Eye of Mordor
Sparktech

June 24, 2022

1 Team members

- Miguel Rodriguez

- Shao-Wen Chang

- Asadujaman Nur

- Vincent Obigwe

2 Introduction

The population of human civilization is increasing rapidly. With this growing population, we need many lands for farming, resident and industrial purposes. Thus, we have been chopping down the forests to support this huge demand for the lands. Although it is solving many problems for us, it is creating many issues by endangering the animal life and forcing them to relocate their natural habitats, which leads them to move closer and closer to people and our habitat. Which causes unexpected fatal accidents.

The problem we are facing is, due to lack of their natural habitats, they, for example deer, are living near firms and woods. In addition, sometime they like to hide or take shelter from people if they get scared or are if they are injured. Sometime they sleep and rest inside those lands. Which can be a huge problem. Since they are deep inside the firming lands, the firmer and the animal rescuers do not know their stances; therefore they cannot provide

support or perform rescue operation in case they are in need of support. Even worst, during the harvesting season, if they are injured(Animals) and decides to have shelter inside the fields such as corn ones and cannot move due to their injuries or too weak, and farmers starts to harvest the corps, sometimes they just run over their tractor on the poor animals without knowing their presence and killing them as a result. This can be avoid by the help of technology.

To solve this issue and save the wildlife, we are going to develop a system, which will detect the animals hiding inside the fields and transmit the data to a receiver to either the farmer or the animal rescuer. So that they can take necessary action to save their life. Here in this project we are going to develop a prototype model using sensors and micro controllers to approach this problem. We will discuss more about the components we will use, methods and implementation in details below.
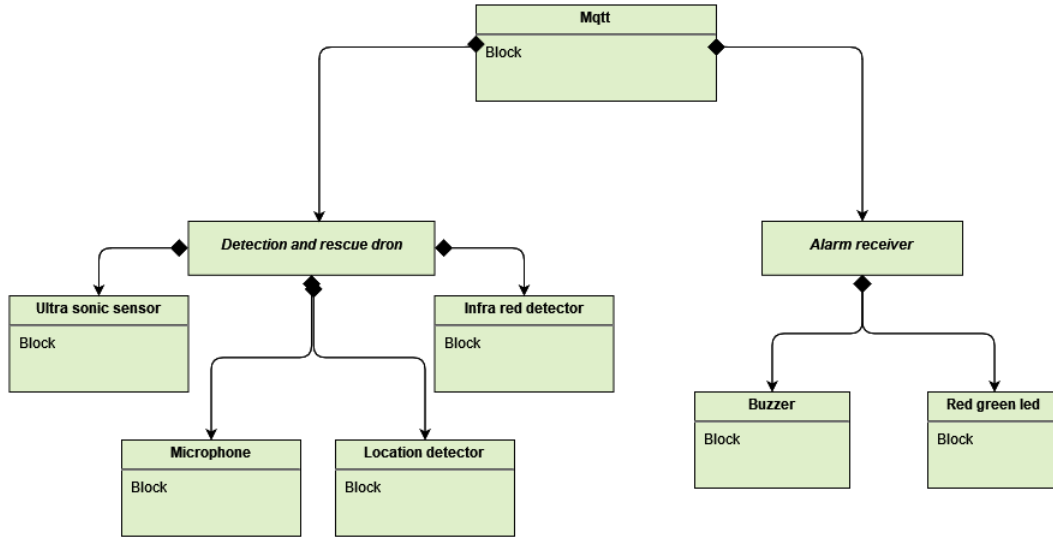
3   Concept Description

Figure 1: Block Diagram

The project has three main components, a Raspberry pi 3B that works as the broker, an Arduino Uno Wifi Rev 2, that acts as the transmitter and

the receiver, and a mobile phone that is able to receive the data produced by the broker, and also return an acknowledgement command. Figure 1.

