University of Waterloo
Faculty of Engineering
Department of Computer and Electrical Engineering

Garden Sentry

Detailed Design and Timeline Report

Group 2020.21

Prepared By:
Ryan Tjhie rtjhie 20603162,
Sharon Zheng s52zheng 20620125,
Lucas Fayoux lpfayoux 20626344,
Monta Gao mzgao 20620168,
Jonathan Xue j32xue 20614443

Consultant: Nachiket Kapre

28 June 2019

Table of Contents

1 High Level Description

This section provides an overview of the Garden Sentry fourth year design project with motivations, objectives and a high-level overview of the system.

1.1 Motivation

Agriculture has been part of human society since the dawn of civilization and has remained important to humanity since. As such, there have been many methods developed to deter pests from impeding the yields of the harvest. What was once a simple scarecrow has now evolved into mass distribution of chemical pesticides. Although effective, it has been observed that these pesticides have proven to exhibit devastating effects on the environment [1]. These also leave a possibility for damage to the consumer, as residues from the pesticides can remain in the produce when it is consumed [2]. This has led to an increase in the chemical-free pesticide market [1].

An electronic system could satisfy this demand as it will be chemical-free and thus would not present any of the previously highlighted defects.

1.2 Project Objective

The objective of the Garden Sentry project is to design a garden protection system that automatically detects and removes the presence of common garden pests. This product will not only protect client gardens, it will also avoid common drawbacks of more traditional pesticides, such as destruction of the environment and excessive damage to the surrounding wildlife.

1.3 Block Diagram

The proposed solution consists of main three subsystems: the web application portion which allows the user to interface with the recorded video snapshots and see the results of the garden sentry, the garden sentry hub which aggregates video and sensor data for the microcontroller to process, and the mechanical subsystem of the water gun which relies on the hub's control signals to be operated. This high level overview of the Garden Sentry system is shown in Figure 1 and Figure 2 below.

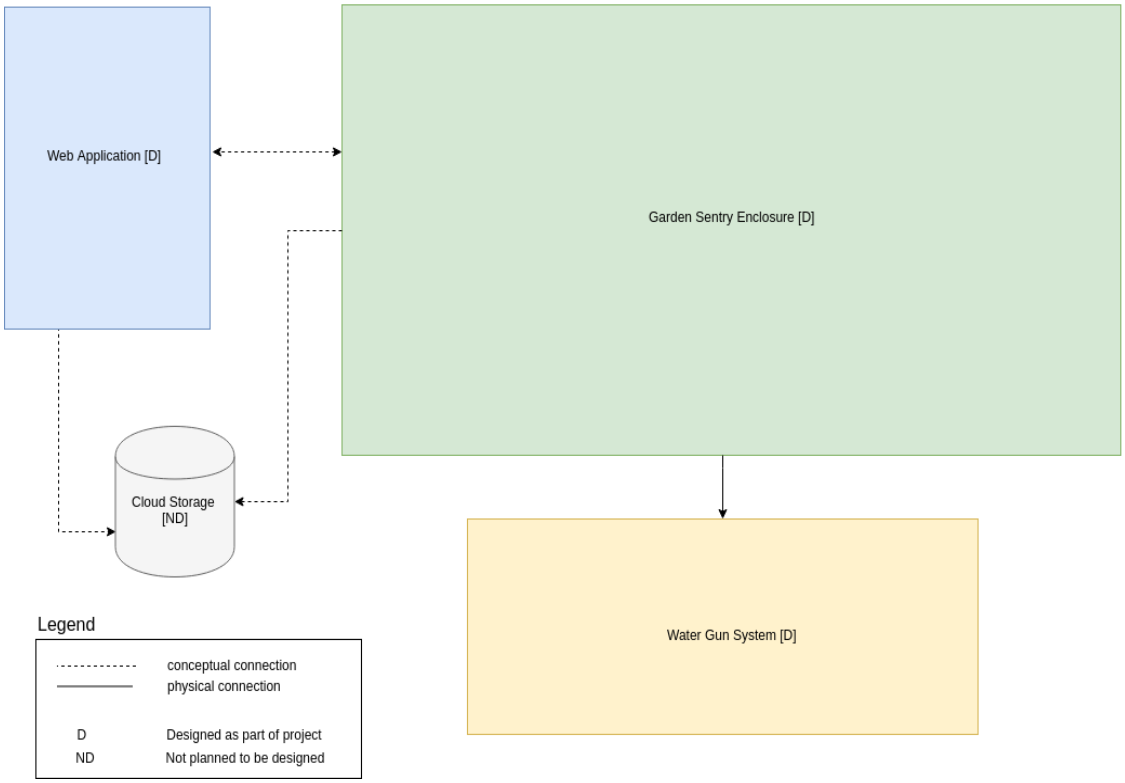

Figure 1. Simplified Block Diagram of the Garden Sentry project

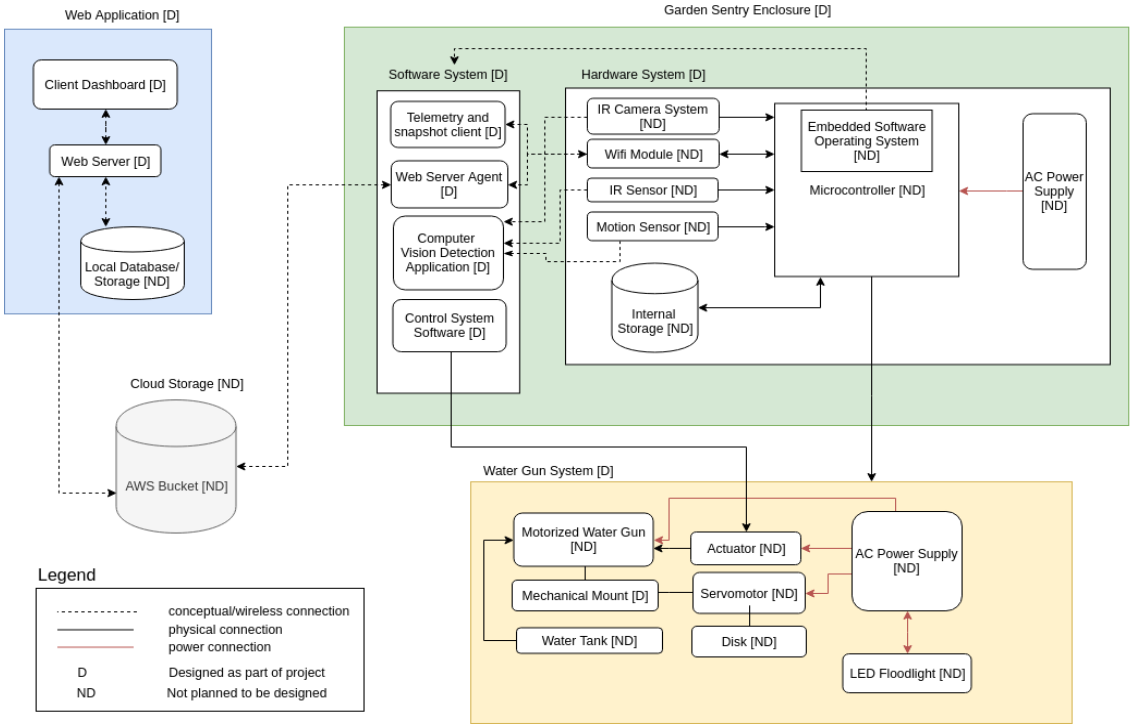

Figure 2. Detailed Block Diagram of the Garden Sentry Project

2 Project Specifications

The project specifications are divided into two categories: functional specifications and non-functional specifications. These are both outlined in the following two tables.

