

DESIGN & ANALYSIS OF MONKEY DETERRENT SYSTEM

Dynamics Control & Instrumentation
Coursework presentation

Jamil R Haidar
HAI26891532

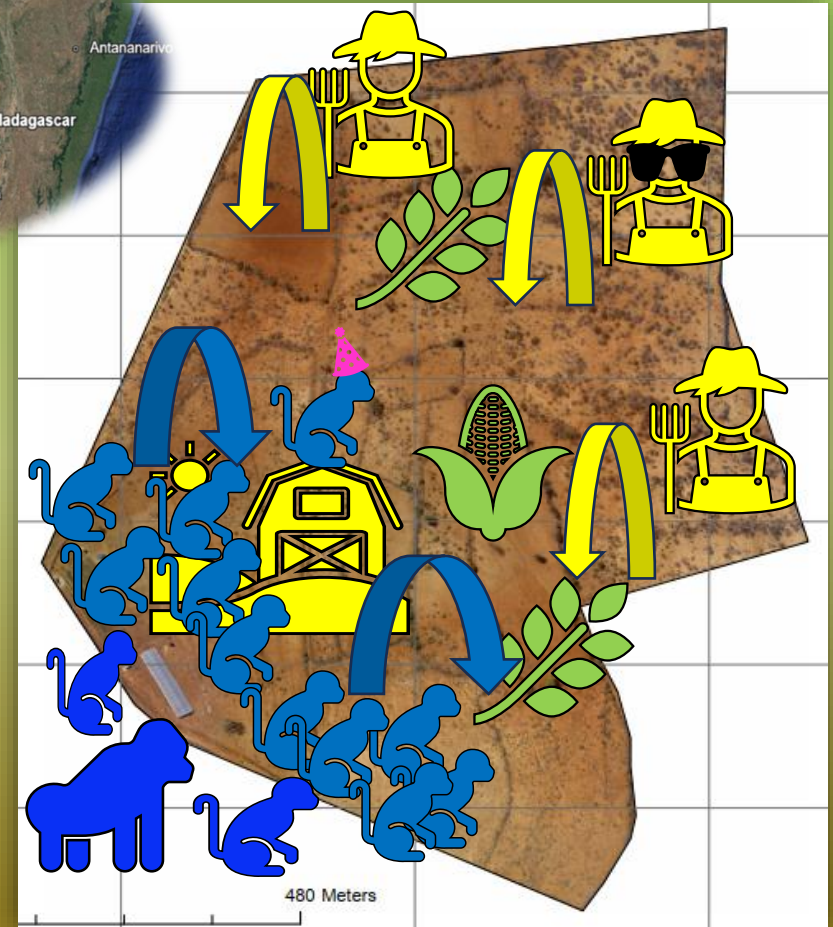


**THE CHOMZANGARI WOMEN'S COOPERATIVE IN ZIMBABWE IS UNDER ATTACK
MONKEYS ARE DAMAGING FARMERS CROPS! THE CWC NEEDS A SOLUTION!!**