The the devices, the Raspberry pi, the Arduino Uno and the mobile are connected to the same LAN network, in some cases we can also use the mobile phone as a hotspot to interconnect the other two devices. Fig 2.
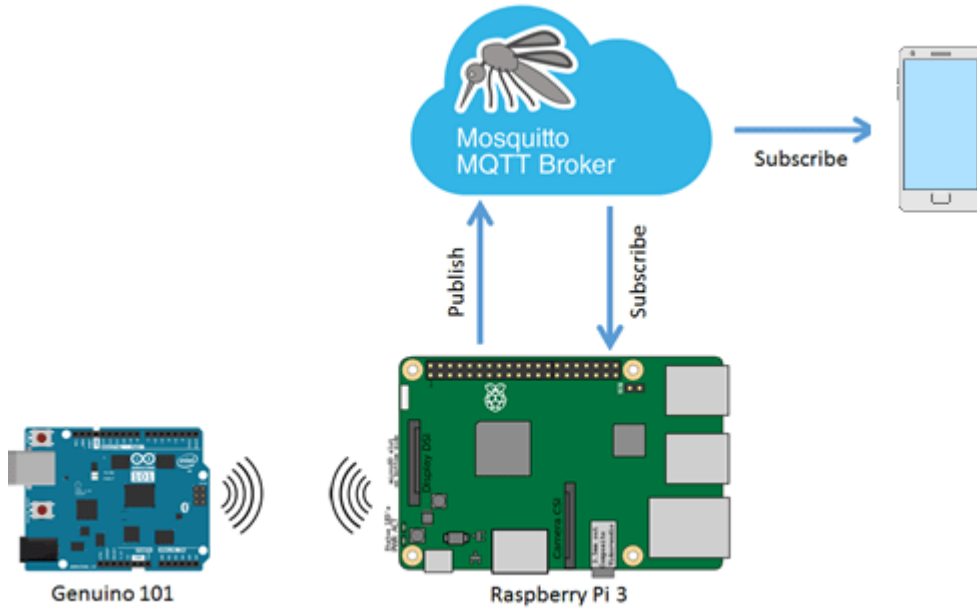

Figure 2: Broker connection through LAN

On the Raspberry Pi 3 we installed Mosquitto, the broker that will receive the information proceeding from the sensors, analyse this data and return the calculated results to both the mobile phone and the Arduino Uno, that depending on the received data will behave in different ways that will be deeply explained in section 6.

The Arduino Uno is equipped with an ultrasonic sensor, that will act as the detector for animals. For the purposes of the simulation was not possible to use all the desired sensor. The Arduino is also connected with five leds: while to let us visualize the Wifi connection, the white led will be static on when the Wifi connection is not working and it will blink when the connection is established. The blue led will keep on when the broker is disconnected, and it will turn of when the broker is connected. The green led is on when

3

the detections are far from the sensor or there is not a detection, in the case of our simulation more than 12 cm, the yellow led will turn on when the detection is between 6 and 12 cm, and when the detection is smaller than 6 cm both the red led and a buzzer to simulate an alarm system. The alarm then can be silenced with the acknowledge button on the mobile app. The schematics of the connection Arduino can be seen on Figure 3
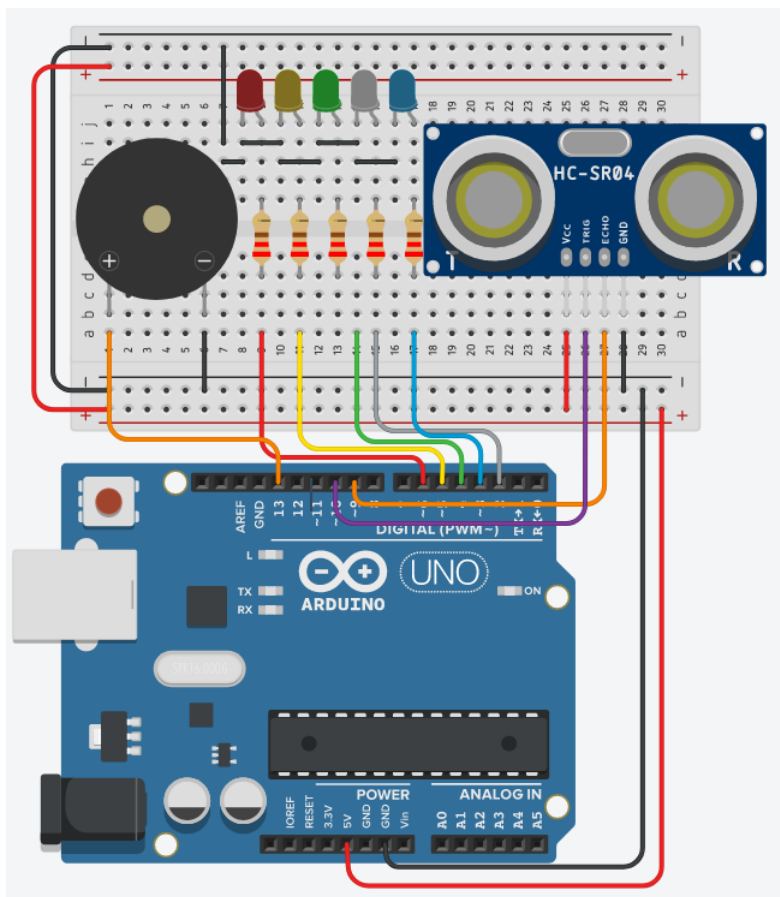

