A blue and white logo

Description automatically generated

EET411

Senior Project

Instructor: Dr.Satinder Gill

**Project Title:** The Smart Bird Feeder

**Team Members:**

* Jeffery Wheeler: Project Lead
* Sean Copple: Team Member
* Joshua Tilson: Team Member
* Valentin Wolf: Team Member

***Honor Pledge:***

*I pledge to support the Honor System of ECPI. I will refrain from any form of academic dishonesty or deception, such as cheating or plagiarism. I am aware that as a member of the academic community, it is my responsibility to turn in all suspected violators of the honor code. I understand that any failure on my part to support the Honor System will be turned over to a Judicial Review Board for determination. I will report to the Judicial Review Board hearing if summoned.*

**Date:** 5/11/2025

Contents

[Abstract 3](#_Toc198993257)

[Introduction 3](#_Toc198993258)

[System Design 4](#_Toc198993259)

[Hardware Description 6](#_Toc198993260)

[Software Description 8](#_Toc198993261)

[Implementation & Testing 9](#_Toc198993262)

[Construction 10](#_Toc198993263)

[Testing Process 11](#_Toc198993264)

[Data Interpretation & Analysis 12](#_Toc198993265)

Proof Of Concept…………………………………………………………………………………………………………………………………..14

[Conclusion 21](#_Toc198993266)

[Future Work 22](#_Toc198993267)

[References 23](#_Toc198993268)

[Appendices 24](#_Toc198993269)

[Appendix A: Block Diagram 24](#_Toc198993270)

[Appendix B: Trade Study 24](#_Toc198993271)

[Appendix C: Budget 25](#_Toc198993272)

[Budget 25](#_Toc198993273)

[Parts List 25](#_Toc198993274)

[Bill of Materials (BOM) 25](#_Toc198993275)

[Appendix D: Schedule/Gantt Chart 25](#_Toc198993276)

[Appendix E: Code 28](#_Toc198993277)

[Flowchart 28](#_Toc198993278)

[Appendix F: Wiring Diagrams/Parts Design 33](#_Toc198993279)

# Abstract

This project focuses on the design and development of a smart bird feeder that improves upon existing feeders by addressing a common issue: unwanted access by animals such as raccoons. The main objective is to create a bird feeder that limits food access to birds only, using a controllable cover mechanism. The initial approach involves opening and closing the feeder based on day and night cycles. Future improvements include the integration of a load cell to detect the weight of visitors and deny access if the detected weight exceeds that of a typical bird, helping to distinguish between birds and other animals. To prevent from intruders of birds getting hurt, safety sensors are installed. The food will be supplied in small portions using a feeding wheel that is controlled by a servo motor. This mechanism ensures that the food always stays fresh.

Additional features planned include a camera for capturing images of feeding birds, a food level sensor to monitor and report when the food supply is low, and a solar panel with automatic sun tracking to power the device independently. The camera and load cell may also support bird identification and behavior analysis. The system will eventually be connected to a mobile app for live updates and alerts. The project uses a mix of sensors, automation, and app integration to enhance usability, sustainability, and data collection.

# Introduction

Bird feeding is a popular hobby that allows people to connect with nature, observe wildlife, and support local bird populations. However, many bird feeders on the market face a common problem: they are easily accessed by animals such as raccoons, squirrels, or larger pests that consume the bird food and often damage the feeder. This not only increases the cost of feeding birds but also limits the effectiveness and enjoyment of bird watching.

The main problem this project addresses is the lack of secure access control in traditional bird feeders. In many areas, especially near wooded or rural locations, raccoons and other animals frequently disrupt feeders. These intrusions reduce food availability for birds and can lead to frequent refilling and repair. While some commercial feeders offer partial solutions like baffles or weight-triggered perches, they are often not customizable, lack automation, and provide limited feedback to the user.

This project aims to design a smart bird feeder that not only prevents access by unwanted animals but also adds modern features such as automation, remote monitoring, and solar charging. By combining hardware like servo-controlled covers, load cells for weight detection, cameras, and level sensors, the feeder can intelligently respond to its environment. A mobile app will allow users to view bird photos, receive alerts about low food levels, and monitor feeder activity.

The goal of this project is to build a functional prototype of a smart bird feeder that solves the problem of animal intrusion while offering additional features to enhance the bird feeding experience. The project also explores ways to make the system energy-efficient and autonomous by incorporating solar power. Through this work, the team hopes to provide an innovative, practical solution for bird lovers and hobbyists.

# System Design

**Block Diagram Overview**

**Power Input**:

* The system is powered by a 5V 500mA solar cell.
* It collects energy from the sun to charge the batteries and run the system.

**Power Management**:

* The 5V relay protects the circuit and manages the power flow.
* The TP4056 charging module charges two 18650 lithium-ion batteries while allowing load sharing to avoid system shutdown during charging.
* The MT3608 boost converter steps up the battery voltage to a stable 5V output.
* A cooling system will be included to help manage heat from the charging process.

**System Controller**:

* A Raspberry Pi Zero is used to control the entire system.
* It processes sensor data, controls motors and LEDs, and handles any wireless communication or image capture if the camera is installed.

