IoT Core also requires secure device authentication, which uses public key cryptography (JWT tokens) to authenticate device identity. This keeps unauthorized devices from accessing the system, lowering the possibility of spoofing or tampering.

Another important security feature is VPC Service Controls, which secure sensitive data from Firestore and Cloud Storages from unwanted and unauthorized access. Firestore also supports role-based access control and rules-based security, which enable developers to limit read and write access depending on device identity or user authentication. This is a great measure to take in securing security, and also ensuring that the microcontrollers will have the proper read and write authorization in order to interact with the database as needed. Google Cloud offers DDoS protection with its Cloud Armor service, which is similar to AWS Shield. This prevents malicious actors from flooding the system with requests that impair garage door functionality. Cloud audit logs provide complete tracking of all access and change activity, assisting developers in detecting and responding to security issues.

For our garage door status system, we can use Google Cloud IoT Core to manage our devices connectivity, authentication, and messaging. The MCUs would securely connect to Google Cloud via JWT authentication and MQTT for real-time communication, allowing the MCUs to safely transmit and receive messages, resulting in seamless interaction between the garage door sensor, control logic, and web application. Firestore's real-time sync feature ensures that any changes to the garage door's state are reflected in the user interface in real time. This eliminates the need for polling, which saves power on the MCUs while increasing responsiveness.

Cloud Functions can be used to trigger notifications when the status changes, such as sending alerts to a mobile app or updating a dashboard. Similar to how AWS S3 stores captured camera images, Google Cloud Storage can be used. The ESP32 can upload images when an object is obstructing the garage door and therefore blocking the garage door from being closed. Using Cloud Storage lifecycle rules, old images can be automatically deleted after a set period to optimize storage costs.

Google Cloud also offers multiple pricing models. The Pay-As-You-Go model allows developers to pay only for the resources they use, making it ideal for IoT projects like ours. Unlike AWS, Firestore has a more predictable pricing structure since it charges based on a set document reads, writes, and storage, rather than a set capacity. Google Cloud also offers a free tier, consisting of up to 50,000 Firestore document reads per month and 5GB free Cloud Storage. This makes Google Cloud a cost-effective choice for prototyping and small-scale projects such as ours.

Microsoft Azure

Microsoft Azure is another leading cloud provider offering a vast array of services tailored for businesses, developers, and IoT applications. Azure provides scalable and secure solutions for hosting web applications, managing IoT devices, and processing real-time data efficiently. One of the core services in Azure is Azure Virtual Machines (VMs), which function similarly to AWS EC2 and Google Compute Engine (GCE). These VMs provide scalable compute power, supporting various workloads, including web hosting, AI processing, and IoT applications. Azure also offers Azure Blob Storage, an object storage service comparable to AWS S3 and Google Cloud Storage. Blob Storage supports high-performance file storage with lifecycle management, making it suitable for managing logs, images, and backups.

For database management, Azure provides Azure Cosmos DB, a NoSQL database optimized for real-time applications. Cosmos DB is well-suited for IoT applications like our garage door system due to its low-latency data access, real-time updates, and automatic scaling. Unlike Firestore, which is optimized for hierarchical document storage, Cosmos DB offers multi-model support, allowing it to work with document, key-value, graph, and column-family data structures. It provides real-time data synchronization between IoT devices and the cloud, reducing the need for polling and saving energy on the ESP32 microcontrollers. This also ensures that any change in the garage door status is instantly reflected across all connected devices.

Another key service, Azure Functions, is Azure's serverless computing solution, similar to AWS Lambda and Google Cloud Functions. Azure Functions allow developers to execute event-driven code in response to triggers, making it ideal for tasks such as updating Cosmos DB when an ESP32 device sends a status update. Azure IoT Hub serves as the central service for managing IoT devices, enabling secure, two-way communication between ESP32 microcontrollers and Azure's cloud infrastructure. It supports MQTT, HTTP, and AMQP protocols, ensuring compatibility with ESP32's built-in networking capabilities.

In addition to Cosmos DB, Azure also provides Azure SQL Database, a fully managed relational database service similar to AWS RDS. If our garage door system requires structured data storage with advanced querying capabilities, Azure SQL could be a viable alternative. However, for real-time synchronization and low-latency performance, Cosmos DB remains the preferred option.

Table 3.10.1: Cloud Server Provider Comparison

	AWS	Google Cloud	Microsoft Azure
Security	Multi-Layered, AWS Shield	Cloud Armor	Pay as you go
Pricing	Tiered pricing: Pay as you go, spot pricing, on-demand	Tiered pricing: Per-second billing, Free tiers	Tiered pricing: Pay as you go, reserved instances
Databases	DynamoDB	Firestore, RTDB	Cosmos DB
Services	AWS IoT Core, Lambda	Google Cloud IoT Core, auto update Firestore	Google Compute Engine

3.11.1 Database Types

There are a number of different database types, each useful in their own way depending on the data being stored and how one wants to access said data. In order to choose the right type of database for our project, we will view factors such as data structure, query complexity, and real-time system capabilities necessary to our system.

Relational SQL

Relational SQL databases are designed to handle more structured data within a well-defined schema. It organizes information into tables of rows and columns. This structured approach allows for complex relationships between data points through primary and foreign keys, allowing for advanced queries with high accuracy. The use of SQL as a query language provides powerful data write and read capabilities, making these databases particularly effective for applications where data is more complex – like address, employee id, salary — such as company records. These databases have vertical scaling and have limited support for real-time sync. Offline support requires additional configuring.

Although they are not optimized for real-time updates, it is possible to implement instant updates by polling in relational databases. However, polling increases power consumption and therefore temperature, as the devices will have to constantly be checking for updates to the database.

Non-Relational NoSQL

Non-Relational NoSQL Databases are more flexible in comparison to relational databases, having a varied data structure depending on the type of data being used, such as documents, wide-column stores, and key-value, just to name a few. Instead of the fixed schema or predefined structure of a relational, these are more dynamic in terms of data organization. This characteristic makes NoSQL ideal for applications that need to handle large volumes of unstructured or rapidly changing information. Query languages vary between the type of database structure used, such as JSON or key-value lookups. This type of database is more suited to applications that implement real-time systems and IoT networks where performance is prioritized over strict relational integrity due to their horizontal scaling.

Due to event-driven updates, the need to poll for data updates is non-existent for NoSQL databases. This leads to a lower power consumption, which can help increase the battery life for our system. Lower power consumption will also allow for a more stable and lower temperature environment for our device.

It is possible to implement both a relational and non relational database depending on the data we are storing for our web application. For example, we can store more structured and less likely to change data such as username and password and object detection logs in relational databases, and store real-time garage door status changes to a non-relational database. However, we will be implementing a NoSQL database for it's built-in real-time syncs, as it is the backbone and a necessary component for our system.

Table 3.11.1 Relational Vs Non-Relational DB

	Relational	Non-Relational
Data Structure	Fixed schema	Flexible schema
Best Use	Structured data like user accounts or logs	Real-time updates, IoT network, event-based changes
Query	Complex queries	Optimized for speed
Scalability	Vertical	Horizontal
Real-Time Sync	Requires polling for instant updates	Built-in
Offline Support	Requires additional configurations	Offline syncing
Power Consumption	Higher	Lower

3.11.2 Database Implementation

For database management, Google Cloud Platform has different databases implementing NoSQL. These include: Cloud Bigtable, Cloud Datastore, Firestore and Firebase Realtime Database.

Cloud Bigtable

Cloud Bigtable database is designed mainly for large-scale applications such as IoT data storage and time series analytics. Bigtable organizes data into rows and columns within tables, using unique keys to identify each row. Due to its data structure nature, it is extremely useful in storing massive amounts of structured data. This structure allows for super fast read and write operations, optimized for single-row lookups. Querying in Bigtable relies on primary key lookups and range scans, meaning it lacks the flexibility of SQL-like queries but excels in efficiently handling vast amounts of data.Bigtable does not support offline data synchronization or real-time updates. Instead, it is best suited for applications where large scale data is used over real-time synchronization.

Cloud Datastore

Cloud Datastore uses an entity-based structure where data is stored in key-value pairs within hierarchical schemas. Datastore supports more advanced querying such as indexing and SQL-like queries by implementing Google’s GQL. It scales automatically, which helps to ensure data consistency across environments. Additionally, it features some offline support, although limited, in Datastore mode. This allows both mobile and web applications to cache data and sync it back once the connection is restored. Datastore does not include support for real-time updates.

Firestore

Firestore is a document and collections based database, great for IoT projects such as our own due to its low-latency fast reads/writes and automatic synchronization. This makes it ideal for applications where live and accurate updates are crucial. Firestore's built-in real-time data updates ensure that the door's status is instantly reflected across all connected devices and web applications, reducing the need for polling, and therefore the need for the devices to constantly be powered on even when not in use, resulting in a decrease of power consumption, extended battery life.

Firebase Realtime Database (RTDB)

RTDB is also a database that enables applications to store and sync data in real time. However, unlike Firestore, RTDB stores data as a JSON tree rather than a

documents based data structure. One of RTDB's key advantages over Firestore is its lower latency. These two share the ability to push real-time updates to all connected clients/devices whenever data changes, eliminating the need for polling. This greatly reduces battery consumption as we can have it so that the MCUs only read and write data to and from the database when a change has been noted by the web application. RTDB is optimized for low-latency performance, ensuring that data is instantly available to users on the web browser. However, it lacks some of Firestore's advanced querying capabilities due to the difference in their data structures, as all data resides in a single location rather than being distributed across multiple areas.

For our project, RTDB will efficiently store and synchronize garage door status updates in real time between the ESP32 devices and the web application. Firestore is better suited for structured data and projects with advanced queries. Since the main backbone of our data consists of a simple data variable "door-status," we can ultimately use RTDB for our project. Alternatively, we can either focus solely on implementing RTDB for all data variables, or use both Firestore and RTDB databases simultaneously for different purposes. For instance, Firestore can be used for storing values such as user login information, Wi-Fi SSID and Password and notification configurations, whereas the real time variables such as garage door status can be stored in the RTDB. Splitting the use of the databases as such can lead to better structured data pulls and updates but might result in a lag of communication as the MCUs have to interact with two databases simultaneously.

Table 3.11.2 Google Cloud Platform Database Implementations

	Cloud Bigtable	Cloud Datastore	Firestore	Firebase RTDB
Data Synchronization	No real time	No real time	Real-time	Real-time
Offline Support	None	Limited	Stronger, automatic syncing	Basic
Data Structure	Key-value Wide column	Entities and Properties	Documents and collections	JSON tree
Query	Single-row	Simple,	Advanced,	Simple

		indexing	indexing	filters

3.12.1 APK Application Technology

When developing an application that interacts with a cloud server, effectiveness and cost actually vary significantly between iOS, Android, and web-based APK applications. iOS applications, due to Apple's fairly strict developing environment, often require a higher investment due to things like licensing fees and device specific tools like Xcode. Furthermore, actually publishing an app to the App Store requires the use of a Mac device. The App Store review process can also be time-consuming, delaying updates and increasing maintenance efforts, which will cause a dent on our project timeline. Further, an additional $99/year is required for an App Store developer account. Despite this additional cost, iOS apps are known to have great security and smooth performance. In terms of maintenance and app updates, iOS applications require periodic updates that must go through Apple's review process, slowing down deployment. Cloud integration is seamless within Apple's ecosystem but can be restrictive when integrating with third-party cloud services. Performance-wise, iOS apps tend to be highly optimized for Apple devices, ensuring a smooth and responsive user experience with their ecosystem. This could be extremely limiting to not only users – therefore hindering our universal and user friendly goal – but in terms of development, one of the four team members is an Android user.

Comparatively, Android applications generally have lower development costs, as they can be built on various platforms. Additionally, the Google Play Store is less restrictive and therefore more affordable when it comes to publishing an app, sitting at a one time $25 fee for a publishing account. Android applications are relatively easier to maintain compared to iOS since updates can be rolled out more quickly without a lengthy and strict approval process. Android offers greater flexibility with their own cloud-based interactions, making it easier to integrate with third-party cloud services without restrictions. However, due to the variety of devices that run android, such as Google Pixel or Samsung to name a few, and operating system versions, ensuring consistent performance across different hardware configurations can be a challenge in a grander scope.

Web-based applications eliminate the need for company specific app store distributions and can be updated instantly. This makes them super cost effective when compared to the previous APK options for applications Furthermore, these

APKs can be accessed on both iOS and Android devices via a browser, making them more universally inclusive to our users. Additionally, due to their single codebase and cross-platform compatibility, web based apps provide ease of development. However, web apps can have limited access to device-specific features such as Bluetooth or push notifications, which could impact crucial overall functionality concerning real time updates for our users. The performance is also dependent on the browser connectivity, which could cause a delay in the app and real time access to data. Overall maintenance is the easiest among the three, as updates are deployed instantly on the server, making this perfect for development and update rollouts. Cloud integration is essential for web-based apps, as they rely entirely on these remote servers to function. In the case for our project, this is not a downside, as being able to use any third party cloud servers allows great flexibility with choosing a cloud server provider with lower costs and fast response times without device specific restrictions. However, performance may suffer in areas with poor network connectivity, as the application depends solely on the browser's ability to retrieve and process data for real time updates.

Table 3.12.1 APK Application Comparison

	iOS	Android	Web-Based
Development Cost	High, requires Mac, Xcode, fees	Lower	Lowest
Publishing Cost	$99/year	$25 flat fee	No app store distribution
Cloud Server Integration	Seamless with Apple only	Easy integration with third-party cloud services	Fully dependent on cloud-based interactions
App Store Restrictions	Strict and time-consuming	Faster approvals, less restrictive	N/A
Cross-Platform Compatibility	Limited	Limited	High
Performance	Optimized for Apple devices	Varies based on hardware	Dependent on browser performance

Security	High	Moderate	Moderate
Access to Device Features (Bluetooth, push notifications…)	Full access	Full access	Limited
Maintenance	Requires app store approval	Easier than iOS but still requires app store approval	Easiest with instant updates

3.13.1 Communication between Microcontrollers

Based on the communication protocols chosen, the data flow transfer will be affected. This will affect how quickly data is transferred from one microcontroller to the other creating latency between the two and changing response times. Along with standard wired communications, microcontrollers, such as the ESP32, come with the WiFi and Bluetooth module built-in and are capable of acting as a local server without the need of additional external hardware. As such, to generate faster communication times, we can utilize the built-in modules for local control since no cloud communication will be required. This will also decrease installation time since when initially configuring the device to the garage door, the user will be present and connected to their local home WiFi network.

In order to further minimize communication times, we will utilize the Wi-Fi and Bluetooth capabilities of the microcontroller. For our system, we will use two microcontrollers – a Main/Sensor MCU responsible for processing input from the reed switch sensor and the camera module and the Servo MCU in which through a Motor Controller controls the Servo Motor to open and close the garage door. Since the two MCUs to be placed some x amount of range from one another but still within the bounds of the same room, we can establish Bluetooth communication between them. This will cause a low latency and allows the garage door to be opened or closed locally despite lack of Wi-Fi connectivity.

While wireless communication offers flexibility and ease of installation, a standard wired communication approach could also be considered for reliability and security. Using wired connections, such as UART or I2C, between the Main MCU and the Servo MCU can ensure a stable and interference-free

communication channel. However, a wired setup introduces physical constraints, limiting the placement of the microcontrollers within the garage due to the necessity of running cables between them. This could make installation more complex, increasing installation time and decreasing user friendliness, and limits physical device placements on the garage. For the way our device is currently set up, it is crucial the Servo Motor MCU sits near the garage button and the Main MCU to sit near the garage door path to monitor object blockage. Wired communication can reduce latency significantly and eliminate concerns about wireless connectivity drops, but the physical limitations are impossible to move past without changing the way our device will interact with the physical garage.

Depending on the results of the testing phase, our team may adjust our approach to local communication, either by relying on the built-in local server, using the home Wi-Fi network, or Bluetooth connectivity. During testing, we will evaluate which method minimizes communication latency and ensures the fastest response times, then implement the local communication accordingly. Since our system specifications, as outlined in Section 2.5.1, require low notification delays and high-accuracy status detection, achieving fast and reliable responses is critical to the success of our project.

Table 3.13.1: Local Communication Comparison

	Latency	Power Consumption	User Setup	Reliability
Bluetooth	Low, ideal for real-time communications	Lower than Wi-Fi	Pair the 2 MCUs for direct wireless communication	Reliable through Wi-Fi shortages;
Built-In Local Server MCU	Higher than Bluetooth	Higher than Bluetooth as it is always active	Connects to home network, no additional pairing	Reliable through longer distances, dependent on Wi-Fi stability

Wired	Lowest	Lowest due to direct electrical signals	Increase installation time due to wiring	Most reliable communications

3.14.1 AI-Based Object Detection

For our DoorSure system, we needed a way to detect if something was in the way before closing the garage door. Since we're using the ESP32-CAM, we looked into a few lightweight object detection options that could work with our limited hardware. We found three leading platforms: TensorFlow Lite, OpenCV, and Edge Impulse. Each one has its own strengths and drawbacks, so we spent time looking at what they offer and how easy they are to set up.

TensorFlow Lite is a popular lightweight framework from Google that's made for running ML models on mobile and embedded devices. The advantage is that it supports offline inference, runs on lower-powered systems, and has access to a variety of pre-trained models like MobileNet. But once we dug deeper, we realized that the version made for microcontrollers, called TensorFlow Lite for Microcontrollers (TELite Micro), is very limited. It can only handle very simple image classification models, and anything more complex - like very limited. It can only handle very simple image classification models, and anything more complex - like object detection with MobileNet SSD - does not fit the memory of the ESP32-CAM. On top of this, the setup and training process requires a lot of extra tools and know-how. In the end, we decided not to use TensorFlow Lite because it doesn't meet our needs in this project.

OpenCV is a really powerful computer vision library used for everything from face detection to advanced image processing. It has tons of features and can definitely handle object detection, but not on the ESP32-CAM. OpenCV is meant for devices that have an operating system, more RAM, and a lot more processing power. The ESP32-CAM just isn't built for that. We'd have to send image data to a Raspberry Pi or another external device to process it, which adds extra cost and complexity of the project. So even though OpenCV is a solid tool in general, we decided against using it because it does not fit our low-power setup.

Edge Impulse is made specifically for embedded devices and edge AI applications, and it works really well with microcontrollers like the ESP32-CAM. What we liked most about it was how easy it is to use. One can collect data,

train your model, and deploy it all through their web interface, without needing to know a ton about machine learning. It even optimizes models automatically to fit on smaller devices. It does require internet access during training, the final model runs completely offline on ESP32-CAM. Overall, Edge Impulse fit our needs for this project with its compatibility and ease of use.