CWC PLOT MAP



CWC ISSUE

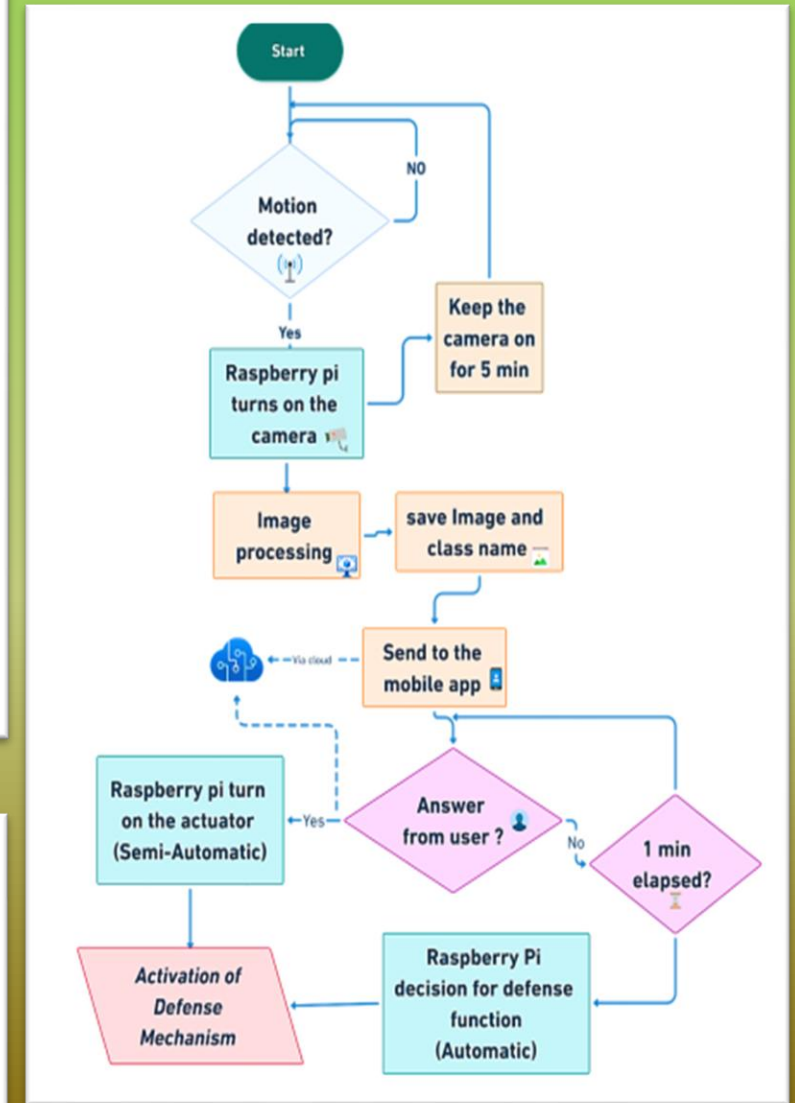
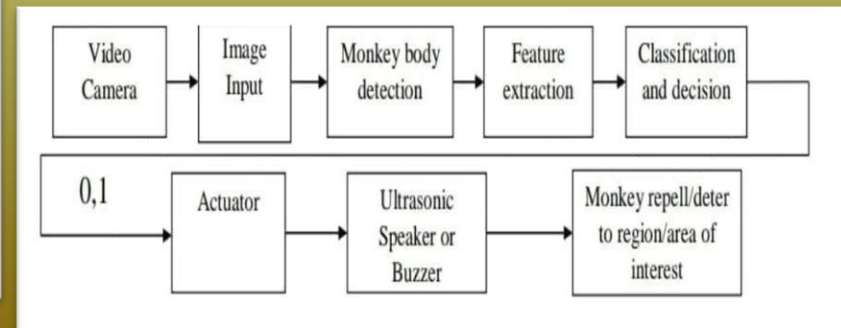
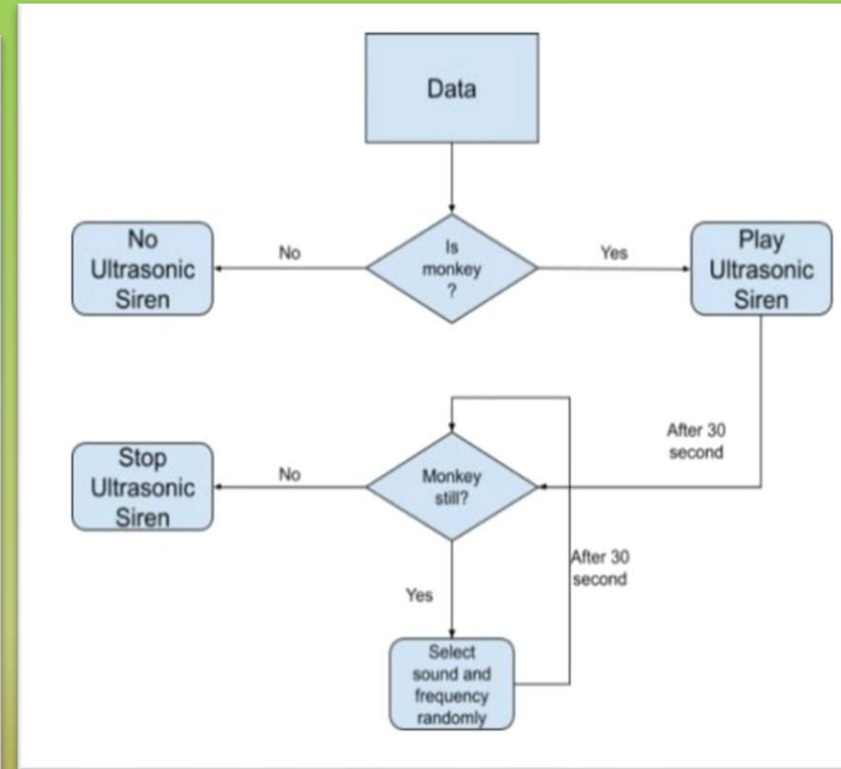
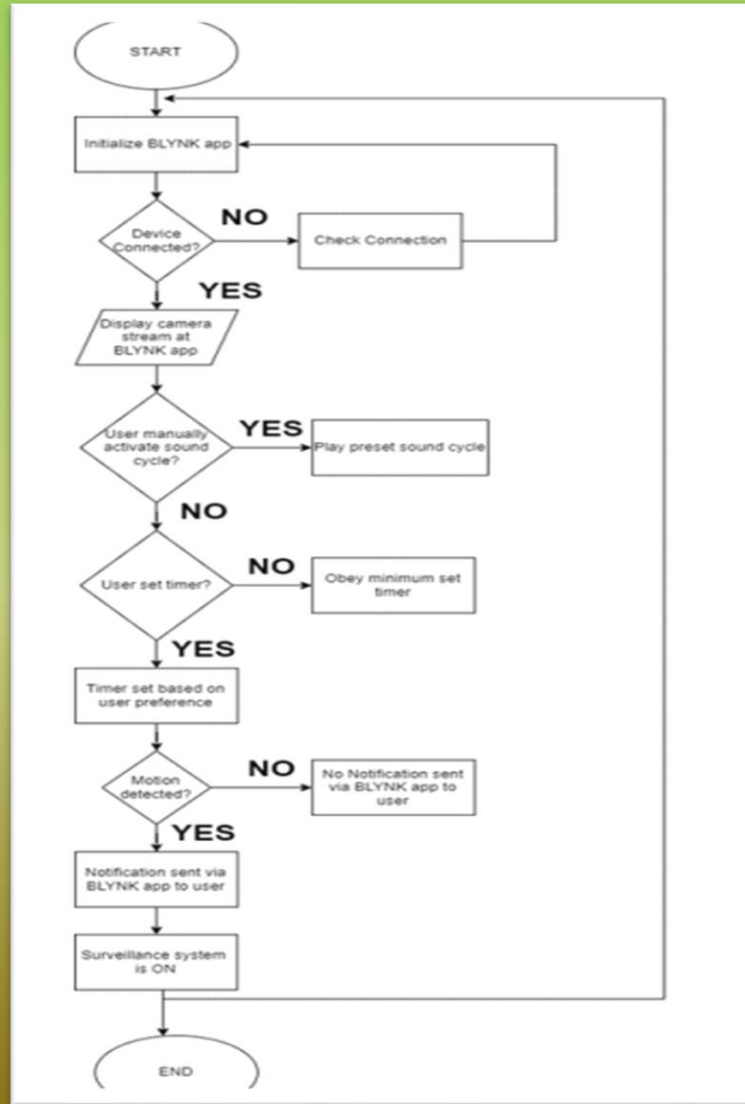