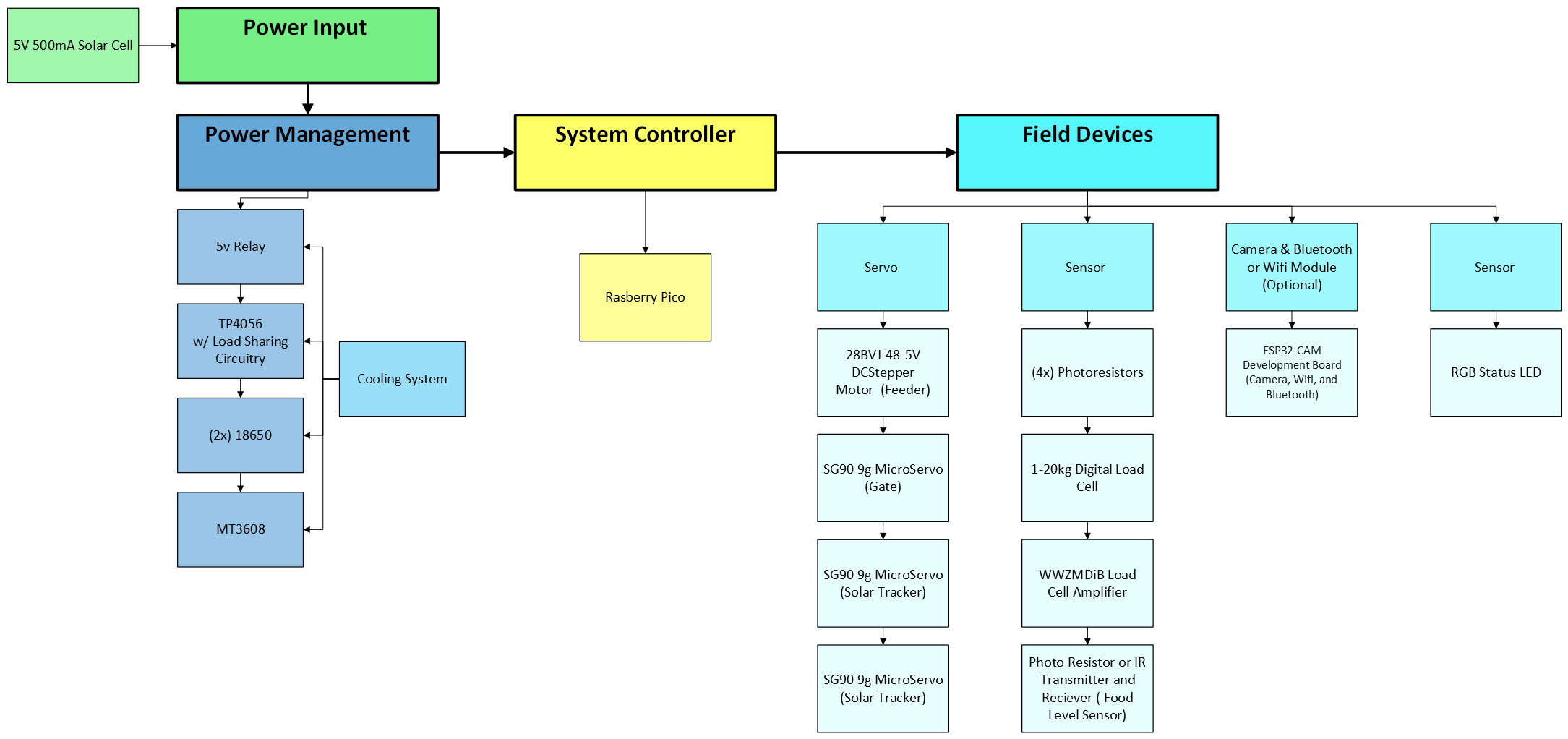
**Field Devices**:

* **Servos**:
  + 28BYJ-48 stepper motor rotates the bird feeder container.
  + SG90 micro servos are used for the food protection gate and solar tracking on two axes.
* **Sensors**:
  + Four photocell resistors detect ambient light for solar tracking and day/night detection.
  + A 1–20kg load cell with an amplifier monitors bird weight to determine if an animal is too heavy.
  + A photoresistor or IR sensor setup is used to detect when the food level is low.
  + An Ultrasonic sensor checks the food level inside the storage container
* **Optional Camera and Wireless Module**:
  + The ESP32-CAM module adds the ability to take pictures of birds and send data wirelessly.
* **RGB Status LED**:
  + Used to show system status like low battery, low food, or normal operation.

**Key Design Decisions and Considerations**

* **Solar Power**:  
  The use of a solar panel combined with load-sharing circuitry ensures that the bird feeder can operate independently of any external power sources.
* **Energy Efficiency**:  
  The system limits power use at night by turning off non-essential devices like the camera and solar trackers when sunlight is not available.
* **Animal Detection**:  
  The load cell allows the system to distinguish between birds and larger animals like raccoons, helping to protect the bird food.
* **Weather Protection**:  
  All electronics are housed in a custom-printed, weatherproof enclosure to withstand outdoor conditions.
* **Expandability**:  
  The design allows for future upgrades, such as sending bird photos through Wi-Fi or Bluetooth using the ESP32-CAM.

*Block Diagram 1*



## Hardware Description

The smart bird feeder project uses a combination of commercially available hardware and custom-designed parts to achieve its goals. Each component was selected based on performance, availability, and compatibility with the system requirements.

**Microcontrollers**

* **Raspberry Pi Zero**  
  The Raspberry Pi Zero is the main system controller. It manages data from the sensors, controls the field devices, and handles communication tasks such as Wi-Fi and Bluetooth.
  + **Specifications**: 1GHz single-core CPU, 512MB RAM, Wi-Fi, Bluetooth, mini HDMI, micro USB ports.
  + **Selection Justification**: Chosen for its compact size, low power consumption, and ability to handle more complex processing tasks like camera image handling and wireless communication. The Raspberry Pi Zero uses Python programming language which will be more suitable for more complex tasks.

**Sensors**

* **Photoresistors**   
  Four photocell resistors are used to detect ambient light. They help determine day/night conditions for opening or closing the feeder and support solar tracking.
  + **Selection Justification**: Simple, low-cost, and reliable for detecting light changes.
* **Load Cell with Amplifier (1–20kg range + HX711 amplifier)**  
  The load cell measures the weight of birds or animals at the feeder. It is sensitive enough to differentiate between birds and larger unwanted animals.
  + **Selection Justification**: Wide measurement range with good resolution. The amplifier boosts the small signal from the load cell to a readable level for the microcontrollers.
* **Infrared Transmitter and Receiver / Photocell Food Sensor**  
  This sensor setup detects when the food level drops below a certain threshold inside the feeder.
  + **Selection Justification**: Simple solution to monitor food levels without requiring complex electronics.