Table 3.14.1: AI-Based Object Detection

Feature	TensorFlow Lite	OpenCV	Edge Impulse
ESP32-CAM Compatibility	Limited (TELite Micro too constrained for object detection)	Not compatible (requires more RAM/OS than ESP32-CAM has)	Designed for microcontrollers like ESP32-CAM
Offline Inference	Yes (for simple models)	No (requires external processing)	Yes (after training)
Model Complexity Support	Only very simple classification models	Not on ESP32-CAM	Automatically optimized for microcontrollers
Ease of Use / Setup	Requires many tools and technical knowledge	Complex setup and external hardware needed	User-friendly web interface; minimal ML knowledge needed
Training Requirements	Requires external tools and expertise	External training required	Online training via browser
Cost & Resource Overhead	Yes	No	Yes

3.15.1 Embedded Firmware

To bring our hardware to life, we need the right development environment to program the ESP32-CAM. There are a couple of solid options for writing and deploying firmware. We focused on two main features: ESP-IDF and

Arduino-based firmware. ESP-IDF is the official development framework from Espressif. On the other hand, Arduino-based firmware offers a very different experience.

ESP-IDF is the official SDK provided by Espressif, and it gives full control of the ESP32's features. It allows highly optimized applications with low-level functionality. It can be useful for advanced configurations and performance tuning. However, it also comes with a learning curve, embedded C and command-line knowledge is needed. It can take more time both setup and troubleshooting, it's not ideal for fast prototyping.

On the other hand, Arduino-based firmware uses the Arduino IDE, offering a more beginner-friendly environment. It supports a massive number of libraries and examples, including many to the ESP32-CAM. The community support is also huge, making it an easier solution. Additionally, it allows us to test features without spending too much time on setup. While it may not provide the same level of low-level control or optimization like ESP-IDF, it has more than enough for projects like ours.

Table 3.15.1: Embedded Firmware

Feature	ESP-IDF	Arduino-Based Firmware
Official Status	Official SDK from Espressif	Community-supported platform
Control & Optimization	Full low-level control; great for performance tuning	Limited low-level access; less optimized
Ease of Use	Steeper learning curve; requires knowledge of C and CLI	Beginner-friendly; uses Arduino IDE
Library & Example Support	Decent, but less user-friendly	Extensive libraries and examples available
Community Support	Smaller, more technical community	Large and active community
Setup Time	Longer setup and	Quick setup and testing

	configuration time	
Prototyping Speed	Slower; better for production-ready firmware	Faster iteration; ideal for prototyping

Chapter 4. Standards and Design Constraints

When starting a project, standards and design constraints should be kept in mind at all times to ensure that our project not only meets requirements and standards but also functions within the constraints enforced on us such as cost and time duration. Standards document the characteristics of the product such as dimensions and most importantly safety.

4.1 Standards

Standards can be found in every aspect of our lives as simple as modern conveniences such as USB cables that allow devices to connect to each other, not only does these standards make devices work together but also share guidelines to make these technologies more accessible and reliable, throughout the next pages we will explore relevant standards to our project, we have considered the latest guidelines to ensure optimal performance, by integrating the up to date standards we would lay the foundation for a solid project that we can innovate and advance to reach a market ready product capable of competing with the competition.

4.1.1 PCB Design

First of all, we need to define PCB, according to IEEE, PCB is a printed circuit board which is used to interconnect electronic components constructed of 1 layer or possibly multiple layers of dielectric and conductive materials that are designed to form conductive paths, in our project, we will be using JLC PCB, a trusted manufacturer with more than 9 million PCBs produced per year, to ensure our design is implemented up to the following standards.

For the PCB design, we are going to take in consideration both the professor's standards from the University of Central Florida and IEEE standards, by integrating both guidelines we ensure that our PCB design meets the global guidelines and ensure that our design will be approved locally by UCF, the university guidelines are incredibly useful as they consist of previous senior

design experience which would make the implementation of our PCB in our project easier and robust.

For the professors' guidelines, we would need to use no smaller capacitors than 0802 capacitors and avoid using any IC with smaller pins spacing with 0.5 mm and they would recommend using LM25xx or LM 26xx as they are more robust and easier to use more than newer regulators from Texas Instruments as they are very sensitive to board layout citing previous senior design projects' experience, moreover, voltage regulators have to be on on daughter boards which would make it easier to switch in case the voltage regulator burned, additionally, the professor arthur weeks from the university of central florida recommends having LEDs on the voltage regulators in order of making testing and debugging easier, furthermore, in order for our design to be approved by the professors we would need to include at least one MCU, a power supply unit and several peripherals integrated with the MCU, also we would have to replace the development boards with customized boards.

We are going to be using the IPC-2221A which identifies generic physical design principles and is supplemented by various sectional documents including IPC-2222 and IPC-2223 which would be illustrated in the following figure:

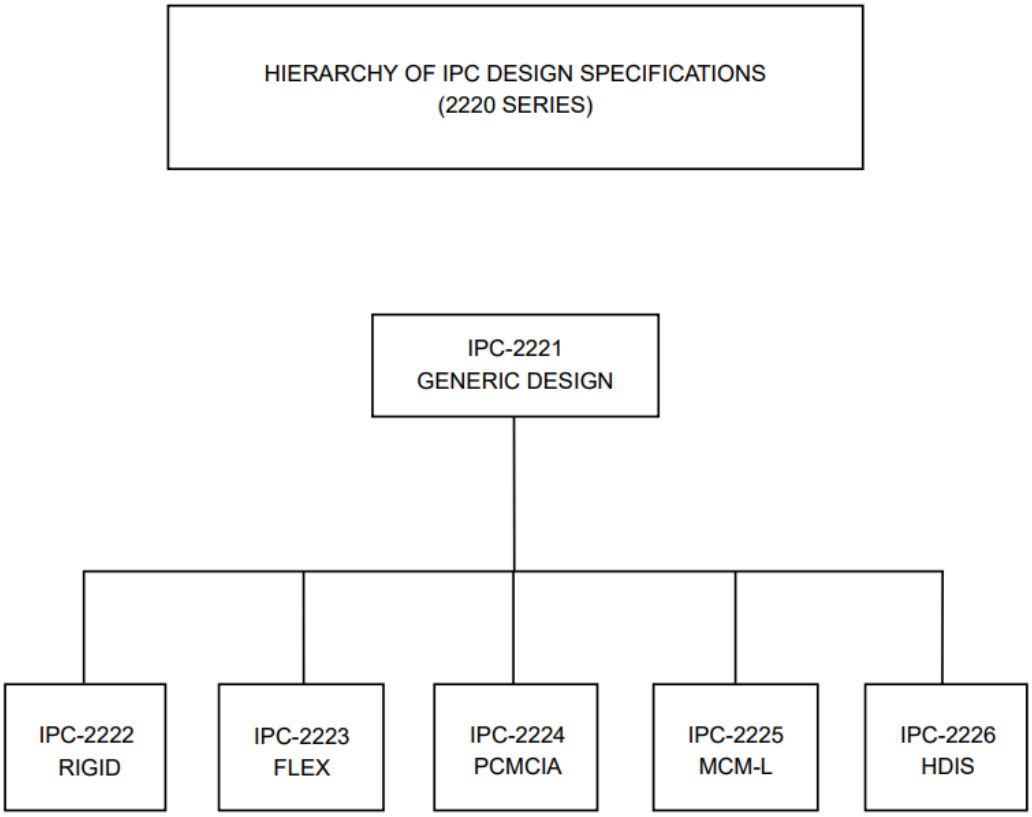


Figure 4.1.1 Hierarchy of IPC Design specification (2220 series)

This standard is intended to provide information on the generic requirements for organic printed board design,

More standards are provided by JLC PCB which we need to adhere by in order to make our circuit realistic and doable, those standards include hole size and drill diameter which we will be specifying below:

- Drill parameter for multilayer: 0.15-6.3 mm
- Min. via hole size/diameter for multilayer: 0.15 mm hole size/ 0.25 mm via diameter
- PCB thickness: 0.4 - 4.5 mm
- Via hole to hole spacing: 0.2 mm
- Via hole to track: 0.2 mm
- Copper clearance from routed board edges: ≥ 0.2 mm
- Minimum trace width: 0.16 mm

By implementing these guidelines by JLC PCB, the professors of the University of Central Florida and IPC- 2221A, we ensure that not only our design is doable and realistic but also follow all recent guidelines and up to standards.

4.1.2 WI-FI

As the DoorSure devices will use wireless connectivity to interact with the web application, we need to make sure they follow these standards for wireless communication. One such way for wireless interaction is the use of Wi-Fi, which follows the standard IEEE 802.11. IEEE 802.11 was introduced in 1997, being the pioneer of 2.4 GHz wifi. This has been the basis of future wifi infrastructures and many international industries adopt this standard. Currently, the protocol has been upgraded to IEEE 802.11ax, or Wi-Fi 6, which supports data rates up to 9.6 Gbit/s and improves performance in data-heavy areas, both the ESP 32 Development board and the ESP 32 camera development board are fully compliant WI-FI 802.11 b/g/n, we will be talking about the history and their applications, starting with the 802.11b or WI-FI 1 achieving higher data rates than and less interference from cordless phones and baby monitors, it was introduced to the market in 1999 from Apple, it would incorporate various modulation schemes such as direct-sequence spread spectrum/complementary code keying (DSSS/CCK), WIFI 1 was an early version only working at distances of 38 meters indoors and 140 meters outdoors, for WI-FI 3 or IEEE 802.11g, it would allow for faster rates up to 54 Mbits/s in a 2.4 GHZ frequency band, lastly, the IEEE 802.11n or WI-FI 4 supports both 2.4 GHZ and 5 GHZ supporting up to

600 Mbit/s data rates, it enabled the use of the WLAN networks in place of wired networks, which enabled the wide use of the WIFI, since both development boards support up to WIFI 4, they can reliably connect and update the database and consequently our web app with sufficient data rate, this standard would offer a balance between power efficiency, speed and range, which is widely used in IOT applications since the widespread compatibility of IEEE 802.11b/g/n.

Below is the evolution of IEEE standards for WI-FI technology.

Figure 4.1.2 Evolution of IEEE Standards For WI-FI Technology

4.1.3 Bluetooth

Another way the DoorSure devices can connect with each other is through bluetooth. This standard enables wireless communications through short ranges, operating in the 2.4GHz unlicensed industrial, scientific, and medical frequency band. Having bluetooth gives us an alternative way for the devices to connect to the web app in a situation where the wifi connectivity is unable to work, allowing the devices to be flexible when linking to each other, the ESP 32 development board that we chose supports both classic and Low Energy (LE) which would be a subset of the bluetooth 4.0 stack, it surfaced as an appealing alternative providing a low power mechanism for sensor data collection, which would provide us with an alternative for our devices to provide each other information such as the reed switch status and object detection status using the ESP 32 camera development board, which would be using Bluetooth 4.2 which would offer additional features making bluetooth LE a competitive candidate among the available low power communication technology such as IP profile and enhanced security.

On the other hand, Bluetooth v4.0 has multiple appealing features and modes, offering unidirectional communication between two or more bluetooth devices using advertising events, which as a result, we can achieve a communication solution without entering a bonded solution, this manner would be more energy efficient but unfortunately due to the absence of a handshake process between the two devices, this manner is subject and highly vulnerable to security threats, moreover, Bluetooth LE broadcast is not a reliable communication mechanism as the transmitter wouldn't know whether the data actually reaches any observers at all, V4.2 would be more appealing offering secure bluetooth communications and additional capabilities such as bluetooth smart which would offer features appealing to IOT applications and it would be considered relatively new with some issues still need to be solved which would be useful in our research and testing.

Additionally, there are efforts to connect Bluetooth LE devices with the internet but there are disagreements of a standardized way of transmitting transmitting IPV6 packets over the internet, although Bluetooth smart is a really interesting field which we identified and may be used in our project as a supplementary channel, since it is still a developing technology with its challenges we have decided to use WIFI connections since its widespread support.

4.2 Design Constraints

Every project has constraints which are an inevitable reality affecting every aspect of the project. The following constraints played a significant role in shaping our project, from storming ideas til testing and integration of the overall system together, constraint not only define the boundaries of what can be achieved but also gives us the motivation to innovation and creative problem solving techniques to optimize our approach to problems, they push us to find ways and be resourceful around these following constraints, whether it is working under pressure to meet tight deadlines or overcoming technical challenges that will meet us along the way.
Ultimately, every one of the following constraints has driven us to refine our ideas and build the project under the following constraints.

4.2.1 Time

In senior design, time would be our main constraint that would shape every aspect of the project, with the limited time frame of two academic terms which has one of them as summer semester giving us even less time which would mean that every decision must be made with efficiency in mind which would mean we need to be wise choosing our components and constructing the final

design to avoid repeating steps, Having open communication with team members and having solid deadlines on each tasks helped us maintain our progress, since there are no extensions on the provided deadlines, we had to set our own deadlines before to review the document and ensure the submitted document will be the best possible work to decrease the amount of work we have to do when we get the feedback.

These internal deadlines early on in our journey not only helped us maintain our progress but also taught us accountability since we all depend on each other in this project, with the recommendation of senior design professors, we had a leader step up from our team to be responsible for setting those deadlines and keeping us all working with the same pace so that we all would be done with our individual tasks in time,since the repercussions of falling behind will be severe for everyone in the team risking to take senior design again so falling behind is not an option.

Due to limited project time frame, we had to adopt rapid prototyping and testing cycles, knowing that beforehand, we adopted parallel task execution so that CPE students will be working on the software while EE students will be working on the hardware, then we can working on integrating all the project together, although, this isn't the ideal work environment but with motivation and guidance from professors gave us the push we needed to finish in time.

Ordering parts and PCB design would be one of the most affected aspects in our project due to time constraints. Early on in our journey, we experienced delays for more than a week when ordering parts from amazon, to offset these delays and prevent them from happening again in the future, we ordered backup components for every component in our project and we plan to order multiple PCBs and construct multiple PCBs as backup for our main PCB, also, we intended to take professor week's recommendation as adding LEDs in multiple parts of the PCB and putting the voltage regulators as separate PCBs and connect them to the main board by pin headers.

Time constraints will cause us to take more precautions such as working during spring break and any time available to ensure we have a fully functional prototype in senior design 1 and give us a head start in senior design 2 to avoid having any delays in terms of components or PCB deliveries which usually take more than a week, all these precautions are absolutely necessary with the given time frame.

Although, time constraint heavily impacted our project, but we recognize that in a work environment time constraint is an everyday reality, limited time frames

and the urgency to learn new skills needed during the project help prepare us to professional work environment after graduation, this intense pace required us to learn flexibility and adaptability to whatever happens during these two terms, and since most of use are taking other courses with senior design, adaptability and flexibility is key. The result of time constraint can be devastating if not taken seriously such as a rushed project with not enough time for testing and debugging so adaptability and flexibility is key to pass this class and have a working project that can be our stepping stone toward a job after graduation.

By recognizing the trade off between time constraint and final product quality we have embraced the values of time management and open communication between team members, time constraint will not only motivate us toward working efficiently towards our end goal which is having a working project with multiple backups that would help acquire skills that would help us even after graduation but also this tight schedule will motivate us to work harder and reinforce the project's reliability to minimize any risk of redoing already done parts, after taking these precautions and having this vision in mind, we will be able to meet all the deadlines and produce a fully functional project.

4.2.2 Economic Constraint

Another main constraint for our project is economic constraint, since we don't have a sponsor for the project, we are relying for our own funds to cover the expenses, which in return imposes a strict budget. Our team agreed that each member can spend up to $200, meaning our total budget will be $800, although the budget seems to be higher than the total cost in the bill of material but we had to order ahead with faster delivery options, since we suffered from delays since the beginning of the project, which encouraged us to order extra components as backup and some components so that each team member can work individually from home without waiting for critical components from each other, so that we can ensure continuous progress and avoid any further delays.

Moreover, since we plan on having multiple backup PCB boards with faster delivery to ensure continuous progress throughout senior design 1 and 2, extra funds are needed, balancing the extra cost of faster delivery and extra components with our limited budget requires us to make careful decisions when spending money from that budget , keeping in mind efficiency is key.

Since the team consists of 4 undergraduate students, cost is a critical factor to consider in every decision we make, with limited funds, the trade off between quality and cost would cause us to make a decision every time we spend money whether it is components or faster delivery, balancing between the two factors, for example when considering different Microcontroller choices, we had to

choose the best option as illustrated above in the microcontroller comparison since the microcontroller is the main component in our circuit.

Moreover , we chose the Esp32 since it has many built in capabilities such as a built in wifi and bluetooth modules which would would make capable of acting as a local server without the need of additional hardware, therefore choosing the esp 32 would make the perfect balance between quality and cost since it is a high quality microcontroller with lower cost since it has built in modules saving the cost of buying these modules separately.

On the software side, Esp32's support for ESP-NOW is another key advantage of choosing this microcontroller which would allow us to connect the ESP 32s together wirelessly, by maintaining efficient and a reliable communications between the ESPs we can enhance the performance and lower the cost of the precautions we would need to take in case of unstable communication.

Since we are aware of existing products with similar capabilities in the market discussed above in existing products chapter 2, that will enforce an economic constraint on us, although our project won't be manufactured and sold on a market level, but keeping other existing products in mind let's us have a reasonable price range as our project should cost since we don't want our project to be out of range compared to its market comparison to ensure that our design remains competitive and better than other products.

Overall , economic constraint is a significant limitation to our project driving us to be resourceful in using resources that are available to us and be mindful of that limitation when choosing our components balancing between performance and cost, also, balancing between delivery options would be key to balancing between the cheaper and slow option and the faster and the more expensive option to prevent any delays in our timeline.

4.2.3 Safety Constraint

Safety is always the number 1 priority when implementing any project, especially one involving an automated system with moving parts like our garage door project so we need to have a safety constraint and implement fail safe mechanisms in case of a network failure we have implemented an option to connect to your device using bluetooth module which is already built in the esp 32.

A failure to consider safety could have severe consequences including injury or property damages, for example if there is a long delay in door opening or

closure it may cause unauthorized access to your property or even theft, so we need to have rapid response especially for door closure, moreover the accuracy when checking the door status needs to be incredibly high (≥ 95% Accuracy) to guarantee that the system actually reflects real time state of the door, also, a low notification delay (≤ 2 seconds) is critical in providing near-instant security alerts, which will enable the user have the peace of mind in terms of getting a notification whenever an anomaly is detected, the user can also set multiple anomalies, for example, door opened late at night or leaving the door open for a long period of time, a failure in controlling the door can result in the garage door closing accidentally which can cause damages.