Figure 3: Schematics of Arduino connections

4   Project / Team management

In carrying out our project, we consider project management as an important aspect that helps us in realizing the goal and objective faster. Project

management benign the application of various methods, skill knowledge and experience to achieve a detailed object that has already been set and marked as the project acceptance benchmark. It is also very important that time and budget are considered. As students we embarked on this project with the focus on managing time and delivering the best result given the limited amount of time that and the no budget we deal with. Having this in mind, we decided to approach the whole project using the agile project management.

Agile project management is known as an iterative means of delivering a project throughout a given life cycle. There life cycle are made up of several iterations that are all geared towards the completion of the project. This leads to us having weekly meetings to analyze tasks and evaluate progress. With Agile project management, continuous improvement and development is the goal, to enable the project to get better on further iterations. While using Agile project management, we made use of Kanban framework to be able to achieve our project.

Kanban is a framework on agile project management that is deals with growth changes and the need for continuous process improvement. The core practices involved in using Kanban are:

1. Visualization of tasks in a board like manner with tools such as excel.
2. Reduction of work in project by introducing changes incrementally.
3. Management of the flow of to-dos.
4. Defining processes and tasks clearly.
5. Enhancing the process of feedback to enable improvement of the system.
6. Improving work flow all together.

To be able to properly visualize our tasks and clearly see what each member of the group has to do, we made use of GitHub projects. GitHub projects can be seen as a customizable spreadsheet that helps us integrate tasks with GitHub in our repository. It empowers more customization by enabling filtering; sorting, grouping and working with GitHUb pull requests. It further looks static with the addition of colors to determine the various stage each task is.

5 Technologies

In this section, we explained the technologies used in our project. We split the technology into two namely hardware and software. The reason for the

split is due to the fact that our project contains both software and hardware components that are linked to work together. Fig 4
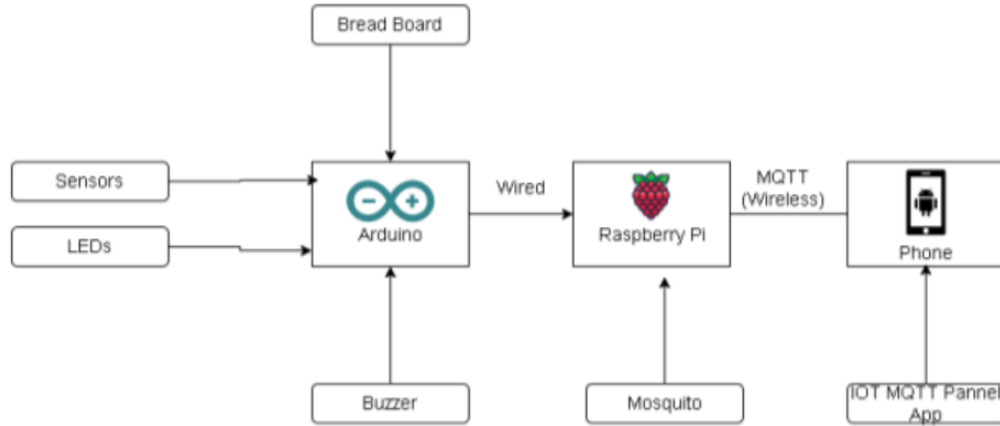

Figure 4: System Architecture

As can be seen by the diagram above which explains the basic connection between the sensors, led, Arduino raspberry pi, phones, etc. This will further be explained in the forthcoming subsections.

5.1   Hardware

In carrying out our project, a wide range of hardware was used in accomplishing the task. The hardware components used include but are not limited to:

5.1.1   Sensors

We made use of a wide range of sensors such as ultrasonic sensors that enables us to detect when a farm instrument is approaching the animal.