2.1 Functional Specifications

Specification	Description	Needed
Detect Pest	Detect pests 5m in front of the device (rabbits, squirrels, other) in a 70° field of view	Yes
24-hour Usage	Day and Night	Yes
Scare Pests	Shoot water within a 5m range at pest until it is no longer detected	Yes
Data Retention	Video clips or photos of pest detection are stored online	Yes
Security System	Detect entry into garden by intruders (rabbits or otherwise)	No
Plant Watering	Water plants when shooting at pests	No

2.2 Non-functional Specifications

Specification	Description	Needed
Price	Within budget	Yes
Quiet	Noise within reason and legal limits for gardens and yards. Must be under 85dBA according to City of Toronto	Yes
Water resistant	Hardware is protected from water and potential short circuits	Yes
Size	Less than $1m^3$ in volume and 1mx1mx1m in dimensions	Yes
Wifi Updates	Update firmware over WiFi	No
Aesthetically Pleasing	Aligned with typical garden decor	No
Weight	Less than 50kg	Yes

3 Detailed Design

3.1 Embedded Software Subsystem

The embedded software system consists of several parts. First, there is the motor control system, which handles moving the water gun to an appropriate position and activating it. Second, there is the pest detection system, which utilizes infrared sensors in combination with computer vision to detect pests within the garden. Third, there is the control software which takes as input the video feed and sensor feed, passes it to the pest detection system and uses its outputs to determine when to use the motor control system to use the water gun to scare pests. The chosen microcontroller is the Nvidia Jetson Nano, the microcontroller uses the Ubuntu operating system, an environment which many members are familiar with. Furthermore, the Nvidia Jetson Nano is optimized for Artifical Intelligence (AI) tasks like Computer Vision which is used for the Rabbit Detection System. It is common to program in Python on embedded systems such as this due to its ease of use and speed of development and many common libraries provided for controlling the microcontroller are written in Python.

In Figure 3 below, the control system is described in an overall flow. The day mode and night mode deal with how we should use the camera and the infrared sensor. During the day, we can simply poll the camera and during night mode we can use the infrared sensor to turn on the lights when movement is detected, in combination with the camera. The video feed is processed by the computer vision system and will return the pests location within the frame. Once a pest has been detected without motion for 3 seconds, the motors and servos will be repositioned to the appropriate angle and the water gun will be activated by turning on the pump. A sliding window of video is stored with a maximum capacity of 20 seconds and 10 seconds after shooting the water gun the video will be taken and sent to the server. The Jetson Nano uses a Wi-Fi connection to make a simple HTTP Post request to the Web Server to upload and store the video.

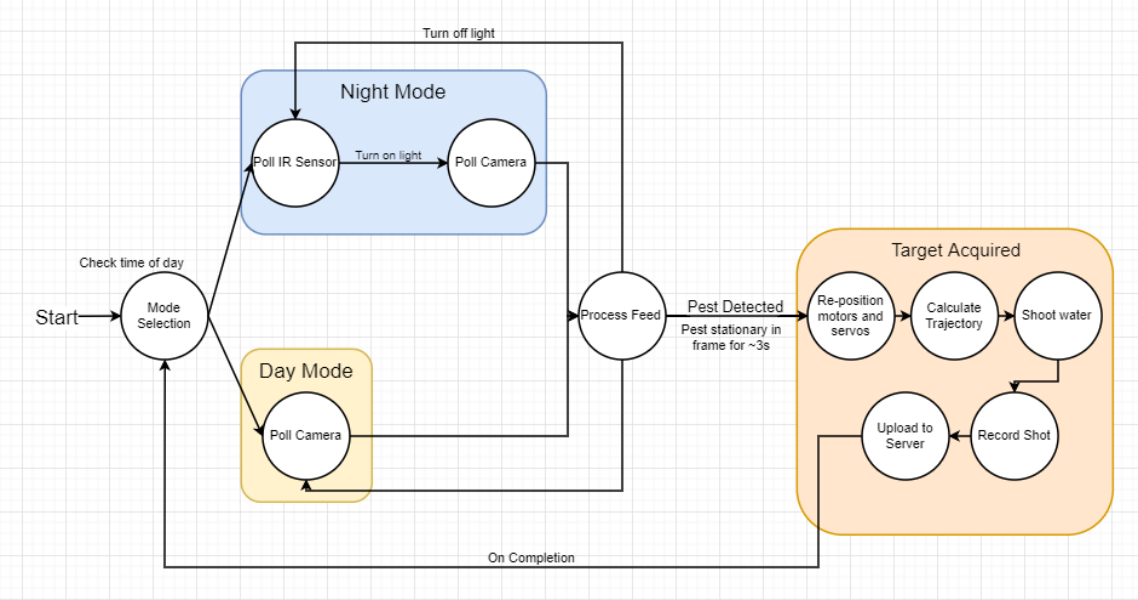

Figure 3: Overall Control System

3.1.1 Rabbit Detection Subsystem

The rabbit detection system is designed to use infrared sensors in combination with computer vision to detect pests within the garden. The computer vision system uses the You Only Look Once Version 3 (YOLOV3) object detection model to detect pests. In order to detect pests, the model must be trained on approximately one thousand images of the desired pest. The reason for this number of images is because the object detection model must also be able to recognize the pest at a variety of angles and distances and the model requires many images to achieve a high level of accuracy.

The infrared sensor is used at night, where the camera will likely be unable to detect the pest. When pests walk in front of the Garden Sentry, the infrared sensor will detect the movement and shine a light towards the pest. The light will remain on for 30 seconds after detecting motion, after which it will shut off again. The light should be sufficient for the camera to detect the pest in front of it, or to even scare it away on its own. If the infrared sensor and light was not used, then the system could not be used at night due to poor lighting. An alternative to the infrared sensor would be to turn a light on all night in the garden, however this was considered a poor choice due to power consumption and the light could be considered an annoyance by owners of the product. An alternative to the object detection system was considered, such as a simpler image classifier, however this solution does not provide the system with the location of the pest and the water gun could not be fired in the direction of the pest to scare it away.