Most importantly, a failure to consider safety can result in injury, which is why intensive testing and debugging are incredibly important in this project, this trade off between testing, cost and time will cause us to carefully balance between doing intensive testing and cost within the time constraint, to achieve this balance, we have to divide up tasks among team members and working on individual parts and then integrate them all together, although this approach will be effective in terms of time constraint while doing intensive testing but will have impact our budget.

Safety is the only constraint that we can absolutely not compromise on, no matter the trade off between cost and time, ensuring the safety for our project's users is the absolute priority, that will affect our choice of components which have lower tolerance which would be more expensive, choosing the camera for instance, will also require us to choose a high quality camera with the option of viewing live feed (advanced goal).

Safety constraint affected our design from the very beginning, for example, we chose a camera (720+p) with AI capabilities instead of an ultrasonic sensor to detect objects in way of the garage door and possibility identifying them (stretch goal), although this approach will have some privacy concerns and high power consumption compared to other sensors, but the camera approach will offer more visual information that will increase security and justify this trade off, this approach will give us a significant advantage compared to other market example such as the Chamberlain MyQ Smart Garage Door Sensor and the Genie's Aladdin connect, positioning of the camera will be crucial in terms of detecting the objects in garage door way, not only will incorporating this camera with Ai capabilities improve security measure but will give us the advantage over our competition to improve system accuracy and maintain the user trust in our product.

4.2.4 User friendly Constraint

User friendly constraint focuses on designing our system to be user friendly for everyone. Since our project's idea is to provide a peace of mind to people, we intend to make our project comfortable and accessible, this means designing an interface that would minimize complexity of our system that would include software and hardware.

For the software, we intend to make our application as user friendly as possible by designing a simple interface that would facilitate easy navigation through the app, moreover we intended to have clear labeling with a step by step guide in the app if needed, we would then have intensive testing so that we can get feedback of users which would play a crucial role in shaping the most accessible way our app would be to serve everyone.

On the other hand, for the hardware part, we would design our system as user friendly as possible in terms of maintenance and assembly by ensuring the reliability of our components against various environmental conditions, also we would focus on the quality of components such the camera quality to ensure the system can detect objects and deliver high resolution video feed when the user wants and would also be minimize the frequent maintenance.

Since our project is still in the prototype phase, we are prioritizing flexibility and ease of use of our product to make it more comfortable and user friendly compared to the competition since our product will be used everyday by our clients at least once.

We recognize that our project will not be offered on the market but being aware of the competition and designing our project to surpass them in terms of user friendly design is key to build a market ready product, overall, user friendly constraints will shape our design making it better in many ways than the market competition.

4.2.5 Manufacturability Constraint

Although our project won't be manufactured and sold on large scale, but manufacturability constraint shaped our project from the beginning as the idea of our project came from the needs of one of our team members, so we decided to implement this idea to give him a peace of mind, while having other products on the market with similar capabilities but our project will have the best features of these products such as price, live camera feed and object detection using camera with AI capabilities.

The prototype we are building is limited by the constraint of the lab equipment and available tools for undergraduate students, moreover, since we are ordering our PCB from JLCPCB we are limited by their manufacturing capabilities that would also lead us to have specific trace width and components sizes ensuring our design will be successful and without issues, to avoid revising our design and ordering again, we will be asking for help from senior design professors to ensure that our design can be produced but also meets all senior design requirements.

This experience of working as a team under manufacturability constraint gives us real world scenarios as there are manufacturability limitations in every project whether it is equipment limitation or supply chain limitations.

The lessons learned from project will be incredibly useful in case one of the team members wanted to begin a startup company or even develop this project more to have it ready for mass production, since we are aware of the competition in the market, we are designing our product to be better in many ways that would outperform the competition.

In conclusion, manufacturability constraint would provide an aspect to consider when it comes to developing the initial idea with respect to the available equipment and PCB refinement and revision. By having this limitation in mind from the beginning of the project, we will ensure the design is innovative and realistic.

4.2.6 Sustainability Constraint

The sustainability constraint of this project will not be as critical as safety constraint or time constraint but it will still play a role in shaping our project especially if we decided to develop it into a market ready product, we will need to consider the wear and tear on our components since they are going to be operating in a garage affected by multiple environmental conditions such as temperature and humidity as the operating temperature will typically be -20°C to 60°C and the humidity tolerance will be 10% - 90% RH.

Therefore, choosing durable components that would withstand these environmental factors, in result we can reduce the frequency of maintenance and replacement of these components, although, some components will be replaced no matter the precautions we take such as batteries but we can extend the battery life so that it will last 3 months or more by lowering the overall power consumption to 2 watts to 6 watts, that will cause us to lower the frequency of maintenance.

For senior design we will be designing a prototype of a garage so we can represent it in the demo, so it would demonstrate our system's capabilities by controlling the whole garage from a user friendly interface, by designing this prototype, it gives us complete control over how the garage is built from integrating safety features and optimizing performance, this approach will allow us to tune our design based on that specific design as a proof of concept, then we can use this prototype and ultimately we will produce a product that can be implemented over any garage door.

We would also have to consider the wear and tear for the prototype of that garage as we intended to simulate long term operation on that prototype by repeated use over short period of time, this operation would be challenging since team members lack mechanical design and testing of the garage door, we will ask faculty members for help in that field.

4.2.7 Live Demonstration Model Creation

A small scale demonstration model of a garage door will need to be created to simulate the installation of the DoorSure system and what the proper behavior should look like. We intend for this model to mimic real life garage doors, achieving key features as illustrated in table 2.5.1 and 2.5.1, our specifications tables. Our project will need to meet at least our three highlighted specifications — garage door status accuracy, battery life, object detection accuracy — to demonstrate successful completion of the project and course.

Since the small scale model needs to mimic the entire experience of a garage door opening and closing similar to a full scale testing and operation of a garage door in real life, we need to allocate the proper amount of time and resources to make this happen. That means that our time constraint is put under more stress, as we need to ensure proper time to design and build a functioning prototype of a garage that will open and close at the press of a button to showcase our device's capabilities.

Failure to meet this constraint will lead to failure in the Senior Design course, as we will not have been able to properly showcase a live demo of our device working to our mentoring advisor and review committee.

In conclusion, sustainability wouldn't be critical to our design but it remains a key consideration especially if we are designing a prototype of the garage that can possibly be the foundation into a market ready product, that would affect the choosing of components and as a result affect our whole PCB design.

Chapter 5. Comparison of ChatGPT with other Similar Platforms

With the increasing popularity of Artificial Intelligence, our team has benefitted from using ChatGPT as a resource to guide us throughout our project. From searching for bugs in our code and proofreading our technical documentation, to researching hardware and software technology, ChatGPT has provided insights that have cut the research period almost in half, allowing the team to focus on the hardware and software. By utilizing AI-powered assistance, we have been able to enhance our research, improve efficiency, and ensure a more well-rounded approach for our project.

Case Study 1

Prompt: “What kind of motor would be efficient to use to push a garage door button?”

When inputting this question into ChatGPT, the responses received were mostly accurate. The model listed out 3 of the motors that could be implemented in our project, linear actuators, solenoids, and servo motors. We chose the servo motor as our choice early in our project through our own research and all of the characteristics shown were accurate in its results. However there are a few caveats. ChatGPT only lists these three models in its results and no other motors like DC, AC, their brushed or brushless counterparts, or stepper motors are listed. In addition only their general specifications are listed along with their main strengths, no disadvantages or limitations are listed at all along with any specific engineering specifications. Cost is not listed which is a major factor in deciding which component to get in order to cut costs.

Its main recommendations at the end were either a linear actuator or a servo motor. To conserve costs, the servo motor would be the better choice as linear actuators are expensive to acquire compared to getting servo motors. While ChatGPT does help in deciding which part would be preferable to get to press the garage door button, the information generated is left to be desired.

When applying the same prompt into Gemini, their response is more detailed than ChatGPT’s, as each component has pros, cons, their ideal form in use of real-world applications, along with major key considerations in case if the part they suggested is not what we want to choose. While Gemini’s response is good, the problems that are explained in ChatGPT’s response can also be

applied to Gemini as well. There is no mention of cost for all of the components and the list of components are also limited to 3. Servo motors are not mentioned as well, which adds to the confusion on Gemini or even ChatGPT added more than 3 motors to their list.

Microsoft Copilot is even worse than both models as only two motors are listed (which were small DC motors and linear actuators), while giving them general descriptions on why to use them. They do not break down the advantages and disadvantages of each component or the cost of obtaining them.

Case Study 2

Prompt: “For a system that requires low power when in use and when it's idle, along with long battery life, what size of battery should I consider using? (AA, AAA, 9V, etc.)”

This prompt was to make sure the model would properly reply with the battery size types, along with the purpose of what the system should do in respect to power operations. In the case of ChatGPT’s response, their list of battery size types is mostly accurate to the results found in Section 3.2.1.2, along with major key considerations. The prompt only lists the kinds of batteries that are commonly used while this report also uses batteries that are uncommon, such as 123, CR2, and N batteries. Their list does not go over if there are alkaline or lithium versions of the respective battery, along with not going into detail over the pros and cons of the type. At the end the model suggests different kinds of batteries depending on the situation, such as making the system ultra-low power, size and weight constraints, and rechargeability.

For Gemini’s response, the style of their prompt is similar to ChatGPT’s. Both include a limitation for 9V batteries in terms of power efficiency and the recommendations are similar. One interesting thing to note on Gemini’s side is the inclusion of a lithium battery known as “LiFePO4”. Although this is useful to know, we intentionally did not include other lithium battery types as we wanted our system to use replaceable batteries for the sake of convenience.

Like the first prompt, Copilot has the worst response of the three, only mentioning the three battery size types that are arguably the most commonly used. Only AA, AAA, and 9V are listed, while the other two models list out more of a variety like button cells and other lithium type batteries. Each battery listed specifies their capacity, size, lifespan, and use cases, except that 9V batteries only list their capacity and use cases for some reason. This inaccuracy within

their response shows the unreliability that AI models can have and why it is important to seek the direct sources for research instead of overly relying on AI.

Case Study 3

Prompt: “I’m using ESP32-CAM to do an object detection project. I need the camera to recognize objects in the way when the garage is open. What are my choices?”

ChatGPT offered me opinions to choose from: TensorFlow Lite for Microcontrollers, Edge Impulse, OpenMV, and YOLO-based External Processing. On the other hand, Deepseek gave me seven opinions, divided into different implementation methods. They were both very helpful and provided step-by-step guides on how to accomplish it, even giving me recommendations on which one works in what scenarios.

These prompts bring insight to a few things. For starters there might be moments where the prompt needs more detail in order for the response to be plausible to believe. Otherwise the response will be somewhat limited. Plus, this shows that AI response models cannot be overly relied on because their results can be made out of random jargon and have minor mistakes in providing facts to components.

Case Study 4

Prompt: “I’m developing an accompanying application to a device that needs to work across multiple platforms. I want to understand the differences in building an APK for Android vs iOS vs web based.”

When typing the prompt into ChatGPT, the responses were given in a very structured manner. It provided me with three primary approaches: native development, cross-platform frameworks, and progressive web apps (PWAs). It correctly outlined the strengths of Android’s Java/Kotlin, iOS’s Swift, and cross-platform solutions like Flutter and React Native.

It gave me detailed information on Apple’s security policies, but failed to cite any resources. When promoted to cite where the information was found, the engine fed me back a link to an irrelevant website of Apple. When prompted to give a valid citation, it then linked an unreputable source. This is a setback on the technology, as there is a lot of either phony or outdated information in the internet, and if ChatGPT cannot differentiate between them, it will very easily

spew out this information as solid truths to the prompter. If I had been just a curious person typing the prompt in for more information on how application development worked, it would have been very difficult to understand and process the information that was given. The answer seemed long winded and filled with filler content, some of which ended up not being completely accurate. ChatGPT primarily focused on general characteristics and strengths but did not list significant disadvantages of each method. For example, it did not mention the challenges of App Store approval for iOS due to their strict policies – especially citing what these strict policies are and how to develop your app to pass them, or performance trade-offs with hybrid frameworks. Another key omission was cost considerations, such as Apple's developer fees, the expenses of maintaining native apps versus web-based solutions and the comparison with Android APK. ChatGPT ultimately leaned towards cross-platform development (Flutter/React Native) as the best choice but did not provide a scenario-based recommendation tailored to different app needs.

When applying the same prompt into Gemini, the response was more detailed than ChatGPT's. It broke down pros and cons for each approach and provided real-world application examples. Gemini also highlighted key considerations, such as device compatibility, performance needs, and long-term maintenance. However, the same issues persisted—cost was not mentioned, and the list of development approaches was somewhat limited. It lacked a strong emphasis on offline capabilities and app store publishing hurdles, which are critical factors in deciding between native and web-based solutions. However, one pro was when given additional information about my situation, such as "I want to create an application that manages door status and the user can alter the status from the phone," the engine mentioned more situation based answers such as discussing the limitations of the APKS in terms of device access and how it could potentially affect the response times and database updates and requests with this limitation.

When prompted, Microsoft Copilot only listed native Android (APK) and iOS development, with brief descriptions of why to use them. There was no mention of cross-platform frameworks, PWAs, or costs. Copilot also failed to break down the advantages and disadvantages of each approach or consider real-world scenarios such as scalability, app store restrictions, or cost efficiency. This made the response less useful for decision-making, as it lacked depth and alternative solutions. It gave a clear cut answer without much detail. In order to gain more information and insight with this engine, the user would have to kind of bombard the system with multiple questions to force a more useful answer from it. The detail that was given by ChatGPT and Gemini in comparison fared better in terms of comparisons between the APKs.

Chapter 6. Hardware Design

This section will discuss the systems that will be implemented for DoorSure. Each system is important in making the devices fully operational and show the reasoning behind why we chose these systems and their contribution to the overall project. Along with the hardware design systems the components used for each system will be explained in detail and the reason behind choosing them. Each component will then be displayed on a schematic for each device.

6.1 Subsystems

Garage Status System

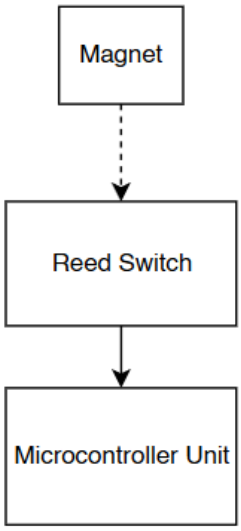


Figure 6.1.1: Garage Status Subsystem

The garage status subsystem is responsible for detecting whether or not the garage door is open or closed. This is one of the main important functions for DoorSure as the user needs to be informed if the garage door has been interacted with as they are preparing to leave their household. The main component used to detect the status is a reed switch, an electromechanical switch that behaves in different properties depending on the proximity of the magnet from the switch.

For the reed switch to properly function, a magnet will be attached to the side of the garage door. As the main device that contains the reed switch will be mounted alongside the break beam box, the placement allows the reed switch

to be in full proximity of the magnet's presence. When the user interacts with the web application to either close or open the garage door, the magnet will either be in range or move away from the reed switch. The switch then sends data to the microcontroller, which decides the status of the garage door. The information of the garage door's current status then gets sent to the web app for the user to see that the garage door is either open or closed after that interaction.

Object Detection System

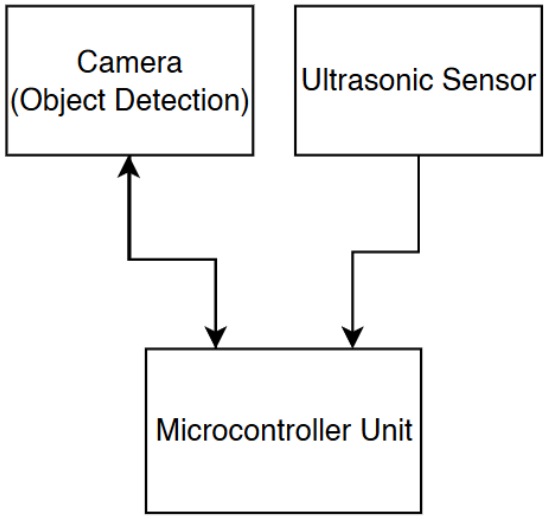


Figure 6.1.2: Object Detection Subsystem

The object detection subsystem is involved in detecting any objects that are obstructing the path of the garage door. Two components are used for this system. The main component that oversees the object detection is the OV2640 camera. The OV2640 camera is currently installed on an ESP32-CAM, a device that integrates the ESP32 development hardware with the OV2640 camera. The camera requires at least 3.3V of power to be operated. A two way connection is formed between the microcontroller and camera; the microcontroller powers the camera's functionalities while the camera sends information to the microcontroller for any objects that are detected. The camera is programmed through computer vision to detect objects that are within the garage door's path. Although programming the camera to detect objects in its field of vision can dedicate a great amount of time, we decided to limit the camera to view objects that are mainly common in blocking a garage door's path, like small moving objects and spherical objects.

In the case that the camera fails to function, the ultrasonic sensor operates as a backup or in tandem with the camera. The ultrasonic sensor we are using is the HC-SR04, requiring 5V to power the component via the Vcc pin. The trigger and echo pins are wired to the digital pins on the microcontroller. The HC-SR04 uses ultrasonic waves to measure the distance from where the sensor is emitting. If an object is within the distance that is measured, the ultrasonic sensor will send information to the microcontroller, which is then sent to the web app to notify the user that something was blocking the way of the garage door.

Camera Feed System

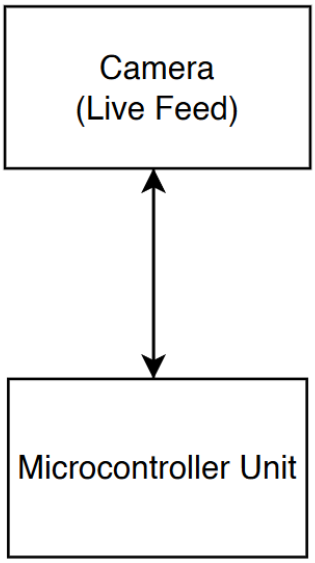


Figure 6.1.3: Camera Feed Subsystem

The camera system is a subsystem that is part of the main system. There is a two way communication for the microprocessor to power the camera while the camera sends data to the microprocessor to send information to the web application. The camera being used for this system is another OV2640 camera module, being installed into the main device with the sole purpose of streaming a live feed of the garage door for it to be viewed on the web app.

While having both camera feed and object detection functions in a single OV2640 would work, in the long run this is unfeasible because one of the functions will have to be limited in quality in order to keep the performance of the camera being stable without being overstrained. By having two OV2640s the camera functions can be separated, giving each camera high performance due to only being in charge of one singular purpose.

