Squirrely Bird Feeder

Design Document

Team Number sddec26-03

Faculty Client:

Randall Geiger

Team Members/Roles:

Miles Nichols

Wyatt Sinclair

Kenny Tran

Ben Bartels

Jack Morrison

Nolan Honert

Revised: 5/3/2026

Executive Summary

The Squirrely Bird Feeder project is designed to solve a common problem for backyard birdwatchers: squirrels stealing birdseed, damaging feeders, and disrupting the birdwatching experience. Traditional squirrel-proof feeders often rely on passive mechanical barriers such as domes, slippery poles, or weight-based mechanisms. These solutions can be inconsistent and do not provide any smart monitoring or digital interaction. Our project addresses this issue by combining artificial intelligence, motion detection, mechanical control, and cloud-based image storage into one smart bird feeder system.

The main goal of the project is to create a humane, automated feeder that can distinguish between birds and squirrels and respond accordingly. Key design requirements include detecting motion within a reasonable radius of the feeder, capturing images when motion is detected, classifying the animal as a bird or squirrel with at least 70% accuracy, allowing birds to access the seed, and closing a mechanical gate when a squirrel or unwanted animal is detected. The system must also be low-maintenance, safe for wildlife, weather-resistant in outdoor conditions, and stay within a prototype budget of $500.

Our proposed design uses a Raspberry Pi as the main controller for the feeder. A PIR motion sensor detects nearby movement and triggers a camera to capture images. These images are then processed using a machine learning model trained to identify birds, squirrels, and empty feeder states. If the system detects a squirrel, the Raspberry Pi sends a signal to a servo-controlled mechanical gate that closes access to the seed. If a bird is detected, the feeder remains open. The design also includes cloud storage and a user interface component, allowing the user to save and view bird images later through a simple dashboard or app.

So far, the team has made progress on the early prototype and data collection stage. A motion-activated camera case has been designed and 3D-printed to serve as a trail camera for capturing images of squirrels. The motion-triggered camera system is working as intended, but outdoor image collection and full feeder integration still need to be completed. The next major steps are to gather real feeder images, train and improve the classification model, build the full mechanical feeder gate, integrate all hardware and software components, and begin system-level testing.

Overall, the design meets the main user needs by providing a humane, automated, and more intelligent way to protect birdseed while also giving users a digital birdwatching experience. As testing continues, success will be measured through classification accuracy, gate response time, motion detection reliability, cost control, and user feedback.

Learning Summary

Development Standards & Practices Used

- Hardware & Circuit Practices: GPIO pin interfacing for sensors and actuators, pulse-width modulation (PWM) for servo motor control, rapid prototyping using 3D printing for the camera enclosure, and weatherproof housing design for outdoor electronics.
- Software Practices: Edge computing for local machine learning inference, cloud-based data routing (MQTT/REST APIs), version control using Git/GitHub, and Agile/Kanban methodology for continuous integration and task management.
- Engineering Standards Considered: * IEEE/ISO/IEC 29148-2018 (Requirements Engineering) for strict, testable system requirements.
- IEEE 802.11-2020 (Wi-Fi standard) for reliable edge-to-cloud network communication.
- IEEE 2413 (IoT Architecture) for standardizing the physical, network, and application layers of our smart feeder system.

Summary of Requirements

- Functional: Detect motion within a 1-2 foot radius, capture images, classify the animal (bird vs. squirrel) with at least 70% accuracy, and close a mechanical gate if a squirrel or unwanted pest is detected.
- Performance & Environmental: Operate seamlessly in a continuous loop without user intervention, function outdoors in mild weather (32°F to 90°F), and deter pests humanely without trapping or causing physical harm.
- User & Cloud: Send and store captured bird images to the cloud, display images on a simple user dashboard/app, and remain low-maintenance for the casual birdwatcher.
- Economic: Complete the prototype within a $500 budget constraint.

Applicable Courses from Iowa State University Curriculum

- CprE 288 (Embedded Systems): Provided the foundational knowledge for programming the Raspberry Pi, reading PIR sensor data, and driving the mechanical servo gate.
- ComS 309 (Software Development Practices): Taught the team how to utilize Git, manage a frontend/backend software architecture, and work effectively as a team using Agile principles.

New Skills/Knowledge acquired that were not taught in courses

- Edge AI Deployment: While ISU courses cover machine learning, we had to learn how to optimize and deploy a lightweight, highly efficient machine learning model specifically for an edge device (Raspberry Pi) with limited compute power.

- Cloud IoT Architecture: We acquired new skills in setting up cloud services specifically for IoT devices, including handling secure image transmission, MQTT message routing, and cloud database storage.
- Weatherproof Mechanical Design: We had to independently research and learn how to design, 3D print, and assemble physical enclosures that can protect sensitive electronics and camera lenses from outdoor elements and moisture.
- Custom Dataset Curation: We learned the practical challenges of capturing, cleaning, and labeling a custom dataset of local backyard wildlife from scratch, rather than relying on pre-packaged academic datasets.

Table of Contents

1. Introduction

1.1. Problem Statement

Birdfeeding is a common form of entertainment for many people, allowing them to connect with local ecosystems. Without this interaction, many would lose a vital source of peace and environmental engagement. The question is: How do we maintain the integrity of these feeding environments when our resources are being depleted by invasive and relentless squirrel populations? There are many proposed solutions on the market, but most are just plain ineffective. Relying on ineffective plastic domes and slippery poles pushes homeowners into an endless, expensive, and futile cycle against highly intelligent pests. This current approach is a disaster for bird enthusiasts who invest significant time and funds into their hobby.

Firstly, let's identify the biggest obstacle in this domain: the agility of the determined squirrel. Traditional mechanical defenses simply cannot keep up with these acrobatic pests, resulting in wasted premium birdseed and damaged property. Our project, the Squirrely Bird Feeder, seeks to engineer a definitive solution to this problem. We are designing a smart camera system that uses artificial intelligence to distinguish between a bird and a squirrel.

By accurately identifying the target, the system will trigger a safe, immediate closing mechanical gate that prohibits all access to the bird seed. Ultimately, this approach will protect the user's financial investment, prevent resource depletion, and transform a simple bird feeder into an interactive, digital wildlife experience.

1.2. Intended Users

Furthermore, it is vital to acknowledge the primary demographic affected by this issue: the casual backyard birdwatcher. These individuals rely on leisurely birdwatching. It is obvious that as the cost of premium seed rises, the financial burden of feeding neighborhood squirrels becomes an unsustainable strain for the average homeowner. They require a reliable, automated solution. By integrating an active deterrent alongside a "digital doorbell" that captures photos of visiting birds, our project offers these users a delightful, effortless way to engage with nature while securing their peace of mind and financial investment.

We must also expect to see significant utility for the serious avian enthusiast. It is a much larger obstacle for enthusiasts to attract specific, sometimes rare, birds when aggressive mammals dominate the feeder. This demographic necessitates a highly selective feeding environment. It's implausible to expect serious birders to achieve their observation and documentation goals if their feeders are constantly overrun by pests.

Lastly, a significant requirement for our intelligent design is humane wildlife management, which directly appeals to the eco-conscious homeowner. Due to increased environmental awareness, people want to support local biodiversity without resorting to lethal or toxic pest control methods. Generally, the persistent presence of nuisance wildlife increases the risk that homeowners will resort to dangerous chemicals or traps out of sheer frustration. However, our system trains the squirrels to stay away without causing lasting harm. This ensures that the local ecosystem remains balanced and safe.

