AI & SUSTAINABILITY IN VET EDUCATION

Name of the Author/s: Sucessos Criativos, Lda

Title of WP: WP3 – AI-SAVES@ProjectHub

Activity Reference: 3.2 – Project template for AI-SAVES@ProjectHub

Logo:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Table of Contents

1. Project Title: 4
2. Project Category: 4
3. Project Description: 4
4. Keywords: 4
5. Target Audience: 4
6. Project Objectives: 4
7. Project Timeline: 5
8. Resources Provided: 5
9. Tools and Technologies Used: 5
10. Step-by-Step Implementation Plan: 5
11. Evaluation Criteria: 13
12. Languages Available: 13
13. Contributors: 14
14. Feedback and Continuous Improvement: 14

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

1. Project Title:

AI-Based Computer Vision for Security Monitoring

---

2. Project Category:

AI in Robotics

---

3. Project Description:

This project focuses on the development of a simple yet effective AI-based security monitoring system using a Raspberry Pi and a camera module. The system continuously analyses the camera feed using an object detection model to identify the presence of people in the scene. Unlike basic motion detection systems, this solution introduces the concept of a defined detection zone, represented by a virtual bounding box corresponding to a specific real-world area of interest.

A notification is triggered only when a person is detected inside this predefined zone, reducing false alerts caused by irrelevant movement outside the monitored area. When such an event occurs, the system sends a WhatsApp message via the CallMeBot API, simulating a real-time security alert mechanism. This project demonstrates how computer vision and messaging APIs can be combined to create low-cost, intelligent monitoring solutions suitable for access control, restricted areas, or surveillance applications.

---

4. Keywords:

Raspberry Pi, Person Detection, Computer Vision, Security Monitoring

---

5. Target Audience:

Both Educators and Students

---

6. Project Objectives:

- Implement real-time person detection using a pre-trained object detection model.

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- Define and visualize a virtual detection zone corresponding to a real-world monitored area.
- Trigger alerts only when a person enters the specified zone, minimizing false positives.
- Integrate WhatsApp notifications using the CallMeBot API.
- Understand how spatial constraints can improve the reliability of AI-based detection systems.
- Explore practical security-oriented applications of computer vision on edge devices.

---

7. Project Timeline:

With the provided resources, this project should take approximately 2-3 hours or less to be done calmly.

---

8. Resources Provided:

- Python code implementing person detection, zone intersection logic, and WhatsApp notifications
- Demo video showcasing the result

---

9. Tools and Technologies Used:

- Raspberry Pi 5
- Raspberry Pi Camera Module 4 v2
- Python programming language
- OpenCV for camera input and image processing
- Pre-trained Object Detection Model (YOLO11-Nano)
- CallMeBot API for WhatsApp notifications via HTTP request

---

10. Step-by-Step Implementation Plan:

1. **Hardware Preparation:** Connect Raspberry Pi with the camera module

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

2. **Software Setup:** Install the necessary software on Raspberry Pi, including Raspbian OS and Python libraries including OpenCV, Picamera2, Ultralytics YOLO and Requests API.

3. **CallMeBot API Setup:** Follow the setup guide of CallMeBot (https://www.callmebot.com/blog/free-api-whatsapp-messages/) and save your Api Key.