Furthermore, in the case that one of the cameras fails to function, a backup situation has been planned. The ultrasonic sensor will still be in operation, taking over the responsibility of detecting objects, while the other working camera will switch to view the garage as a live feed camera instead. Although this setup is not as reliable as having a camera dedicated for object detection compared to the accuracy of the ultrasonic sensor, the system will not cease to function since both camera feed and object detection systems are still active even if one camera is unavailable.

Temperature and Humidity System

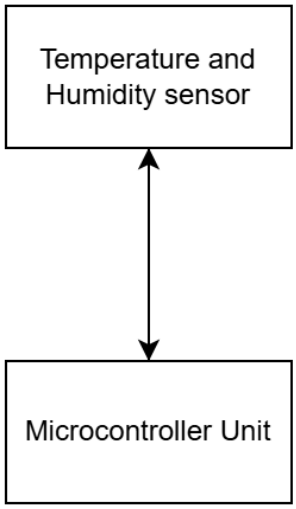


Figure 6.1.4: Temperature Subsystem

The temperature and humidity system serves to analyze the temperature and humidity of the garage room, making sure that both the main and servo devices are still functional should the room reach abnormal conditions. When certain temperatures or humid values are met, the sensor will send the information to the microcontroller, alerting the user that the devices may risk damage under these conditions and will enter sleep mode until the conditions return to normal. The reason why temperature and humidity are in the same block is that the device that will be mentioned have both functions integrated into a single device.

The DHT11 temperature and humidity sensor will be used for both the main and servo device, requiring 3.3-5V to operate and a data pin that can be connected to a digital pin on the microcontroller. The range of operation in temperature is

0-50 degrees Celsius (32-122 degrees Fahrenheit), and has an accuracy of around 2 degrees celsius. Additionally, the DHT11 provides relative humidity value in percentage, having a range of operation of 20 to 90 percent RH with an accuracy of 5 percent RH, the DHT11 uses a resistive humidity measurement component and NTC temperature measurement component, the high accuracy of the sensor makes the system feasible to implement in alerting the user about the status of the devices through the room temperatures and humidity.

The bidirectional arrows in the above flowchart visualize the communication protocol between the MCU and the DHT11, as the MCU is expected to send a start pulse to the DHT11 in the beginning of the communication processes which would last 18ms by pulling down the data pin. The DHT11 will respond with a response pulse which indicates that the sensor received the start pulse, then sending the data frame containing the temperature value in celsius and relative humidity percentage value followed by a checksum value to confirm the correct transfer of data to the MCU. After sending the necessary information to the MCU the DHT11 goes into low power mode until the next start pulse, which is a highly desirable feature for our project as our devices need to operate with minimal energy being wasted.

Buzzer System

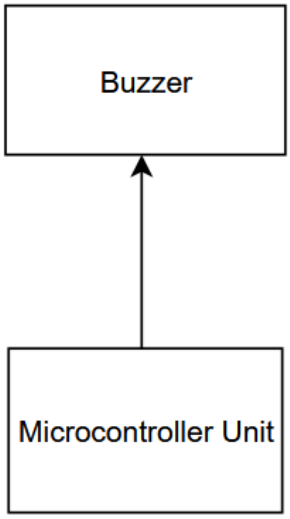


Figure 6.1.6: Buzzer Subsystem

The buzzer subsystem contains the buzzer component for when the device becomes active in use. A sound will initiate indicating that the buzzer is in use

until the device finishes its main function. Both devices will have a piezoelectric buzzer installed, being configured to make a sound when, for the main device, the reed switch is being interacted with, whether the magnet attached to the garage door enters or leaves the proximity of the reed switch. For the servo device, when the user interacts with the web application to initiate the button press, the buzzer will go off when the servo motor is initiated. The piezoelectric buzzer has two pins; one for data and the other for ground. The data pin can be wired to a GPIO pin that is capable of transmitting pulse width modulation signals. This is so the microcontroller unit can send PWM signals for the piezo element to vibrate at different oscillations in order to create the buzzer noise.

Servo Motor System

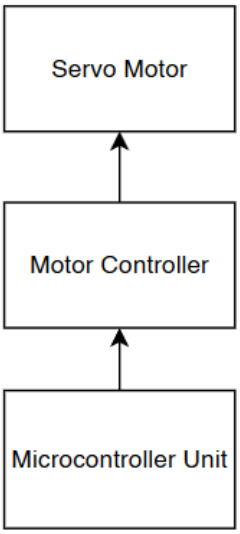


Figure 6.1.7: Servo Motor Subsystem

On the servo device, the servo motor system enables the use of the servo motor to press the garage door button. The subsystem consists of an MG90 servo motor, which has a motor controller built in. This is powered by the microcontroller, providing the necessary directions to the servo motor to properly function. We chose this servo motor due to having its advantages over the related SG90 servo; the MG90 contains metal gears which maximize

longevity and minimize maintenance frequency. Despite the inclusion of the metal gears the component is light in weight and manages to provide a stall torque of 1.8kg/cm in a small size, which is sufficient enough for our button pressing device. The servo motor has 3 pins: Vin, Gnd and PWM. As the minimum voltage required for the MG90 motor to operate is 4.8V, we will be providing the servo with regulated 5V of power, with common ground being wired to the ground pin on the MCU. For PWM, this can be connected through a GPIO data pin. As the MG90 has the motor controller built in, it provides stable data signals when the user opens or closes the door on their application.

On the servo schematic that will be discussed later on in the report, the MG90 can be seen as the male pinhead; the servo motor chosen uses female jumpers for pin connections, which is easier to connect to the servo system PCB as we can install male headers on the PCB for simple hardwiring.

Communication System

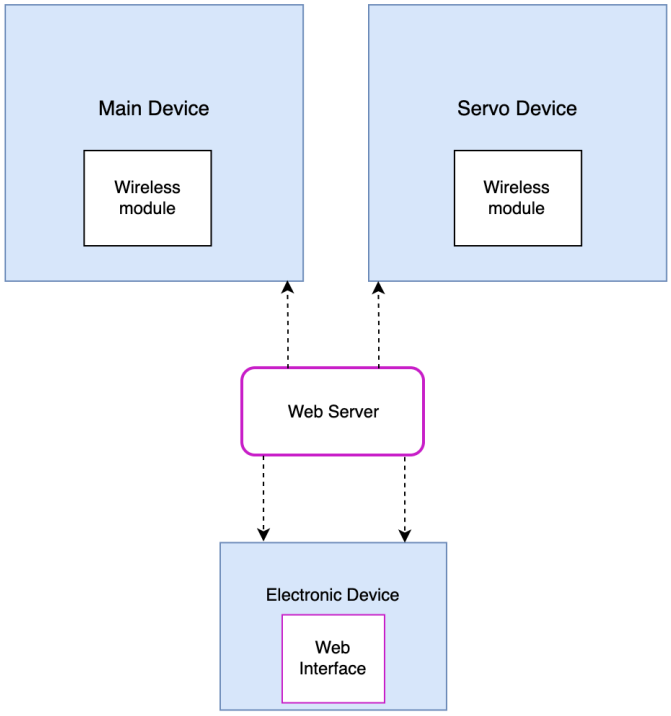


Figure 6.1.8: Communication Subsystem

The communication subsystem for DoorSure allows each device to communicate to the web server when the user interacts with the web application. Each device is equipped with an ESP32 microprocessor, which has integrated wireless communications. Although the devices can connect to each

other through applications like ESP-NOW, this would sacrifice the ability to connect to the web server. Through intensive testing and research we learned that there would be no possible way to communicate with the web server while communicating with each other, as only one kind of communication can only be active while the system is in use, meaning that if we want to incorporate the ability for the devices to connect to each other and use other wireless communications to the web server, the system must immediately shut down one kind of communication before immediately switching to the other type, which is not viable for a system that needs to have a quick response time for notifying the user in relations to the garage door. The most efficient, practical and feasible solution is to have each device communicate directly to the web server instead of each other. This solves the issue of the method of communication as instead of using two different communication styles to interact with the devices to the web app, having the devices communicate directly to the user application with only one method eliminates the need of delay switching to another communication method and in the long run will not complicate in further troubleshooting as using two different methods would result in more microprocessors to be added to accommodate the two wireless setups.

In our wireless communication system, the main source of communication between the ESP32 processors in the main and servo devices with the web application is wifi, as this allows the user to connect to the devices through the web app at a different location if the user is not present as their household. In terms of other communication types like JTAG, UART and I2C, a prototype has been made to test the efficiency of battery life using I2C via an LED panel, connecting to the ESP32 through the SDA and SCL pins. The OV2640 camera module is connected to the main device's microprocessor through the RX and TX pins, allowing asynchronous serial communication to occur between the two devices through UART.

Power Management System

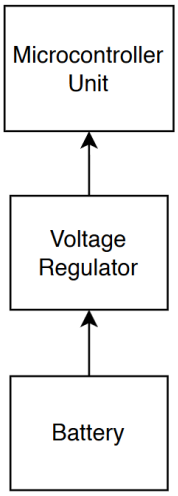


Figure 6.1.9: Power Subsystem (general)

The main and servo design will utilize their own power systems as they are not sharing the same MCU. To make troubleshooting any issues with power management less troubling and time consuming they will be assembled on a daughter board for each respective system. This will allow for the separate system to be easily replaced as they are not integrated into a main board, compared to if the power system was fully integrated on the main pcb. The construction of the power system will be identical for both devices to reduce any chance of unique variety and to make debugging and replacing parts less time intensive.

Battery Choice

For consistency reasons the main and servo devices will be powered by the same battery. As listed in Chapter 3 in the batteries section, lithium batteries were the main choice of primary battery to use. Compared to carbon-zinc and alkaline, lithium batteries have a longer capacity and while slightly expensive to buy, they will last for a good amount of time in the long run. As for the battery size, we decided to choose 123 batteries as the main source of power. This can be attributed to a few aspects to our reasoning for choosing an obscure battery size. Compared to AA and AAA batteries, which can only hold up to 1.5V, 123 batteries can hold up 3V, requiring less batteries to be used to power the devices. The 123 batteries also have a small form factor, meaning that we can use at most two 123 batteries for each device without taking up much space. The downside of using the 123 batteries is their prices as lithium batteries are

more expensive compared to alkaline batteries, although the price paid for a two pack of the 123 batteries will be worth the longer battery life of the devices.

Power Regulation

Both main and servo devices will require the use of a voltage regulator in order to supply a constant voltage to the devices while minimizing any risk of potential voltage spikes or drops during operation. We will prioritize on selecting switching regulators over linear regulators as switching regulators are highly efficient in power consumption and produce less heat when dissipating.

We will be using Texas Instruments' TPS613221ADBVR and TPS613222ADBVR boost regulators to generate a 3.3V and 5V output respectively. Using these boost regulators allow us to use one battery to power the whole system, which while is very efficient in creating a smaller footprint in design, this can lead to a lower operating life compared to installing two batteries. Although we are aware of the downside of using a boost converter, the reason we chose these regulators is our previous experience and familiarity using them in past projects, allowing installation and configuration of the converters to be easy to manage. Since these will be installed on the daughter boards, we can easily switch them out for other switching regulators in the case that the boost converters do not work in our favor in the future, or quickly replace or configure them for easy debugging and maintenance.

Table 6.1.10 Switching Regulators

Parameters	TPS613221ADBVR	TPS613222ADBVR
Input voltage	0.9V-5.5V	0.9V-5.5V
Output Voltage	3.3V	5V
Minimum switch peak-current limit	0.75A	1.10A
Output Voltage ripple	(-10mV) - (+10mV)	(-10mV) - (+10mV)

For a general outlook of the power systems as shown on figure 6.7, a battery is connected to the voltage regulators (3.3v and 5v), which would then power the microcontroller using the 3.3v regulator. The regulators will power other components as well depending on the voltage input required to operate these components reliably, which are specified in their respective datasheet. For

instance, some peripherals would require 5v such as the servo motor while the microcontroller would require 3.3v. The specific components used for the power system have been explained earlier in this section.

6.2 USB to UART bridge

The CP2102 is the USB to UART bridge in our ESP32 development board that we will be using initially in prototype and developing our project, the USB to UART bridge allows a straightforward connection between the the computer's USB to the ESP32 serial pins, since we have the CP2102 chip on our board, it is more convenient compared to a separate external USB to serial adapter, the development board would include an on board voltage regulator as well which would be the AMS1117 ensuring a stable power supply, overall, the CP2102 and the voltage regulator allow us to use a USB connection for power and data, which would simplify the process of programming our development board.

Figure 6.5 in (appendix C) is a schematic of the CP2102 from Silicon Labs showing how to the chip connects to a USB port on the left side of the figure, the VBUS pin would be the 5V from the USB port, and D+ and D- would be the data connection from the USB port. We can observe optional components such as the 4.7Kohm pull up resistor as option 1 which can be added to increase noise immunity, option 2 would give us the ability to power other devices from the on-chip regulator by adding the 4.7 microFarad capacitor. In conclusion, these optional components offer more flexibility and improved performance to our overall circuit.

6.3 JTAG and UART

UART, which stands for universal asynchronous receiver transmitter, is a hardware feature that handles asynchronous serial communication. In our project, we will be using the UART pins to program the ESP32 chip. First, we set the communication parameters which includes baud rate. We then assign pins for connection which in our case would be the pins showcased in the UART connection table below which are critical for transmitting and receiving data. Before initiating UART communications, we would need to install the driver specified by the datasheet for our CP2102 USB to UART bridge, which converts the USB signal to serial signal that the ESP32 chip understands, then we can initiate the UART connection, transmitting and receiving data from the MCU and USB connection. We will be using the UART connection for programming of the MCU and consequently our entire board.

Below is the UART connection to our microcontroller as specified by the ESP32 datasheet.

Table 6.3.1: UART Connections

Pins	Name	Functionality	Connection
1	TX	Transmit data between the MCU and USB to serial converter	TXD0 on the MCU
2	RX	Receive data between the MCU and USB to serial converter	RXD0 on the MCU
3	GND	Ground	Ground

Additionally we included JTAG (Joint Test Action Group) pins to our MCUs for debugging. Although we can debug without the need to setup JTAG by using the idf.py monitor and connecting to the ESP32 through the UART pins, we chose to add JTAG pins as an additional method of debugging. Both methods are integrated under the Eclipse ide in order to provide a quick and easy transition between compiling and debugging code with its built-in capabilities other than directly from the terminal window. JTAG provides additional advantages as well, as it lets us pause code execution and inspect variables in real time which would be a major advantage especially during the prototype and testing phase. Since the idf.py monitor route uses the UART pins, we can use the JTAG pins to debug in case of damage occurring to the UART pins after programming. This ensures our project is able to withstand damages and continue to function as usual while allowing us to debug. We may remove these pins in future versions of our project, since we would be more confident in our code and our design but for the prototype phase, we recognize that solving errors that may face us is part of the process for developing a product.

Below is the JTAG connection to our microcontroller as specified by the ESP32 datasheet.

Table 6.3.2: JTAG Connections

Pins	Name	Functionality	Connection
1	3.3v	Power supply for the JTAG interface	3.3V regulator
2	MTD0	Test data out	(IO15) pin 23 on the MCU

3	MTCK	Test Clock	(IO13) pin 16 on the MCU
4	MTDI	Test data in	(IO12) pin 14 on the MCU
5	MTMS	Test mode select	(IO14) pin 13 on the MCU
6	GND	Ground	Ground

6.4 Schematic Design

Main System Schematic

This schematic is composed of the main device, which contains the components that have been explained earlier in the main system. The heart of the main system schematic is the ESP-WROOM-32, acting as the main source of data communication between the components along with wireless communication to the user web application. A component that was not discussed earlier was the buzzer, which is located on the bottom left of the schematic; its purpose is to create a sound when the main device is currently active in use as the user interacts with the web app. The buzzer has two pins, one for data and the other being connected to ground. As such the buzzer can be connected to the ESP32's GPIO pins. The same is applied to the led installation, with the cathode side being wired to the GPIO pins of the ESP32 and the ESP32-CAMs and the anode side being connected to ground. A resistor will be added alongside the LED to make sure that the LEDs do not short out. The value of the resistor depends on how bright we want the LED to be; the shorter the resistance the brighter. The reed switch also only has two pins, being data and ground. The data wire is connected to another GPIO pin. The DTH11 temperature, while having four pins, one of the pins are unused, with the remaining three being used for power, data, and ground. The data pin on the DTH11 can be connected to a GPIO pin on the WROOM.

For power connections, the ESP32 and ESP32-CAMs share the same junction, which is powered by a 5V source. The power source that is listed on the schematic will be listed later in this section when we describe the schematic for the power system. The ESP-WROOM-32 on the schematic does not have a VIN (5V) pin whereas in real life the component does have a 5V input pin due to the usage of the AMS1117 regulator that is built into the microcontroller units. As such the three ESP32 components are connected together using the 3V3 pin. To control the OV2640 modules directly from the WROOM processor the

ESP32-CAMs have a U0R and U0T pin, which can be wired into the RX and TX pins on the WROOM respectively. Both cam devices have the R and T pins junctioned together and connected to the RX and TX pins on the WROOM. Besides data connections from GPIO, there are other modules connected to the WROOM through other methods, in terms of the ESP32-CAMs the OV2640 can communicate through I2C and SPI.

As mentioned above, we included UART and JTAG pins for programming and debugging respectively.