**Actuators**

* **SG90 9g Micro Servo Motors (4 units)**  
  Used for various mechanical movements:
  + One for rotating the food container.
  + One for opening and closing the food protection gate.
  + Two for tracking the solar panel in two axes (horizontal and vertical).
  + **Specifications**: 4.8V–6V operating voltage
  + **Selection Justification**: Lightweight, affordable, and widely supported by Arduino and Raspberry Pi libraries.
* **28BYJ-48 Stepper Motor with ULN2003 Driver**   
  Considered for smooth and accurate rotational control of the feeder.

**Other Hardware Components**

* **5V 500mA Solar Panel**  
  Provides renewable energy for battery charging, ensuring off-grid operation.
* **TP4056 Charging Module with Load Sharing Circuit**  
  Manages safe charging of the batteries while allowing the system to continue running during charge cycles.
* **18650 Lithium-Ion Batteries (2 units)**  
  Store energy collected from the solar panel. Connected in parallel for longer runtime with 5000mAh combined capacity.
* **MT3608 Boost Converter**  
  Increases the battery voltage up to 5V needed by most components.
* **RGB Status LED**  
  Used to visually indicate different system statuses, such as food low, battery low, or normal operation.
* **ESP32-CAM Module**

Adds a camera, Wi-Fi, and Bluetooth capabilities to the system.

**Custom-Designed Hardware Elements**

* **Feeder Mechanism**:  
  A rotating food container divided into four pockets, dispensing food every 90 degrees of rotation.
* **Custom Base**:  
  Designed to securely hold all circuit boards and field devices inside a weatherproof enclosure.
* **Solar Tracker Mount**:  
  A mechanical base built to allow the solar panel to rotate and track the sun for better charging efficiency.
* **Load Cell Mounting Structure**:  
  Custom mounting for the load cell to ensure accurate and protected weight measurements.
* **Bird Perch and Food Access Door**:  
  A landing perch for birds and a servo-controlled door that opens or closes to allow access to food based on sensor inputs.
* **Acrylic Food Container**:  
  A tube that stores the bird food and makes it easy to check food levels visually.

## Software Description

The software for the smart bird feeder is still being developed. Right now, the goal is to create a simple program that controls the motors, sensors, and lights based on sensor readings.

The software will have three basic parts:

* **Setup**: Turn on and prepare all the parts like sensors, servos, and lights.
* **Main Loop**: Keep checking the sensors and make the feeder open or close based on the readings.
* **Power Saving**: Turn off parts of the system at night to save battery power.

**Functions the Software Will Perform**

* Open the food cover when it is daytime.
* Close the food cover when it is dark or when a heavy animal is detected.
* Turn on a light if food is running low.
* Move the solar panel to follow the sun (solar tracker).
* (Later) Take pictures of birds and send them over Wi-Fi.

**Libraries and Tools Used**

* **PyCharm and micro python IDE** for writing and loading the programs.
* **Servo Library** for moving the servo motors.
* **ESP32-CAM Library** (later) for taking pictures and sending them.

# Implementation & Testing

The implementation process for the smart bird feeder project started with designing and preparing the mechanical and electrical systems separately. This helped make sure that each part would fit and work correctly before full assembly.

**System assembly and hardware integration**

The mechanical structure of the bird feeder is being designed to be fully 3D printed. Special attention was given to leave enough space for all electrical components, as well as any future upgrades. Once the mechanical parts are finished, they are tested without any electronics installed to make sure everything fits and moves correctly. After the mechanical testing is complete, the electrical components like the Raspberry Pi, servos, sensors, and solar panel will be installed. The system will be assembled carefully to protect the wiring and to make sure that all parts are easy to access if repairs or upgrades are needed later.

**Software development and debugging**

The software development is done using simulation tools. Because many of the components were not available in Tinkercad, Wokwi was used to model and test the system. Debugging will continue as real components are installed, with special focus on testing the servo movements, sensor readings, and communication between parts.

**Testing procedures and verification methods**

Different parts of the system will be tested individually first:

* Solar Cells: Tested outside during daylight to check charging performance.
* Photosensors (Light Sensors): Tested outside to verify they correctly detect day and night conditions.
* Load Cell: Will be tested by placing different weights to make sure the readings match expected values.
* Food Level Sensor: Will be tested by adjusting food levels and checking if the sensor triggers correctly.
* Servos: Movements will be tested manually to ensure they open and close the gate properly and rotate the solar panel.
* Food dispenser will be tested first without and then with use of the stepper motor.

Once each part is tested, the full system will be tested as a whole, simulating real use by placing weights, blocking light, and adjusting food levels.

**Challenges and implemented solutions**

One challenge was designing the mechanical structure to fit all components while still keeping the feeder small and good-looking. Careful planning of the 3D-printed parts and early mechanical testing helped solve this issue. Another challenge was the limitation of only having three analog inputs. Temperature could also become a concern. Different 3D printing materials and colors are considered because some materials may warp or weaken in outdoor heat. Furthermore, it will be important to choose a printing material that is environmentally friendly and not harmful to the birds or other animals.

## Construction

The construction process for the smart bird feeder focuses on building strong mechanical parts, preparing for electrical integration, and testing subsystems before final assembly.

**Fabrication Methods and Materials Used**

Most of the mechanical components for the feeder are 3D printed. Using 3D printing allows the team to quickly create parts, test different designs, and adjust as needed.

**Prototyping and Final Product Construction Process**

Multiple mechanical prototypes are being made to make sure all electrical components fit properly. Since the final list of electronic parts might change, the design allows extra space for possible future upgrades.

The team divided the system into smaller sections to make testing easier. Subsystems like the solar tracker, food dispensing gate, and load cell assembly are built and tested separately. Once each subsystem works correctly, they will all be combined into one complete system.

**Calibration and Fine-Tuning Steps**

After assembling the full system, calibration and fine-tuning will be done. The servos will be adjusted to make sure they open and close the gates correctly. The load cell will be calibrated to measure bird weights accurately. Sensor thresholds, such as light levels for opening the gate, will be adjusted based on real outdoor testing. Software fine-tuning will also happen after full system assembly to make sure that all parts work together smoothly.

## Testing Process

The testing process for the smart bird feeder project includes a variety of tests to make sure that all functions work correctly, and that the system can survive outdoor conditions. Testing is being done step-by-step as the project is developed and assembled.