Autonomous
Sustainable
Low-cost
Scalable
Humane



Researched Models

Logic Sequences of Peer Reviewed Models



CWC MODEL

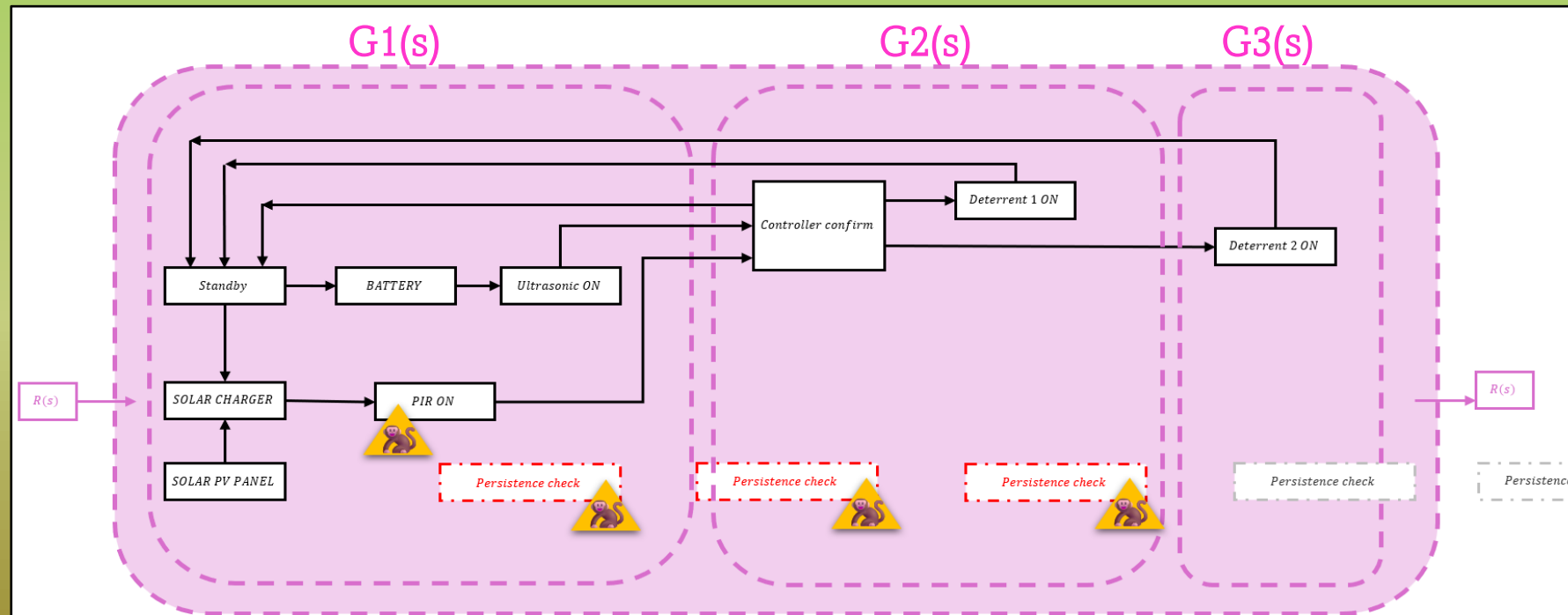
Logic Sequences Requirements

Monkey Behavior

A fixed single-mode deterrent risks habituation, so staged response matters.

Cost & Sourcing

It had to stay locally practical and realistically replicable.



Sustainability

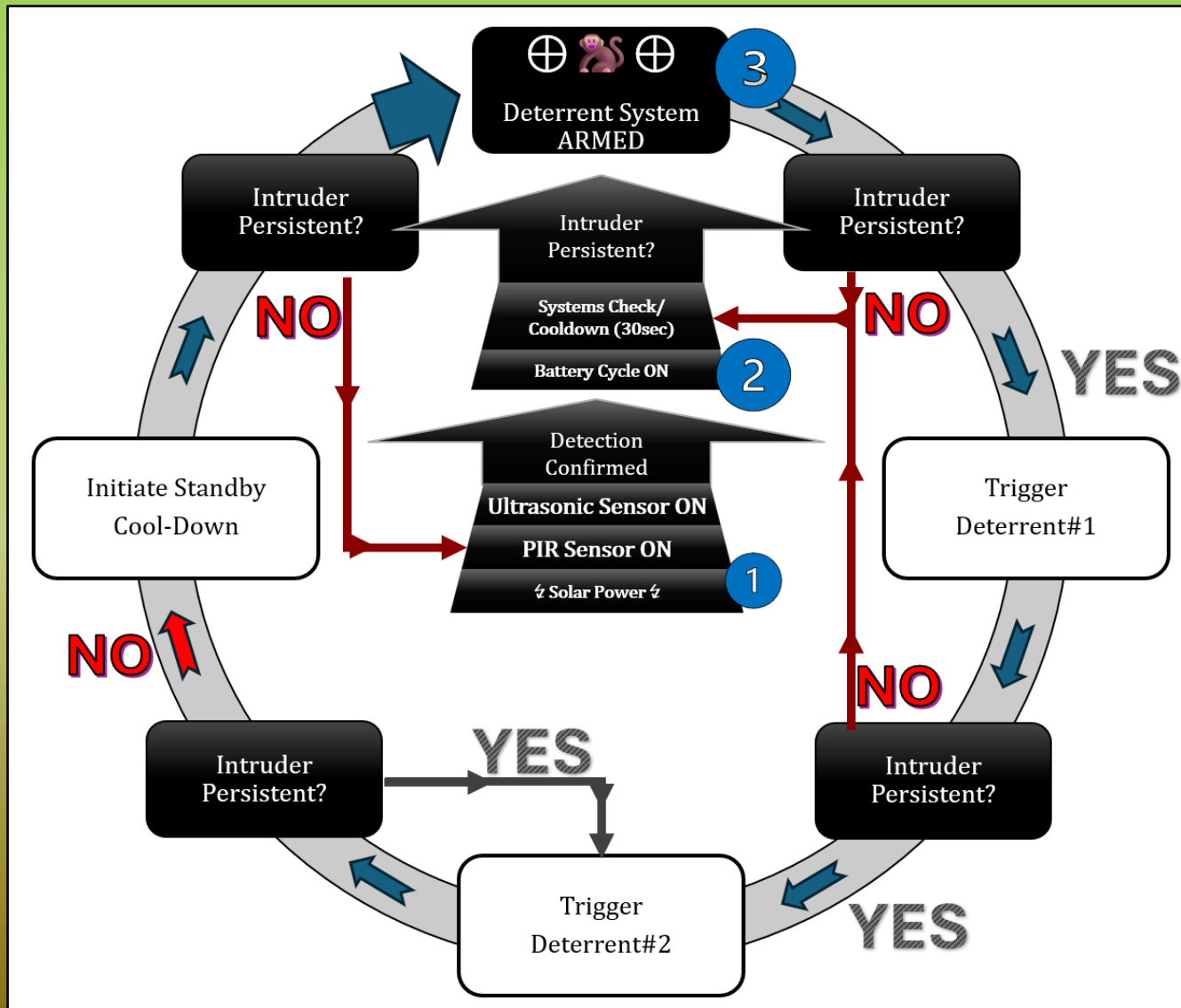
Power use had to stay low enough to justify an off-grid future build.