In terms of speed, object detection is reported to process images at many different rates depending on the resolution of the images sent to the computer vision model as well as the computer vision model used. The most relevant model recorded is the Tiny-YOLOV3 model which is optimized for running on embedded systems or systems with limited resources. The model was recorded to run at 25 frames per second with a resolution of 416x416 pixels [3]. Further testing will need to be done to see if the resolution is sufficient to detect the pests or if a higher resolution will be required. Regardless, the tradeoff in this design will be between the resolution and the speed of the computer vision model and the currently estimated 416x416 pixels will be used for training the model.

Figure 4 below is an example of the output of the YOLOV3 model. There are boxes around each predicted object as well as a label of what was predicted. Furthermore, the system works even when the object is not fully in view as seen with the dog partially blocking the view of the bicycle.

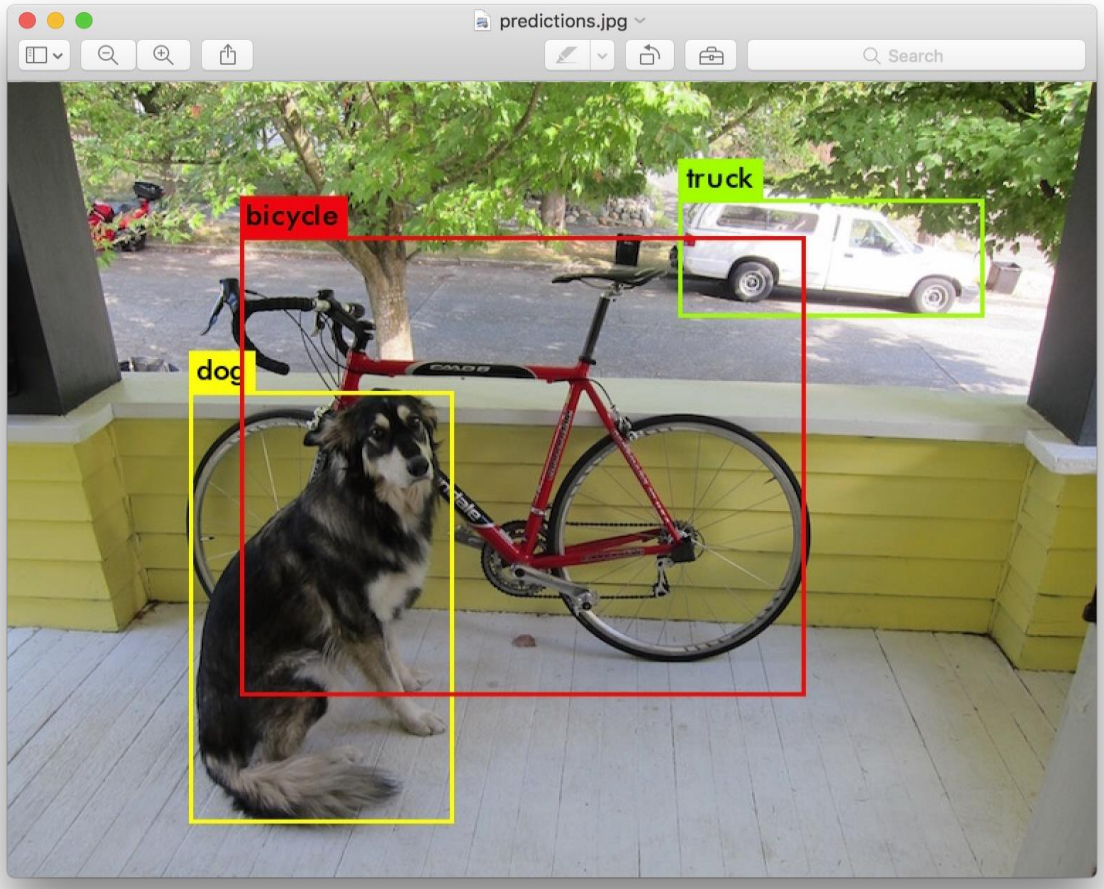

Figure 4. YOLOV3 Object Detection *[4]*

3.3 Web Application Subsystem

The Garden Sentry requires a user facing website for customers to both look at their analytics as well as view snapshots of the Sentry in action, in order to paint a better picture of their Garden Defense system. It also requires a user facing front-end and its corresponding application architecture as well as a back-end server architecture and storage subsystems to operate as desired, with these subsystems interacting cohesively and properly scaling as the userbase grows.

3.3.1 Client-facing Architecture

There are a few choices for client-facing architectures, with a variety of different frameworks available. Most modern frameworks follow a model-view-controller (MVC) design pattern for UI, which abstracts the data (the model) away from the visual components (view) by limiting their interaction strictly through the software which manages the logic between the data and the visual interfaces (the controller). Some examples of such frameworks include React, Angular, Vue.js and Ember.js. As some of their names imply, these frameworks are all Javascript web applications.

React is a popular framework which acts as the view in standard MVC principles, maintained mostly by Facebook. React's main advantages include its performance, simplicity due to increased abstractions, and wide adoption and documentation. The stand-out feature of React is its use of a virtual DOM, which allows for more efficient browser rendering, since it keeps a cached structure of the page and re-renders components only on changes. It uses one-way data binding which allows for more easily maintainable code and arguably more logical software design patterns. Some of the disadvantages include the high memory cost the framework requires due to its use of a virtual DOM abstraction, which it keeps in memory and syncs with the page's real DOM.

The client facing architecture of choice is React, mainly due to its extensibility, ease of use, popularity, and existing familiarity among engineering team members.

3.3.2 Client-facing Interface

The interface has several requirements which correspond to several user use cases. The first use case is when a user wishes to see a brief overview of the sentry, including when the last defense event occurred was, how many events have occurred in given timeframes, and the uptime of the sentry, as well as any diagnostic information available. A user can also share their stats and show off their defense statistics on various social media.

The second use case is when a user wishes to see the actual footage of defense incidents, in the form of short clips. The interface displays a nice gallery of incident snapshots which can be sorted by date and can

be selected to be played using the video player of the user's browser. A drawn design of a single page application which satisfies both the use cases above can be seen below.

