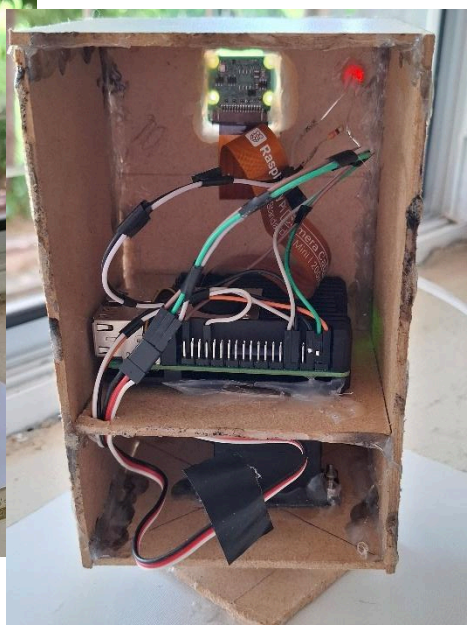
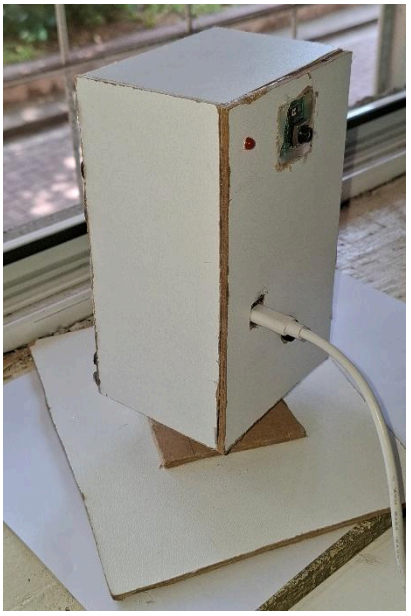
AI Dog Chaser

This project aims to develop an intelligent surveillance system using Raspberry Pi and object detection technology to monitor restricted outdoor areas and prevent stray dogs from entering. The system operates by continuously rotating a camera to scan the environment. When a dog is detected, the camera halts its movement, a high-frequency sound is emitted to deter the animal, and an LED light is activated as a visual alert. These deterrents remain active until the dog is no longer detected, after which the system resumes its scanning routine. The goal is to provide a cost-effective, automated, and humane solution for maintaining animal-free zones in public or private spaces.







Pictures of the final project

OBJECTIVES

- Develop practical skills in using Raspberry Pi for real-time automation.
- Implement object detection techniques to recognize and track animals.
- Create a responsive system that combines mechanical motion and audio-visual alerts.
- Understand the integration of hardware (camera, motors, sensors) and software (Python, object detection models).
- Explore the use of AI in monitoring and deterrent systems.

TARGET LEVEL

- High school students engaged in practical engineering or automation projects.
- Learners enrolled in vocational programs focused on embedded systems, AI applications, or digital electronics.
- Young innovators or tech enthusiasts with foundational skills in coding and circuit design.

PRE-REQUISITE KNOWLEDGE:

- Basic understanding of Python programming
- Familiarity with Raspberry Pi setup and general hardware interfacing
- Introductory knowledge of object detection and camera modules
- Ability to read simple wiring diagrams and connect electronic components such as LEDs and buzzers

HARDWARE

MATERIAL	
Raspberry Pi 5	
27W USB Type-C Power Supply for Raspberry Pi 5	
Flat Jumper Wires	
Servo Motor DS04-NFC 360° Continuous Rotation	

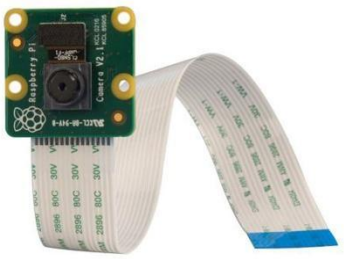
Raspberry Pi Camera Module 4	
Modulo Buzzer (FC-49)	
Red LED Diode	
Wood planks for the physical frame	