Scalability

The logic should work as a node that can later be replicated.

CWC MODEL

Escalation Logic Model



Three-stage operating idea:

Stage 1:
PIR sees motion and the ultrasonic sensor confirms presence within range.

Stage 2:
the controller checks readiness and arms Deterrent 1.

Stage 3:
If the intrusion persists, the controller escalates activates Deterrent 1 and then escalates to arm Deterrent 2; otherwise, it cools down back to standby.

Why I like this logic

It is simple enough to build, selective enough to save power, and structured enough to model mathematically.

SIMULINK TESTING

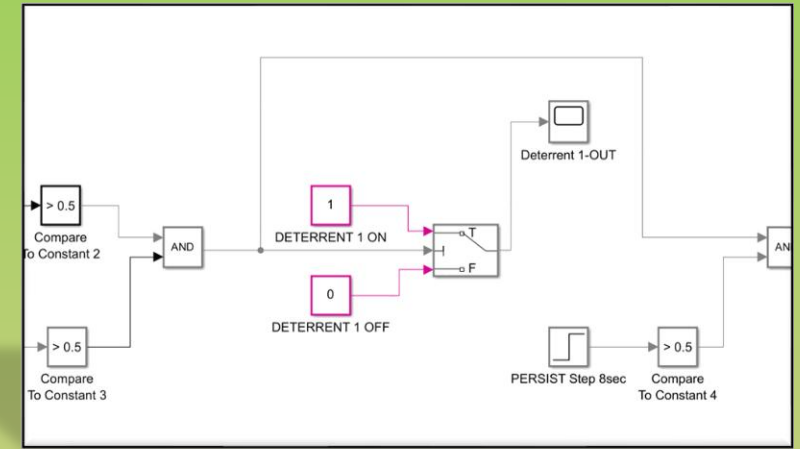
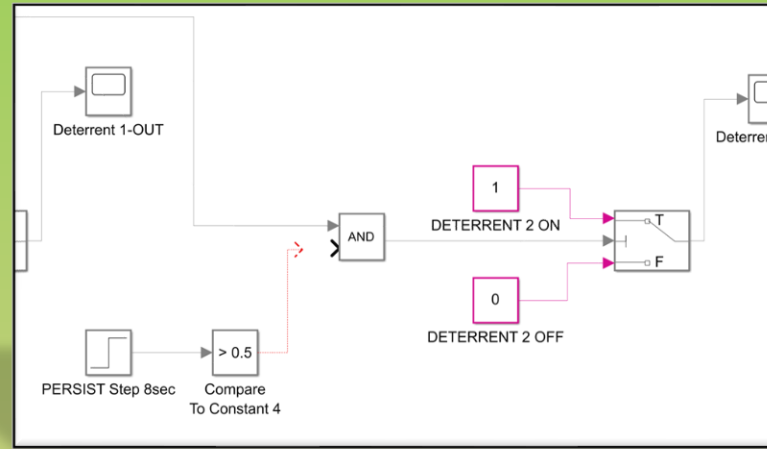
Transfer System Logic

Applied first-order form

$$G(s) \approx \frac{K}{\tau_s + 1}$$

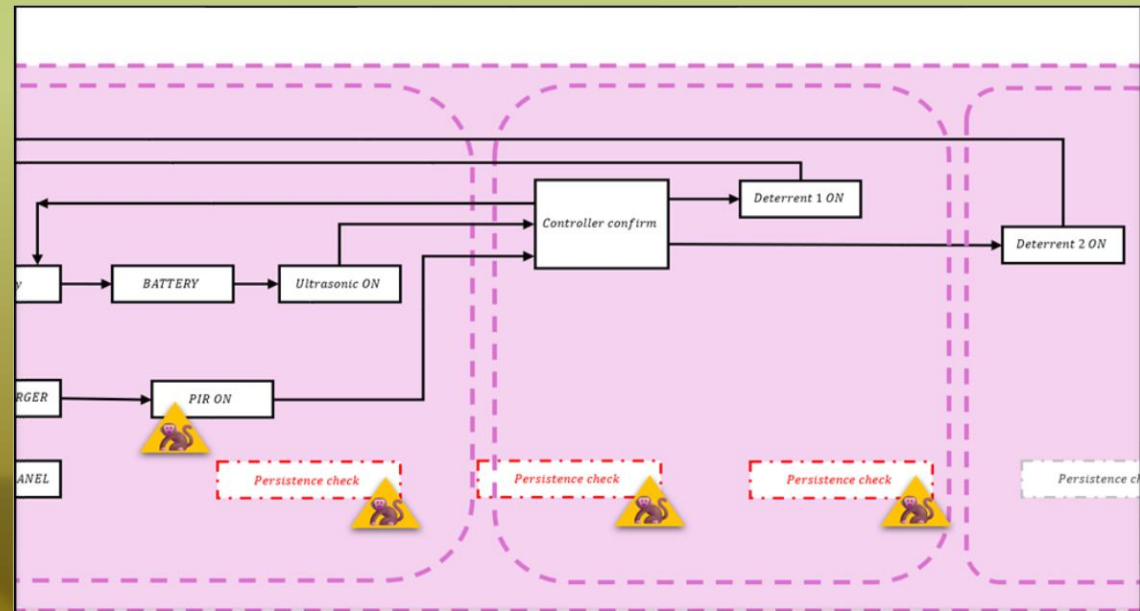
$$G(s) \approx \frac{1}{0.3s + 1}$$

$K \approx 1$ and $\tau \approx 0.3$ s from the measured confirmation delay.