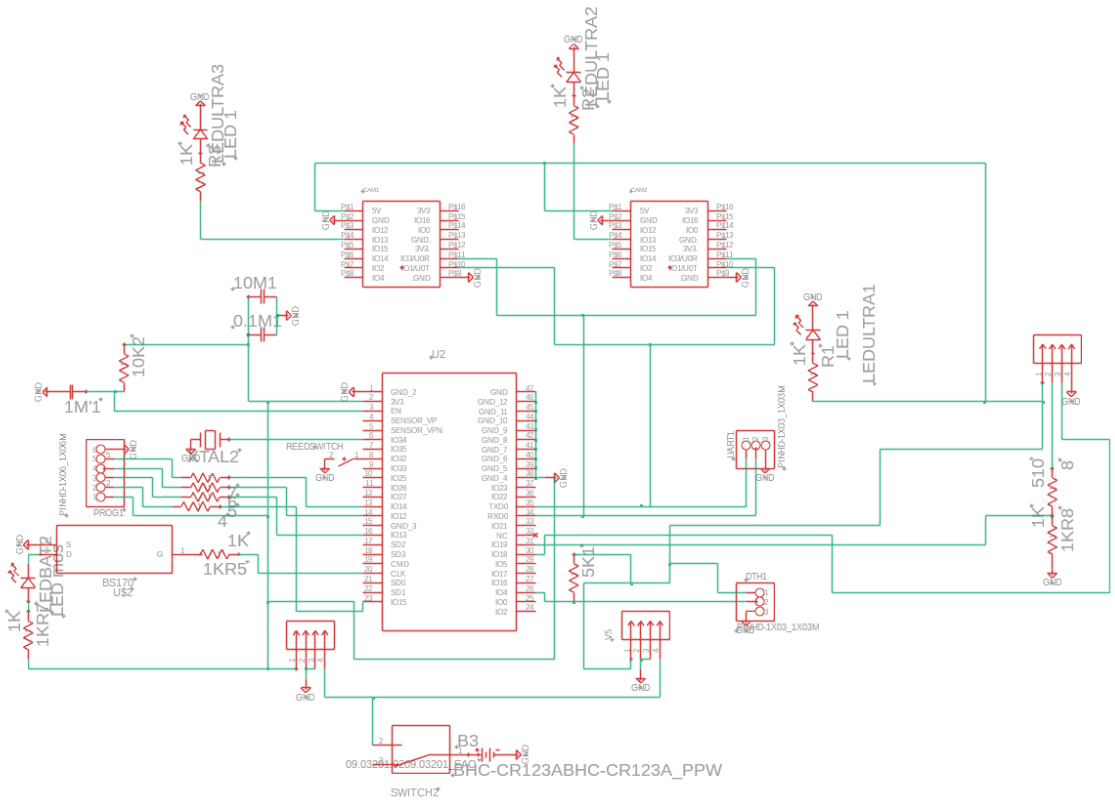


Figure 6.4.1: Main Device Schematic

Servo System Schematic

The schematic for the servo system contains the components that are used in the servo device. Compared to the main system, as the servo device is only used to initiate pressing the garage door button, there are fewer components installed in the servo schematic, we will explain the integration of the whole system servo together. Like the main device, we will be powering the servo

device using a single 123 size battery, which can be illustrated on the servo system schematic as BHC-CR123, single battery holder meant for containing 123 sized batteries. The battery holder would allow us to switch the batteries easily, allowing us to offer easier maintenance and quick debugging and testing. A switch is installed so we can physically turn the whole system ON/OFF. This will offer convenience to our users and in turn also help in troubleshooting should a component malfunction from an electrical short.

We would then use a 3.3V regulator using TPS613221ADBVR and a 5v regulator using TPS613222ADBVR as illustrated in the voltage regulator schematic to offer a stable voltage supply to components in order to avoid spikes and damages to our board. In addition to using the regulators we will also include the BS170, a transistor serving as an efficient low power switcher to control auxiliary components. This will be connected with an LED (used for debugging purposes) through the CLK pin on the ESP32 MCU. Due to its fast switching performance this prevents potential power overloading on the microcontroller and easily controls components that require more power input compared to the MCU.

For the servo motor unit, we added a male header which would make connecting to the MG90S easier as the MG90S wiring ends off with female headers, which is convenient for when we need to switch servo motors for maintenance and testing purposes. The servo motor is powered by a regulated 5V from the daughter board as illustrated in the schematic, which is connected from the Vcc pin on the servo motor to the power source. The second pin is used for the PWM control signal from the ESP32's GPIO pin 18 and the last pin is the GND pin, having it as common ground with the MCU. For the DHT11 connection, this follows the same pin layout as the MG90S (Vcc, Data, Ground, the fourth pin is unused), but for the data pin, it is connected to pin 25 in order to output the value of both temperature and humidity as serial data. Additionally a 5k pull-up resistor is connected to the 5V regulator due to having a connecting cable shorter than 20 meters as instructed by the datasheet.

According to the ESP32-WROOM-32 datasheet, it is advised to add an RC circuit delay of a resistance of 10k Ohm and a capacitance of 1 microFarad at the EN pin to ensure the power supply to the ESP32 during power-up; additionally we would add 0.1 microFarads which would act as a decoupling capacitor, since there may be a sudden increase in the current draw, this can potentially cause power rail collapse, therefore it is recommended to add the 10 microFarads to the power rail.

Like on the main system, a buzzer will be installed on the servo system, which will be used to create sound giving us an additional advantage in debugging and

testing; it can also be considered as another safety layer as we may program it to produce a sound after a small delay from the servo motor's motion so that it can be when the garage door is in motion providing an audible feedback, offering a clear and a real time indication of movement.

Lastly, we added UART and JTAG pins in our design since it is a common practice and recommended in the ESP32 datasheet, which was explained more in the JTAG and UART section. The UART is connected by TXD0, RXDO and GND in this order, the JTAG is connected to IO14,IO12, IO13 and IO15 via 100 ohm resistors for each line. We added a 3.3V pin as a reference for the debugger and GND pin as well to the JTAG pins. UART pins are added to our schematic since it is simpler and is used by the built in bootloader while the JTAG allows true hardware debugging which may help us in debugging and testing our project as a whole.

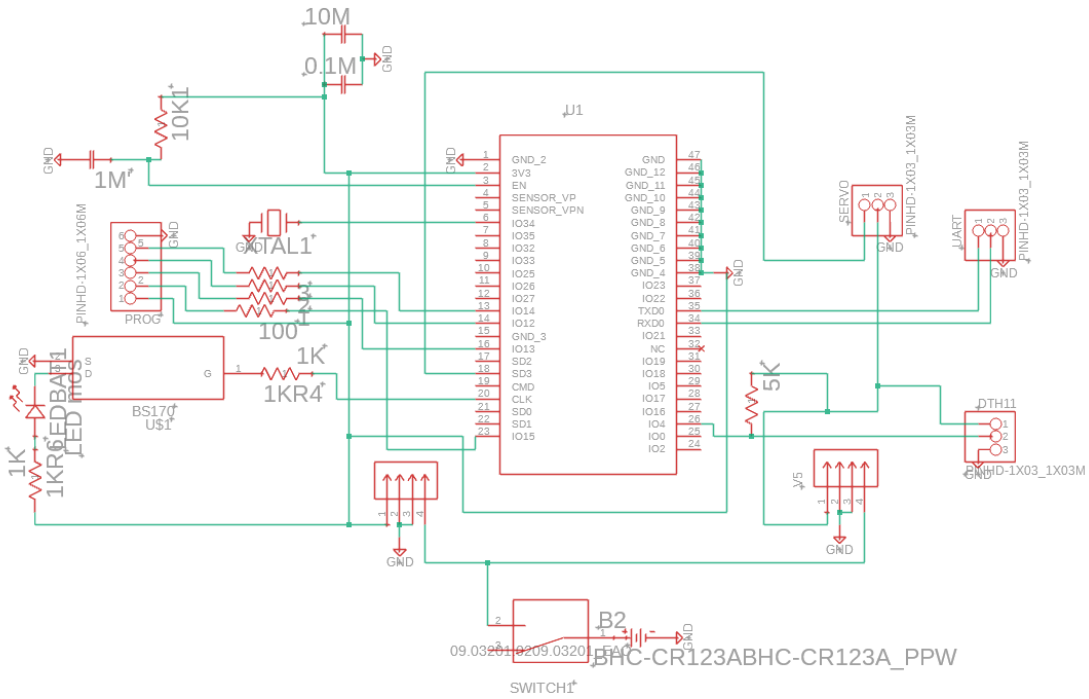


Figure 6.4.2: Servo Device Schematic

Power Systems Schematic

Below is a schematic of the voltage regulators we are going to be using in our project. Both schematics have the same design overall but the regulators are swapped between TPS613221ADBVR, which would be the 3.3V regulator unit and TPS613222ADBVR, which would be the 5V regulator unit. These schematics are illustrated in figure 6.4.3 and figure 6.4.4 respectively.

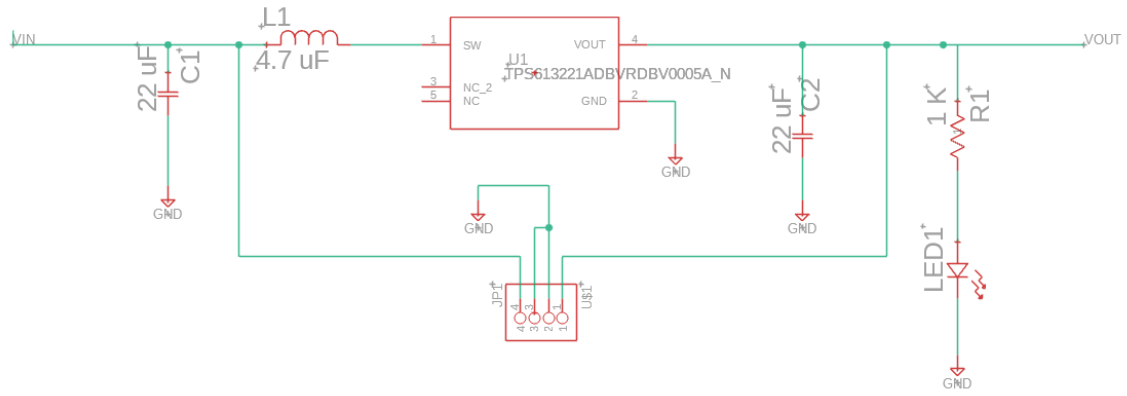


Figure 6.4.3: 3.3V regulator Schematic

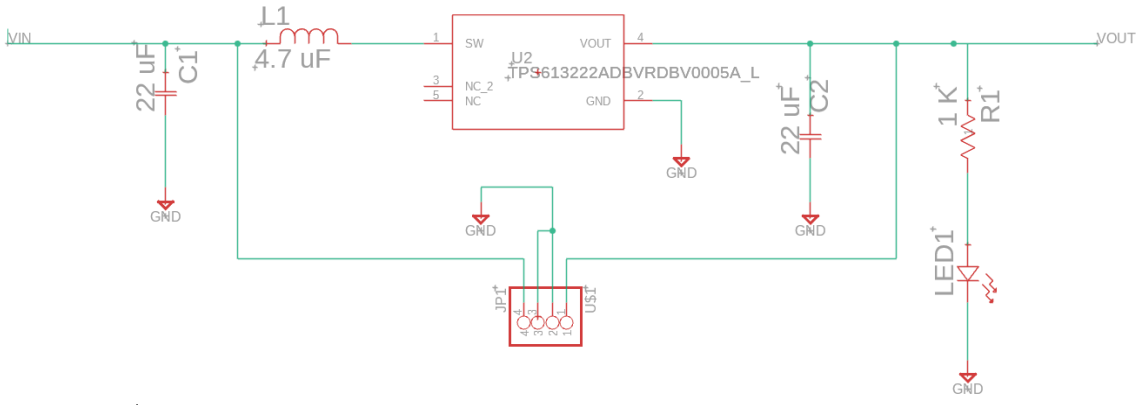


Figure 6.4.4: 5V regulator Schematic

Chapter 7. Software Design

7.1 Use Case Diagram

Prior to beginning the actual coding of the software aspect of our system, it is important to initially understand the actual system we want to create. This is where the Use Case Diagram will be greatly beneficial in breaking down what we want our application to deploy and how it will interact with the device systems and user to create the system we desire.

In Figure 7.1, we can see the system will have a total of three actors - the user, Firebase, and the garage system which is made up of the two individual ESP32s. As outlined in Chapter 2, our team aims to create a web application in which the user will do most of the interaction with the system. In this web application, the user will be able to perform a list of actions, including the most important which is opening and closing the garage door system.

For simplification, the two separate ESP32 microcontrollers are merged into a single garage door system actor. It is important to note that although the MCUs work together to form the device, they are not physically attached together as one. They communicate wirelessly with one another and the Firebase to create the working system. The separate responsibilities for each ESP32, one for the servo motor and another for the reed switch and camera, are detailed in other figures and sections of this report.

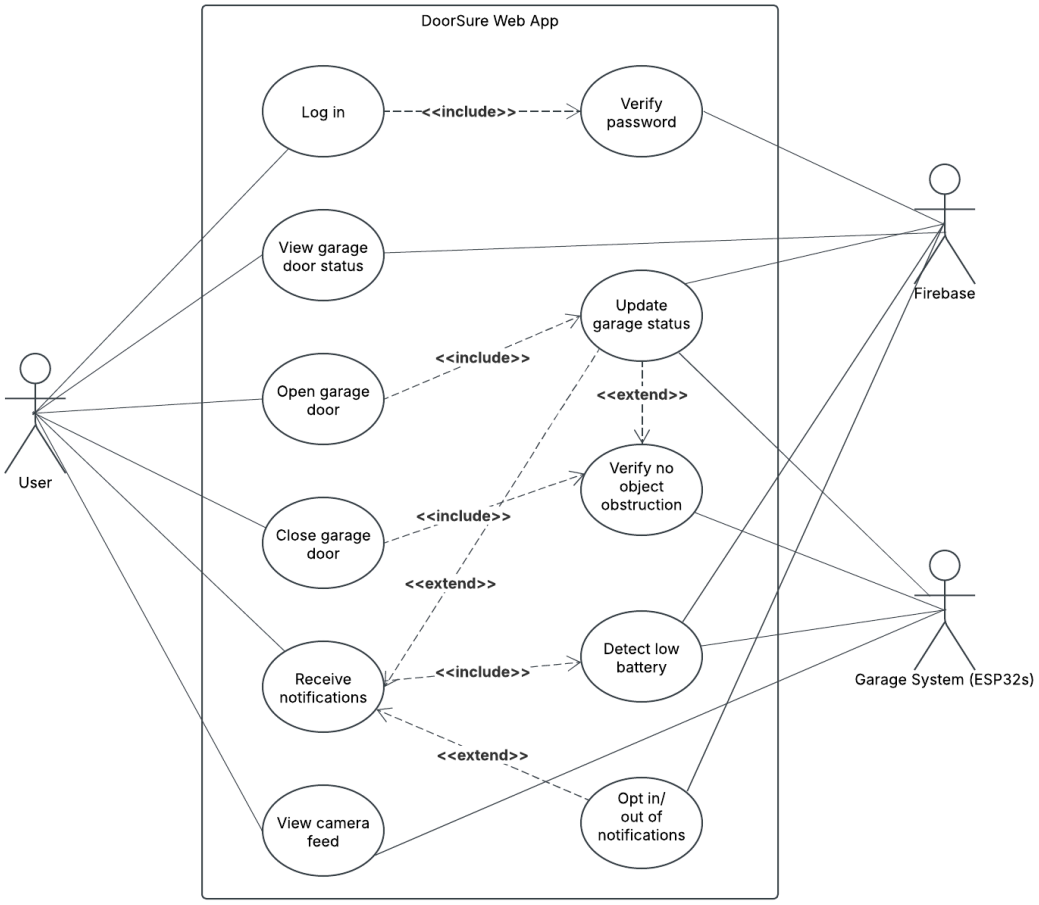


Figure 7.1: Use Case Diagram

7.2 Entity-Relationship Diagram

Before setting up the database, we need to understand how different entities interact with one another. The Entity-Relationship Diagram helps us lay out the relationships between key components, ensuring that all the data is stored correctly.

As shown in Figure 7.2, The Garage is the core of the system, linking everything together. Each User has a unique User_ID, which connects to a Garage_ID, allowing them to control. The Sensor entity tracks whether the garage door is open or an object is in the way, ensuring that users get real-time updates.
The system also includes a Camera linked to the garage for live video monitoring. The Button Device represents the physical button that users can press to open or close the garage. Additionally, the Battery Status entity tracks battery type, charge level, and the last updated time. This helps the system track the battery level and alert the user. Finally, the Notification entity keeps users

informed with all kinds of messages, such as garage status changes, sensor detections, and low battery alerts.

This database structure organizes the system's hardware and software and supports the full functionality of the DoorSure System.

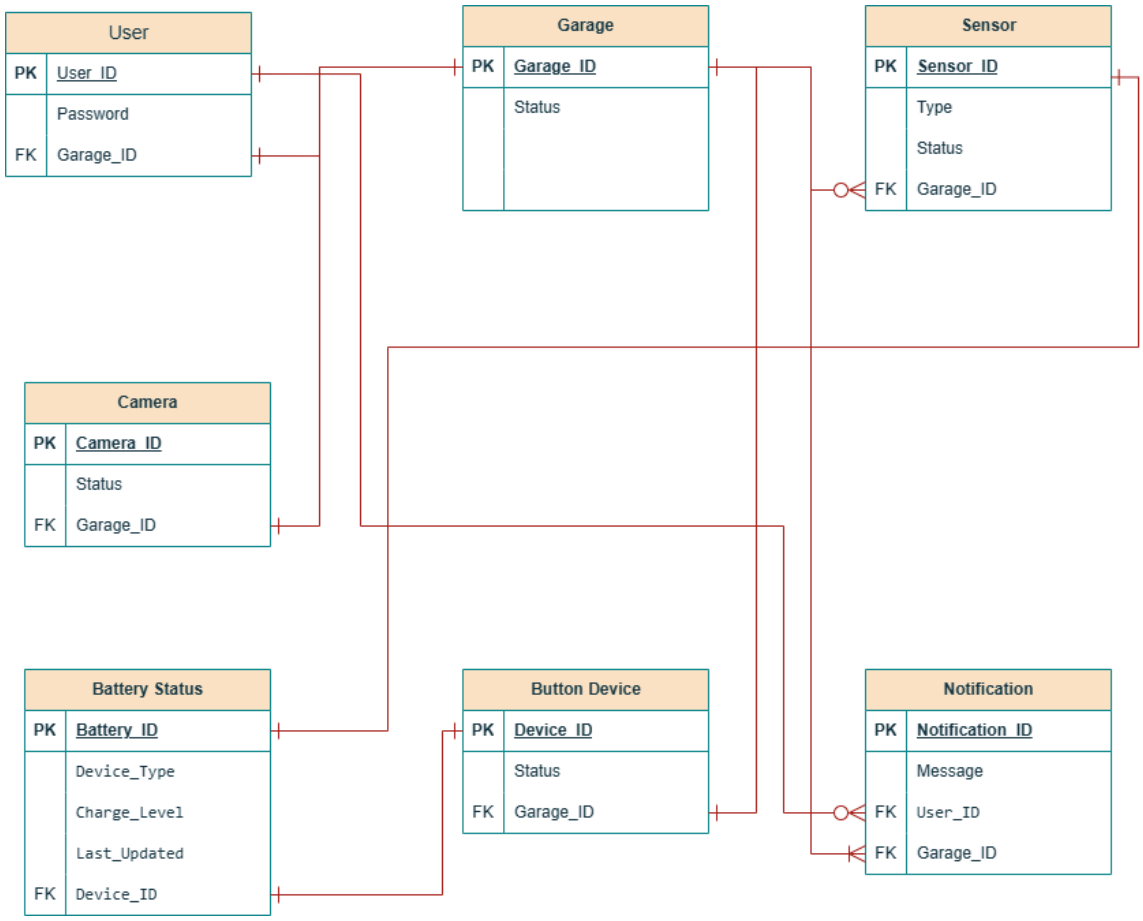


Figure 7.2: Entity-Relationship Diagram

7.3 Class Diagram

Before starting the software development process, it's crucial to outline the different parts of our system and their interactions. This diagram helps us organize the structure by breaking down key components and their relationships. This diagram ensures that everything works together before we start implementing the system.