Ultimately, it is unavoidable to acknowledge the state of current bird-feeding technology. Between the financial drain, the lack of feeder exclusivity, and the risk of inhumane retaliation, sticking to traditional, passive bird feeders is irresponsible and unrealistic. It's time to implement modern, intelligent engineering solutions to outsmart the squirrels effectively and sustainably.

User Group 1: Casual Yard Birdwatcher

Persona: A homeowner, renter, family, or person who enjoys watching birds as a hobby. Likely to have a feeder placed in their yard, deck, or out a window that allows them to see the bird feeder.

Needs:

- Prevent squirrels from stealing bird feed
- Low-maintenance solution
- Still be low effort to replace feed and keep squirrels away

Value: Our bird feeder protects it from unwanted animals while still allowing the user to enjoy their hobby. Preventing these animals saves the user money and preserves their current bird population.

Justification: This directly addresses the issue of birdfeed going away too fast and the frustration that comes with it. It helps make the birdwatching experience peaceful and as intended.

User Group 2: Serious Bird Watchers

Persona: An experienced bird watcher who is trying to attract certain species of birds. This user may track when a bird visits and keep records of what birds have been seen.

Needs:

- Selective feeding environment for specific bird types
- Ability to record and see all bird activity
- High level of control over what can get into the feeder.

Value: The system enables targeted feeding and only allows specific species to eat from the feeder. The built-in camera system provides data collection and an alternative tool to see birds when the user is not home.

Justification: This aligns with the problem of unwanted animals stealing bird feed. It allows these extreme enthusiasts to track their birds at a professional level without any outside interference.

User Group 3: Eco-Conscious Homeowner (Eco Warrior)

Persona: An environmentally aware homeowner or wildlife advocate who is passionate about supporting local biodiversity. They actively want to feed native bird populations but oppose lethal, toxic, or otherwise harmful pest control methods to manage nuisance animals.

Needs:

- A 100% humane, non-lethal, and non-toxic method to deter squirrels.
- A solution that promotes natural ecosystem balance without permanently displacing or harming native mammals.
- A sustainable approach that reduces the environmental and financial waste of spilled or stolen birdseed.

Value: Our system provides a safe, behavioral-training approach to wildlife management. Using a harmless mechanical gate triggered by AI effectively trains squirrels to stay away without causing them any physical distress or injury, allowing the user to protect their local bird population ethically.

Justification: This directly addresses the ethical dilemma of backyard wildlife management. It prevents the escalation to dangerous traps or chemical deterrents out of sheer frustration, offering a modern, cruelty-free technological alternative that respects and protects the entire local ecosystem.

2. Requirements, Constraints, And Standards

Functional requirements:

1. Detects motion within a 1 to 2-foot radius of the feeder via a motion sensor. **(constraint)**
2. Capture images when the sensor detects motion.
3. Classify the animal in the image as a bird or a squirrel with at least a 70% accuracy rate **(constraint)**
4. Allow access to the feed when the detected animal is a bird.
5. Close a mechanical gate when the detected animal is unwanted (e.g., squirrel, unwanted bird/similar).
6. The device will reopen once the squirrel is no longer detected and the time threshold has passed.
7. Store capture bird images so the user can view them.
8. The user application displays recent bird images.
9. Log device and cloud errors for debugging and maintenance.

Performance Requirements:

1. Respond promptly before an unwanted animal can reach the feed.
2. Motion detection -> AI classification -> gate response -> image capture shall all work in a single continuous operation, with smooth transitions & without requiring user help.

Physical Requirements:

1. The feeder includes a Raspberry Pi.
2. The camera is positioned to capture images of the animals as they access the feed.
3. The feeder includes a motion sensor such as a PIR sensor.
4. The feeder includes a mechanical gate to block access to the feed. **(constraint)**
5. The feeder includes a servo that opens and closes the gate.
6. A feeder protects electronics from outdoor weather conditions.
7. Allows the user easy access to refill the seed without altering the camera or other devices.
8. The cost of the prototype parts and services must cost no more than 500$. **(constraint)**

Environmental Requirements:

1. Operates outdoors during mild weather conditions, specifically above freezing (32°F to 90°F), as extreme cold can drain batteries and freeze standard servos. **(constraint)**
2. Doesn't disturb other wildlife with loud noises.
3. A humane deterrence method that does not trap the unwanted animal inside or injure the animal. **(constraint)**

User Requirements:

1. Be low-maintenance for the casual user.
2. Easy to set up with minimal directions/tools, taking under an hour to assemble.
3. A simple way to look at past bird photos on a basic dashboard or app.
4. Little interaction past refilling feed and cleaning.
5. The user should be able to view images taken by the Pi Camera.

2.1. Engineering Standards

Engineering standards matter because they make systems safer, more reliable, easier to build, and more compatible. The IEEE standards emphasize that the standards help engineers work with teams, clients, customers, and technologies. They are also useful for solving environmental, economic, and technical problems. These all relate back to our project, as the feed includes hardware, software, cloud storage, and user experience, which we will need to take into account during development and after implementation.

1. **IEEE/ISO/IEC 29148-2018** [IEEE SA - IEEE/ISO/IEC 29148-2018]

   This standard dictates how you should gather, write, and manage the requirements of a new software or hardware system. This is helpful because it provides direct guidance, with strict rules for engineers, rather than leaving engineers to guess what the product should do. It outlines how to talk to stakeholders to figure out what they need, how to translate those needs into clear, testable engineering requirements (avoiding vague language), and how to document them properly so the whole team understands the goal. Its main objective is to prevent project failures that occur when the final product doesn't match what the client or user originally wanted.

2. **IEEE 802.11-2020/Cor 1-2022** [IEEE SA - IEEE 802.11-2020/Cor 1-2022]

   This is a specific technical correction (a corrigendum) to the foundational IEEE 802.11-2020 standard, which defines the medium access control (MAC) and physical layer (PHY) specifications for Wi-Fi networks. Published in 2022, this brief document aims to rectify a minor but important error in the IEEE 802.11ay amendment, which governs extremely high-throughput wireless operations in the 60 GHz band. Specifically, it corrects a mistake in how the "Protected Announce Support bit" was documented, ensuring that engineers and developers building Wi-Fi hardware and software have the exact specifications needed for secure signaling and reliable device interoperability.

3. **IEEE/ISO/IEC 29148-2018** [IEEE SA - IEEE/ISO/IEC 29148-2018]

   IEEE 2413 is the universal blueprint for building an Internet of Things (IoT) system. Because the IoT world exploded so quickly, many companies were building smart devices in completely different, incompatible ways. This standard was created to stop that fragmentation. It outlines a common, standardized framework that defines the different "layers" of an IoT system. It starts from the physical hardware at the edge, moving up through the networking/communication layer, and finally to the cloud and software applications. Its primary goal is to ensure that all IoT devices are designed with a consistent architecture so they can operate securely, reliably, and seamlessly, regardless of who makes them.

1. **IEEE/ISO/IEC 29148 (Requirements Engineering):** Yes, this is very relevant. Our project has a lot of moving parts such as the motion sensor, the AI classification, and the mechanical gate. Instead of just guessing what the feeder should do or using vague phrases

like "it works outside," this standard provides a clear guide for writing strict, testable rules. It will help me make sure that what we build actually matches what a user would want from a smart bird feeder.