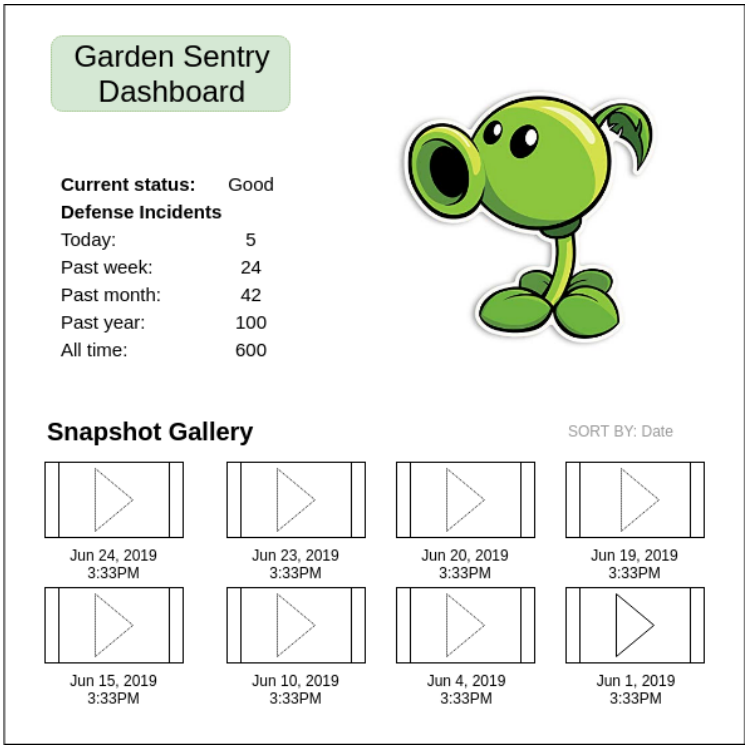

Figure 5. Mockup of User Interface

3.3.3 Web Server Architecture

The server application serves as the backend component for the overall web application, whose main purpose is to serve the user facing website content to the client, through database fetches and minor logical computation and authentication. The server application is designed scale reliably to accommodate a growing user base and large bandwidth of data, due to the video streaming aspect of the web application.

As in the case of the front-end frameworks, there are many widely used backend frameworks available for building web applications and are written for a variety of programming languages as opposed to only JavaScript like in the case of the front-end frameworks. Some popular ones include Django which uses Python, Express which is a library on top of the Node.js runtime, and Ruby on Rails, which uses Ruby. This project uses Express due to its simplicity and ease of integration with the client-facing architecture (React). A high-level overview of the three tier architecture can be seen in the figure below.

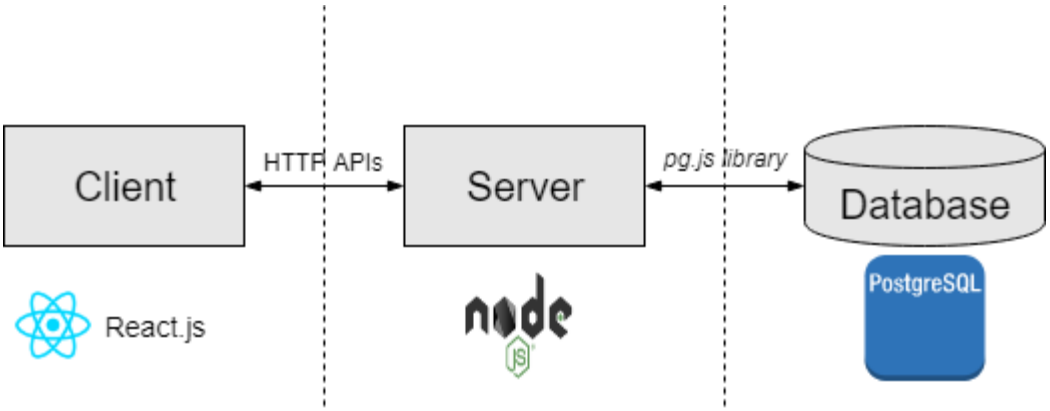

Figure 6. Three Tier Architecture Overview

3.3.4 Database Design

Garden Sentry generates a lot of data and is meant to be scalable, allowing users to view a variety of information about their Garden Sentry from a single web page. This requires that sentry data, including event logging, sentry metadata (geolocation, id, name, serial number) be stored in a database and easily queried by the web server. The database of choice in this project is PostgreSQL, a relational database management system which emphasizes robustness and extensibility

event		
PK	eventId	int
	eventType	eventType
	eventMessage	text
	timestamp	ts with timezone
FK	deviceId	int

device		
PK	deviceId	int
	version	text
	location	JSON {lat: , long:}
	lastSeen	ts with timezone
	deviceName	text
	status	text

Figure 7. Event and device relational tables

A row in the event table represents a meaningful event, categorized by *eventType*, and detailed through the *eventMessage* fields. Along with the event data, the database store some metadata, including the event's timestamp as well as the unique id of the sentry which logged the event, named *deviceId*.

A row in the device table represents a sentry, which includes the sentry's firmware *version*, *location* in terms of latitude and longitude, the last time it was seen as *lastSeen*, an optional user given *deviceName*, and the current operational status as *status*. The goal for the database schema is to keep it lightweight and extensible if new features need to be added, which is why the fields are somewhat minimal.

These two tables and their fields provide all that is needed to populate our database and build our front-end user interface.

3.3.5 Cloud Server Provider

The web application requires servers for hosting our web application as well as storing videos generated by sentries, and there are a multitude of services available. While it is possible to deploy the web applications and host data through one's own on-premise servers, it is more logical to deploy through a cloud service provider, due to reliability and ease of use. For the project to scale, there must be reliable storage available to host a potential high volume of video. If we assume an average of 6 incidents a day, considering the average crop growing season to be 120 days with each incident snapshot video to be around 40MB, then for a userbase of 1000 we would need 6*120*40*1000 MB or 28.8TB of cloud storage a year (ignoring data pruning). Scaling for a userbase of 10000 users we would need 288.8TB of storage a year.

One popular cloud service provider is Amazon's AWS, which provides a web service called Elastic Compute Cloud or Amazon EC2 for deployment of web applications, as well as a storage service called Amazon S3. An alternative to AWS as a cloud service provider is Microsoft's Azure, which provides cloud computing/hosting for web applications in the form of Azure virtual machines. Azure also provides object storage in the form of Azure Blob.

As storage is the key criteria, emphasis is placed on comparing storage costs of the two providers - a pricing comparison of the two providers can be found in the tables below.

Table 1. Cost comparison of AWS and Azure cloud storage services

Monthly Storage costs [per GB]	First 50 TB	Next 450 TB	Over 500 TB
AWS S3	$0.0230	$0.0220	$0.0210
Azure Storage-Blobs	$0.0208	$0.0200	$0.0192

For the application's desired range of 28.8 to 288.8 TB, Azure's storage costs are roughly 10% cheaper than AWS, making it more desirable to the project's storage needs.