5.1.2   Buzzer

The buzzer acts as an actuator that informs the farmer of an animal's presence. The buzzer produces a sound that acts as a signal.

5.1.3 Breadboard

A breadboard is a tool/device that is used to seamlessly connect wires and other supporting electronic components while prototyping [5]. Since the Eye of Mandor is a prototype, made use of breadboard to connect the sensors, resistors, etc with the Arduino.

5.1.4 Led

In our project, LED is used to signify the position of the tractor or farming device to an animal or obstruction. The light turns GREEN when there is no obstruction or the tractor is nowhere close. YELLOW when there is a construction, but the farming instrument is still in far proximity to the animal. The light turns RED when it is now in close proximity to the animal.

The LED serves as an additional precaution with the buzzer to offer multiple layers of protection for the animal.

5.1.5 Arduino

Arduino is an open-source platform that is commonly used in the process of building and programming electronics. It has the capability of sending and receiving information from devices [6].

In our project, Arduino was used to connecting all the sensors, buzzers,s and the rest of the electric component. It transmits the data to the Raspberry Pi through a wired connection.

5.1.6 Raspberry Pi

Raspberry Pi is a low-cost mini-computer that was developed by the Raspberry Pi Foundation UK. It can serve as a computer CPU and perform various computer functions [7]. In our project, Raspberry Pi was used for a wide range of tasks which includes installing a mosquito for a connection with the mobile phone through MQTT and getting data from the Arduino.

5.1.7 Phone

The mobile phone was used to establish a connection with the Rasberry Pi wirelessly using MQTT. This is done through software.

5.2 Software

There are also software components of our application. The software includes but is not limited to:

5.2.1 Mosquitto

Mosquitto is a popular opensource MQTT broker that permits the process of connecting to sensors, devices, and apps in publish and subscribe architectures [8]. In our project, a mosquito was used as the MQTT broker. It was installed on the raspberry pi.

5.2.2 MQTT Protocol

Message Queuing Telemetry Transport (MQTT) is an open publish and subscribe protocol that is designed for constrained devices and is used for telemetry purposes [5]. In our project, MQTT was used with the help of mosquitoes to connect to the mobile app in a subscribe and publish manner.

5.2.3 IoT MQTT Panel App

This is an android application that enables us to subscribe to and publish topics through the MQTT protocol. This app communicated with the Rasberry pi with the help of mosquito which is installed in the raspberry pi.

6 Implementation

There were three main components to implement Eye of Mordor, the Arduino, the Raspberry Pi and the MQTT mobile app.

6.1 Raspberry Pi

To realize the connection between client and server, the first step was to install Mosquitto broker in the Raspberry Pi. By default. A broker program written in C++ runs on the Raspberry pi to decode information under the MQTT protocol. The c broker on the Raspberry Pi has two functions *on_connect* and *on_message*. The *on_message* function handles all the incoming messages from the Arduino, it makes the computation and then return