2. **IEEE 802.11-2020/Cor 1-2022 (Wi-Fi Correction):** This one is only slightly relevant. The Raspberry Pi definitely needs Wi-Fi to send the bird pictures to the cloud and the user's phone, so the general Wi-Fi rules matter. However, this specific document addresses a highly technical error in super-fast 60 GHz networks. A simple bird feeder sending pictures from a backyard doesn't need that kind of crazy speed, so this specific update doesn't really impact the day-to-day project design.
3. **IEEE 2413 (IoT Architecture):** Yes, this is highly relevant. The bird feeder is basically a classic Internet of Things (IoT) device. It has physical hardware in the yard (sensors and a motor), a brain (the Raspberry Pi), and a cloud application (where users view the photos). This standard gives us a blueprint for organizing these different pieces so they can talk to each other smoothly without the whole system becoming a disorganized mess.

When discussing this with my team, they focused on standards that handled different functional parts of the feeder. For example, one teammate chose a standard related to battery safety and power management, since we have to figure out how to keep the feeder powered outside. Another teammate chose a standard focused on data privacy to make sure that the app and the cloud database storing the user's bird photos are secure and cannot be easily hacked.

To incorporate these standards, I plan to make a few specific changes to my project:

For the Requirements Standard (29148): I will rewrite my current project requirements to be much more specific. Instead of vague goals like "operates in all weather," I will change it to a measurable rule, such as specifying the exact temperature range the feeder must withstand.

For the Wi-Fi Standard (802.11): Instead of worrying about ultra-fast speeds, I will modify my design plans to specify that the Raspberry Pi should just connect to a standard 2.4 GHz home Wi-Fi network. This will give the feeder a better range through the yard and walls.

For the IoT Standard (2413): I will follow the standard procedure by formally organizing my project's architecture into three distinct layers: the physical layer (the camera, sensor, and gate), the network layer (the Wi-Fi connection), and the software layer (the cloud storage and app). Structuring my documentation and design process this way will help me keep the code and the physical build standardized and organized.

3 Project Plan

3.1 Project Management/Tracking Procedures

We will adopt a Kanban project management style. Given the distinct but highly integrated software, hardware, and machine learning components of the Squirrely Bird Feeder, Kanban provides a continuous, flexible flow of development rather than restricting us to rigid sequential

phases or time-boxed sprints. This approach allows the team to build, test, and refine features in parallel based on team capacity. For example, tasks related to engineering the physical gate and weatherproofing can flow through our pipeline while the machine learning model is continuously trained and improved, with team members pulling new tasks as they finish previous ones.

To track progress and visualize our workflow, our team will use a Kanban board (such as GitHub Projects or a similar tool) to move tasks through stages like "To Do," "In Progress," and "Done." Alongside this visual board, we will use Git for code version control, Discord for day-to-day communication and coordination, Google Drive for managing group resources and design documents, and the team website to publicly update our progress..

3.2 Task Decomposition

1. Hardware
    a. Bird feeder Structure and design
    b. Integrating the Raspberry Pi and the Passive Infrared Sensor(PIR)
    c. Servo-controlled gate implementation
2. Machine Learning
    a. Collecting our own test images for the bird and squirrel dataset with an empty feeder
    b. Train the model on the dataset
    c. Optimize and maintain the model's accuracy and reliability
3. Software & Backend
    a. Image processing with classification integration
    b. Storing images and timestamps in the cloud
    c. Safe error logging
    d. Image capture
4. User Interface
    a. Website dashboard/app on phone
    b. Display captured bird images
    c. Keep a healthy, error-free page

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

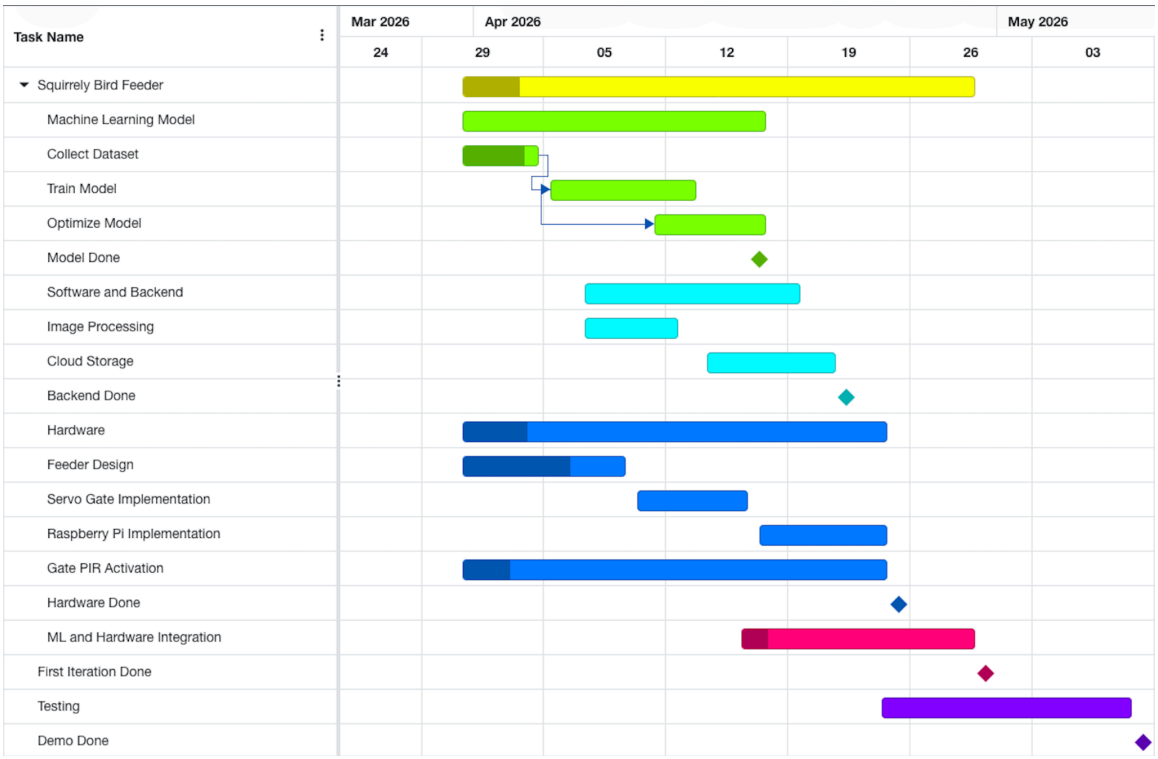
For the semester, we will focus on the prototype having the core functionalities and implementation of the entire pipeline: from identifying movement, capturing it, classifying it as a squirrel or a bird, closing the gate to deny access to the feed if it is a squirrel, and opening after detecting the absence of the squirrel. Deliverables include a finalized physical prototype (housing, Pi, sensors, and servo), a base ML model capable of distinguishing local squirrels from common birds, and local-only operation testing.

Evaluation Criteria & Metrics:

- **Motion Detection:** The PIR sensor must reliably detect motion within a 1-foot radius of the feeder.

- **Classification Accuracy:** The machine learning algorithm must classify the animal in the image as a bird or a squirrel with at least a 70% accuracy rate.
- **System Latency:** The continuous operation of motion detection, AI classification, and gate response must occur promptly enough to close the gate before the squirrel reaches the feed.
- **Budget Compliance:** The total cost of the prototype parts and services must not exceed $500.

3.4 Project Timeline/Schedule



3.5 Risks and Risk Management/Mitigation

Risk	Severity	Mitigation
Low model accuracy	High	Increase dataset volume, use pretrained models
Slow inference	High	Optimize our model/use as

		lightweight models as possible
Hardware failure	High	Perform maintenance checks to prevent problems before they occur.
Weather damage	Medium	Have our equipment waterproofed or placed inside a weatherproof birdhouse.
Integration issues	High	Make sure components work entirely before integrating. Test after integrating each component.