As shown in Figure 7.3, we can see the system is built on multiple key classes, each representing a specific aspect of the garage door detection system. The User class handles authentication, allowing users to log in and manage their garage. The Garage class is responsible for checking the door's current status and providing a method to open and close it. The ButtonDevice class simulates the physical garage button, enabling users to operate the door remotely. Safety is our top priority, which is why the Sensor class is included to detect obstacles and prevent the door from closing on objects. Additionally, the Camera class allows users to monitor their garage with live video feed.

Power management is another important aspect of our project. The BatteryStatus class keeps track of battery levels, making sure the system remains operational. Finally, the Notification class keeps users informed, sending alerts with all kinds of information.

The DoorSure system seamlessly integrates hardware and software. This well-defined class diagram not only helps us organize the garage door system efficiently but also monitors and controls every process.

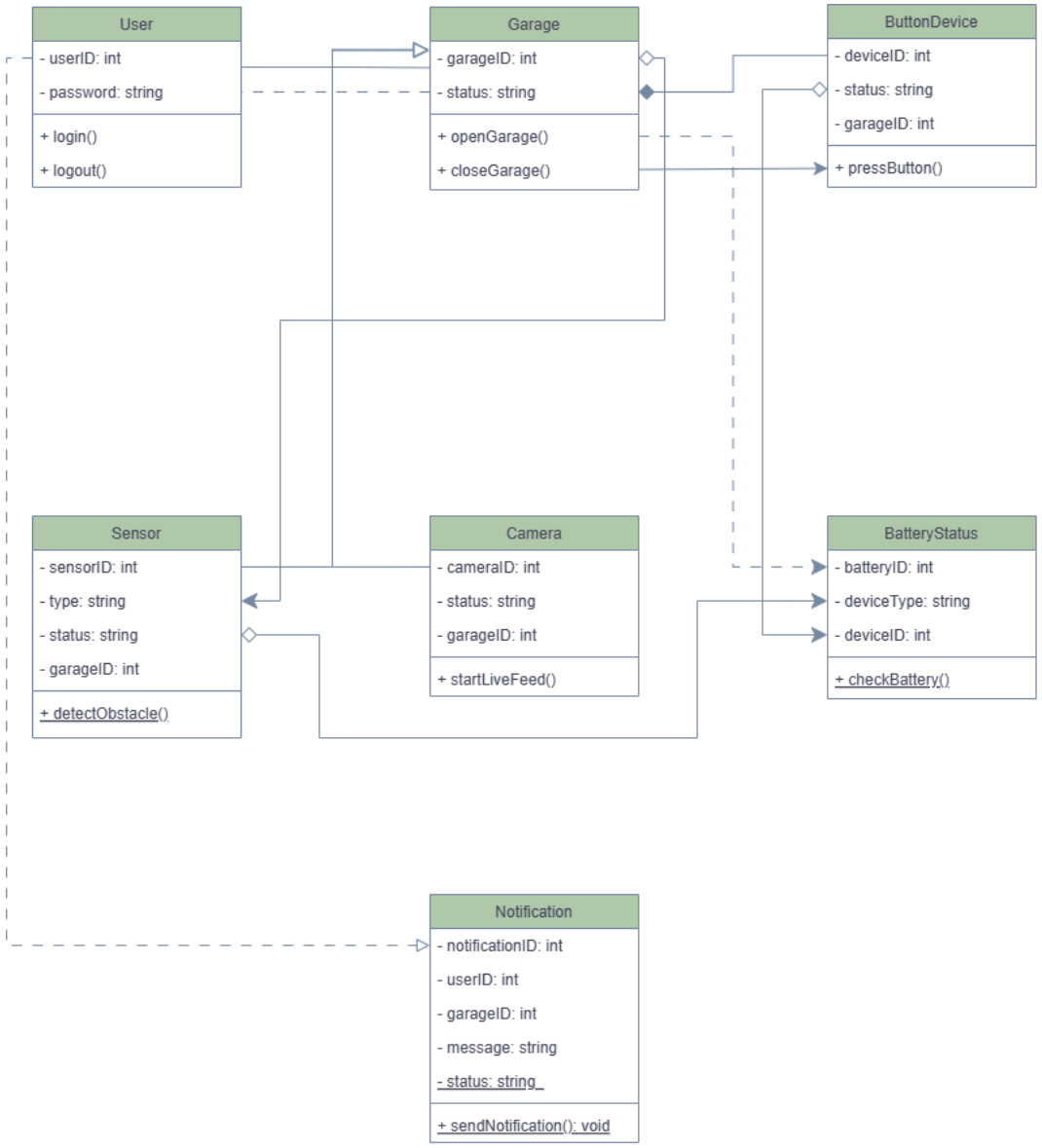


Figure 7.3: Class Diagram

7.4 State Diagram

The State Diagram in Figure 7.4 breaks down how the DoorSure system responds to different states based on user actions. The system starts in an Idle state, continuously waiting for user input. From this state, it can transition based on what happens next. If the user enables the live feed, the system moves to the Live Feed state, allowing real-time video streaming. If the battery is running low,

the system enters the Low Battery Warning state to notify the user to change the battery.

The primary interaction happens when the user presses the button to open or close the garage. If the door is closed and the button is pressed, the system moves to Garage Opening and then to Garage Opened. If the door is opened and the button is pressed, the system moves to Garage Closing and then to Garage Closed. However, if the system detects an object in the way, it stops and notifies the user instead of proceeding.

This state diagram helps us map out exactly how the system should react in different situations, making it smooth and safe. By outlining these different states and their interactions, the diagram provides an overview of how the system should respond to real-world scenarios.

Figure 7.4: State Diagram

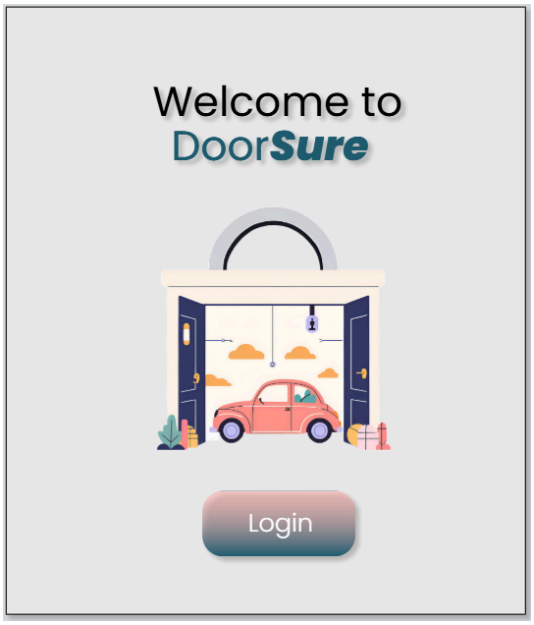
7.5 User Interface

As stated previously, almost all of the user interactions with our system will be made through our web application. As such, it is absolutely crucial that our user interface is clean cut and user friendly. We aim for our application to be as intuitive and easy to understand as possible to allow for an easy user experience for our customers.

Utilizing Penpot to generate the prototype for our web application, we can simulate how the web application will appear in looks, as well as how the user will be able to navigate across screens through interactions. Our web application will consist of three main html pages: a login, a home page, and a live camera feed page.

The login page, shown in Figure 7.6.1 shown in the left, will be simple, consisting of our project logo, a welcome text, and a login button. We will implement google login for our web application, so pressing the button will lead the user to a new webpage where they will be able to login accordingly.

The settings page, also shown in the right of Figure 7.6.1 will consist of two main sections – the settings and the notification preferences. From this page, the user will be able to change their Wi-Fi settings, notification preferences. The user will also be able to set a custom amount from a drop down menu for notifications after their garage door has been left open for x amount of minutes.



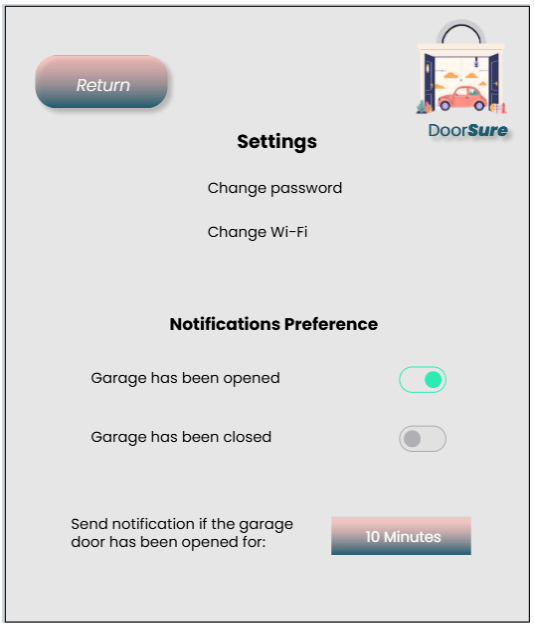


Figure 7.5.1 Login Page (left), Settings Page (right)

The home page will showcase the current status of the garage, grabbed from the RTDB. From this page, the user will be able to press a single button. If the current status is “open” the user will press the button to close. If the current status is “closed” the user will press the same button to open. Each button press will ultimately result in a change to the database accordingly, granted no objects are blocking the path of the garage door.

Along with this function, the home page will also have an image of a gear that can be pressed to bring up the settings HTML. The user will also be able to press a button “view live feed” which will then navigate to the life feed page, pictured on the right of Figure 7.5.2.

The live feed page will showcase the livestream footage from the ESP-CAM. The footage will only be displayed in the app on this page. If there is an obstruction that is causing the garage door to not be allowed to close, the object that is causing this will be displayed in the recent log area.

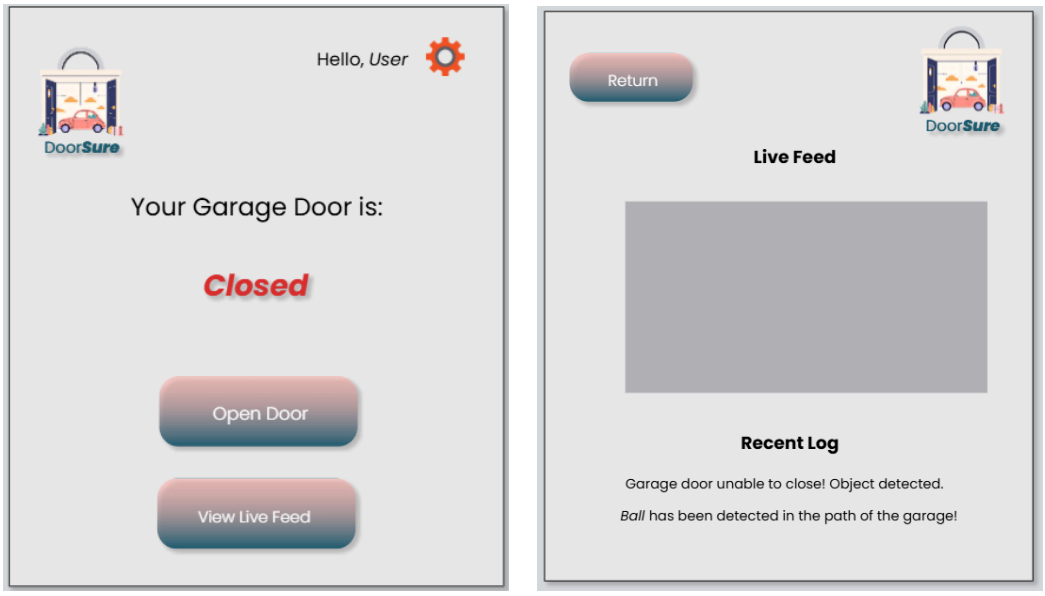


Figure 7.5.2 Home Page (left), Live Feed Page (right)

7.6 Development Environments and Tools

7.6.1 Programming Languages

Our web application system will be programmed using the standard languages – HTML, CSS, and JavaScript language. This will ensure cross-browser compatibility as we will be following modern web development standards. The program code for the chosen microcontrollers, the ESP32, will be written in C.

7.6.2 IDEs

As ESP32s are the microcontroller chosen for our project, we will be utilizing Arduino IDE for all source code written and uploaded to the microcontrollers. Along with using Arduino IDE, we will also be utilizing VSCode for all programming not related to the ESP32. Although our web application is directly linked and deployed using Firebase for testing purposes, we will require the IDE to write programs and debug.

7.6.3 Libraries and Dependencies

Within the Arduino IDE, we will be utilizing various libraries provided through both GitHub and the built in library manager in Arduino. The libraries used are as follows:

- ArduinoJson by Benoît Blanchon
    - This library can be accessed by including <ArduinoJson.h>
    - A powerful JSON serialization/deserialization library for embedded systems like the ESP32. It is used to parse, generate, and manipulate JSON data efficiently. In this project, it plays a key role in formatting data exchanged between the ESP32 and cloud services such as Firebase, ensuring proper data structure for communication.

- Async TCP by ESP32Async
    - This library can be accessed by including <AsyncTCP.h>
    - This library works in tandem with the ESP Async WebServer to create an object that listens to the server port.

- ESP Async WebServer by ESP32Async
    - This library can be accessed by including <ESPAsyncWebServer.h>

    - This library works in tandem with the Async TCP, responsible for creating an Async WebServer object to create an object that listens to a server port number. This allows communications with the web application.

- Wi-Fi ESP32 Library
    - This library can be accessed by including <WiFi.h> after adding the ESP32 into device manager in Arduino.
    - This library was used to connect the ESP32s to the WiFi network. It was also used to identify the IP Address of the WiFi connection, allowing a websocket to then be generated with the web application for communications with Firebase.

- Firebase Arduino Client Library For Arduino Devices by Mobizt
    - This library can be accessed by including <Firebase_ESP_Client.h>.
    - This library allows us to define our Firebase project configurations and authentication tokens, allowing the ESP32 access to write and read data from the Firebase RTDB.

- ESP32Servo by Kevin Harrington and John K
    - This library can be accessed using <ESP32Servo.h>.
    - This library was used to create a servo object in the program code, which then allowed us to program angle movements on the servo.

7.6.4 Web Application Deployment and GitHub

Our web application will be tested and deployed directly through Google Cloud Platform's Firebase. We will be utilizing both RTDB and Firestore according to the data types needed to perform our system actions. The RTDB will be used to store variables that are expected to change frequently, such as the garage door status. The Firestore database will be used to store variables that can change but are much less frequent to change, like Wi-Fi SSID, Wi-Fi password, user password, notification preferences, etc…

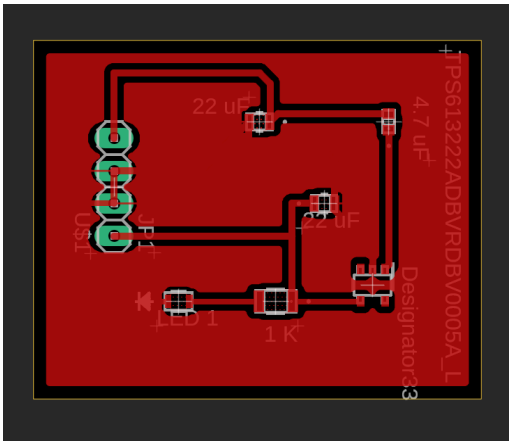
For testing updates before deployment we will use a combination of VSCode, the command terminal and local host. In order to secure version history and backups of our work, in case of any software breaking bugs or accidental code deletions, we will utilize a GitHub Repository. This also ensures that all four team members will have constant and up to date access to all source code concerning the project.

Chapter 8. System Fabrication/ Prototype Construction

Live Demonstration Model Creation

A small scale demonstration model of a garage door will be created to simulate the installation of the DoorSure system and what the proper behavior should look like, we intend for this model to mimic real life garage doors, achieving key features as illustrated in table 2.51 our project such as garage door status accuracy, battery life, object detection accuracy and object identification accuracy, with this small scale model we intend to mimic the entire experience of owning our system from installation to full scale testing and operation, this will enable us to evaluate the whole system's performance under realistic conditions with different real life scenarios to ensure our system's reliability and functionality.

Voltage regulator PCB



Chapter 9. System Testing and Evaluation

Chapter 10. Administrative Content

10.1 Budget

After researching the project and discussing the allocated budget for each team member, we agreed that every team member can spend up to $200 which will be divided among the components in the bill of materials. This list is subject to changes based on time and money constraints. Since our team is composed of 4 team members, our total budget is $800. The huge gap in the total amount spent on components illustrated in the bill of material and allocated funds by each team member, allows us to have faster delivery for the components and PCBs. Additionally, BOM gives us more liberty when it comes to brainstorming and executing more creative and innovative solutions to problems we may find along the way for this project. We recognize we may accidentally damage components along the way during testing and prototyping so these extra funds will give us the opportunity to replace those components as well as add any additional components as needed.

10.2 Bill of materials

Table 10.2: Bill of Materials

Components	Details	Vendor	Quantity	Unit price	Total price
Microcontroller	ESP 32s 2.4 GHZ dual mode WIFI + bluetooth dual-core microcontroller integrated with antenna RF AMP filter	Amazon	2	$8.99	$17.98
Camera	ESP 32-cam MB 5v wifi bluetooth CH340 OV2640/8225N	Amazon	2	$13.99	$27.98

	camera				
Reed switch	4 pcs door sensor magnetic switch	Amazon	1	$7.99	$7.99
Replaceable batteries	Duracell coppertop AA	Amazon	4	$1.18	$4.72
Servo motor	MG90 180-degree servo motor 4PK	Amazon	1	$13.99	$13.99
Colored LEDs	100PK of LEDs for prototyping	Amazon	1	$4.99	$4.99
Total:					$77.65

10.3 Work Distribution

After multiple meetings with the team and assessment of everyone's personal and technical skills, we have created the following work distributions table. As we are all current students juggling differing work and class schedules, it is important to note that we are open to helping one another out with tasks as needed. If someone is struggling with a certain aspect or needs assistance, as we are a team working towards one goal, we will help one another in any way we can. Regardless of that fact, members are still expected to make continuous progress in their assigned tasks in order to ensure we meet the deadlines specified in our milestones. Senior Design is a rigorous course and as such we must keep one another in check to ensure proper completion of our project. Furthermore, our team will have two meetings per week to ensure workload is sufficient for each member and to have continuous progress checks. This is done to ensure the project is moving ahead with each passing week and to adjust tasks as necessary based on actual progress and outcomes.