**Comprehensive Testing Process**

Testing is being performed on both mechanical and electrical systems. The motors, sensors, and software are tested to make sure they respond properly to different conditions. 3D printed parts are tested for correct fit, durability, and weather resistance. Electrical testing includes checking that the power system supplies enough energy and that sensor data is accurate.

Mechanical movements, waterproofing, and cooling of the electronics are also part of the testing process. The system is tested first in small parts and then as a complete unit once everything is installed.

**Test Cases**

**Test Case 1: CAD Design and 3D Printing Test**

Most of the parts have been redesigned to increase sturdiness, correct the sizing to connect pieces together, and added tubing along with other routing for wires to connect the top components to the bottom of the base where the Raspberry Pico and other circuit boards are placed. A latching mechanism was also implemented to hold the LDR sensors in place firmly.

**Test Case 2: Virtual Testing with WOKWI**

Component connections and code are tested virtually using WOKWI to find problems early before installing real hardware.

**Test Case 3: Hands-on Component Testing**

Real components are wired together and tested using a multimeter, checking sensor readings and making sure motors move correctly.

An Ultrasonic sensor was tested similarly to the IR sensor by adjusting the distance after wiring to ensure proper operation as well as setting a distance limit to determine storage level, separate from food bowl.

**Test Case 4: Overtemperature Prevention**

The system is tested by varying the temperature around it to check if the cooling system, including fans and thermistors, keeps the electronics from overheating.

**Test Case 5: Water Resistance Test**

The case is sprayed with water while using water test strips inside to check for any leaks.

**Test Case 6: Material Sunlight Resistance**

Different 3D printed materials are exposed to sunlight to see how well they resist warping or becoming brittle.

**Test Case 7: Weather Condition Testing**

The feeder is tested under simulated heavy rain and wind conditions to check stability and water resistance.

**Test Case 8: Environmental Material Testing**

Different materials are tested to find out which ones last the longest in outdoor conditions while being environmentally friendly.

# Data Interpretation & Analysis

* Present and analyze the data collected during testing and project operation.
  + The data collected during the testing phases included values such as the overall mechanical measurements and how things would fit together after 3D printing, as well as circuit component compatibility and their operations defined by the code in microPython. Most components were originally based on one thing like the size of the pico and how many ports it had along with the power requirements. This lead to what kind of motors to use, the batteries, solar cell size and the relay for it, how much light triggers detection, and what weight to sense for a certain bird.
  + Raw Data:
    - Project Size: 4 inches Wide x 2 feet Tall
    - Bird Weight: 600 grams
    - Light Detection from ADC: 800
    - System Power requirements: 5V, 2 Amps
    - Charging requirements: 5V, 600 milliamps
    - Motor uses: Uln2003 driver, stepper, and 3x, 9-gram servos
    - IR Sensor Distance: 0.5 inches
    - Ultrasonic Sensor trigger distance: 60cm
* Discuss the meaning and implications of the obtained results.
  + Due to some of the results in the initial design, parts had to be redesigned in CAD to make sure they fit correctly and fixed some of the issues that occurred like the feeder bowl shape, component placement, and mechanical joints for both the lid and solar tracker. In addition, the Raspberry Pico only contained 3 analog to digital converters which forced us to use a digital light detecting resistor rather than simply using a photoresistor. This occurred again when trying to use the servo to rotate 360 degrees, because it can only move 180 degrees, meaning the uln2003 had to be implemented to work with the Raspberry Pico, driver a stepper motor. Lastly, the TP4056 charger would trigger overcurrent protection through its outputs, requiring the MT3608 to be connected directly to the lithium-ion batteries instead of through the charger’s output.
* Compare your results with theoretical expectations and existing research.
  + We expected not to get the results we were looking for in the first trials because it is difficult to do so on the first try. So constant adaptations of the project slowly developed though the month, resulting in major changes while maintaining the original idea. Online there are research documents on building similar products, however they do not fully imitate the work we wanted to do. Other feeders dispense food or have a camera, but do not offer solar rechargeability, food protection, or AI imaging assistance. Fewer others even used a Raspberry Pico on such a small scale so much of the final collected data like the physical measurements, working circuit, and code was done overtime as we figured out what worked and what didn’t.

# Proof Of Concept:

# A hand holding a metal object AI-generated content may be incorrect.

# A close up of a device AI-generated content may be incorrect.A group of electronic devices AI-generated content may be incorrect.A black pipe with a blue circuit board AI-generated content may be incorrect.A black and silver object on a concrete surface AI-generated content may be incorrect.

A computer desk with wires and wires

AI-generated content may be incorrect.A hand holding a solar panel

AI-generated content may be incorrect.

# Conclusion

* Summarize the key findings and achievements of your project.
  + Few parts needed to be modified to work well with each other such as the physical dimensions, circuit connections, and code requirements.
  + Product size is needed to withstand the bird, other animals, contain components, and actuate without resistance.
  + Sensors must be within researched values like bird weight, light intensity, detection distances, and power requirements.
  + Power system needed to charge and discharge at proper voltages and current while also having a long working life.
* Restate the project objectives and evaluate their fulfillment.
  + The project objectives are to have a bird feeder that can dispense food to the correct bird based on weight, during the day only. It will be able to connect to Wi-Fi, track the sun, detect bird species, and sense when the bowl or the storage container is empty.
* Discuss the overall success of your project and its limitations.
  + Currently we have been very successful in creating a working product, however it is not assembled yet and still requires three more components to be implemented before its completion. The limitation for this system is feeding crows and keeping out other scavengers. With small changes in the design though, it can be used to feed other animals, charge a larger battery with MPPT, or be utilized in a factory to dispense other materials while keeping track of specific variables.

# Future Work

At this stage, the overall design remains consistent with our original plan, and that’s intentional. We’re now fully engaged in the build phase, installing components, testing circuits, and verifying mechanical movements. Tasks include wiring the load cell and stepper motor, calibrating sensor inputs, and confirming servo behavior in real-world conditions. We’ve also started 3D printing parts for the enclosure and feeder mechanism, refining each iteration for better fit and function.