3.4 Physical Design

The physical design of the system is divided in two major parts. The first being the embedded hardware system and the second being the mechanical subsystem. The hardware subsystem deals with inputs and houses the requisite software while the mechanical subsystem deals with the movement and delivery of the payload. The former is stored inside a rectangular container while the latter is stored atop. This is to keep the two systems separated, preventing interference between the two.

3.5 Embedded Hardware Subsystem

The hardware subsystem is the system that houses the embedded software subsystem and provides the connections the former requires to perform its tasks. These include the actual inputs to the product, the camera and IR sensors whose data will be used to detect the actual target. Also included is a Wi-fi module that allows for communication with the web application. Finally, there is the actual microcontroller, the Nvidia Jetson Nano which will control each of these various parts and perform the required operations.

3.5.1 Motor Control

The aiming of the firing apparatus is performed by a motor that will swivel the device to an appropriate position. Two types of motors are possible for this application- a servo and a stepper motor. A servo motor is able to retain a position because of the closed loop configuration of the system while a stepper motor only knows to increment or decrement by the amount inputted. The former is chosen because although stepper motors are very accurate, it lacks in speed and servo motors are able to retain a constant position.

This servo is controlled via pulse width modulation (PWM), with the input signals arriving from the control device, the Jetson Nano. Instead of having the servo connect directly to the computer, an intermediate PWM driver is used. This allows for easier implementation of additional components, such as the PWM controlled water pump. Each additional component will require some general-purpose input and output (GPIO) pins on the Jetson, so the driver saves on pin usage as it will be the only device using these pins. The driver connects to the cables of each device and will then connect to the appropriate pins of the Jetson. An example of this is shown in Figure 4, where 4 different motors are connected to the driver which would then connect to the computer. The chosen driver is an 8-channel PWM [5] and can be used to control the motors using the Adafruit "ServoKit" Python library [6] because motors cannot be directly connected to the Jetson Nano.

Mapping the pins to the driver and configuring the Jetson for PWM would normally be a complicated task, but Nvidia has provided provisions to simplify this task. A python library has been provided that simplifies many common tasks regarding GPIO. One such operation is PWM, so the usually difficult task of implementing PWM is made trivial.

```
import time
from adafruit_servokit import ServoKit

kit = ServoKit(channels=8)
kit.servo[0].angle = CALCULATED_ANGLE
```

Figure 8. Sample code for Adafruit "ServoKit" Python library

When a pest is detected, the coordinates of the location of the pest will be provided and an angle for the motor is determined. As the camera and the water gun are vertically aligned, the water gun is at the center of the camera's view. With this information and trigonometric identities, we can determine the angle the water gun is to be adjusted. Figure 3 shows an example of two position coordinates.

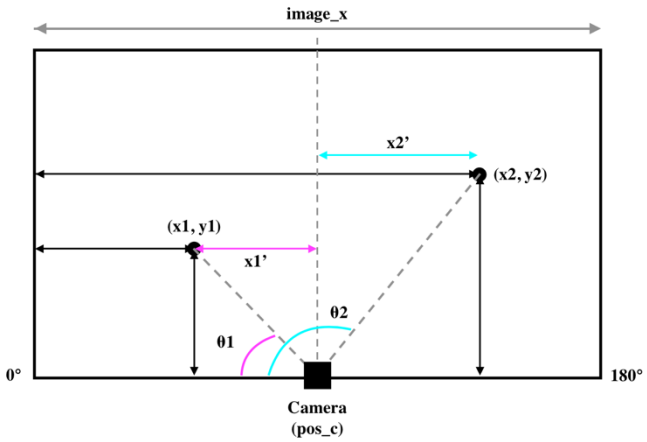

Figure 9. Example of two position points received

Coordinates can be divided into two groups: points left and right of the camera (pos_c). Points to the left (as depicted by (x1, y1) in pink) can simply use the tan inverse formula to determine the angle θ1 needed for the camera. Points to the right of the camera (such as (x2, y2) in blue) require one extra step of calculating the supplementary angle.

After acquiring the angle needed, the servo motor still needs the duty cycle to understand where to angle itself. Using the datasheet for the servo motor, Figure 4 shows the duty cycles needed for 0 degrees is 0.5ms, a 1.5ms duty cycle to represent 90 degrees, and a 2.5ms duty cycle that represents 180 degrees. This information can be used to determine the pulse width of any angle between 0 and 180 degrees, shown in Equation 1. With a PWM period of 20ms, the duty cycle supplied to the servo motor is the percentage of pulse width to PWM period.

Equation 1. Pulse width equation and duty cycle equation

$$Pulse\ Width = \frac{1}{90}\theta + 0.5$$

$$Duty\ Cycle = \frac{Pulse\ Width}{20ms} * 100\%$$

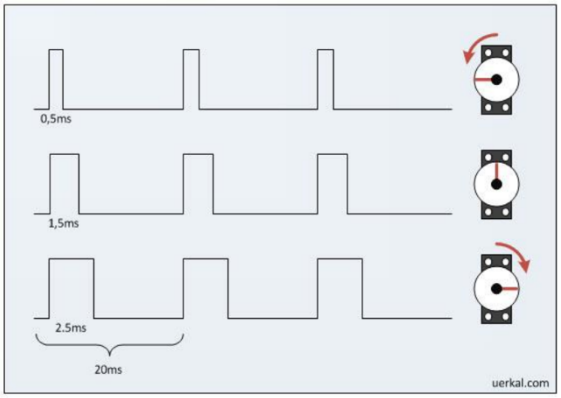

Figure *10.* Duty cycle vs. angle *[5]*

3.5.2 Camera and Sensor

The primary inputs to this system are a camera and an infrared sensor. The camera selected is the RaspPi NoIR V2 [7]. This camera meets the requirements highlighted in the Bunny Detection Subsystem, and easily connects to the Jetson. The Jetson Nano has a camera serial interface (CSI) that is compatible with this camera and this port is specifically designed to interface with cameras. Other camera modules considered either were too high in megapixels or drew too much power, both which were unnecessary for the project specification.

The sensor's only goal is to detect the presence of a potential pest at night. They signal to the system that the flood lights to be turned on, so that the camera may can see the target. As such, the accuracy of the sensor is not too important and further investment in a higher quality sensor would be a waste of resources.

3.6 Mechanical Subsystem

The mechanical subsystem is comprised of several functional and non-functional components, some of which operate together. A water pump is used to create a high-pressure chamber which can be used to shoot water at high velocities towards a target. The water is piped to a flow control system which will adjust the flow rate of the water, this can be realized using a variable control valve. After the valve the

water is sent to a nozzle which controls the size and shape of the opening which will affect the final projectile velocity and accuracy. Each of these components have a multitude of options, each with their own specifications which must be accounted for.