3.6 Personnel Effort Requirements

Task	**Estimated Hours**	**Explination**
Hardware Development	80hrs	Have to come up with a design, implement it, ensure the parts run smoothly, and then deploy to test whether they work consistently in a feeder without human interference.
ML Model Development	80hrs	Collecting and organizing a labels dataset for birds, squirrels, and empty feeder images, then train the model. This includes preprocessing images, tuning hyperparameters, retraining for improved accuracy, and optimization.
Software/Backend	60hrs	Developing a pipeline that connects the motion detection, image capture, and classification results. Having scripts to trigger the camera, send images, and store results. Handles errors and debugging.
UI Development	30hrs	Creating an app interface to see pictures of a bird from a

		camera is fairly straightforward. This means displaying images from the cloud and an easy-to-use interface.
Testing	60hrs	Testing all parts to make sure they work independently and together.
Documentation	40hrs	Document designs, test results, hardware setup, system architecture, and final products.

3.7 Other Resource Requirements

Hardware

- Raspberry Pi
- Servo
- Motion sensor(PIR)
- Feeder materials.

Software

- Cloud services (AWS/GCP)
- TensorFlow / PyTorch

Other

- Development tools (VS Code, GitHub)
- Dataset (Images captured during the testing period)

4 Design

4.1 Design Context

4.1.1 Broader Context

Our design problem is situated within the intersection of residential wildlife management and eco-conscious hobbies. The primary communities affected by this design are casual backyard birdwatchers, serious avian enthusiasts, and eco-conscious homeowners.

Public health, safety, and welfare: The persistent presence of nuisance wildlife often increases the risk that frustrated homeowners will resort to dangerous chemicals or traps. Our system provides a safe, non-toxic alternative that prevents the escalation to dangerous pest control methods.

Economic: As the cost of premium seed rises, feeding neighborhood squirrels becomes an unsustainable financial strain. This project reduces the financial waste of spilled or stolen birdseed.

Environmental: Due to increased environmental awareness, people want to support local biodiversity. The design promotes natural ecosystem balance by effectively training squirrels to stay away without causing lasting harm or permanent displacement.

4.1.2 Prior Work/Solutions

The Squirrely Bird Feeder builds upon established IoT and machine learning principles. Passive Infrared (PIR) sensors provide a proven, low-power method to trigger outdoor electronics (like our camera) immediately upon detecting motion [1]. For species identification, recent IoT research highlights the efficacy of integrating edge devices with cloud-based computer vision models to perform real-time wildlife classification without overburdening local hardware [2]. Finally, ongoing research emphasizes that continuous data collection and model retraining are crucial for maintaining high computer vision accuracy in dynamic outdoor environments [3].

Existing Market Solutions & Differentiation

Current market solutions fall into two categories:

- **Mechanical Feeders (e.g., Squirrel Buster):** Use weight-activated springs to block access.
    - Pros: Effective physical deterrence; requires no power.
    - Cons: Cannot differentiate between squirrels and large desirable birds; offers zero digital or photographic experience.
- **Smart Camera Feeders (e.g., Bird Buddy):** Use Wi-Fi and AI to identify birds and send app notifications.
    - Pros: Excellent user experience and data collection.
    - Cons: Completely lacks a physical mechanism to stop pests from eating the feed.

Our system bridges this gap by combining the intelligence of a smart monitor with active, mechanical deterrence. The table below compares the advantages and shortcomings of our target solution against existing products:

Feature / Metric	**Squirrely Bird Feeder (Our Solution)**	**Smart Camera Feeders**	**Mechanical Feeders**
Pest Deterrence	**Active (AI-Triggered Gate)**	None (Passive alerts only)	Active (Weight-triggered)

Species Selectivity	**High** (AI differentiation)	None (Cannot block pests)	**Low** (Excludes by weight)
Digital Experience	**High** (App, photo gallery)	**High** (App, photo gallery)	None
Humane Approach	**Yes** (Behavioral training)	N/A	**Yes** (Mechanical block)
Technical Complexity	**High** (IoT, AI, and Motors)	Medium (IoT and AI only)	Low (Mechanical only)

8.3 References

[1] C. Ogbonnaya, L. Paish, and C. Njoku, "Conserving Rare Birds in an Ecosystem of Direct Competition for Food with Squirrels Using Squirrel-Proof Bird Feeder," Preprints, 2026. [Online]. Available: https://www.preprints.org/manuscript/202601.0635

[2] D. Gupta, "Motion detector using pir sensor," Electronics Maker, Dec. 2011. [Online]. Available: https://www.electronicsmaker.com/em/admin/pdf/construction/Motion.pdf

[3] T. Sardana and S. Bitragunta, "Deep Learning-Enabled Squirrel Detecting Prototype Development and Processing Analysis," in 2023 IEEE 20th India Council International Conference (INDICON), 2023, pp. 1-6.

4.1.3 Technical Complexity

The problem scope contains multiple challenging requirements that exceed current solutions. The project requires seamless integration of hardware and software, demanding that motion detection, AI classification, and a mechanical gate response operate promptly in a single continuous operation. The machine learning algorithm must accurately classify animals with at least a 70% accuracy rate , and the system must securely transmit and store images via cloud services.

4.2 Design Exploration

4.2.1 Design Decisions

Core Processing Unit: We decided to use a Raspberry Pi as the feeder's brain to control the camera, passive IR (PIR) sensor, and door servo. This ensures we have enough local compute power to interface with cloud services.

Deterrence Mechanism: We opted for a mechanical gate powered by a servo. This ensures a 100% humane, non-lethal, and non-toxic way to keep unwanted animals out.

Cloud Architecture: We decided to offload heavy processing to Cloud Services, utilizing an Inference Service, Species ID Service, and an MQTT Service for message routing. This decision prevents the Raspberry Pi from being overburdened and allows users to access their images remotely.

4.2.2 Ideation

When ideating the best deterrence method for the feeder, we considered the following five options based on current market solutions and user constraints: Ineffective plastic domes:

1. Passive barriers that squirrels easily bypass.
2. Slippery poles: Another passive, traditional mechanical defense that cannot keep up with acrobatic pests
3. .Loud Noises: An active deterrent that would violate our environmental requirement of not disturbing other wildlife.
4. Lethal traps or toxic chemicals: Harmful pest control methods that violate our requirement for humane wildlife management.
5. Smart Mechanical Gate: A motorized door triggered by AI classification to block access.