On the software side, we’re moving from simulations to hardware testing. The control logic is being deployed to the Raspberry Pi Pico W, and we’re actively debugging issues with servo timing, sensor thresholds, and program flow. We’ve also implemented a private GitHub repository to streamline version control and collaboration. This gives the team a centralized place to share STL files, code, and documentation without losing progress or duplicating work. As the system comes together, we’re logging pass/fail results, updating code as needed, and preparing for full system integration and testing.

Looking ahead, there are still several features we plan to implement as time allows. These include AI assistance for species identification, a camera module, a more efficient cooling system, and an automated food level sensor. Future research will focus on expanding wireless functionality using the ESP32 platform to stream video and potentially incorporate AI analytics alongside it.

Beyond bird feeding, this project has broader applications. Every subsystem we’ve developed, from motion control to environmental sensing, could be adapted to other fields. The design could be scaled for use in industrial automation, smart agriculture, or even low-power off-grid systems. Every part of this system was designed in CAD, wired and coded by hand, and serves as proof of our ability to integrate mechanical, electrical, and software systems into a functional, responsive device.

# References

*Raspberry Pi - 28BYJ-48 Stepper Motor ULN2003 Driver | Tutorials for Newbies*. (n.d.). Tutorials for Newbies.

<https://newbiely.com/tutorials/raspberry-pi/raspberry-pi-28byj-48-stepper-motor-uln2003-driver>

*Feeder.ino copy - Wokwi ESP32, STM32, Arduino Simulator*. (n.d.).

<https://wokwi.com/projects/429368894967044097>

[https://docs.arduino.cc/learn/electronics/servo-motorsLinks to an external site.](https://docs.arduino.cc/learn/electronics/servo-motors)

[https://forum.allaboutcircuits.com/threads/charge-two-18650-using-single-tp4056.183851Links to an external site.](https://forum.allaboutcircuits.com/threads/charge-two-18650-using-single-tp4056.183851)

[https://www.best-microcontroller-projects.com/tp4056.htmlLinks to an external site.](https://www.best-microcontroller-projects.com/tp4056.html)

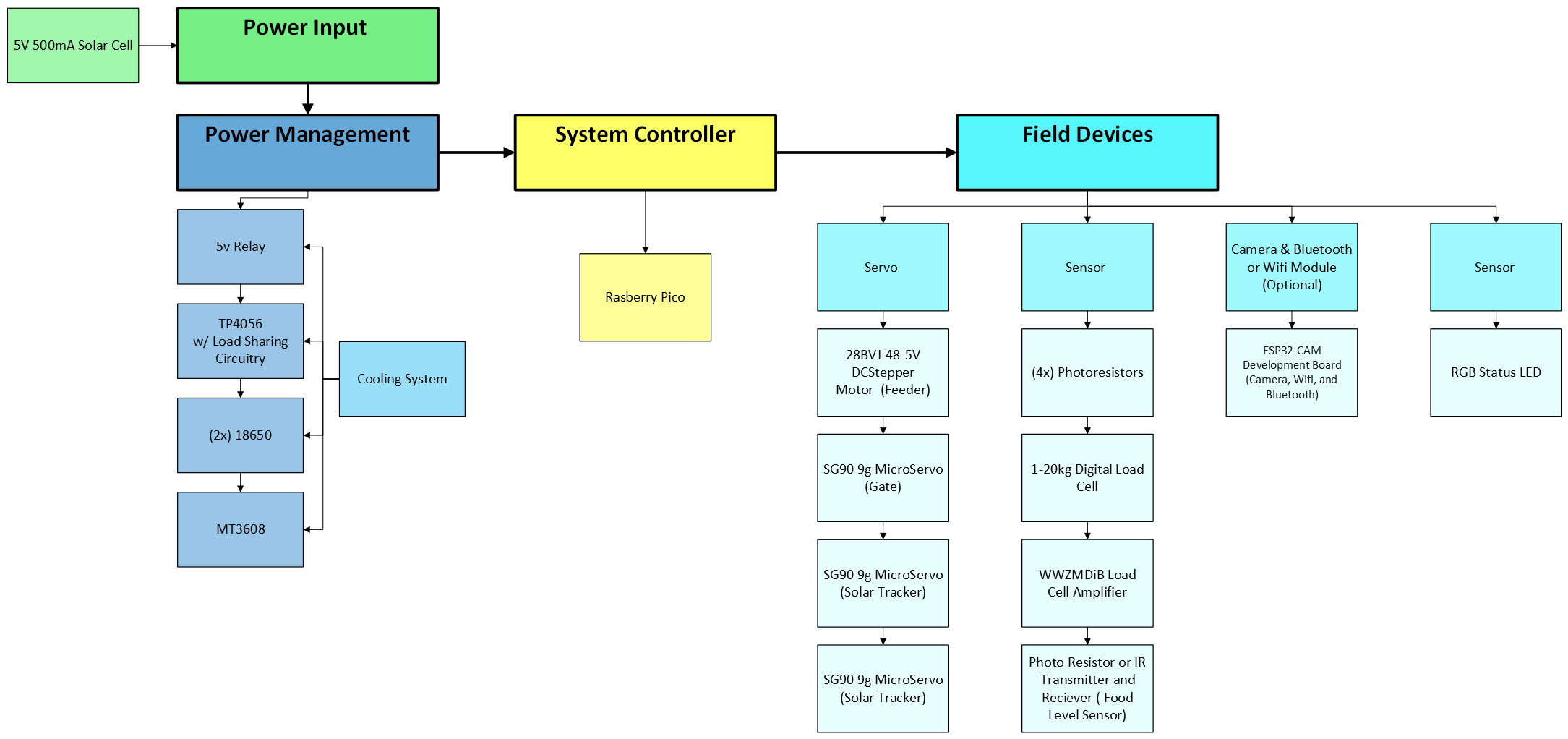
[https://forum.arduino.cc/t/18650-battery-shield-operation-and-modification/688048/6Links to an external site.](https://forum.arduino.cc/t/18650-battery-shield-operation-and-modification/688048/6)

<https://m.youtube.com/watch?v=wMtaWqGTlP4>

# Appendices

## Appendix A: Block Diagram

*Block Diagram 1*



## Appendix B: Trade Study

**Trade Study 1: Animal Identification Method (Jeffery Wheeler)**

Jeffery compared three options to distinguish birds from other animals: a contact sensor, an AI model with a camera, and a load cell. After reviewing accuracy, compatibility, cost, and power needs, the load cell was selected. It offers a good balance of accuracy and low power use, fits well with the system, and can be easily integrated into the perch design.