Using basic kinematics equations, the output velocity of the water gun can be found for a given distance goal. Using the specification distance of 5m and a launch angle of 0 degrees the following time and exit velocity can be found.

Equation 2 Exit velocity of water calculation

$$\mathrm{d} = \mathrm{v_i t} + \frac{1}{2}\mathrm{a t^2}$$

$$0.5 = 0 + \frac{1}{2}(9.8)\mathrm{t^2}$$

$$t = 0.32s$$

$$5 = v_i(0.32) + \frac{1}{2}(0)(0.32)^2$$

$$v_i = 15m/s$$

These calculations are only rough estimates as they do not account for air resistance nor do they factor a launch angle. The air resistance will result in a projectile that travels closer than the desired amount and may cause the water to defuse into a wider stream. The lack of launch angle also reduces the overall distance, but it will allow for compensation if the distance is deemed too short. The exit angle can be adjusted if the design requirements change or to tune the system to the required specifications.

There are three main configurations which can be used for this project with varying advantages and disadvantages. The first configuration is driven by a standalone pump such as an outdoor pump for pools, or a computer pump used for liquid cooling. The second choice is a standard garden hose typically already located near the garden. The final option is a toy water gun which includes all the components needed such as the pump, nozzle and actuator. These options will be evaluated on the merits of flow rate, price, ease of use and ease of implementation.

One of the main advantages of the standalone water pump is the flow rate it can produce. There are a wide variety of pumps available that range from 10 gallons per hour anywhere to 5000 gallons per hour. These pumps will be able to meet our design requirements no matter what distance is chosen if the flow rate of the pump is high enough. The cost of these pumps varies depending on the flow rate but can range anywhere from 25$ to as much as 250$ for the strongest ones. This is much more expensive than the other

pumps, but it is still within the project budget. Another advantage for this option is the portability of pump. Since the pump is only connected to the water gun, the entire system can be moved easily. This is especially useful for when the system must be demonstrated since it can be used anywhere. The product would be much easier to use since it would require no connections to an existing water line. However, this would also add the need of a water reservoir which would need to be periodically filled.

The second option is to use a standard water hose to drive the water gun with the use of a controllable valve. Standard North American garden hoses operate at approximately 24 gallons per minute with a PSI of 40-60 [8]. These values are depending on the length of the hose along with its material and curvature. This would make the system very difficult to tune and adjust since the hose will never be in the exact orientation every time it is attached. The testing would therefore be difficult to perform, and the result would be inconsistent. Many home gardeners also regularly use their hose to water their plants along with driving sprinkler systems. This means that they may not be able to use the garden sentry if their garden already utilizes a hose powered device. The major advantage of this option is the price, since the hose is free the only cost would be a control valve and the nozzle.

The third option is to use a pre-built motorized water gun which includes a pump, reservoir and nozzle. The only additional mechanism needed would be some wire leads to electronically fire the gun. The cost of this toy is approximately 40$ which makes it the cheapest option since no other parts are needed. This also means that the system is easy to implement since all the parts are already assembled and working. The water gun is the weakest of the options in terms of power since it's not powered by a conventional water pump, instead it's driven by a motor which compresses and decompresses a piston filled with water. This ultimately results in a low-pressure and low flow rate gun, which in turn results in a low distance projectile. Due to the nature of this gun being a toy, the reservoir has a limited capacity meaning it would need to be refilled often.

The following weighted decision matrix will evaluate the 3 previously stated designs by comparing their flow rates, prices, ease of use and ease of implementation. Flow rate will have the highest rate of 0.5 since the specification goal of 5m must be met for this project. Ease of implementation will have the next highest weight of 0.3 since this project must be completed before winter due to its nature. Finally, both price and ease of use will be assigned weights of 0.1 since these are non-vital additions which are nice to have. Each option will score the criteria from 0-10 with 0 being the worst score and 10 being the best score.

Table *2*. Configuration decision matrix

		Options		
Criteria	Weight	Water Pump	Hose	Toy Gun
Flow Rate/ Power	0.5	9	8	2
Ease of Implementation	0.3	3	3	9
Ease of Use	0.1	8	5	5
Price	0.1	2	5	8
	Total	6.4	5.9	5

Therefore, it can be shown that the water pump is the best design choice for this project as evaluated based on flow rate, ease of implementation, ease of use and price.

3.6.1 Water Pump

There are many factors that must be considered when choosing an appropriate water pump to drive the water gun. There must be enough water flow in order to keep up with the gun output when firing. We will assume that the nozzle has a diameter of 1mm and the speed of the water is assumed to be 30m/s for a distance goal of 10m. With this the flow rate of the system can be found:

Equation 3. Flow rate out of nozzle calculation

$$flowrate = vA = (15m/s)\left(\pi(0.5mm)\right)^2 = 0.0118L/s = 42.4L/h = 11.2gal/h$$

There are 3 main types of water pumps which will be considered for this project, an outdoor pump such as a pool pump, a voltage-controlled PC cooling pump and a PWM controlled PC cooling pump. Both cooling pumps offer a similar range of flow rates while the pool pumps typically offer a higher flow rate comparatively. However, due to the low flow rate needed any of these pumps will work since even the lowest rated pumps are strong enough to meet our requirements. The length of the pipes for the interconnections are minimal therefore the pressure loss in the system is negligible, meaning the operational pressure of the pump is irrelevant.

The prices for both PC pumps are within the range of 100$ while the outdoor pumps range from 30-200 depending on the flow rate. Since only a minimal amount of flow is required the 30$ pump is adequate. In order to control the flow rate from the outdoor pump a variable actuated valve is needed to adjust the pressure. This part is very expensive since it's used for industrial applications. The usage of valves also adds complexity to the system since the head pressure of the pump would matter and therefore the pump would need to be sized higher than needed in-order to support the valve. For these reasons the PC pumps will be used to drive the water gun.

Finally, there is a choice between using either the PWM controlled pump or the voltage-controlled pump. The voltage-controlled pump would need an analog signal with enough current to drive the pump. This is hard to accomplish on a Jetson Nano board since it does not have any analog voltage pins with a high enough current. To use this type of pump an additional controller would be needed to drive the pump. In comparison, the PWM controlled PC pump can be driven by the same driver that the servo uses. The pump would only need to be supplied by an additional external power source. Since the servo also relies on PWM controls the implementation and operation should be similar making it easy to set up. Therefore, the PWM controlled PC pump is the best pump for driving the water gun.