4.2.3 Decision-Making and Trade-Off

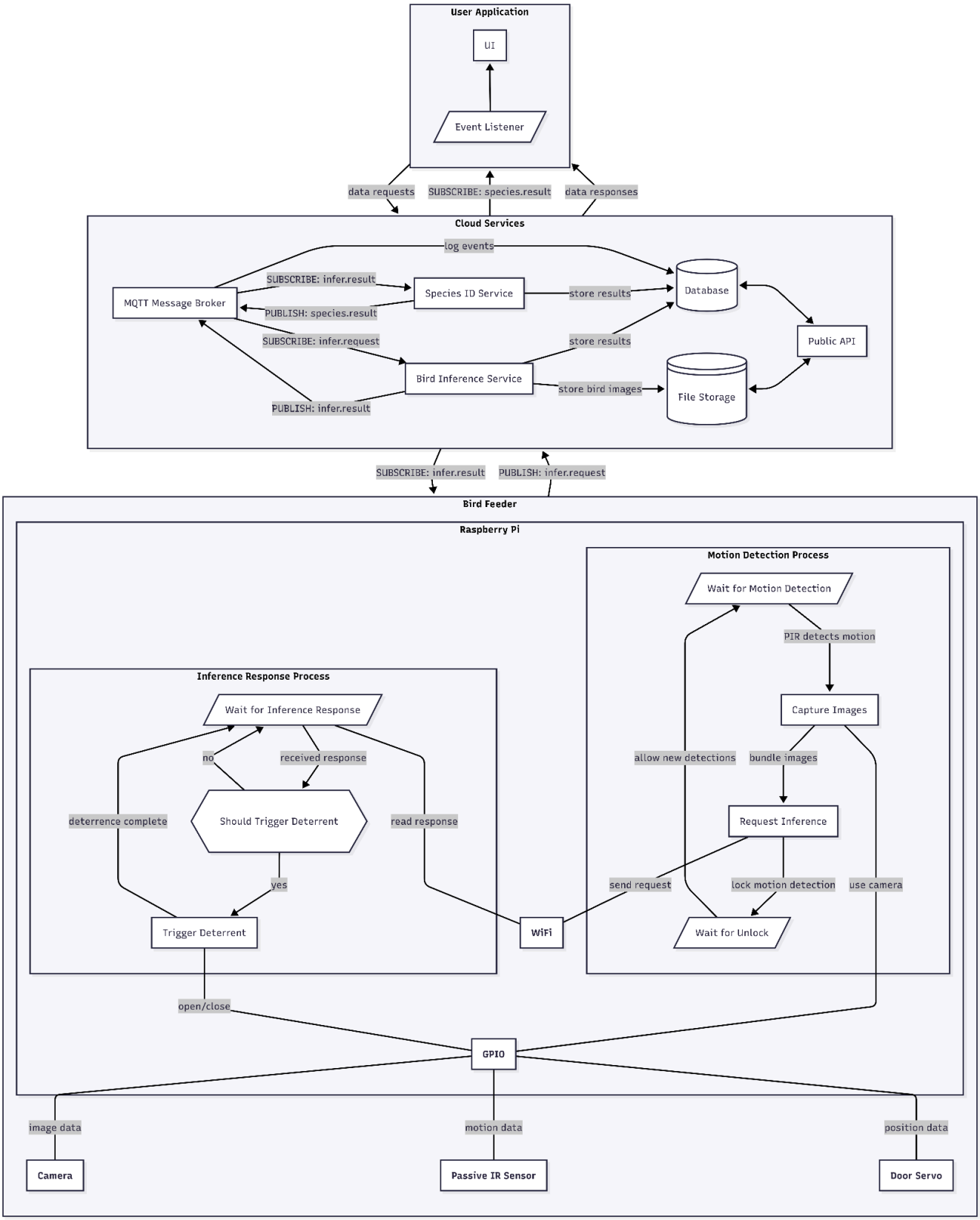
We chose the Smart Mechanical Gate. While traditional methods (domes and poles) are cheaper, they result in wasted premium birdseed and damaged property. Lethal methods and loud noises were immediately disqualified because they contradict the needs of our eco-conscious target users. The smart mechanical gate requires more technical complexity and power, but it is the only option that safely and humanely trains the squirrels to stay away without causing physical distress.

4.3 Proposed Design

4.3.1 Overview

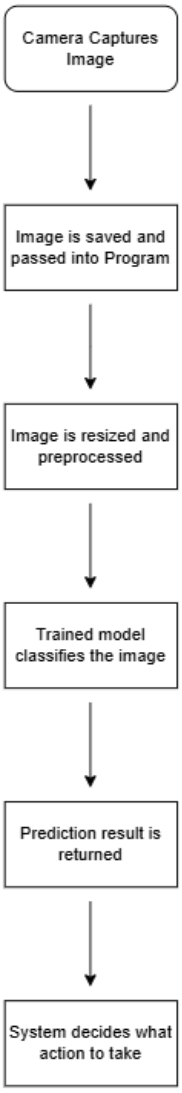
The bird feeder system consists of three major components: The Feeder itself, the Cloud Services, and the User Interface.

4.3.2 Detailed Design and Visual(s)


User Application
UI
Event Listener
data requests
SUBSCRIBE: species.result
data responses
Cloud Services
log events
SUBSCRIBE: infer.result
MQTT Message Broker
Species ID Service
store results
Database
PUBLISH: species.result
SUBSCRIBE: infer.request
store results
Public API
Bird Inference Service
store bird images
File Storage
PUBLISH: infer.result
SUBSCRIBE: infer.result
PUBLISH: infer.request
Bird Feeder
Raspberry Pi
Motion Detection Process
Wait for Motion Detection
PIR detects motion
Capture Images
allow new detections
bundle images
Request Inference
send request
lock motion detection
use camera
Wait for Unlock
Inference Response Process
Wait for Inference Response
no
received response
deterrence complete
Should Trigger Deterrent
read response
yes
Trigger Deterrent
WiFi
open/close
GPIO
image data
motion data
position data
Camera
Passive IR Sensor
Door Servo

Our detailed design is structured around three primary, interconnected subsystems: the Bird Feeder, the Cloud Services, and the User Interface. These components operate in a continuous loop to manage feeder access, process image data, and deliver a seamless user experience. (Please refer to the Squirrely Bird Feeder Vertical Block Diagram for the visual data flow).

AI System Flow:



1. Bird Feeder (Hardware & Edge Processing)

The physical bird feeder is controlled by a Raspberry Pi, which serves as the central processing unit. It interfaces with three main hardware components via GPIO pins: a camera for image data, a passive IR (PIR) sensor for motion data, and a door servo for position data. The Pi manages two distinct concurrent processes: Motion Detection Process: The system idles in a "Wait for Motion Detection" state. When the PIR sensor detects motion, it triggers the camera to capture and bundle a series of images. The Pi then locks further motion detection, requests an inference by sending the images over Wi-Fi, and enters a "Wait for Unlock" state. Inference Response Process: Concurrently, the Pi waits for an inference response from the cloud services. Once a response is received, the Pi evaluates the "Should Trigger Deterrent" logic. If a squirrel is identified, it triggers the deterrent by sending a signal via GPIO to adjust the door servo (open or close) to block access.

2. Cloud Services (Processing & Storage)

The cloud architecture offloads heavy machine learning and data management from the Raspberry Pi, consisting of several microservices.

Message Routing: An MQTT Message Broker handles all message passing, subscriptions, and request routing between the feeder, the cloud services, and the user application.Inference &

Identification: The system utilizes two distinct analytical services. The Bird Inference Service consumes images sent by the feeder and returns confidence scores for what makes up each image. If an image is determined to likely contain a bird, the Species ID Service runs a more intensive analysis to attempt to identify the exact species.

Data Storage: Both the Inference Service and Species ID Service log their metadata results into a Database. Concurrently, the captured images are saved in File Storage for user viewing.

Public API: A simple web service acts as a bridge between cloud storage and the user interface, allowing the mobile application to securely request and interact with the stored image data.

3. User Interface (User Application)

The user interface is the front-end platform where users interact with customization options and image collection features. It will consist of either a native mobile application or a Progressive Web App (PWA) that runs in a browser. An Event Listener within the application subscribes to species identification results from the MQTT broker. The application sends data requests to the Public API and receives responses, updating the UI to display the "digital doorbell" gallery of identified birds and feeder alerts.

4.3.3 Functionality

The Raspberry Pi waits for movement detected by the PIR sensor. This triggers the Pi to capture images with the camera and send them to the cloud inference service to identify whether a squirrel is present. If a bird is detected, access is allowed. If an unwanted animal is detected, the mechanical gate closes to block the feed. The device will reopen once the squirrel is no longer detected and a time threshold passes.