**Trade Study 2: Microcontroller Selection (Sean Copple)**

Sean evaluated the Arduino Uno, Raspberry Pi Zero 2 W, and Raspberry Pi Pico W for controlling the smart bird feeder. The Raspberry Pi Pico W was selected because it has strong wireless capability, good GPIO support for sensors and motors, a small size for the case, and better power efficiency. It also allows faster software development.

**Trade Study 3: Servo Motor Selection for Food Gate (Valentin Wolf)**

Valentin compared three servo motors: the SG90, MG90S, and DS3218. Although the MG90S is stronger and better for outdoor use, the team already owns SG90 servos. The plan is to first test the SG90. If it does not provide enough torque, the MG90S will be used as a backup.

**Trade Study 4: Boost converter selection (Josh Tilson)**

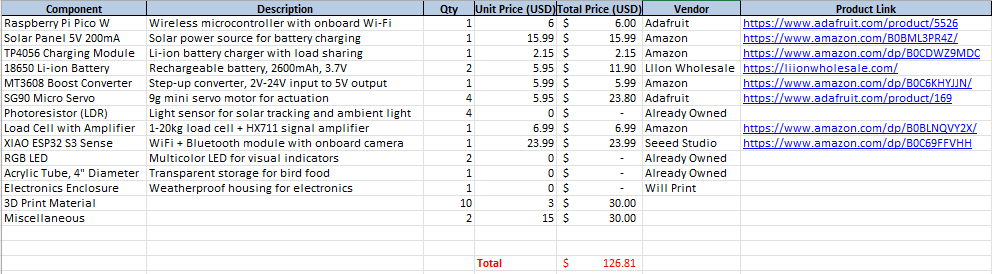
Josh evaluated the boost converter to output more current for the system. His three choices were between MT3608, XL6009, and XL6019. At first the MT3608 was chosen because it was a cheaper well-known option which still works in the circuit now. However, it has a limitation of 2 amps which could be overdrawn if all components are working at the same time. It is unlikely that it will take more than 2 amps during its operation, but if more current is required the XL6019 is a good replacement that is slightly more expensive and is capable of 5 amps instead.

## Appendix C: Budget

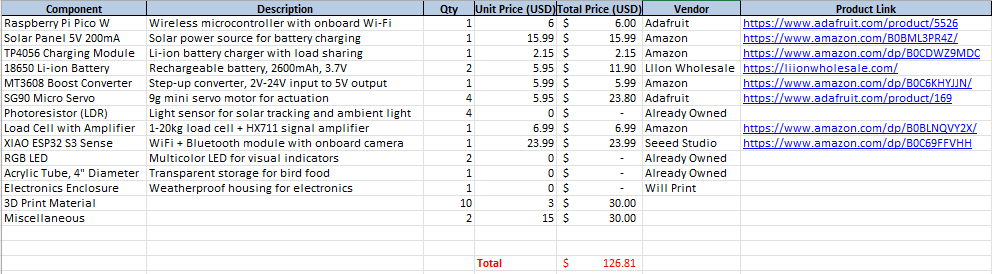
### Budget

The Budget for this project is about 150 Dollars for the major components of the Bird feeder.

### Parts List



### Bill of Materials (BOM)



## Appendix D: Schedule/Gantt Chart

#### Team Structure:

The project team is composed of five members: Jeffery Wheeler, Sean Copple, Joshua Tilson, and Valentin Wolf. Each member brings a unique skill set to the group, allowing for a well-rounded and collaborative approach to the smart bird feeder project. Team roles were assigned based on individual experience and personal strengths.

Jeffery Wheeler is the project lead. He initially proposed the bird feeder idea and has extensive experience in mechanical systems and control design. He also owns a 3D printer and has a clear vision for the final product’s design and functionality. As the team lead, he will guide the project’s direction, coordinate tasks, and make final decisions when needed.

Sean Copple is highly skilled in both electrical and mechanical tasks and also owns a 3D printer. Sean is responsible for identifying and sourcing critical hardware components and assisting with physical assembly and wiring.

Joshua Tilson is part of the Mechatronics program and brings strong experience in building control systems. Like the rest of the group, he has access to a 3D printer and is involved in selecting components for the project. Joshua will also support hardware integration and testing.

Valentin is responsible for the documentation and communication. He is responsible for writing and organizing project materials, scheduling meetings, and coordinating information flow between team members. He also supports all other areas of the project as needed.

Although specific roles have been assigned, all team members contribute across disciplines, including software, hardware, and design. Tasks are divided to maximize efficiency and are reassigned as needed based on availability and progress. All major decisions are made collectively. If any disagreement arises, the project lead will make the final decision to maintain momentum and ensure consistency.

#### Communication Plan:

Effective communication is essential for this project, especially since the team members are located in different time zones. To stay connected and on track, the team has implemented a structured communication plan with multiple tools and channels.

A dedicated group chat was created for quick updates, sharing images and videos, and staying in regular contact. For more formal discussions, the team uses the ECPI Canvas group page, which includes discussion boards for each project topic. This platform is also used for collaboration on weekly assignments and lab reports.