the value of the LED which has to be turned on or off as well when the buzzer buzzes. Figure 2

```cpp
void on_message(struct mosquitto* mosq, void* obj, const struct mosquitto_message* msg)
{
    char* load = (char*) msg->payload;
    std::string dist_st(load);
    int distance = 0;

    try
    {
        distance = std::stoi(dist_st);
        std::cout << "Distance " << msg->topic << ": ";
        std::cout << distance << std::endl;
    }
    catch(const std::exception& e)
    {
        std::cerr << e.what() << std::endl;
    }
    if (distance > 12)
    {
        mosquitto_publish(mosq, NULL, TOPIC_O, 1, "0", 0, false);
    }

    if (distance > 6 && distance < 12)
    {

        mosquitto_publish(mosq, NULL, TOPIC_O, 1, "1", 0, false);
    }
    else if (distance <= 6)
    {
        mosquitto_publish(mosq, NULL, TOPIC_O, 1, "2", 0, false);
    }
    std::string ds_o = std::to_string(distance);
    const char* msg_o_c = ds_o.c_str();
    int lenght = ds_o.length();
    mosquitto_publish(mosq, NULL, TOPIC_D, lenght, msg_o_c, 0, false);
}
```
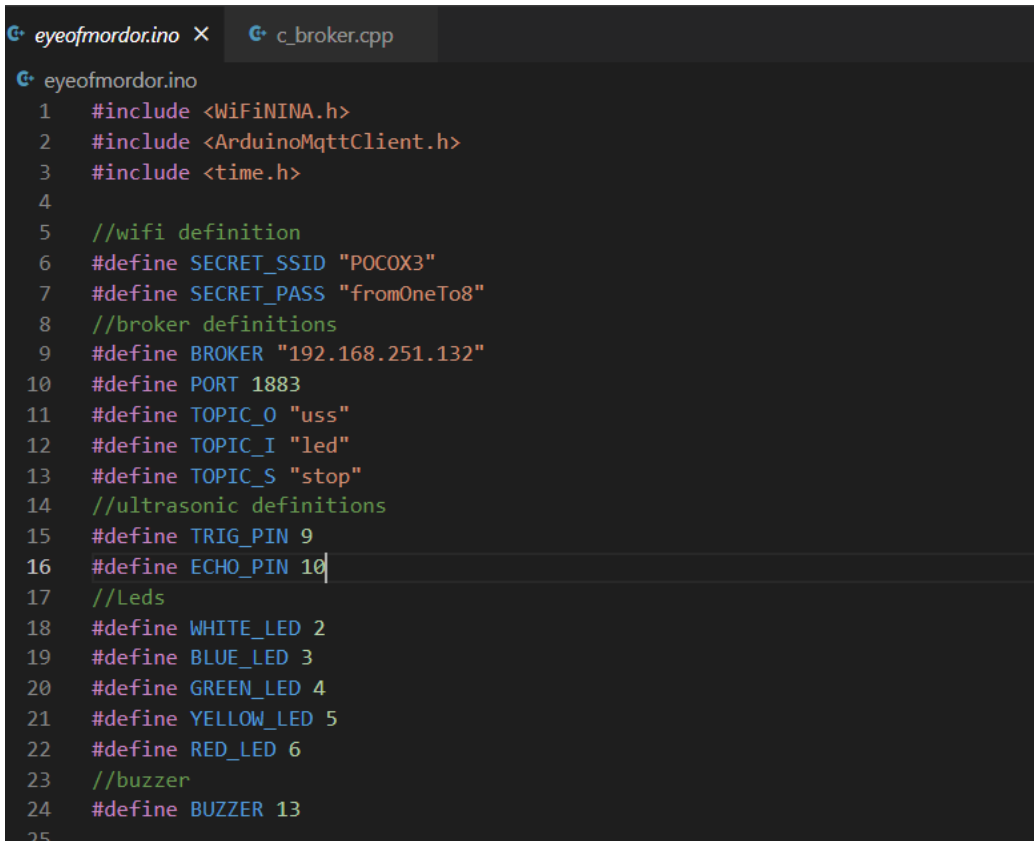
Figure 5: C++ broker

6.2 Arduino

The implementation in the Arduino starts with the connection to the the Internet, this connection was done with the use of the WiFiNINA library,

we declare the IP address of the broker and the port that we will use. The IP address can be taken from the command line on the Raspberry Pi with the command *hostname -I*, and the port is defined as *1883* that is the port defined by the Mosquitto broker. We also define the names of the topics that are used in the implementation: *uss, led* and *stop*. The *uss* topic will send the information of the ultra sonic sensor, the *led* topic will receive the information of which led has to be on and the *stop* topic is used to stop the buzzer using the mobile phone. Moreover, the ports of the Arduino are setted as shown in Figure 6.

```
eyeofmordor.ino ×      c_broker.cpp

eyeofmordor.ino
  1   #include <WiFiNINA.h>
  2   #include <ArduinoMqttClient.h>
  3   #include <time.h>
  4
  5   //wifi definition
  6   #define SECRET_SSID "POCOX3"
  7   #define SECRET_PASS "fromOneTo8"
  8   //broker definitions
  9   #define BROKER "192.168.251.132"
 10   #define PORT 1883
 11   #define TOPIC_O "uss"
 12   #define TOPIC_I "led"
 13   #define TOPIC_S "stop"
 14   //ultrasonic definitions
 15   #define TRIG_PIN 9
 16   #define ECHO_PIN 10
 17   //Leds
 18   #define WHITE_LED 2
 19   #define BLUE_LED 3
 20   #define GREEN_LED 4
 21   #define YELLOW_LED 5
 22   #define RED_LED 6
 23   //buzzer
 24   #define BUZZER 13
 25
```