4.3.4 Areas of Concern and Development

Our primary concerns involve model accuracy, slow inference speeds, and weather damage. To address these, our immediate development plans include increasing the volume of our dataset to improve classification accuracy, optimizing our models to be as lightweight as possible to reduce system latency, and ensuring our equipment is properly waterproofed within the birdhouse.

4.4 Technology Considerations

Our design leverages several distinct technologies:Hardware: We utilize a Raspberry Pi, PIR motion sensors, and servos. The Pi provides excellent flexibility for IoT projects, though it requires careful power management in outdoor environments.

Cloud & ML: We rely on AWS/GCP for cloud services and TensorFlow/PyTorch for machine learning. By hosting heavy ML processing in the cloud, we trade local independence for higher accuracy and better data collection, though this requires a reliable Wi-Fi connection.

4.5 Design Analysis

Our current objective is to use the Raspberry Pi to get images of squirrels, so we have currently designed and 3D printed a case to act as a motion-activated trail cam. With this, we plan to mount it on a post and point it at a bowl of birdseed to collect images, which will be triggered whenever the motion sensor goes off. Hopefully, after a few days, this will produce a good amount of photos. After finding images that are actually squirrels, we'll use them to train the squirrel detection model, so it will have a dataset of photos from the same camera it will regularly receive images from.

Currently, the motion-activated camera works as intended, but we haven't had an opportunity to set it up and test it outside to gather images yet. After we gather enough images and set up the camera, we'll move towards creating the actual bird feeder that will close the gate when it detects a squirrel. As far as the current design iteration goes, we need to make the feeding trough wider so it provides more room for squirrels and birds to get on it and give us a wider picture to see more of the target in the photo. There haven't been any real issues with building it, aside from a couple of design oversights when making a case that could be mounted to a board or post for the trail cam.

But once we have the trail cam case finished and gathering data, we'll move on to working on the actual birdfeeder and printing it and testing it in sections (gate, mounting the bird feeder, feed hopper, securing the lid onto the top and keeping it secured, mounting the camera/sensor, and the Raspberry Pi). As we flesh out each section, we'll put it all together.

5 Testing

- Testing is a critical component of the Squirrely Bird Feeder project, ensuring that our combination of mechanical, software, and AI components functions harmoniously. Our overarching testing philosophy relies on "early and often" iterative testing in real-world environments. A unique challenge to our system is

the unpredictability of outdoor environments and live wild animals. Lighting changes, wind, and background movement have already proven to be significant variables. Therefore, our strategy heavily emphasizes field testing over purely simulated bench testing.

5.1 Unit Testing

- Units Tested: AI recognition model, PIR motion detection script, and servo actuation logic.
- Methods & Tools: The AI model is tested against a validation dataset of captured squirrel images (currently returning 99% confidence). Hardware components like the PIR sensor are tested manually for distance sensitivity triggers, and power draw is measured using a multimeter to ensure the 5W peak target is met.

5.2 Interface Testing

- Interfaces Tested: Communication between the Raspberry Pi and the Camera, PIR sensor, and Servo motor.
- Methods & Tools: We verify that a signal from the PIR properly initiates the camera capture script via GPIO pins. Tools include Python test scripts and system logging to track I/O timing and ensure no dropped signals.

5.3 Integration Testing

- Critical Paths: The primary integration path is: PIR triggers -> Camera captures image -> AI Model processes image -> Logic determines species -> Servo actuates (if squirrel).
- Methods & Tools: This is tested on the bench by simulating motion and providing control images to the camera to ensure the servo gate responds correctly based on the AI's classification.

5.4 System Testing

- Strategy: End-to-end testing by deploying the fully assembled prototype outdoors.
- Methods & Tools: We deploy the feeder with feed (currently corn to attract wildlife, explicitly avoiding rice to protect bird health) and a 64GB SD card. We review system logs and captured images to evaluate system performance as a whole.

5.5 Regression Testing

- Strategy: Ensuring that updates to the AI model or edge computing logic do not degrade previously established accuracy.
- Methods: Every time the AI model is updated to include bird identification or background removal algorithms, it will be re-run against our baseline dataset of 1,500 images to verify accuracy is maintained.

5.6 Acceptance Testing

- Strategy: Demonstrating to our client (Randall Geiger) that the feeder successfully identifies squirrels and closes the gate, while remaining open for birds.
- Client Involvement: We regularly review captured field data and prototype iterations with Geiger (as done in our recent meeting where we reviewed the 600 MB of capture data).

5.7 Security Testing (if applicable)

- Strategy: Securing SSH access to the Raspberry Pi to ensure no unauthorized access occurs while it operates on the ISU network.

5.8 User Testing

- Strategy: In our case, the "user" involves both the backyard birdwatcher and the wildlife. We evaluate usability by seeing how easily the feeder can be deployed, powered (via drill battery), and accessed for data retrieFval. Wildlife usability is tested by observing if animals are deterred by the mechanism or shape of the feeder.

5.9 Results

- Current Results: Our initial deployment resulted in a 99% confidence rate on squirrel detection based on 1,500 collected images. However, we encountered significant false positives (6,480 images due to background movement) and link-layer data overload (8 GB captured in 2.5 days). Furthermore, a bug was identified where sleep mode halts motion detection unless a monitor is attached.
- Next Steps: Implement an algorithm to remove backgrounds, elevate the feeder to attract more birds, apply an IR filter to reduce PIR FOV, and pivot to edge computing on the Pi to reduce OTA data transmission.