Table 1- List of material to implement the project AI Dog Chaser

SOFTWARE

- **Operating System:** Raspberry Pi OS (64-bit)
- **Programming Environment:** Thonny Python IDE or VS Code (via VNC)
- **Python Libraries:** opencv-python, ultralytics, pytorch, gpiozero, lgpio, os, time, sysfsPWM
- **AI Tools:** Roboflow (for dataset annotation and preprocessing), custom-trained YOLOv11 model (157-2MAT.pt)
- **Remote Access Tools:** VNC Viewer and SSH for real-time monitoring, debugging, and control on a local network

CIRCUIT

You will need to connect all components to a common ground (GND) and supply them with the appropriate power from the 3.3V or 5V pin. The **camera module** connects to the CSI port for

image capture. The **servo motor** is powered via the 5V pin and receives its control signal from **GPIO 18**. The **LED** is connected to **GPIO 15**, with a resistor in series to limit current. The **buzzer** is connected to **GPIO 13**, and is activated by the system when needed. Make sure all components share a common ground for stable operation.The circuit for this project is shown below:

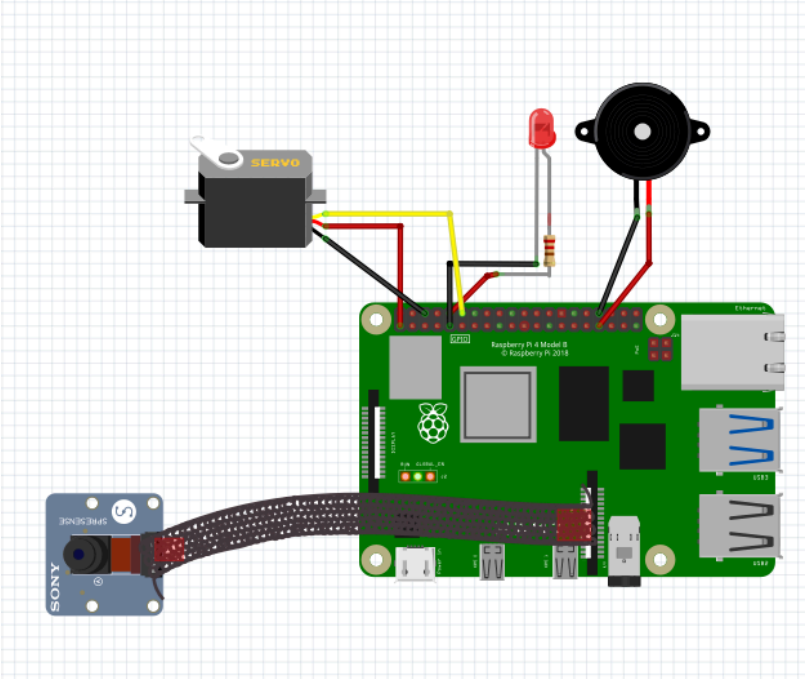


Figure 1 – Circuit diagram of the AI dog chase

STEP-BY-STEP INSTRUCTIONS FOR AI – DOG CHASER

1. Make the circuit

Begin by assembling the hardware components:

- Connect the Raspberry Pi Camera Module 4 to the CSI port on the Raspberry Pi 5.

- Wire the servo motor to enable camera rotation (GPIO 17).