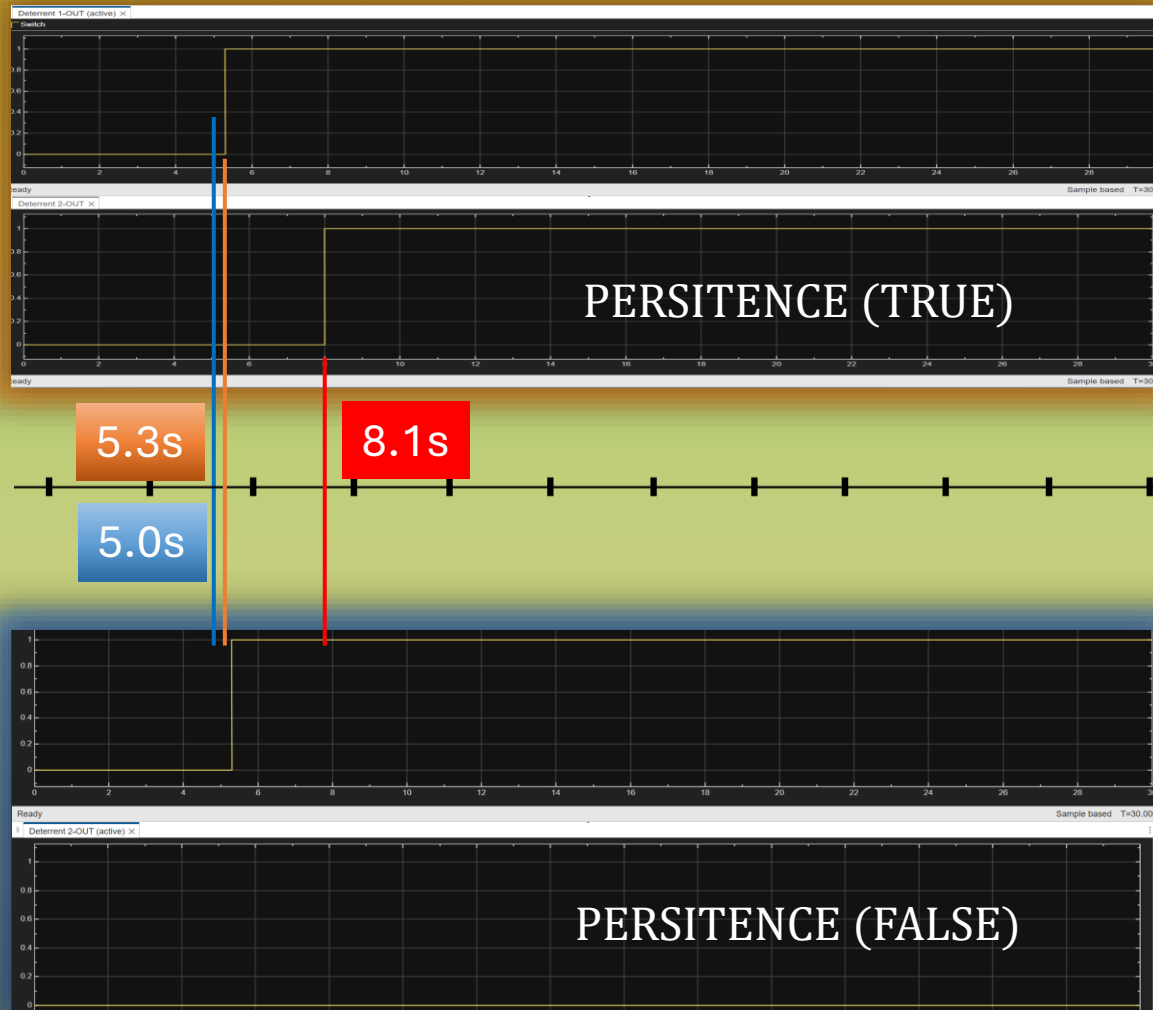
Key timings

- PIR step = 5.0 s
- Ultrasonic step = 5.1 s
- Confirmation delay = 0.3 s
- Deterrent 2 trigger = 8.0 s if persistence remains true



SIMULINK RESULTS

Transfer System Logic



What the plots prove

*IF (P.I.R + U.S = ON) AND (PERSISTENCE = TRUE @ 5.3 s)
ARM DETERRENT 1*

*IF PERSISTENCE = TRUE @ 5.1 s
Deterrent 1 goes active at about 5.3 s
ESCALATE to Deterrent 2 follows at 8.0 s.*

*IF PERSISTENCE = FALSE @ 5.1 s
Deterrent 1 at 5.3 ONLY
Standby at 8.0 s*

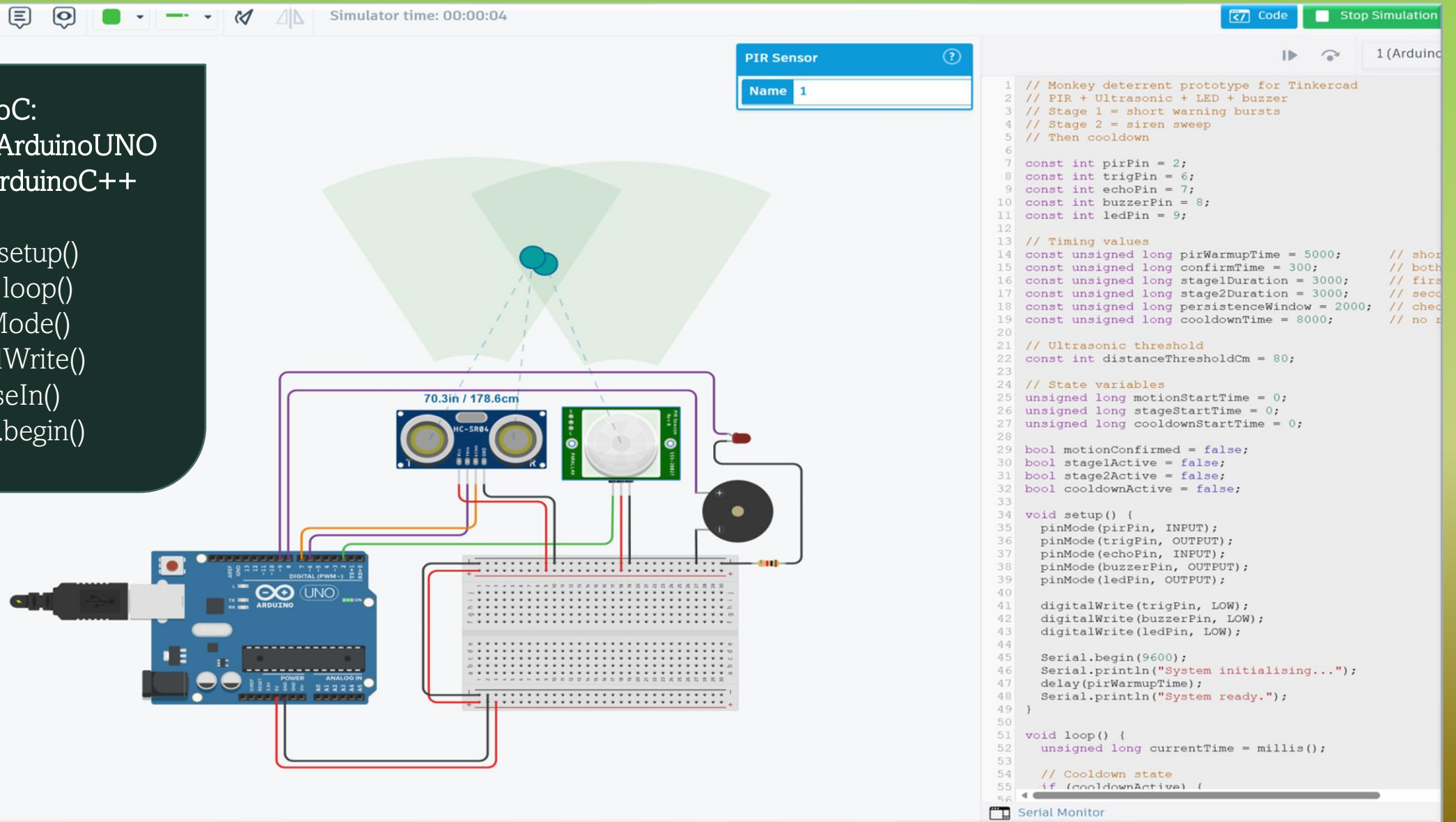
This high-level model shows that power sustainability is being considered as well as monkey behavior.