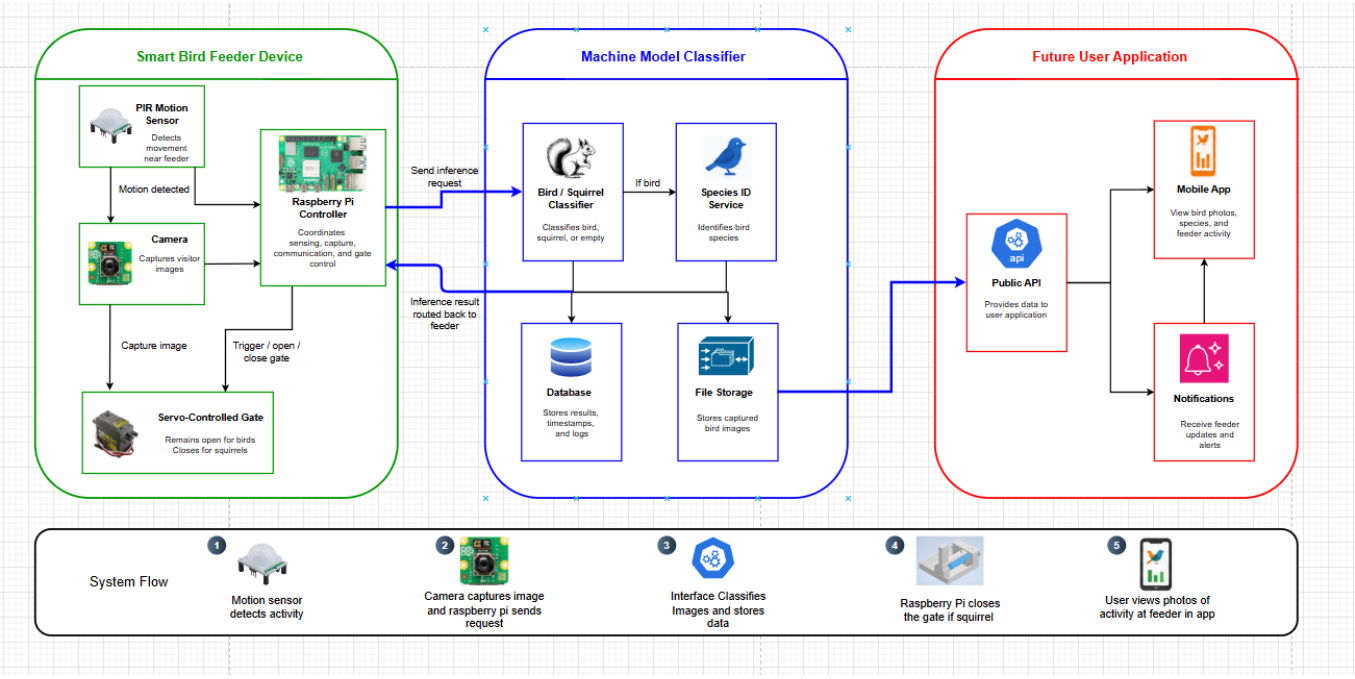
6 Implementation

Preliminary implementation involves our first physical prototype. We currently utilize a Raspberry Pi connected to a camera module and a PIR sensor. Because we are waiting on 3D printing filament, the initial mechanical prototype utilizes an oversized servo mount/hub. It lacks the final deterrence gate but successfully functions as a data-collection platform.

The system is powered by a drill battery, which calculations show supports the 5W peak draw for a full day. Software-wise, the Pi captures images to an SD card upon PIR triggers. Initially, data was sent to an ISU ETG VM, but due to massive data collection from wind/background noise, the software implementation is pivoting to run the recognition model directly on the Pi (Edge computing).

As we transition to this on-device model, a critical design consideration is preventing data leakage during training. Because PIR triggers often capture bursts of nearly identical images, simply randomizing our train/test split would place highly correlated frames into both datasets. This would allow the model to "cheat" and achieve artificially high accuracy by memorizing specific backgrounds, lighting, or motion events rather than genuinely learning the target features. To prevent this and ensure the model performs exactly as intended, we are strictly isolating our training and testing data by distinct trigger events rather than by individual frames.

High Level Design:



7 Ethics and Professional Responsibility

7.1 Areas of Professional Responsibility/Codes of Ethics

Area of Responsibility	Definition	Relevant Item from IEEE Code	Team Interaction

Public Health & Safety	Ensuring our product does not harm the public or environment.	Item 1: "To hold paramount the safety, health, and welfare of the public..."	We actively researched animal safety, determining we must avoid rice as it expands in birds' stomachs and causes health issues.
Technical Competence	Being honest about our system's capabilities and limitations.	Item 3: "To be honest and realistic in stating claims or estimates..."	We acknowledged that our 99% squirrel detection rate might be artificially inflated due to identical background conditions and are acting to fix it.
Technology and Society	Understanding the environmental impact of our device.	Item 5: "To improve the understanding by individuals and society of the capabilities..."	Working with experts like the Audubon society to better understand native Iowa bird species and how to ethically attract them.

- Area of Strong Performance: Health & Safety (Animal Welfare). We are prioritizing the well-being of the wildlife by selecting safe feed (corn over rice) and ensuring the mechanical gate will not injure animals during actuation.
- Area to Improve: Reliability/Competence. The current prototype has bugs (sleep mode requiring a monitor, extreme false positives from the PIR). We must improve our software robustness before finalizing the product.

7.2 Four Principles

Broader Context Area	**Beneficence**	**Nonmaleficence**	**Respect for Autonomy**	**Justice**

Environment / Wildlife	Providing a safe food source for local bird populations.	Ensuring the mechanical gate does not pinch or harm squirrels when closing.	N/A (Animals do not have autonomy in the human sense)	Ensuring fair access to food for the intended target (birds).
Client / End User	Providing an automated, stress-free birdwatching experience.	Ensuring the device does not cause electrical fires or damage property.	Allowing the user to easily turn off the camera or data collection.	Providing an affordable ($500 limit) solution.

- Important Pair (Environment - Nonmaleficence): Our primary goal is to deter squirrels without causing them physical harm. We will ensure this by designing the servo-gate to close gently and safely, using image recognition to trigger before the squirrel is inside the mechanism.
- Lacking Pair (End User - Autonomy): Initially, our cloud-based image collection may strip some privacy from the user's yard. We are mitigating this by moving the image processing to the Edge (locally on the Pi) so private yard images are not transmitted over a network.

7.3 Virtues

1. Innovation

Definition: Approaching traditional problems with modern, out-of-the-box thinking, and being willing to integrate complex technologies (like AI and IoT) into everyday contexts.

How we support this: We actively encourage exploring new frameworks and hardware solutions. During our ideation phase, instead of settling for passive mechanical barriers that currently exist on the market, we collectively supported the ambitious approach of using a Raspberry Pi, edge ML processing, and cloud architecture to create a truly smart, active deterrence system.

2. Accountability

Definition: Taking personal ownership of specific subsystems (Hardware, ML, Software/Backend, UI) while ensuring those components meet the technical standards required for final integration.

How we support this: We utilize a Kanban project management board to track our parallel workflows. Every team member is responsible for moving their tasks from "To Do" to "Done" and reporting any blockers during our communication check-ins on Discord. This ensures no one is left holding up the pipeline.

3. Adaptability

Definition: The ability to pivot and adjust designs when real-world constraints (like weather, hardware limitations, or AI accuracy) challenge our initial assumptions.

How we support this: We approach testing as a learning opportunity rather than a pass/fail metric. For instance, as we transition from our 3D-printed camera case prototype to outdoor image collection, we hold collaborative review sessions to discuss what failed and immediately adjust our physical housing or code without placing blame.

Part 2: Individual Reflections

Wyatt Sinclair

Demonstrated Virtue: Perseverance

Why it is important: Machine learning can take multiple tries, setbacks, or failures. It takes lots of effort time, and learning to achieve the desired goal

How I have demonstrated it: Throughout the project I have continued to improve the bird feeder system even when running into challenges with datasets or the model. Our model showed high confidence, but we had to question whether the results were reliable. I worked to better understand the model's behavior, clean ip the code, and test using different types of images.

Un-demonstrated Virtue: Patience

Why it is important: Machine learning training and outdoor data collection require significant trial and error to get right.

What I might do to demonstrate it: I will practice patience during the upcoming outdoor image collection phase by systematically testing different angles and lighting conditions for the camera, rather than rushing straight to the final integration.

Jack Morrison

Demonstrated Virtue: Diligence

Why it is important: Hardware and software integration requires precise timing, especially for a continuous operation loop like motion detection to image capture.

How I have demonstrated it: I put in the detailed, careful work into getting the PIR motion sensor to successfully trigger the Pi camera, and integrating the presence classification algorithm into the process to reduce false positives.

Un-demonstrated Virtue: Proactive Communication

Why it is important: With parallel Kanban tasks, the software team cannot build the backend effectively if they don't know exactly what the hardware team is outputting.

What I might do to demonstrate it: I will ensure I am thoroughly documenting our hardware pinouts and sensor data structures and sharing them directly with the backend developers before they ask for it.

Miles Nichols

Demonstrated Virtue: Curiosity

Why it is important: Implementing an edge-based machine learning model that hits a 70% accuracy requirement involves tools outside the standard curriculum.

How I have demonstrated it: I took the initiative to research and understand how to properly train and optimize a lightweight classification model that can run efficiently on a Raspberry Pi to distinguish between birds, squirrels, and empty feeders.