3.6.3 Nozzle

The purpose of the nozzle is to direct the water into a desired shape and size for the purpose of increased velocity, accuracy and spread. There are typically 2 shapes found for a hose nozzle, a solid stream and a cone. The stream gives higher accuracy since all the water is directed towards the middle of the nozzle. The cone shape gives lower accuracy since less water is being directed towards the target. However, this has an inverse effect on the spread of the projectile as the cone shape allows the water to hit more area and thus has a higher chance of hitting the target. Both advantages and disadvantages must be weighed when chasing a nozzle type for the water gun in order to maximize the chance of hitting the target. Choosing one type on nozzle will limit the design and may cause problems when implementing the system. Therefore, a hybrid nozzle type will be chosen in order to utilize both benefits of the stream and cone shapes. A hybrid nozzle can gradually change the shape of the water from a stream to a cone so that the optimal shape can be found for our system.

3.6.2 Water Reservoir

There are two main decisions that must be considered for the water reservoir, the size and the placement. The size of the reservoir should be large enough to only need filling every week but small enough to not be a space concern. From section 3.5.1 it was found that the flow rate from the output of the 1mm diameter nozzle with a velocity of 15m/s is 0.0118 L/s. If each shot takes 2 seconds and the expected amount of rabbits is 5/day then the weekly water usage would be:

Equation 4. Reservoir sizing estimation

$$0.0118 L/s * 2s * 5 * 7 = 0.83L$$

Therefore, a 1L reservoir is enough for this system. But, a container this small may be hard to find so any reservoir larger than 1L is appropriate for the water gun.

There are two main options for the placement of the reservoir, either on the ground or at the same height of the gun. If the tank is placed on the ground, then the pump must use some of its energy to pull the

water up from the tank to the gun. This would cause the gun to have a lower pressure drop and thus a lower exit velocity. If the reservoir was placed in line with the pump, then no additional work must be used to pull the water. The reservoir will most likely be placed somewhere in-between in order to control the final velocity and distance of the water and may act as another tuning parameter for the system.

3.6.3 Enclosure

The purpose of the enclosure is to hold the electronic components in one area and to protect them from potential damage. Most notably, the enclosure must be waterproof as this entire project involves shooting water which could damage the entire system. Thus, the Jetson and its various peripherals must be kept inside the container. A model is shown below in Figure 8, with simplified versions of the required connections and components for the design. The camera and sensors are embedded into the front of the enclosure, allowing for them to see outside of the box. These two are also covered by a shade, so that no water will fall from the gun, into the lens.

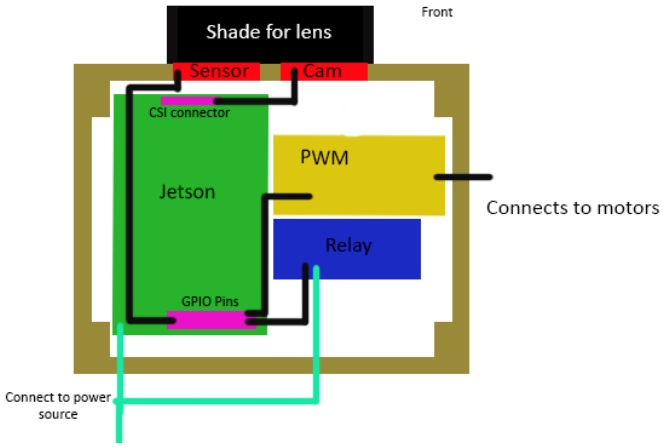

Figure *11*. Basic overview of inside of enclosure

With these components in a confined environment, the heat produced from these components must be considered. Initially, a fan was considered to expulse the hot air from the enclosure. This idea was later abandoned as the space constraints of the enclosure does not allow for the inclusion of a fan. Instead, it was decided to simply implement a series of air holes at the bottom of the enclosure. This solution presents multiple benefits to our system with very little cost. The first being simplicity, adding holes to the bottom is a relatively easy task and does not increase the complexity of the overall design. Additionally, these holes do not take up additional space in the enclosure and are still resistant to water entry.

4 Discussion and Project Timeline

4.1 Evaluation of Final Design

The objective of the project is to design an embedded system that will deter pests from entering gardens. Theoretically, the design proposed for each subsystem will combine to create a functioning system that will deter pests. The design includes detection methods using computer vision and infrared sensors, motor controls to aim the water gun, a water pump system to spray water towards the pests and finally a web server to allow users to view instances of entry. The water gun in theory will scare pests away when they are either hit by the water, feel the spray of the water near it or hear the water gun being fired. The computer vision system will theoretically detect pests entering the garden. The motor controls in combination with the computer vision system will theoretically allow the system to cover a wide range of the garden. Finally, the web server will fulfill our specification of allowing users to view instances of pests entering the garden. Overall, the design meets all of the specifications in theory.

4.2 Use of Advanced Knowledge

The design of the motor control system utilizes knowledge from ECE 380 Control Systems. It uses pulse width modulation to control the angle the servo moves to. The design of the web server uses knowledge from ECE 356 Database Systems to model the database schemas, which will store the previous recordings of rabbits entering the garden. Furthermore, the usage of the Wi-Fi card on the Garden Sentry requires knowledge of networks from ECE 358 Computer Networks. For this design, the Wi-Fi card merely needs connection to the internet to send information to the web server.

4.3 Creativity, Novelty, Elegance

This project is novel because it uses a computer vision module to detect pests, as well as a web server to allow users to see the results of pest deterrence. Traditional pest deterrents merely use an infrared sensor and a simple water spray, this has a limited range and can be easily avoided by rabbits. Furthermore, the instances of pests entering the garden are not recorded in any manner by traditional pest deterrents. Furthermore, by using computer vision as well as an aimable water gun, the Garden Sentry can cover more of the garden than traditional pest deterrents as well as allow users to adjust the positioning of the Garden Sentry to ensure no more pests enter the garden. This idea is novel due to its rarity and usage of newer technology which is only now entering the mainstream of technological projects.

While the ideas used are complex, the elegance of the design is the modularity, which allows for each separate subsystem to exist on its own and then be combined into one system to meet the specifications of the project. The computer vision system can easily be extended to detect more than just rabbits by training

it to detect more objects. This extensibility and modularity allows for more features to easily be added to the Garden Sentry should we choose to do so.

4.4 Student Hours

Table 3 below displays the number of hours worked on the project so far. While the number is not quite the suggested 10 hours per week, this is due to some of the parts still not available for testing.

Table 3. Student Hours

Group Member	# of Hours
Ryan Tjhie	49
Lucas Fayoux	48
Monta Gao	47
Jonathan Xue	42
Sharon Zheng	44

4.5 Potential Safety Hazards

Some potential safety hazards are short circuits if water comes into contact with the circuitry on the embedded system. Since soldering is being done for the wiring, there is a possibility of burns if performed carelessly. Other safety hazards involve errors with wiring and care will need to be taken when wiring the individual pieces together. The water pump also cannot be run dry and if there is no water in the reservoir then the pump can be damaged. The enclosure involves cutting the plastic with a knife or drills, and safety measures will need to be taken such that no one is injured.