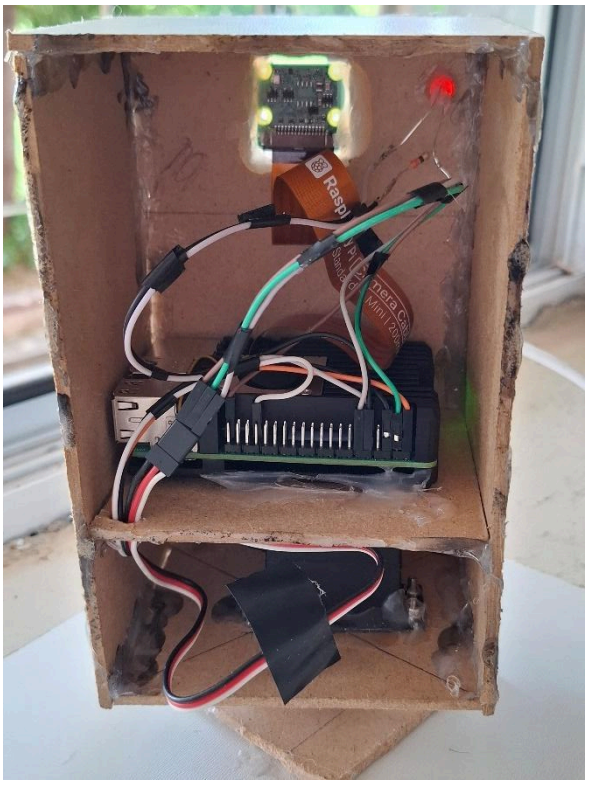
- Connect the LED diode to GPIO 22 with a current-limiting resistor.

- Connect the buzzer to GPIO 23.

- Use the 5V and GND pins to power all components appropriately.

- Ensure that all connections are secure and share a common ground to avoid noise and signal errors

.**Figure 2 – Circuit layout for AI Dog Chaser project: camera, servo, LED, and buzzer connected to Raspberry Pi.**



2. Prepare the software environment

Set up the Raspberry Pi with the following software components:

- Install Raspberry Pi OS (64-bit)

- Install Python 3

- Required libraries: python3-lgpio, python3-libcamera, libcamera, picamera2

- Upload the Python scripts used for object detection and camera control

- Ensure remote access is configured via VNC or SSH

3. Implement object detection and camera control

Develop or load a per-trained object detection model capable of identifying dogs in the video stream.

- Program the servo motor to rotate the camera within a defined scanning range.

- When a dog is detected, stop the motor, trigger the buzzer and LED.

- Resume scanning once no dog is present.