TinkerCAD™ PROTOTYPE MODEL

Proof of Concept

PoC:
Controller: ArduinoUNO
Coding: ArduinoC++

```
void setup()  
void loop()  
pinMode()  
digitalWrite()  
pulseIn()  
Serial.begin()
```



TinkerCAD™ PROTOTYPE CODE

Proof of Concept

```

2 // PIR + Ultrasonic + LED + buzzer
3 // Stage 1 = short warning bursts
4 // Stage 2 = siren sweep
5 // When cooldown
6
7 const int pirPin = 2;
8 const int trigPin = 6;
9 const int echoPin = 7;
10 const int buzzerPin = 8;
11 const int ledPin = 9;
12
13 // Timing values
14 const unsigned long pirWarmupTime = 5000; // shorter for Tinkercad
15 const unsigned long confirmTime = 300; // both PIR and ultrasonic must stay valid
16 const unsigned long stage1Duration = 3000; // first stage duration
17 const unsigned long stage2Duration = 3000; // second stage duration
18 const unsigned long persistenceWindow = 2000; // check for continued presence
19 const unsigned long cooldownTime = 8000; // no retrigger during cooldown
20
21 // Ultrasonic threshold
22 const int distanceThresholdCm = 80;
23
24 // State variables
25 unsigned long motionStartTime = 0;
26 unsigned long stageStartTime = 0;
27 unsigned long cooldownStartTime = 0;
28
29 bool motionConfirmed = false;
30 bool stage1Active = false;
31 bool stage2Active = false;
32 bool cooldownActive = false;
33
34 void setup() {
35   pinMode(pirPin, INPUT);
36   pinMode(trigPin, OUTPUT);
37   pinMode(echoPin, INPUT);
38   pinMode(buzzerPin, OUTPUT);
39   pinMode(ledPin, OUTPUT);
40
41   digitalWrite(trigPin, LOW);
42   digitalWrite(buzzerPin, LOW);
43   digitalWrite(ledPin, LOW);
44
45   Serial.begin(9600);
46   Serial.println("System initialising...");
47   delay(pirWarmupTime);
48   Serial.println("System ready.");
49 }
50
51 void loop() {
52   unsigned long currentTime = millis();
53
54   // Cooldown state
55   if (cooldownActive) {
56     allOutputsOff();
57     if (currentTime - cooldownStartTime >= cooldownTime) {
58       cooldownActive = false;
59       Serial.println("Cooldown ended. Back to standby.");
60     }
61     return;
62   }
63
64   int pirState = digitalRead(pirPin);
65   int distanceCm = readDistanceCm();
66   bool ultrasonicConfirmed = (distanceCm > 0 && distanceCm <= distanceThresholdCm);
67
68   // Standby and confirmation logic
69   if (!stage1Active && !stage2Active) {
70     if (pirState == HIGH && ultrasonicConfirmed) {
71       if (motionStartTime == 0) {
72         motionStartTime = currentTime;
73       }
74

```

```

75   stageActive = true;
76   stageStartTime = currentTime;
77
78   Serial.print("Stage 1 active. Distance = ");
79   Serial.print(distanceCm);
80   Serial.println(" cm");
81
82   } else {
83     motionStartTime = 0;
84     motionConfirmed = false;
85   }
86
87   }
88
89   // Stage 1: warning bursts
90   if (stage1Active) {
91     runStage1Pattern();
92
93     if (currentTime - stageStartTime >= stage1Duration) {
94       allOutputsOff();
95       stage1Active = false;
96
97       bool persistent = checkPersistence();
98
99       if (persistent) {
100         stage2Active = true;
101         stageStartTime = millis();
102         Serial.println("Persistence confirmed. Stage 2 active.");
103       } else {
104         motionConfirmed = false;
105         cooldownActive = true;
106         cooldownStartTime = millis();
107         Serial.println("No persistence. Entering cooldown.");
108       }
109     }
110   }
111
112   // Stage 2: siren sweep
113   if (stage2Active) {
114     runStage2Siren();
115
116     if (currentTime - stageStartTime >= stage2Duration) {
117       allOutputsOff();
118       stage2Active = false;
119       motionConfirmed = false;
120       cooldownActive = true;
121       cooldownStartTime = millis();
122       Serial.println("Stage 2 ended. Entering cooldown.");
123     }
124   }
125
126   // ----- Functions -----
127
128   int readDistanceCm() {
129     digitalWrite(trigPin, LOW);
130     delayMicroseconds(2);
131
132     digitalWrite(trigPin, HIGH);
133     delayMicroseconds(10);
134     digitalWrite(trigPin, LOW);
135
136     long duration = pulseIn(echoPin, HIGH, 30000);
137
138     if (duration == 0) {
139       return -1;
140     }
141
142     int distance = duration * 0.0343 / 2.0;
143     return distance;
144   }
145
146   bool checkPersistence() {
147     unsigned long startCheck = millis();
148