4.6 Project Timeline

In Figure 12 and Figure 13 below, the predicted timelines for both ECE 498A and ECE 498B are shown. Each submodule can be completed in parallel and are expected to take at least a month of work each. At the end of July and into the first weeks of August, the submodules are to be assembled, which is expected to take around 2 weeks. For the prototype demo by July 26th, it is expected that each submodule is in a mostly finished state and is ready to be demonstrated to our consultant.

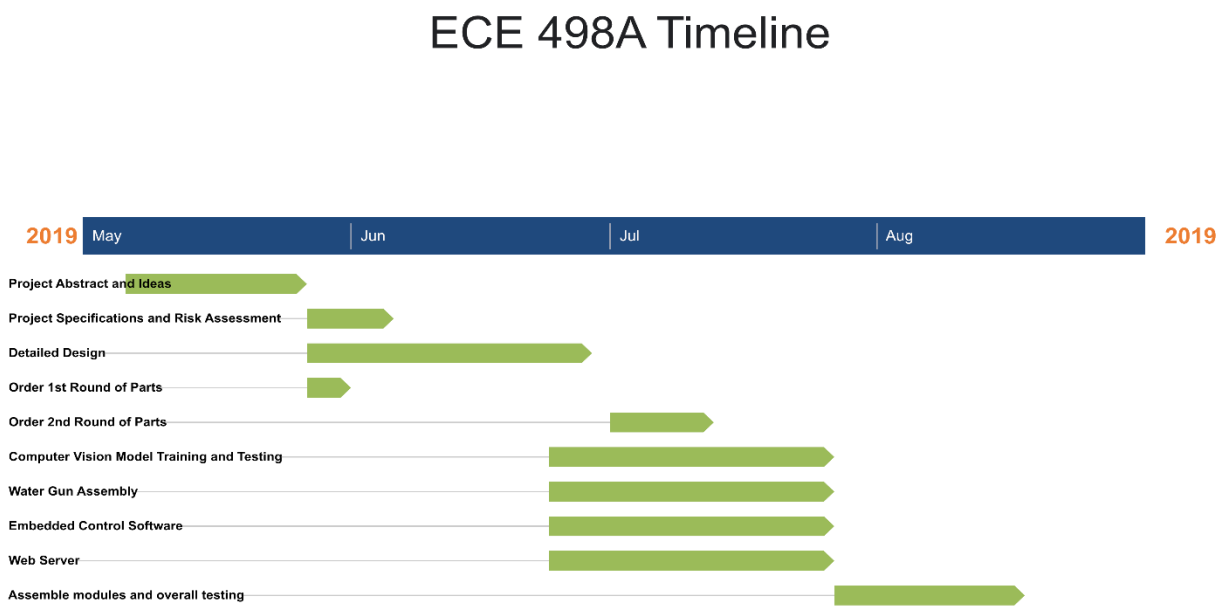

Figure 12. ECE 498A Timeline

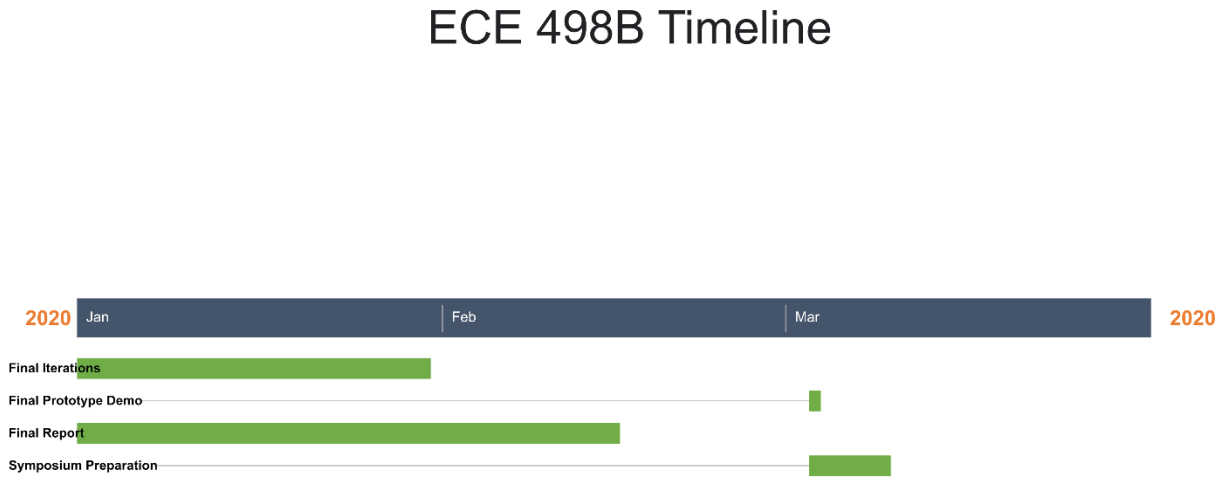

Figure 13. ECE 498B Timeline

References

[1] F. P. Carvalho, "Pesticides, environment, and food safety," *Food and Energy Security,* vol. 6, no. 2, pp. 48-60, 2017.

[2] C. A. Damalas and I. G. Eleftherohorinos, "Pesticide Exposure, Safety Issues, and Risk Assessment Indicators," *Environmental Research and Public Health,* vol. 8, no. 5, pp. 1402-1419, 2011.

[3] NVIDIA, "Jetson Nano Brings AI Computing to Everyone," 18 3 19. [Online]. Available: https://devblogs.nvidia.com/jetson-nano-ai-computing/. [Accessed 23 6 2019].

[4] pjreddie, "YOLO: Real-Time Object Detection," [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed 16 6 2019].

[5] adafruit, "8-Channel PWM or Servo FeatherWing Add-on For All Feather Boards," [Online]. Available: https://www.adafruit.com/product/2928. [Accessed 17 6 2019].

[6] adafruit, "adafruit_servokit - Adafruit ServoKit Library 1.0 documentation," [Online]. Available: https://circuitpython.readthedocs.io/projects/servokit/en/latest/api.html. [Accessed 17 6 2019].

[7] adafruit, "Raspberry Pi NoIR Camera Board v2 - 8 Megapixels," [Online]. Available: https://www.adafruit.com/product/3100. [Accessed 10 6 2019].

[8] R. d. Jauregui, "What Is the Flow Rate of a Garden Hose?," SFGate, 28 November 2018. [Online]. Available: https://homeguides.sfgate.com/flow-rate-garden-hose-82928.html. [Accessed 25 June 2019].