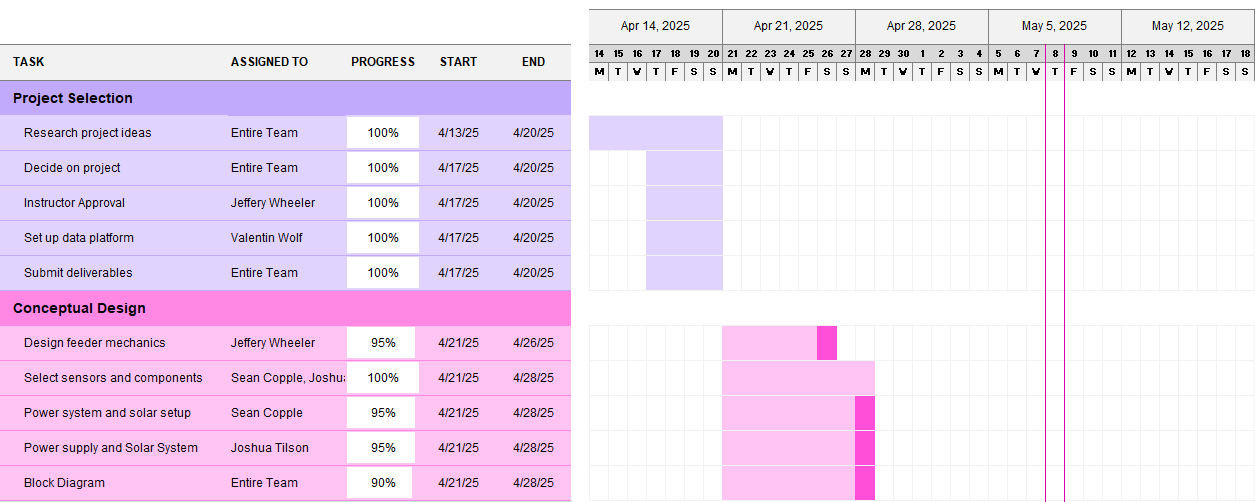
In addition, the team shares and updates project files through a shared folder on the ECPI group page. All files are organized by week and labeled with clear names and timestamps to track progress and maintain version control. To improve collaboration and file management, the team also implemented a GitHub repository. This has become the main hub for storing code, documentation, STL files, and design updates. GitHub allows for version control, easier syncing between members, and reduces the risk of losing progress due to file overwrites or miscommunication.

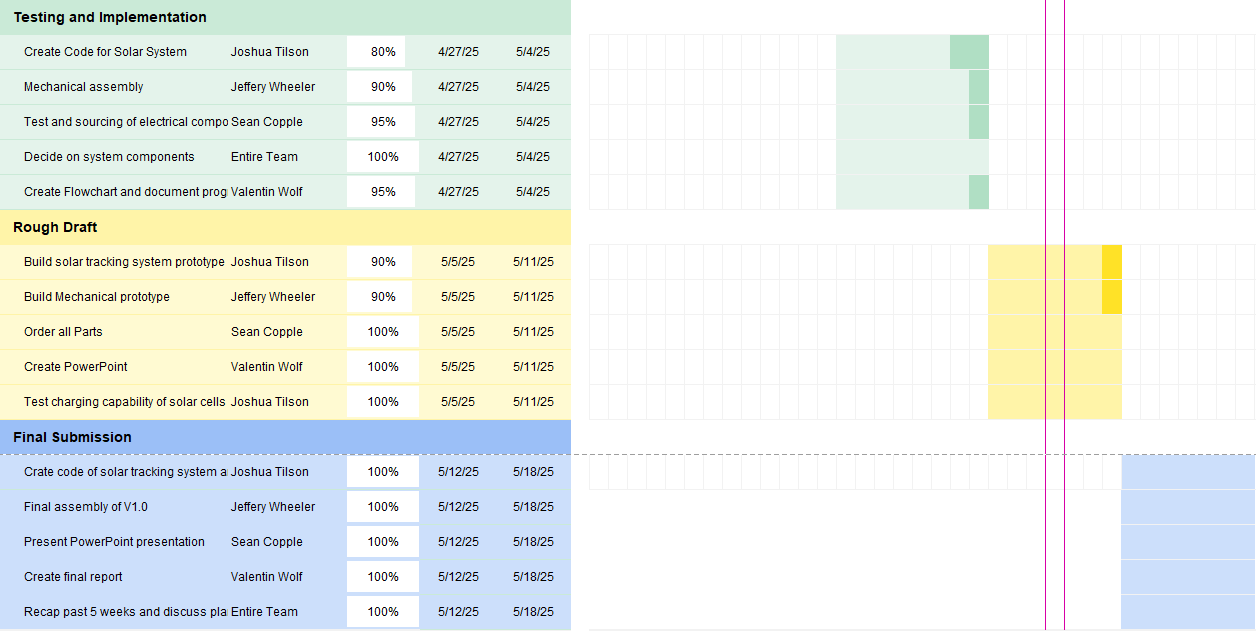
The team meets twice a week. The first meeting takes place at the beginning of the week to plan tasks and delegate responsibilities. A second meeting is held every Sunday to review the week’s progress, discuss any issues, and prepare for the next steps. These meetings are held virtually to accommodate different time zones.

This structured plan ensures that all team members are informed, involved, and able to contribute effectively throughout the five-week project.

#### Gantt Chart:

*Gantt Chart*





## Appendix E: Code

### Flowchart

*Flowchart System Overview*



Code:

**Feeder Section -**

#from hx711\_gpio1 import HX711  
import time  
from machine import Pin,PWM  
  
# Constants for the stepper motor pins  
IN1 = Pin(18, Pin.OUT)  
IN2 = Pin(19, Pin.OUT)  
IN3 = Pin(20, Pin.OUT)  
IN4 = Pin(21, Pin.OUT)  
IRSensor = Pin(16, Pin.IN, Pin.PULL\_UP)  
Photosensor = machine.ADC(28)  
#data\_pin = Pin(14, Pin.IN, Pin.PULL\_DOWN)  
#clock\_pin = Pin(15, Pin.OUT)  
#hx711 = HX711(clock\_pin, data\_pin)  
#hx711.tare()  
lid = PWM(Pin(2))  
lid.freq(50)  
  
# Sequence for moving the stepper motor  
SEQUENCE = [[1,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,0,1]]  
  
# Function to move the stepper motor  
def move\_stepper(direction, steps):  
 # Set the input pins  
 pins = [IN1, IN2, IN3, IN4]  
  