Our team is made of two Electrical Engineers and two Computer Engineers. As such, it was natural for the software and hardware parts to be split accordingly. However, to challenge ourselves and grow our technical skills through this project, we ensured everyone had a mix of work between the two tasks. This is important to not only develop and hone our skills, but also showcase what we have learned throughout our coursework here at University of Central Florida.

Table 10.3: Work Distribution

Hardware		
Task	Primary	Secondary
Prototype Design	Mina Ghandour	Khang Nguyen
PCB Design	Khang Nguyen	All
Power System Engineering	Khang Nguyen	Mina Ghandour
Housing Design	Felix Liu	Khang Nguyen
Microcontroller Selection	Khang Nguyen	Felix Liu
Microcontroller Implementation	Andrea Rosa	Mina Ghandour
Battery Selection	Mina Ghandour	Khang Nguyen
Battery Integration	Khang Nguyen	Mina Ghandour
Camera Selection	Mina Ghandour	Felix Liu
ESP-CAM Integration	Felix Liu	Andrea Rosa
Reed Switch Integration	Khang Nguyen	Andrea Rosa
Servo Motor Integration	Andrea Rosa	Mina Ghandour
Software		
Task	Primary	Secondary
Cloud Server Management	Andrea Rosa	Felix Liu
WebSocket Integration	Andrea Rosa	Felix Liu
RTDB Management	Felix Liu	Andrea Rosa
Firestore Database	Andrea Rosa	Felix Liu
User Authentication	Felix Liu	Andrea Rosa

Notification System	Felix Liu	Andrea Rosa
UI Design	Mina Ghandour	Andrea Rosa
UI Implementation	Andrea Rosa	Felix Liu
Senior Design Project Website	Mina Ghandour	Andrea Rosa
User Setup Bluetooth Integration	Khang Nguyen	Felix Liu

10.4 Milestones

Below are the team's hard deadlines for Senior Design I and Senior Design II. These milestones were generated based on the course outline. In order to ensure our team will meet the deadlines listed and successfully complete the course with a working finished prototype, it is absolutely crucial that our team not only meets the expectations associated with the deadlines, but continues to strive above and beyond what is expected for the project. Additionally, note that no deadlines were written in or meeting dates specified with our review committee, but our team will be ensuring that our reviewers continue to stay up to date with our project and progress as the course continues. Also note that the milestones for Senior Design II are broad as our team begins work for Senior Design II. Based on our progress on hardware and software, we will create the necessary milestones for the Summer term as needed to ensure success in our project.

Table 10.4.1: Milestones for Senior Design I

Task		Start	End	Status
1	Team Formation	Week 1		Completed
2	Project Ideas	Week 2		Completed
3	Divide & Conquer	1/17/2025	1/23/2025	Completed
4	ABET Lectures	1/21/2025	2/13/2025	Completed
5	Divide & Conquer Meeting	1/28/2025		Completed

6	Create and Update the Group Website	1/28/2025	2/7/205	Completed
7	ABET Quizzes (A to G)	2/8/2025	3/7/2025	Completed
8	Spring Break	3/17/2025	3/22/2025	N/A
9	Midterm Report	2/7/2025	3/25/2025	In progress
10	Midterm Report Meeting	3/26/2025	3/28/2025	Completed
11	Update Group Website With Midterm Report	3/28/2025	4/4/2025	Completed
12	Final Report & Mini Demo Video	4/4/2025	4/22/2025	Incomplete

Table 10.4.2: Milestones for Senior Design II

Task		Start	End	Status
1	Build Prototype	May 2025		Incomplete
2	Software Build	June 2025		Incomplete
3	Testing Phase	July 2025		Incomplete
4	Final Presentation	Final Week		Incomplete
5	Final Report	Final Week		Incomplete

Chapter 11. Conclusion

APPENDIX A - References

[1] FBI — Property Crime

[2] US Burglary Statistics 2024

[3] CHAMBERLAIN Smart Garage Control Product
https://www.amazon.com/Smart-Garage-Opener-Chamberlain-myQ-G0401/dp/B08GD3D9YJ?pd_rd_w=wROxZ&content-id=amzn1.sym.55c0153f-1fb7-42ff-8241-d1c0f3732289&pf_rd_p=55c0153f-1fb7-42ff-8241-d1c0f3732289&pf_rd_r=ESZSGMK7RD2924642GRS&pd_rd_wg=mezPF&pd_rd_r=66378763-1e5b-4f0e-94c1-6a46d390b9ba&ref_=sspa_dk_detail_img_1&sp_csd=d2lkZ2V0TmFtZT1zcF9kZXRhaWxfdGhlbWF0aWM&th=1

[4] CHAMBERLAIN myQ Smart Indoor Security Camera
https://www.amazon.com/dp/B0D2LW9M4P/ref=sspa_dk_hqp_detail_aax_0?sp_csd=d2lkZ2V0TmFtZT1zcF9ocXBfc2hhcmVk&th=1

[5] Genie Aladdin Connect
https://www.amazon.com/Genie-Company-ALKT1-R-Smartphone-Anywhere/dp/B016QB7K5O?maas=maas_adg_6D4C85CB50484BEBD447209BF1C8A89E_afap_abs&ref_=aa_maas

[6] Pros and Cons of Cloud Storage | Secure Storage

[7] https://www.ipc.org/TOC/IPC-2221A.pdf

[8] https://jlcpcb.com/capabilities/pcb-capabilities

[9]https://standards.ieee.org/beyond-standards/the-evolution-of-wi-fi-technology-and-standards/#:~:text=IEEE%20802.11g%E2%84%A2%2C%20or,600%20Mbit/s%20data%20rates.

[10]https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

[11]https://github.com/SeeedDocument/forum_doc/blob/master/reg/ESP32_CAM_V1.6.pdf

[12]https://www.handsontec.com/dataspecs/module/ESP32-CAM.pdf

[13]https://www.electronicoscaldas.com/datasheet/MG90S_Tower-Pro.pdf?srsltid=AfmBOoqfugkH59uMXet_zGf2OzQt3r54B38oOEq4il2xz7tb0bQHWPAP

[14]https://media.digikey.com/pdf/Data%20Sheets/DFRobot%20PDFs/DFR0602_Web.pdf

[15]https://www.silabs.com/documents/public/data-sheets/CP2102-9.pdf

BEAUD, D. “LiDAR: What Is It and How Does It Work?” *YellowScan*, https://www.yellowscan.com/knowledge/how-does-lidar-work/. Accessed 4 March 2025.

CD-Team. “IR Sensor | Basics, Types, Circuit, Working, Projects, FAQs.” *Electronics For You*, 28 August 2023, https://www.electronicsforu.com/technology-trends/learn-electronics/ir-led-infrared-sensor-basics. Accessed 4 March 2025.

“A Complete Guide to Reed Switches.” *RS Components*, 1 February 2023, https://uk.rs-online.com/web/content/discovery/ideas-and-advice/reed-switches-guide. Accessed 19 February 2025.

JU. “Hall Effect – Principle, Theory, Formula, Applications & FAQs.” *BYJU'S*, https://byjus.com/physics/hall-effect/. Accessed 27 February 2025.

MaxBotix Inc. *How Ultrasonic Sensors Work*, https://maxbotix.com/blogs/blog/how-ultrasonic-sensors-work?srsltid=AfmBOooYpkQyWUQn5Dv840byDnwifm_vv4nrKygyQtyl_puzLth2fL0y.

Monolithic Power Systems, Inc. “Types and Functions of Sensors in Automotive Systems.” *MPS*, https://www.monolithicpower.com/en/learning/mpscholar/automotive-electronics/automotive-sensing-and-actuators/types-and-functions-of-sensors?srsltid=AfmBOorwYqXal-fPA1_yNcHlcxKkurw55EcwsW5Z3be-07hm5ET5SvlU.

Sparks, Geoffrey. “Database Modelling in UML.” *Methods & Tools*, https://www.methodsandtools.com/archive/archive.php?id=9.

APPENDIX B - Parts

[1] Servo Motor Part
https://www.amazon.com/DIYables-Degree-Arduino-ESP8266-Raspberry/dp/B0BPFXTZ73/ref=asc_df_B0BPFXTZ73?mcid=6caed98ed8e23609961ab0c9ef94c526&hvocijid=12273598725803836168-B0BPFXTZ73-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=12273598725803836168&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9011767&hvtargid=pla-2281435176658&th=1

[2] AA Duracell replaceable batteries
https://www.amazon.com/dp/B00000JHQ6/ref=twister_B0CX2ZM9CF?_encoding=UTF8&th=1

[3]DoorSensor
https://www.amazon.com/DIYables-Magnetic-Arduino-ESP8266-Raspberry/dp/B0B3D7BM4K/ref=asc_df_B0B3D7BM4K?mcid=3da3e052d4453e6d825a02bcd1e45e49&hvocijid=187620812624435686-B0B3D7BM4K-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=187620812624435686&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9011767&hvtargid=pla-2281435177818&psc=1

[4]UltrasonicSensor
https://www.amazon.com/HC-SR04-HC-SR04P-Ultrasonic-Distance-Measuring/dp/B07KNTQ4C2/ref=asc_df_B07KNTQ4C2?mcid=9601b6d8b1df3f319306a5d594025b6e&hvocijid=14937213940166474350-B07KNTQ4C2-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=14937213940166474350&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9057291&hvtargid=pla-2281435178138&psc=1

[5] ESP-32S Development Board (MCU)
https://www.amazon.com/ESP-WROOM-32-Development-Dual-Mode-Microcontroller-Integrated/dp/B07WCG1PLV/ref=asc_df_B07WCG1PLV?mcid=7cc24e50dc423e8e9f1039421f865b8a&hvocijid=3062641011911185649-B07WCG1PLV-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=3062641011911185649&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9011767&hvtargid=pla-2281435177418&th=1

[6] ESP32-CAM-MB 5V WiFi Bluetooth
https://www.amazon.com/ESP32-CAM-ESP32-CAM-MB-Bluetooth-Development-CH340G/dp/B0CBQ7DDM8/ref=asc_df_B0CBQ7DDM8?mcid=e0c8b396601

e33dd9dbe848de410cfea&hvocijid=758869657899358644-B0CBQ7DDM8-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=7588696578993586 44&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9057291&hvtargid=pla-2281435177818&psc=1

[7] Colored LEDs
Amazon.com: (100 Pcs) MCIGICM 5mm LED Light Diodes, LED Circuit Assorted Kit for Science Project Experiment (Multi-Colored - 5 Color) : Industrial & Scientific

Camera development board

https://www.adafruit.com/product/4095

https://www.adafruit.com/product/5955

https://www.aliexpress.com/item/32971057846.html

https://www.amazon.com/dp/B0CX93M5DW/ref=sspa_dk_detail_1?psc=1&pd_rd_i=B0CX93M5DW&pd_rd_w=bnyNA&content-id=amzn1.sym.7446a9d1-25fe-4460-b135-a60336bad2c9&pf_rd_p=7446a9d1-25fe-4460-b135-a60336bad2c9&pf_rd_r=TCX8W2RAC75ZEV0N69FZ&pd_rd_wg=lxlpP&pd_rd_r=9331f95f-ce24-4a98-bbda-07eb9f03fa55&s=photo&sp_csd=d2lkZ2V0TmFtZT1zcF9kZXRhaWw

https://www.amazon.com/Seeed-Studio-XIAO-ESP32-Sense/dp/B0C69FFVHH/ref=dp_prsubs_d_sccl_1/133-6285833-1780855?pd_rd_w=kpYYq&content-id=amzn1.sym.3ad0ccdf-fd9f-4ec9-a400-2b1165fdcd58&pf_rd_p=3ad0ccdf-fd9f-4ec9-a400-2b1165fdcd58&pf_rd_r=TCX8W2RAC75ZEV0N69FZ&pd_rd_wg=trlfY&pd_rd_r=6626049d-9130-4dee-acc2-0a208f942ec3&pd_rd_i=B0C69FFVHH&th=1

Camera modules

https://www.amazon.com/Acxico-OV2640-Wide-Angle-Million-Interface/dp/B09DSZDCTX/ref=asc_df_B09DSZDCTX?mcid=8820535597353f8192dda29338d8c3cd&hvocijid=4629284419412209855-B09DSZDCTX-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=730352155585&hvpos=&hvnetw=g&hvrand=4629284419412209855&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9057291&hvtargid=pla-2281435177858&psc=1

https://www.amazon.com/Acxico-OV2640-Wide-Angle-Million-Interface/dp/B09DSZDCTX/ref=asc_df_B09DSZDCTX?mcid=8820535597353f8192dda29338d8c3cd&hvocijid=4629284419412209855-B09DSZDCTX-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=730352155585&hvpos=&hvnetw=g&hvrand=4629284419412209855&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9057291&hvtargid=pla-2281435177858&psc=1

https://www.ebay.com/itm/394461368220?chn=ps&mkevt=1&mkcid=28&google_free_listing_action=view_item&srsltid=AfmBOorpgIYxkWtnIEJ1Pt8cWfRWSebc-ENZ5UrcLODkEi8wcL-ZGgGrhgk&gQT=2

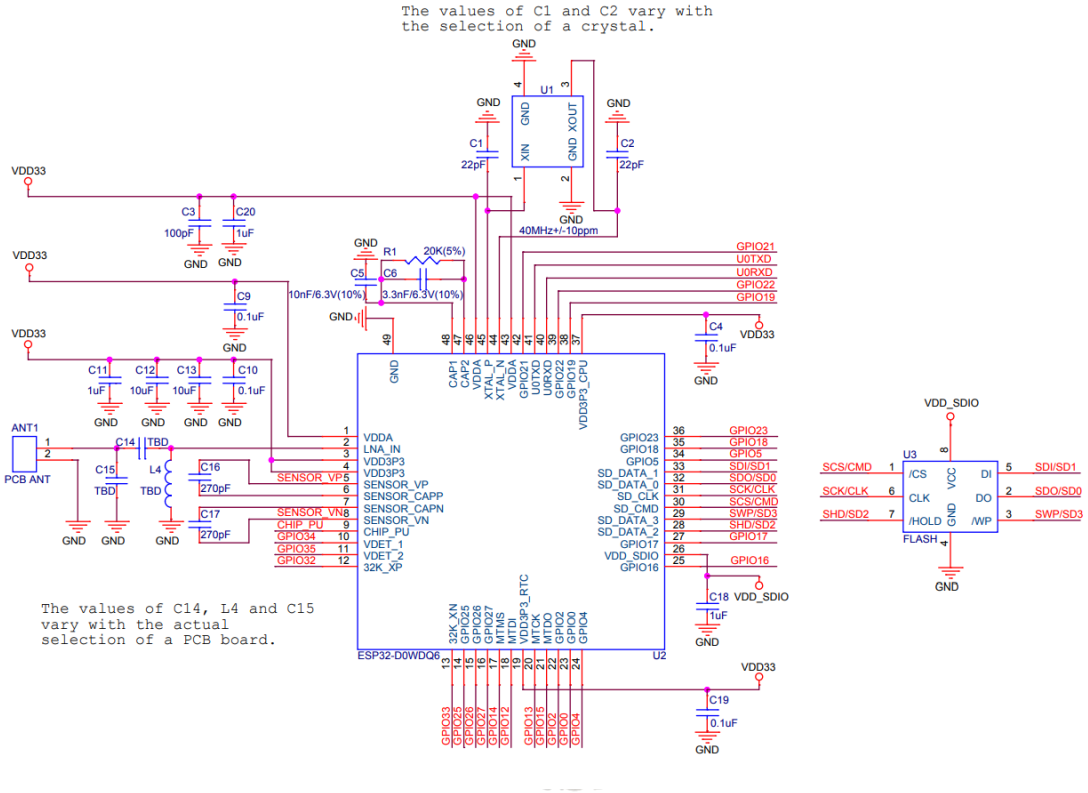
https://www.ebay.com/itm/335617990655?chn=ps&_trkparms=ispr%3D1&amdata=enc%3A17qftNN8zSACBwVsSenjsuQ34&norover=1&mkevt=1&mkrid=711-117182-37290-0&mkcid=2&mkscid=101&itemid=335617990655&targetid=2320093655185&device=c&mktype=pla&googleloc=9057291&poi=&campaignid=21676663813&mkgroupid=175573447188&rlsatarget=pla-2320093655185&abcId=10012304&merchantid=101681819&gad_source=1&gclid=CjwKCAjwvr--BhB5EiwAd5YbXqjR0RGnOY-kwcHpHAlWSZufxXYeCg1BcSH9k0Sq4Heypm9PXilJQxoCfJ4QAvD_BwE

https://www.aliexpress.us/item/3256806050145546.html?src=google&pdp_npi=4%40dis%21USD%214.72%214.72%21%21%21%21%21%40%211200003640879814 1%21ppc%21%21%21&src=google&albch=shopping&acnt=708-803-3821&isdl=y&slnk=&plac=&mtctp=&albbt=Google_7_shopping&aff_platform=google&aff_short_key=UneMJZVf&gclsrc=aw.ds&albagn=888888&ds_e_adid=&ds_e_matchtype=&ds_e_device=c&ds_e_network=x&ds_e_product_group_id=&ds_e_product_id=en3256806050145546&ds_e_product_merchant_id=5295628431&ds_e_product_country=US&ds_e_product_language=en&ds_e_product_channel=online&ds_e_product_store_id=&ds_url_v=2&albcp=20542171673&albag=&isSmbAutoCall=false&needSmbHouyi=false&gad_source=1&gclid=Cj0KCQjw4cS-BhDGARIsABg4_J3R1nl3f9pUUD5bPUe5Bu7yGQe2qRuB4Qx9N7x1lCfZVezF_OE_NygaAvm3EALw_wcB&gatewayAdapt=glo2usa