4. Test and observe system behavior

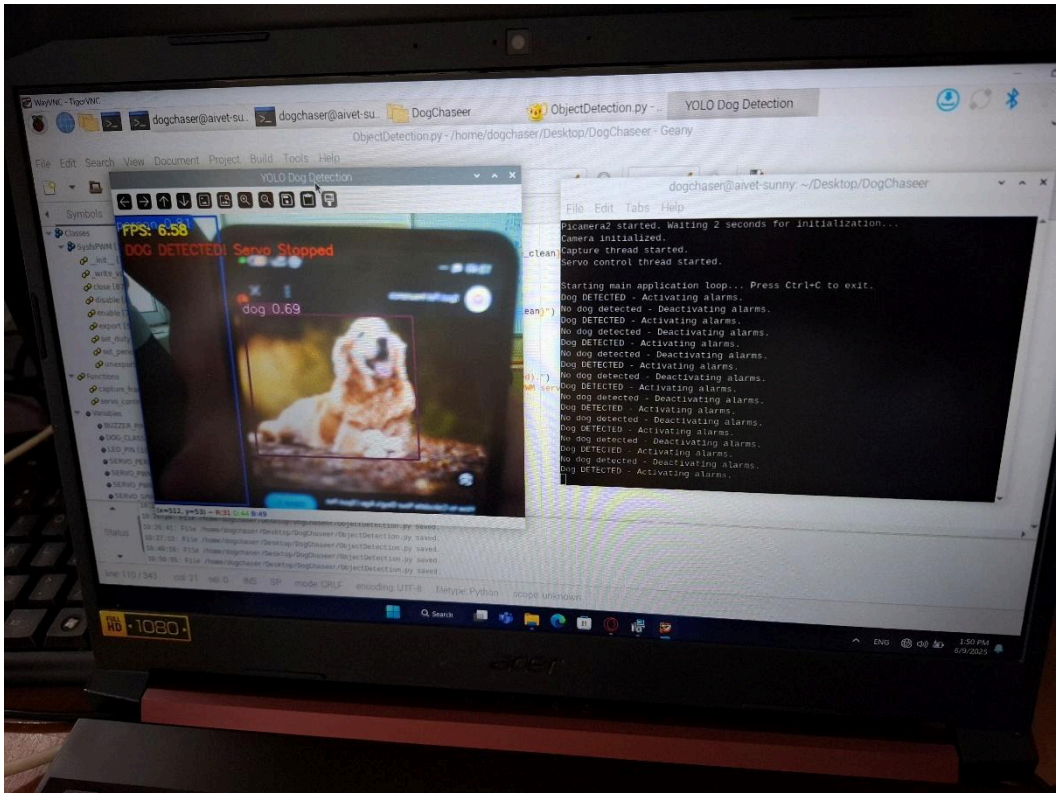


Figure 3 – The system detects a dog and activates the buzzer and LED.

EVALUATION CRITERIA:

System Functionality (40%): The camera rotates within the expected range, accurately detects dogs, and activates both the buzzer and LED alerts consistently.
Detection Accuracy (30%): The object detection model reliably identifies dogs with minimal false positives or missed detections.
System Responsiveness (20%): Fast response time between detection and alert activation, and smooth transition back to scanning when no dog is present.
User Evaluation & Improvement Suggestions (10%): Feedback from users in real-world environments, including suggestions for enhancing performance, reliability, or features.

ESTIMATED DURATION:

- Start Date: April 20, 2025
- End Date: June 1, 2025

LEARNING OUTCOMES:

- Practical skills in using Raspberry Pi for real-time automation and control.
- Knowledge of implementing object detection techniques to recognize and track animals, particularly dogs.

- Understanding of hardware-software integration, combining components like cameras, motors, sensors with Python programming and AI libraries such as OpenCV.
- Experience in developing a responsive alert system, which reacts in real time through audio (buzzer) and visual (LED) signals.
- Insight into the application of artificial intelligence in monitoring and deterrence systems for public or private area management.

SETUP & IMPLEMENTATION STEPS:

1. Hardware Preparation: Assemble the components — connect the camera module, LED, buzzer, and servo motor to the Raspberry Pi.
2. Software Setup: Install the necessary software on Raspberry Pi, including Raspbian OS, Python, and required libraries (OpenCV, TensorFlow or similar).
3. Camera Control & Rotation Logic: Program the servo motor to continuously rotate the camera, scanning the environment in a defined range.
4. Object Detection Integration: Use a pre-trained object detection model to analyze the camera feed in real time and detect the presence of dogs.
5. Trigger Mechanism: When a dog is detected, stop the camera rotation, activate the buzzer (high-frequency sound), and turn on the LED light.
6. Reset Condition: Once the dog is no longer detected, resume the scanning rotation and return the system to its monitoring state.
7. Testing and Calibration: Run multiple tests to adjust camera angle, detection sensitivity, and buzzer sound level for optimal performance.
8. Final Deployment: Mount the system in the desired location for continuous operation and monitoring.

CHALLENGES & TROUBLESHOOTING TIPS:

- **Challenge:** Camera rotation issues
    - **Tip:** Ensure the servo motor is correctly connected and that angle parameters are properly defined in the code. Test the motor independently before full integration.
- **Challenge:** Slow or delayed response to detection
    - **Tip:** Check the video feed processing speed. Lower the camera resolution or frame rate if the Raspberry Pi struggles with real-time performance.
- **Challenge:** False positives or missed detections
    - **Tip:** Place the system in a well-lit environment and minimize obstructions in the camera's view. Calibrate detection sensitivity and run multiple field tests.
- **Challenge:** Inconsistent long-term performance
    - **Tip:** Mount the system securely and conduct routine hardware checks. Consider implementing an automatic reset mechanism in the code to handle crashes or system hang-ups.