 # Set the direction of the sequence  
 if direction == 'forward':  
 sequence = SEQUENCE  
 elif direction == 'backward':  
 sequence = list(reversed(SEQUENCE))  
  
 # Loop through the specified number of steps  
 for i in range(steps):  
  
 # Set the input pins based on the current step  
 for j in range(len(pins)):  
 pins[j].value(sequence[i%4][j])  
  
 # Delay between steps  
 time.sleep(0.002)  
  
# Main loop  
while True:  
 print(IRSensor.value())  
 print("ADC: ", Photosensor.read\_u16())  
 #raw\_wt = hx711.read()  
 #sf = 340/350000  
 #weight = raw\_wt\*sf  
  
 if (IRSensor.value() == 0 and Photosensor.read\_u16() <= 800): # if bowl is empty (IR) and is daytime(photosensor)  
 move\_stepper('forward', 512)  
 time.sleep(2)  
  
 if (weight.value() <= 600 and Photosensor.read\_u16() <= 800): # if weight is in range and is daytime  
 for position in range(500000, 2500000, 5000): # open lid  
 lid.duty\_ns(position)  
 time.sleep(0.01)  
  
 else:  
 for position in range(2500000, 500000, -5000): # close lid  
 lid.duty\_ns(position)  
 time.sleep(0.01)

**Solar Tracker Section -**

import time  
from machine import Pin,PWM  
 #Looking at the panel with sensors facing you  
TL = Pin(3, Pin.IN) #LDR top left  
TR = Pin(4, Pin.IN) #LDR top right  
BL = Pin(5, Pin.IN) #LDR Bottom left  
BR = Pin(6, Pin.IN) #ldr Bottom right  
  
Xaxis = PWM(Pin(2)) # sets x and y axis servo pins as pwm signal  
Yaxis = PWM(Pin(17))  
  
Xaxis.freq(50) # sets x and y axis pwm output frequency  
Yaxis.freq(50)  
  
positionx = 1500000 # start each servo to their midpoint position in nanseconds as duty cycle  
positiony = 1500000  
  
while True: # loop section  
  
 if (TL.value() == 1 and TR.value() == 1): #if top left and top right LDR sensors reach their preset values  
 Yaxis.duty\_ns(positiony) # move y axis servo to saved variable  
 positiony = positiony + 5000 # add 5000 nanoseconds to duty cycle and change the variable value  
 time.sleep(0.01) # wait 10 milliseconds between each step  
  
 if (TL.value() == 1 and BL.value() == 1): # same sequences as before with different sensors sets  
 Xaxis.duty\_ns(positionx)  
 positionx = positionx - 5000 # negative indicates reverse direction  
 time.sleep(0.01)  
  
 if (BL.value() == 1 and BR.value() == 1):  
 Yaxis.duty\_ns(positiony)  
 positiony = positiony - 5000  
 time.sleep(0.01)  
  
 if (TR.value() == 1 and BR.value() == 1):  
 Xaxis.duty\_ns(positionx)  
 positionx = positionx + 5000  
 time.sleep(0.01)

**Ultrasonic Storage Level-**

from machine import Pin

import time

trigger = Pin(2, Pin.OUT)

echo = Pin(3, Pin.IN)

def ultra():

trigger.low()

time.sleep\_us(2)

trigger.high()

time.sleep\_us(5)

trigger.low()

while echo.value() == 0:

signaloff = time.ticks\_us()

while echo.value() == 1:

signalon = time.ticks\_us()

timepassed = signalon - signaloff

distance = (timepassed \* 0.0343) / 2

print("Distance:", distance, "cm")

if distance > 60:

print("Storage Level Low")

while True:

ultra()

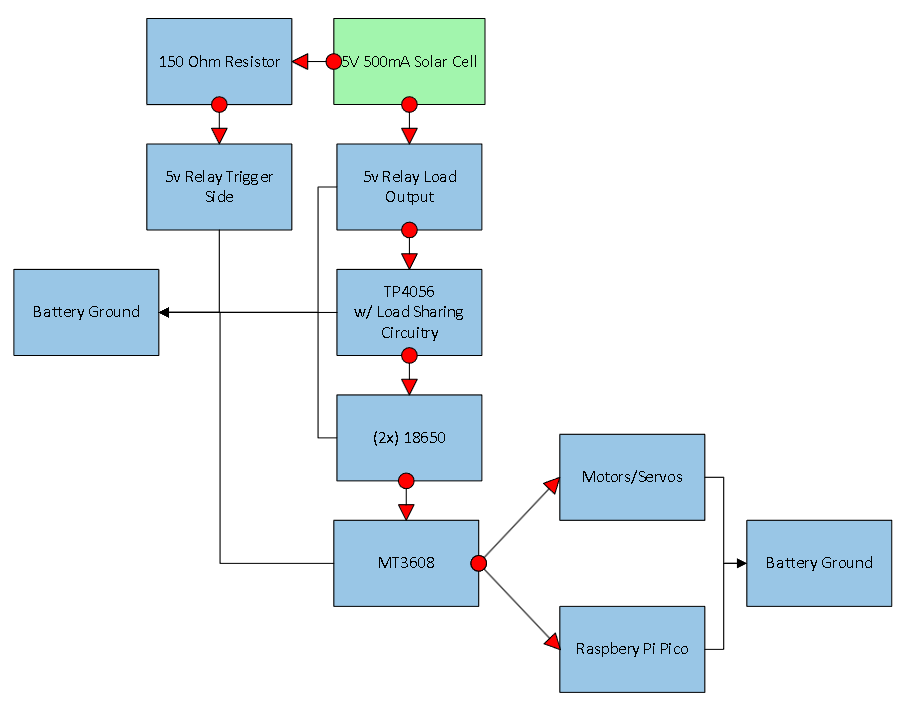
time.sleep(0.1)

## Appendix F: Wiring Diagrams/Parts Design

*Wiring and Parts for System*

A diagram of a machine

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a blue object

AI-generated content may be incorrect.

A computer screen shot of a blue object

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a blue object

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.