```

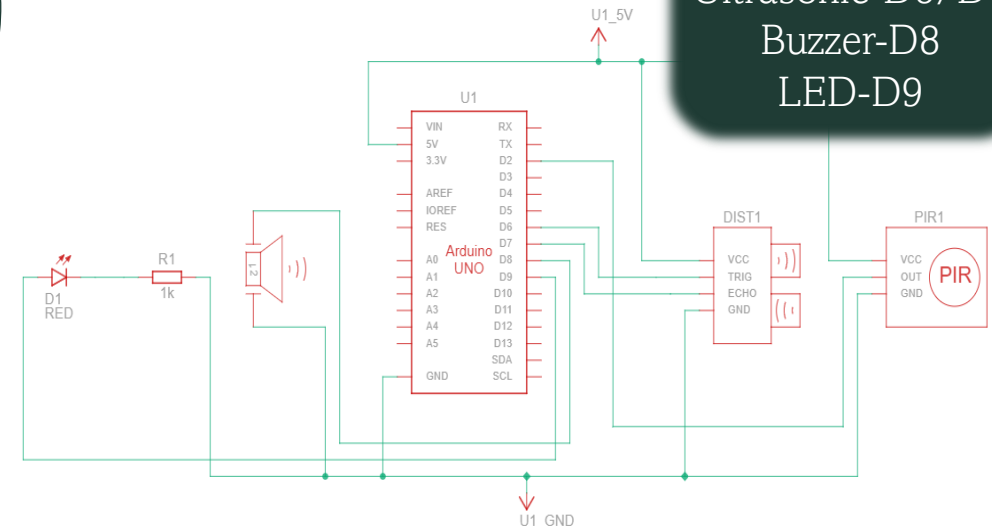
```

149   while (millis() - startCheck < persistenceWindow) {
150     int pirState = digitalRead(pirPin);
151     int distanceCm = readDistanceCm();
152
153     if (pirState == HIGH && distanceCm > 0 && distanceCm <= distanceThresholdCm) {
154       return true;
155     }
156     return false;
157   }
158
159   void runStage1Pattern() {
160     // short alarm bursts
161     digitalWrite(ledPin, HIGH);
162     tone(buzzerPin, 1200);
163   }
164
165   int readDistanceCm() {
166     digitalWrite(trigPin, LOW);
167     delayMicroseconds(2);
168
169     digitalWrite(trigPin, HIGH);
170     delayMicroseconds(10);
171     digitalWrite(trigPin, LOW);
172
173     long duration = pulseIn(echoPin, HIGH, 30000);
174
175     if (duration == 0) {
176       return -1;
177     }
178
179     int distance = duration * 0.0343 / 2.0;
180     return distance;
181   }
182
183   bool checkPersistence() {
184     unsigned long startCheck = millis();
185

```

PINS:

PIR-D2
Ultrasonic-D6/D7
Buzzer-D8
LED-D9



TinkerCAD™ RESULTS

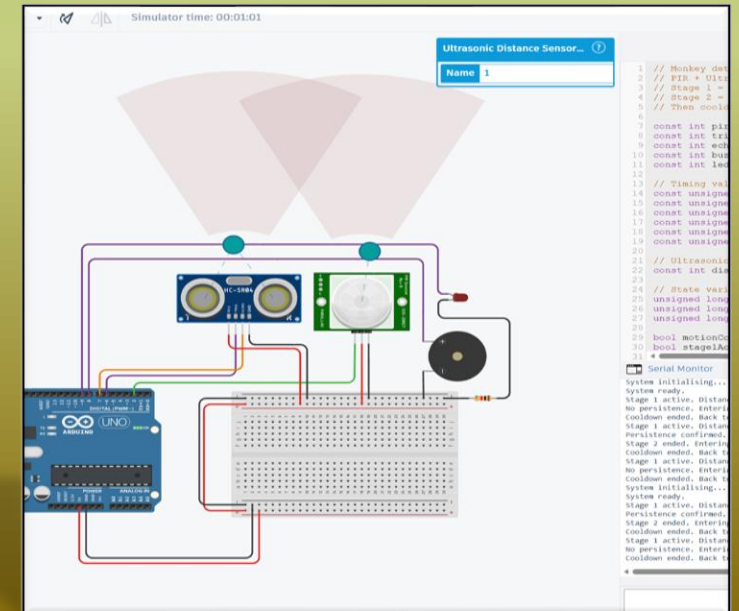
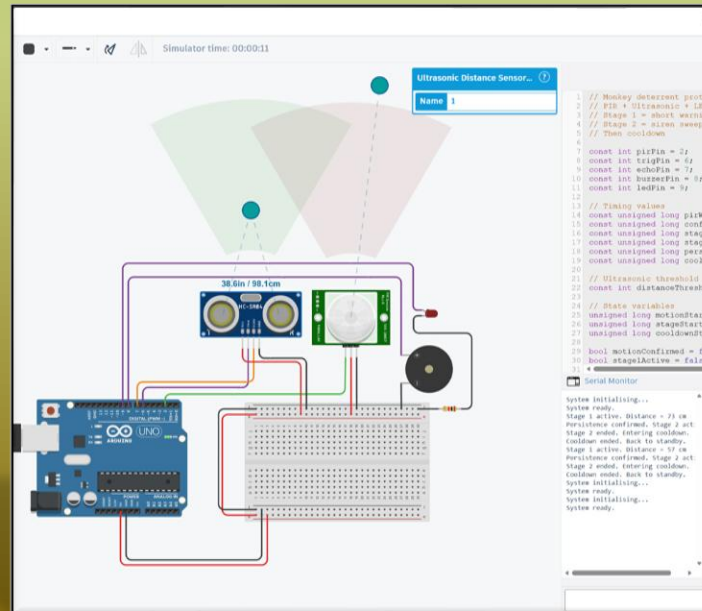
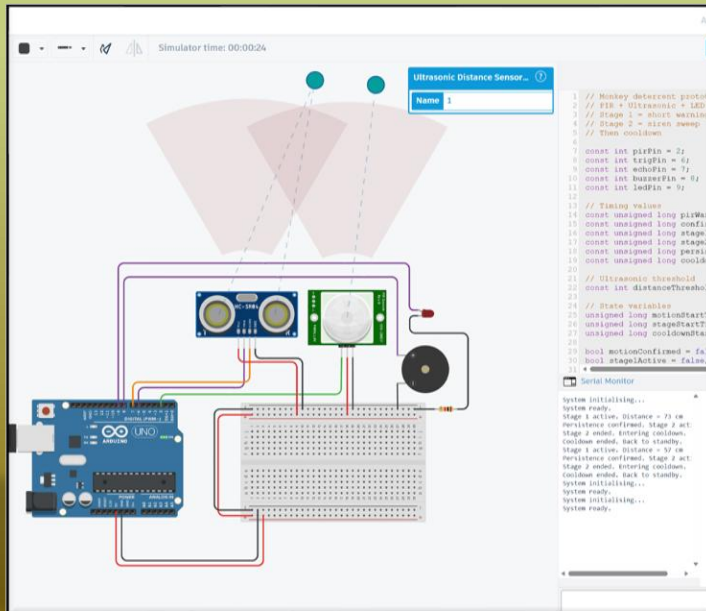
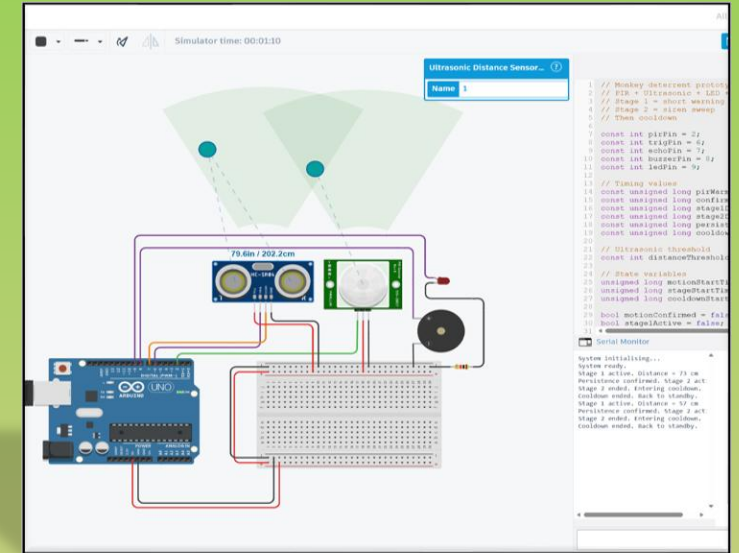
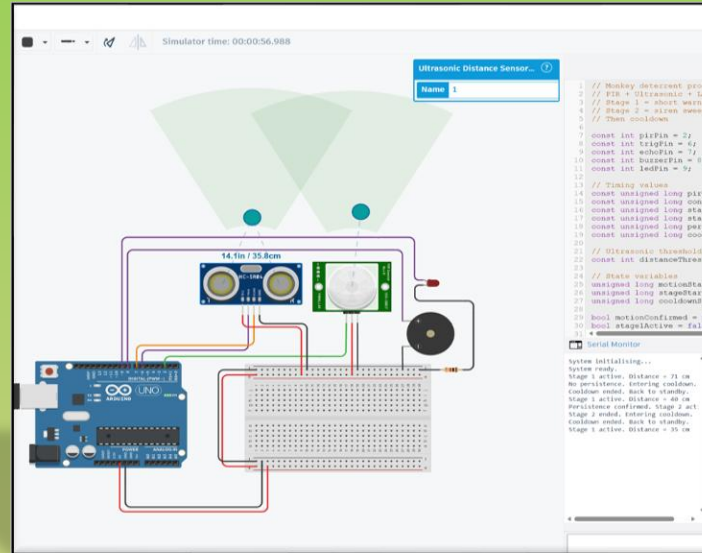
Logic and Deployment Readiness

Strengths & Advantages

- Selective triggering
- Repeatable stage logic
- Power-conscious escalation
- Simple enough to scale as a node

Weaknesses & Limitations

- Still a prototype, not a full field deployment
- No species identification
- Small-scale simulation timings
- Real environment would need ruggedization



Real World Equivalent Prototype

ROM BASIS

Approx. £415.00

- ✓ Ruggedized real-life equivalents to the Prototype.
- ✓ Sourced at consumer-market level (Amazon).
- ✓ “Bang-for-Buck” Pricing for Consumer confidence.

Ruggedized Single-node Deterrent

- ✓ Battery sizing based on optimal conditions.
- ✓ IP-Rated Hardware and Enclosure.
- ✓ Logic Controller saves on unnecessary uptime.

PROTOTYPE (SIMULATION)	PROTOTYPE (REAL-LIFE EQUIV.)	BRAND / MODEL	QTY	APPROX. PRICE.	WHY IT IS THERE
Arduino Uno	IP-Rated Industrial controller	WAVESHARE ESP32-S3-ETH-8DI-8RO	1	81.00	Main controller, digital I/O, relay outputs, rugged field control
PIR sensor	IP-Rated PIR detector	Generic 12V Outdoor PIR	1	5.00	First motion trigger
Ultrasonic sensor	IP-Rated ultrasonic sensor	JSN-SR04T	1	9.00	Secondary confirmation of intrusion distance
Piezo buzzer	Weatherproof horn speaker	Pyle PHSP131T	1	40.00	Main audible deterrent output
–	60 W amplifier module	TPA3118 Mono 60W	1	6.00	Drives horn output stage
LED	12 V strobe / beacon	Generic LED Strobe Siren Beacon	1	15.00	Visual deterrent output
–	Solar panel	Renogy 100 W 12 V mono panel	1	75.00	Off-grid charging source
–	MPPT charge controller	EPEVER Tracer 2210AN	1	77.00	Solar regulation and battery charging
–	Battery	12 V 100 Ah LiFeP04	1	160.00	Evening / disturbance-period energy storage
Breadboard	IP68 enclosure	Generic polycarbonate enclosure	1	12.00	Outdoor housing for control equipment
Jumper wires	Outdoor wiring set	Generic field wiring set	1	10.00	Interconnection and installation
Serial monitor	USB / config access	Included with controller setup	1	Included	Commissioning and diagnostics
Cooldown logic in code	Embedded firmware function	Included in controller	1	Included	Prevents repeated nuisance retriggering

Questions?

Thank you.