Figure 6: Arduino configuration

We define three states that will represent the distance of the detection, the first state for detections farther than 12 cm, the second for detections between 6 and 12 cm and the third one for detections closer than 6 cm. The

initial state is state 1 as we supposed that at the first moment there will not near detection Fig 7.

```
25
26    enum prox
27    {
28        st00 = 0,
29        st01 = 1,
30        st02 = 2
31    };
32
```

Figure 7: Arduino states

The Arduino code is based in six functions *wifisetup(), onMqttMessage(int messageSize), ultrasonic_pub(), reconect(), setup()* and *loop()*. The *wifiesetup* function is in charge to connect with the wifi connection as well to connect with the broker and to subscribe to the different topics, the *ultrasonic_pub* function is in charge of publish the information of the ultra sonic sensor through the *uss* topic. The *reconnect* function will be called in case that the broker or the wifi connection fail and will reconnect the system. the *onMqttMessage(int messageSize)* function get the information coming from the different topics and process it, this function, depending on the received message will change the status of the system, turn on or turn of the different leds and turn on or turn off the buzzer depending on the communication with the Raspberry pi and the mobile app. The *setup* function is a basic Arduino function where we set up our board ant the *loop* function is also a basic Arduino function where we run an infinite loop, this function will call in an infinite loop the publisher and the subscriber. It also checks if the connection is still active, in case is not active will call the *reconnect* function.

6.3   MQTT Mobile app

The MQTT mobile app is configured with the IP address, and the port with the same data used in the Arduino (Section 6.2). The mobile app subscribes only to the topic *distance* to get the distance of the detections, and publishes in the topic *stop* to stop the buzzer when beeping when the detection is still close to the sensor.
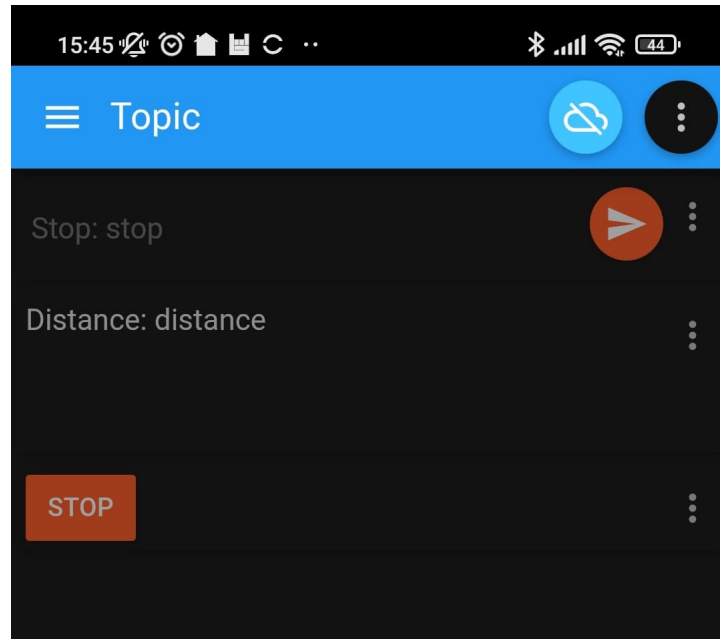
Figure 8: Mobile app

7 Use Cases

From the previous sections, we got to learn all the technical method and the technologies that being used to realize the project. Here in this section we will talk more about how the project can be realize in real life scenarios.

As discussed in "Introduction and Concept" Sections, We are trying to Develop a system that can be used to save and rescue animal life hiding inside field from unwanted accident primarily. Which can

be extended later on to help and assist in resque operation in search of lost human and animal in the wilderness aswell as the injured animal from the poacher. The primary idea is to place necessary sensors on a drone, then use the drone for search. However just like discussed before we developed a working prototype.