Un-demonstrated Virtue: Prudence (Time Management)

Why it is important: Collecting and labeling a custom dataset for machine learning is a massive, time-consuming task that cannot be rushed at the last minute.

What I might do to demonstrate it: I will break down the dataset processing and model training into strict daily quotas to ensure we have a robust model ready in time for our system-level testing.

Kenny Tran

Demonstrated Virtue: Accountability

Why it is important: The mechanical gate is the core safety requirement of the project; it must operate flawlessly to avoid trapping or injuring wildlife.

How I have demonstrated it: I took ownership of the servo-controlled gate mechanism, ensuring it aligns with our ethical requirement to humanely deter pests without causing physical distress.

Un-demonstrated Virtue: Flexibility

Why it is important: Real-world outdoor conditions (like temperature fluctuations or rain) will likely expose flaws in our initial mechanical designs.

What I might do to demonstrate it: I will remain open to redesigning the gate structure or servo placement if our upcoming environmental testing reveals that the current setup cannot withstand outdoor use.

Benjamin Bartels

Demonstrated Virtue: Detail-Orientation

Why it is important: Moving data from an edge device to the cloud requires secure, error-free pathways, especially when handling user image data.

How I have demonstrated it: I focused heavily on the meticulous setup of our cloud architecture and backend communication, ensuring the image data and timestamps are properly routed and stored.

Un-demonstrated Virtue: Assertiveness

Why it is important: To prevent integration bottlenecks, team members must push back if a proposed hardware design makes software integration too difficult.

What I might do to demonstrate it: I will actively speak up in our next integration meeting if the data formats coming from the Pi are not optimized for our cloud database, rather than just trying to quietly fix it on the backend.

Nolan Hoenert

Demonstrated Virtue: Foresight

Why it is important: A complex IoT system requires a strong architectural blueprint (like IEEE 2413) to ensure the physical, network, and software layers communicate seamlessly.

How I have demonstrated it: I helped keep the big picture in focus, looking ahead at how the hardware components, UI dashboard, and cloud services will eventually tie together to create a smooth user experience for the casual birdwatcher.

Un-demonstrated Virtue: Empathy (User-Centricity)

Why it is important: The final dashboard must be low-maintenance and accessible for casual users who just want to see their bird pictures.

What I might do to demonstrate it: I will spend time reviewing the UI/app design specifically from the perspective of a non-technical homeowner, ensuring we prioritize ease-of-use over overly complex data displays.

8 Closing Material

8.1 Conclusion

The goal of the Squirrely Bird Feeder is to create an automated, AI-driven feeder that deters squirrels while allowing birds to feed safely. Thus far, we have successfully built a data-gathering prototype, integrated the Raspberry Pi, Camera, and PIR sensor, and trained an initial model with 99% confidence on squirrels. However, we have been constrained by environmental noise (PIR false positives) and high data transmission costs. Moving forward, we will implement edge processing on the Pi, deploy the feeder in an elevated location to attract actual birds, physically restrict the PIR sensor's FOV, and print the final, weather-proof enclosure.

8.2 References

List technical references and related work / market survey references. Do professional citation style (ex. IEEE). See link:
https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf

8.3 References

[1] C. Ogbonnaya, L. Paish, and C. Njoku, "Conserving Rare Birds in an Ecosystem of Direct Competition for Food with Squirrels Using Squirrel-Proof Bird Feeder," Preprints, 2026. [Online]. Available: https://www.preprints.org/manuscript/202601.063 5

[2] D. Gupta, "Motion detector using pir sensor," Electronics Maker, Dec. 2011. [Online]. Available: https://www.electronicsmaker.com/em/admin/pdf/construction/Motion.pdf

[3] T. Sardana and S. Bitragunta, "Deep Learning-Enabled Squirrel Detecting Prototype Development and Processing Analysis," in 2023 IEEE 20th India Council International Conference (INDICON), 2023, pp. 1-6.

9 Team

Complete each section as completely and concisely as possible. We strongly recommend using tables or bulleted lists when applicable.

9.1 Team Members

9.2 Required Skill Sets for Your Project

(if feasible – tie them to the requirements)

9.3 Skill Sets covered by the Team

(for each skill, state which team member(s) cover it)

9.4 Project Management Style Adopted by the team

Typically, Waterfall or Agile for project management.

9.5 Initial Project Management Roles

(Enumerate which team member plays what role)

9.6 Team Contract

Team Members:

1. Wyatt Sinclair
2. Jack Morrison
3. Miles Nichols
4. Kenny Tran
5. Benjamin Bartels
6. Nolan Hoenert

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

Virtual or in-person meetings as needed

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g.,

e-mail, phone, app, face-to-face):

Discord channel

3. Decision-making policy (e.g., consensus, majority vote):

Majority vote 4/6

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be

shared/archived):

A shared link in the Discord with an AI transcriber.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

Attend all meetings with our faculty member, show up, and be attentive during team meetings. Do not do other work and respect the time of the other members.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

If you volunteer for something, you'll be expected to get it done within the time constraints. If unable to get it done in time, reach out at least a day before the deadline so others can help.

3. Expected level of communication with other team members:

Give a day time for a response. Check Discord daily.

4. Expected level of commitment to team decisions and tasks:

Everyone has a vote/say in how the team makes decisions & assigning tasks.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

Wyatt Sinclair - Team organization

Miles Nichols - design

Benjamin Bartels - 3D modeling

Jack Morrison - testing

2. Strategies for supporting and guiding the work of all team members:

Give people the work they want and are interested in first, and evenly distribute if two people want to do the same thing.

3. Strategies for recognizing the contributions of all team members:

Have a weekly text stating what you have done, what you are doing, and what you plan to do.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

Wyatt Sinclair - Good at managing a team. Work this summer as a manager of a hardware team.

Miles Nichols - Can work in diverse groups and lead conversation. Engineering experience in software and a bit of hardware.

Kenny Tran - Experience with computer vision and machine learning from undergraduate research.

Benjamin Bartels - Good at extrapolating different ideas and roadblocks that alternative ideas could make. Experienced with programming. Experienced with designing 3D models and parts. A little experience designing small electrical projects.

Jack Morrison - Computer and software design experience as well as experience working on projects with deadlines.

2. Strategies for encouraging and supporting contributions and ideas from all team members:

Normalize incomplete ideas as suggestions in discussion and have the entire group build on them to avoid staying silent.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

Reach out to everyone with a text message stating their concern and their solution. Allow others to way in on how to fix the solutions. Having a majority agreement of 4/6 on how to fix the problem if people are split.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

Have a prototype design that can detect squirrels and differentiate between animals.

2. Strategies for planning and assigning individual and team work:

Have people volunteer to start with. Try to have 2 people working on the same thing.

Assign work to the suit majors when possible if it makes sense.

3. Strategies for keeping on task:

Do not plan meetings to last too long. Have the first 10 minutes of the meeting for an external talk

Consequences for Not Adhering to the Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

Address at team meetings with everyone present. Go over how the member is not adhering, issue a warning, and make sure they understand the other team members' frustrations. Write down the altercation time and what was discussed.

2. What will your team do if the infractions continue?

Report to our faculty and professors with evidence of what has happened.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Wyatt Sinclair	DATE 5/1/26
2) Jack Morrison	DATE 5/1/26
3) Miles Nichols	DATE 5/1/26
4) Kenny Tran	DATE 5/1/26
5) Benjamin Bartels	DATE 5/1/26
6) Nolan Hoenert	DATE 5/1/26