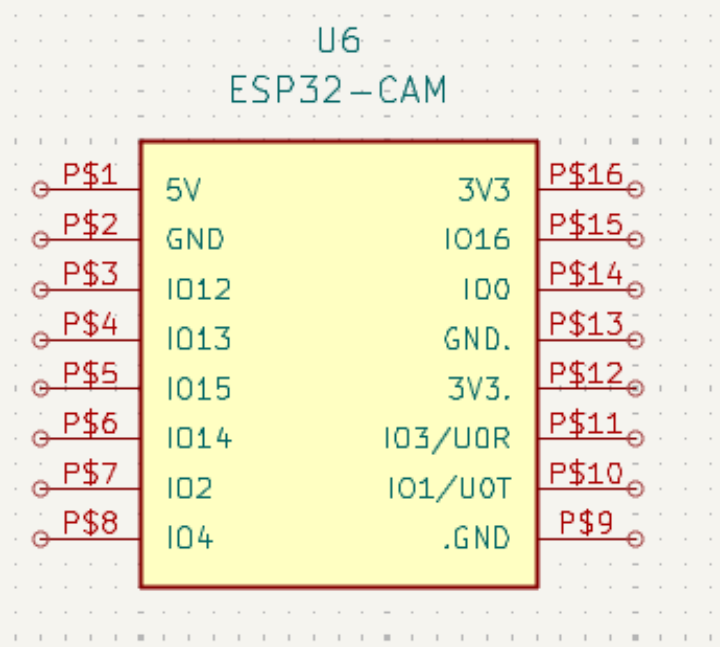
https://www.amazon.com/Taidacent-Interface-Sensor-NT99141-fisheye/dp/B097872PWY?th=1

APPENDIX C - Data sheet

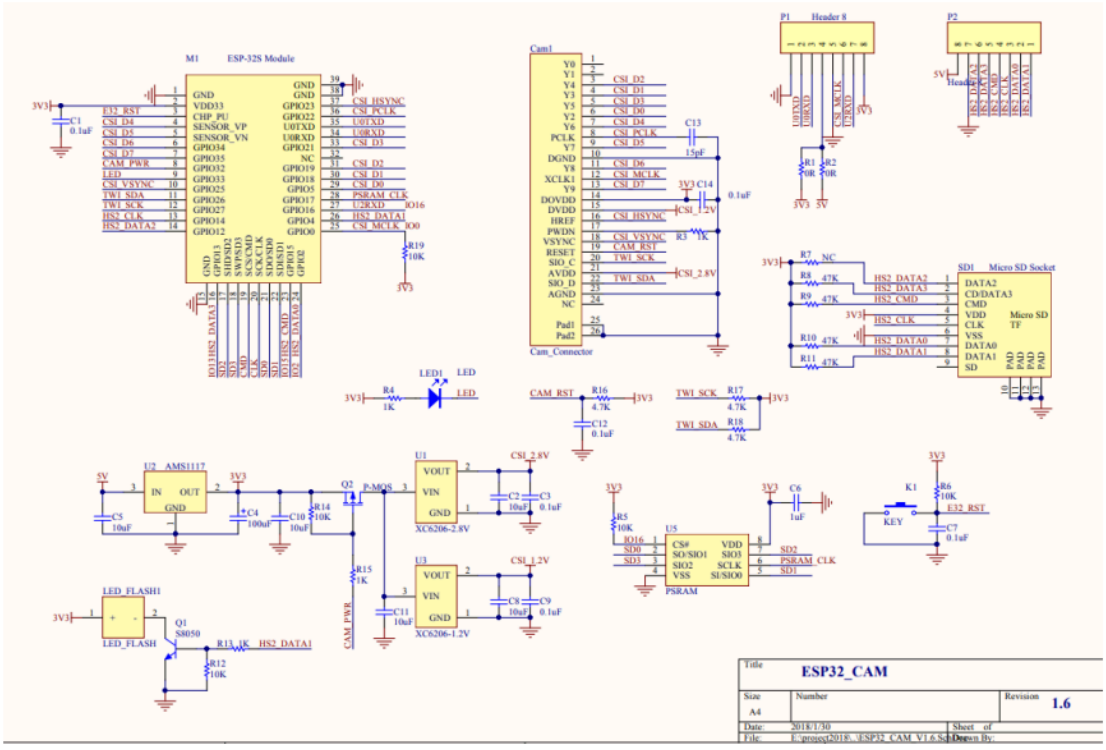
6.1.1: Schematics of ESP32-WROOM-32 from Espressif



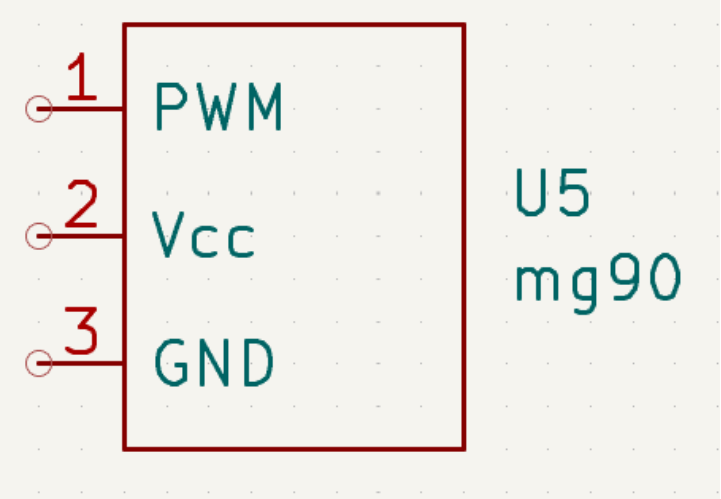
6.1.2: Symbol of ESP32-CAM



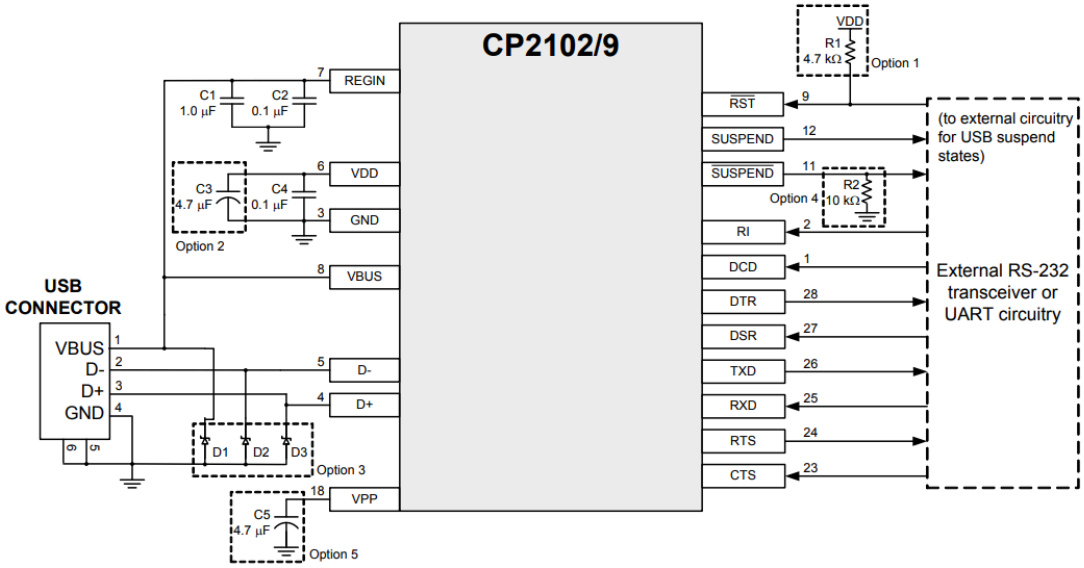
6.1.3: Schematic of ESP32-CAM



6.2.1: Symbol of MG90s Servo Motor



6.5: CP2102 schematic from Silicon Labs



APPENDIX D - LLM Prompts

Prompt 1:
What kind of motor would be efficient to use to push the garage door button?

ChatGPT response:
A small linear actuator or a servo motor would be efficient for pushing a garage door button. Here's why:

Servo Motor

Precise control over movement and force.
Can be programmed to push and release the button reliably.
Compact and low power consumption (e.g., a small 9g or 20g servo).
Linear Actuator

Converts rotational motion into a straight push.
Provides consistent force to press the button.
Available in various stroke lengths to match the button's travel distance.
Solenoid Actuator

Directly pushes and retracts when powered.
Simple and fast operation, but limited control over force and speed.
A small servo motor with a pushrod or a short-stroke linear actuator would be the best balance of efficiency, precision, and ease of control with an ESP32 or other microcontroller. Do you have any size or power constraints?

Gemini Response:
When considering a motor to efficiently push a garage door button, several factors come into play, including precision, power, and size. Here's a breakdown of suitable options:

1. Solenoids:

Pros:
Simple and cost-effective.
Provide a linear push motion, ideal for buttons.
Quick response time.
Cons:
Limited stroke length.
May require careful mounting for precise alignment.

They are either on, or off. There is no inbetween.
Ideal for:
Short, direct button presses.
2. Linear Actuators:

Pros:
Precise control over stroke length and speed.
Available in various sizes and power ratings.
Durable and reliable.
Cons:
Generally more expensive than solenoids.
May require more space.
Ideal for:
Situations requiring adjustable or controlled button presses.
3. Stepper Motors:

Pros:
Extremely precise positioning.
Excellent for applications requiring repeatable movements.
Can be used to turn a screw type mechanism to create linear motion.
Cons:
Require more complex control circuitry.
May be overkill for a simple button press.
Ideal for:
Applications where precise control and repeatability are paramount.
Key Considerations:

Force:
Determine the force required to depress the garage door button.
Stroke Length:
Measure the distance the button needs to be pressed.
Control:
Consider how you will control the motor (e.g., relay, microcontroller).
Power Source:
Ensure the motor's voltage and current requirements are compatible with your power source.
Mounting:
How will the motor be mounted, and how much space is available.
In most cases, for a simple garage door button push, a solenoid or a small linear actuator would be the most efficient and practical choice.

I hope this information is helpful.

Copilot response:
To press a garage door button efficiently, you would want to use a small DC motor or a linear actuator. Here's a breakdown:

Small DC Motor: They are compact, lightweight, and energy-efficient. Ideal for creating a mechanism with gears and levers to simulate the pushing motion. Examples include micro gear motors or hobby motors.

Linear Actuator: They convert rotary motion into linear motion, providing precise control over the push action. They're slightly more complex but offer reliable performance for this purpose.

A microcontroller like an Arduino can easily control both of these motors, allowing you to automate the process.

Does that sound like what you need, or would you like more information on either of these options?
Prompt 2:
for a system that requires low power when in use and when it's idle, along with long battery life, what size of battery should I consider using? (AA, AAA, 9V, etc.)

ChatGPT response:

The battery size depends on your system's power requirements (both active and idle), expected runtime, and physical constraints. Here's a breakdown of common options:

Key Considerations:

1. **Power Consumption:** Measure the system's current draw in both active and idle states.
2. **Voltage Requirements:** Ensure the battery voltage matches your system (or use a regulator).
3. **Capacity (mAh or Wh):** Higher capacity means longer runtime.
4. **Self-Discharge Rate:** Lower self-discharge is better for long-term idle systems.

Battery Options:

- **AAA (1.5V, ~1000 mAh)** – Small, lightweight, good for ultra-low-power applications.

- **AA (1.5V, ~2500 mAh)** – Better for moderate-power devices, lasts longer than AAA.
- **9V (~500 mAh)** – Higher voltage, but low capacity, not ideal for long battery life.
- **Lithium Coin Cells (e.g., CR2032, 3V, ~240 mAh)** – Good for very low-power applications.
- **Li-ion (e.g., 18650, 3.7V, ~3000 mAh)** – Rechargeable, high capacity, suitable for ESP32-based projects.
- **LiPo (e.g., 3.7V, 1000-5000 mAh)** – Great for portable devices, but needs careful handling.

Best Choice for Low Power & Long Battery Life:

- If your system is **ultra-low power** (e.g., ESP32 in deep sleep most of the time), **AA lithium batteries** or **Li-ion (18650)** would be ideal.
- If weight and size are a concern, consider **AAA lithium** or **coin cells** for very low power use.
- For a **rechargeable solution**, **Li-ion or LiPo** is best.

Do you have an estimate of your system's power consumption? That would help narrow down the best battery choice.

Gemini Response:

When choosing a battery for a system that demands low power consumption and long battery life, both during active use and idle states, several factors come into play. Here's a breakdown of considerations and common battery types:

Key Considerations:

- **Power Consumption:**
  - The system's current draw (measured in milliamps, mA) is crucial. Lower current draw translates to longer battery life.
  - Consider peak current demands versus average current draw.
- **Idle Current:**
  - Even in idle mode, some systems consume power. Minimizing this is essential for long-term battery life.
- **Battery Capacity:**
  - Measured in milliamp-hours (mAh), capacity determines how much current a battery can deliver over time. Higher mAh means longer runtime.
  -

- **Battery Chemistry:**
    - Different chemistries offer varying characteristics:
        - **Alkaline (AA, AAA, etc.):** Inexpensive and readily available, suitable for low-drain devices.
        -
        - **Lithium-ion (Li-ion):** High energy density, rechargeable, ideal for devices with higher power needs.
        -
        - **Lithium Iron Phosphate (LiFePO4):** Very long cycle life, safe, and stable, suitable for applications needing longevity.
        -
        - **NiMH (Nickel-Metal Hydride):** Rechargeable, lower self-discharge than older NiCd batteries.
        -
    -
    -
- **Self-Discharge:**
    - All batteries lose charge over time, even when not in use. Low self-discharge batteries are essential for long idle periods.
    -

Battery Types:

- **AA and AAA:**
    - Excellent for low-power devices.
    - Widely available and relatively inexpensive.
    - Rechargeable NiMH AA/AAA batteries (like Panasonic Eneloop) offer a good balance of capacity and low self-discharge.
    -
- **9V:**
    - Often used in devices with specific voltage requirements.
    - Generally lower capacity compared to AA or AAA batteries.
    - May not be the most efficient choice for maximizing battery life in low-power applications.
- **Lithium-ion (18650, etc.):**
    - Offer high energy density, making them suitable for devices with varying power needs.
    - Rechargeable, providing long-term cost savings.
    - Require proper charging circuitry and safety precautions.
- **LiFePO4:**
    - These are growing in popularity, and provide a very long life. They are a good choice when longevity is a high priority.

Recommendations:

- For very low-power applications with long idle periods, high-quality alkaline or low self-discharge NiMH (like Eneloop) AA or AAA batteries are often the best choice.
-
- If the device requires a higher amount of power, or is intended to be used very frequently, then a lithium ion battery or a LiFePO4 battery would be a better choice.
- Always consider the device's voltage requirements and current draw when selecting a battery.

To make the best choice, it is very important to know the power consumption requirements of your system.

Copilot Response:

For a low-power system that aims for long battery life, you might want to consider using AA or AAA batteries. Both are commonly used, easily available, and can provide adequate power for many low-power devices. Here's a quick comparison to help you decide:

AA Batteries:

- **Capacity:** Generally higher (around 2000-3000 mAh for standard alkaline)
- **Size:** Larger than AAA
- **Lifespan:** Longer due to higher capacity
- **Use Cases:** Ideal for devices with slightly higher power consumption (e.g., remote controls, portable lights)

AAA Batteries:

- **Capacity:** Lower (around 1000-1200 mAh for standard alkaline)
- **Size:** Smaller, more compact
- **Lifespan:** Shorter compared to AA due to lower capacity
- **Use Cases:** Suitable for smaller devices with lower power requirements (e.g., small electronic gadgets, clocks)

9V Batteries:

- **Capacity:** Typically much lower in terms of milliampere-hours compared to AA or AAA (around 400-600 mAh)

- **Use Cases:** Best suited for devices specifically designed for 9V (e.g., smoke detectors, some walkie-talkies)

Ultimately, the choice depends on your specific power requirements and the physical space available in your device. If you need higher capacity and longer lifespan, AA might be the better option. If space is limited, AAA could be more appropriate. For niche applications requiring 9V, then of course, the 9V battery would be the one to use.

Let me know if there's more detail you'd like!

Case Study 3:

I'm using ESP32-CAM to do an object detection project. I need the camera to recognize objects in the way when the garage is open. What are my choices?"

For your ESP32-CAM object detection project—where the camera checks for **obstacles when the garage is open**—you have several options depending on **accuracy**, **latency**, **power usage**, and **memory constraints**. Here's a breakdown of your main choices:

🔹 1. **Use TensorFlow Lite for Microcontrollers (TFLite Micro)**
✅ Pros:
- Designed to run on resource-constrained devices like ESP32-CAM.
- Lightweight models like **Person Detection** can fit in memory (~250KB RAM used).

❗ Limitations:
- Only supports very **small models** (usually grayscale, 96x96 input).
- Hard to recognize multiple object types unless you customize or retrain the model.

Use When:
- You're only detecting **presence** of an object (e.g., person, blob).

Example:

- [TensorFlow Lite Person Detection demo](https://github.com/tensorflow/tflite-micro/tree/main/tensorflow/lite/micro/examples/person_detection)

🔹 2. **Use ESP-WHO (Espressif's Face/Object Detection Framework)**
✅ Pros:
- Developed by Espressif, optimized for ESP32-CAM.
- Supports **face detection, face recognition**, and **simple object detection**.
- Well-documented examples.

❗ Limitations:
- More about **face/shape recognition** than object classification (no COCO-style labels).
- Detection only, not classification.

Use When:
- You want basic obstacle detection by shape/movement without needing object types.

Example:
- `camera_web_server` or `face_detection` example in [ESP-WHO GitHub](https://github.com/espressif/esp-who)

🔹 3. **Use External Processing (ESP32 sends frames to another device)**
✅ Pros:
- Run **full YOLO, MobileNet, or SSD** on a **Raspberry Pi**, **NVIDIA Jetson**, or even a **cloud API** (like Google Vision, AWS Rekognition).
- Much higher accuracy and flexibility.

❗ Limitations:
- Requires **Wi-Fi bandwidth** and latency.
- Needs a more complex setup.

Use When:
- You want to recognize multiple object types (e.g., person, bike, box, pet).
- ESP32-CAM acts as a **streaming camera**, not the main processor.

🔹 4. **Motion Detection with Basic CV (OpenCV style)**
✅ Pros:
- Lightweight—detect moving objects by comparing frames.
- Doesn't need ML model.

❗ Limitations:
- Only detects **movement**, not object type.

Use When:
- You only care if **something is in the way**, not what it is.

So Which One Should You Use?

Goal	Best Choice
Detect presence of *any* object	ESP-WHO or TFLite Micro
Detect if a *person* is there	TFLite Micro with Person Detection
Detect *specific objects*	Offload frames to external device
Detect *motion only*	Frame diff (basic CV)

Would you like help:
- Picking a pre-trained model?
- Setting up TFLite Micro?
- Sending ESP32-CAM images to a server for external object detection?

Let me know how advanced you want this to be.

Logo image generated using Canva's AI tools.

Declaration: We hereby declare that we have not copied more than 7 pages from the Large Language Model (LLM). We have utilized LLM for drafting, outlining, researching, and proofreading purposes.