The breadboard represents the Drone´s place holder for the sensor . Since it is hosting the sensors together. Arduino and Raspberry Pi is acting as a motherboard where it process the data that being generated from sensor and send it to its base. To capture the presence and track the movement of animals hiding inside the field we are using Infra-red sensor, which is shown
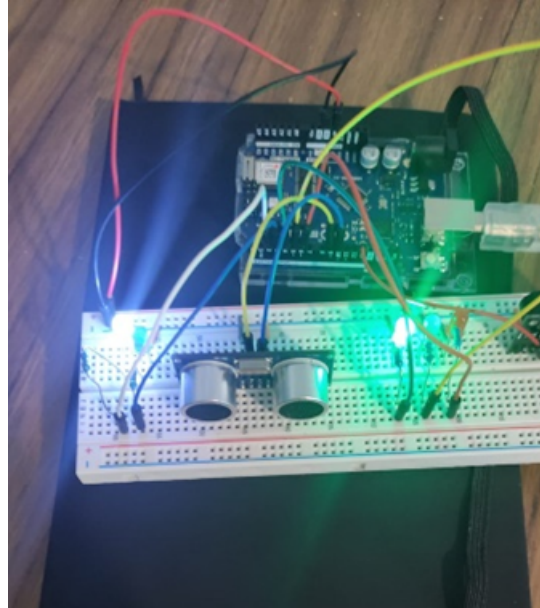
in Figure 9.

Figure 9: Eye of Mordor

To represent the animal we are using the card board also can be shown in the picture. As we can see in the Figure 10 (a) bellow, if we come near to the sensors it starts to read the cards(animals) presence and turn on the yellow LED and send the approximate distance to the hosting device, Figure 10

When it goes real close to the animal it turn on the red LED to notify user and send the distance info to the user so the user can know how close they are from the animal. This demonstration can be followed by the Figure 10.

Once it gather the data it can forward it to its operator as shown in the Figure 10. to ensure that there is an animal. Thus the operator can take necessary actions like whether to rescue, relocate or inform nearby farmer. The possibility is limitless.

Alternatively we can place A device like an "Alarm" or "Notification system on Smart phone" to alert the farmer that there is an animal so that he may take necessary precautions. To represent the scenario in our project, LED and Buzzers been used. Here in the presence of the Card (Animal) the buzzer makes noise and The LED Lights up to represent The Alarm and

(a) Reading Nearby animal
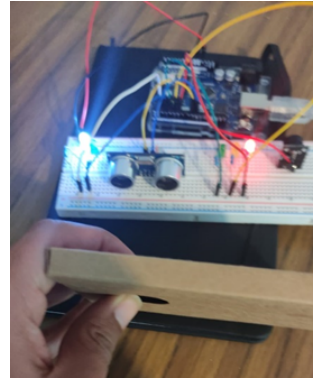
(b) Reading Nearby animal
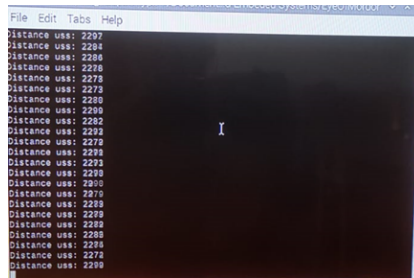
(c) User can check approximate distance

Figure 10: Eye of Mordor detection

notification respectively. Once recognized the condition the user can turn off the buzzer and led by pressing the stop button on the smartphone app as shown in Figure 8.

By realizing this system we believe, we will be able to save a lot of poor innocent souls from being a victim of Human advancement and interventions.

References

[1] Kanvanize. (n.d.). What Is Agile Project Management? A Comprehensive Guide. Kanban Software for Agile Project Management. https://kanbanize.com/agile/project-management

[2] APM. (n.d.). Agile project management. What Is Agile Project Management? https://www.apm.org.uk/resources/find-a-resource/agile-project-management/

[3] APM. (n.d.-b). What is project management? https://www.apm.org.uk/resources/what-is-project-management/

[4] GitHub. (n.d.). About projects. About Projects. https://docs.github.com/en/issues/trying-out-the-new-projects-experience/about-projects

[5] DesPortes, K., Anupam, A., Pathak, N., and DiSalvo, B. (2016, June). BitBlox: a redesign of the breadboard. In Proceedings of The 15th International Conference on Interaction Design and Children (pp. 255-261).

[6] Badamasi, Y. A. (2014, September). The working principle of an Arduino. In 2014 11th international conference on electronics, computer and computation (ICECCO) (pp. 1-4). IEEE

[7] Jolles, J. W. (2021). Broad-scale applications of the Raspberry Pi: A review and guide for biologists. Methods in Ecology and Evolution, 12(9), 1562-1579.

[8] Mosquitto, E. (2018). Eclipse mosquitto. Eclipse Mosquitto.