4. **Execute the Code:** Copy the provided code bellow and save it on your Raspberry Pi. Before running the code, replace the value of "APIKEY" to your Api Key from step 3 and replace the value of "PHONE_NUMBER" to your phone number registered on step 3 (this value should contain your country code).

```
import cv2

import time

import requests

from picamera2 import Picamera2

from ultralytics import YOLO



# ===== CONFIGURATION =====

APIKEY = "1111111"  # Replace with your actual API key

PHONE_NUMBER = "+1123456789"  # Your real phone number with country code
(example for portugal: "+351911234567")

# Detection zone configuration (rectangle in pixels)

# Format: (x1, y1, x2, y2) - top-left corner to bottom-right corner

DETECTION_ZONE = (300, 100, 550, 300)  # Adjust based on your camera view



# Cooldown settings

DETECTION_COOLDOWN = 30  # seconds between notifications



# ===== CAMERA SETUP =====

picam2 = Picamera2()
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
picam2.preview_configuration.main.size = (640, 400)

picam2.preview_configuration.main.format = "RGB888"

picam2.preview_configuration.align()

picam2.configure("preview")

picam2.start()


print("> Camera initialized")


# ===== YOLO MODEL =====

model = YOLO("yolo11n.pt")

print("> YOLO model loaded")


# ===== TRACKING VARIABLES =====

last_notification_time = 0


# ===== FUNCTIONS =====

def send_notification(object_name):

    """Send WhatsApp notification via CallMeBot"""

    try:

        url                                                                    =
f"https://api.callmebot.com/whatsapp.php?phone={PHONE_NUMBER}&text=A+Person
+was+detected+in+the+zone.&apikey={APIKEY}"

        requests.get(url, timeout=5)

        print(f"✓ Notification sent: {object_name}")

    except Exception as e:

        print(f"✗ Failed to send notification: {e}")




def check_intersection(box_coords, zone_coords):
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
    """
    Check if bounding box intersects with detection zone
    box_coords: (x1, y1, x2, y2) - detected object bounding box
    zone_coords: (x1, y1, x2, y2) - detection zone
    """
    bx1, by1, bx2, by2 = box_coords
    zx1, zy1, zx2, zy2 = zone_coords

    # Check if boxes overlap
    if bx2 < zx1 or bx1 > zx2:  # No horizontal overlap
        return False
    if by2 < zy1 or by1 > zy2:  # No vertical overlap
        return False

    return True



def draw_detection_zone(frame, zone_coords):
    """Draw the detection zone on the frame"""
    x1, y1, x2, y2 = zone_coords
    cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 0, 0), 2)  # Blue rectangle
    cv2.putText(
        frame,
        "DETECTION ZONE",
        (x1, y1 - 10),
        cv2.FONT_HERSHEY_SIMPLEX,
        0.5,
        (255, 0, 0),
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        2
    )



# ===== MAIN PROGRAM =====
print("> Person Detection System Started!")
print(f"> Detection zone: {DETECTION_ZONE}")
print(f"> Notification cooldown: {DETECTION_COOLDOWN}s")
print("> Press 'q' to quit\n")


try:
    while True:
        # Capture frame from camera
        frame = picam2.capture_array()


        # Run YOLO inference
        results = model(frame)


        # Create annotated frame
        annotated_frame = frame.copy()
        draw_detection_zone(annotated_frame, DETECTION_ZONE)


        person_in_zone = False


        # Process detections
        for result in results:
            for box in result.boxes:
                cls_id = int(box.cls[0])
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
                label = result.names[cls_id]


                # Check if detected object is a person (class 0 in COCO
dataset)
                if label.lower() == "person":
                    confidence = box.conf[0] * 100
                    x1, y1, x2, y2 = map(int, box.xyxy[0])

                    # Check if person is in detection zone
                    if check_intersection((x1, y1, x2, y2), DETECTION_ZONE):
                        person_in_zone = True

                        # Draw bounding box in red (person in zone)
                        cv2.rectangle(annotated_frame, (x1, y1), (x2, y2),
(0, 0, 255), 2)
                        label_text = f"PERSON IN ZONE {confidence:.1f}%"
                        cv2.putText(
                            annotated_frame,
                            label_text,
                            (x1, y1 - 10),
                            cv2.FONT_HERSHEY_SIMPLEX,
                            0.5,
                            (0, 0, 255),
                            2
                        )

                        # Check cooldown before sending notification
                        current_time = time.time()
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
                        time_since_last = current_time - last_notification_time

                        if time_since_last >= DETECTION_COOLDOWN:
                            print(f"> Person detected in zone! (Confidence: {confidence:.1f}%)")
                            send_notification("Person")
                            last_notification_time = current_time
                        else:
                            remaining = int(DETECTION_COOLDOWN - time_since_last)
                            print(f"> Person in zone (cooldown: {remaining}s remaining)")

                    else:
                        # Draw bounding box in green (person outside zone)
                        cv2.rectangle(annotated_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
                        label_text = f"Person {confidence:.1f}%"
                        cv2.putText(
                            annotated_frame,
                            label_text,
                            (x1, y1 - 10),
                            cv2.FONT_HERSHEY_SIMPLEX,
                            0.5,
                            (0, 255, 0),
                            2
                        )

        # Display status
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        status = "PERSON IN ZONE" if person_in_zone else "Monitoring..."
        color = (0, 0, 255) if person_in_zone else (0, 255, 0)
        cv2.putText(
            annotated_frame,
            status,
            (10, 30),
            cv2.FONT_HERSHEY_SIMPLEX,
            0.7,
            color,
            2
        )

        # Show cooldown timer if active
        if last_notification_time > 0:
            time_since_last = time.time() - last_notification_time
            if time_since_last < DETECTION_COOLDOWN:
                remaining = int(DETECTION_COOLDOWN - time_since_last)
                cv2.putText(
                    annotated_frame,
                    f"Cooldown: {remaining}s",
                    (10, 60),
                    cv2.FONT_HERSHEY_SIMPLEX,
                    0.6,
                    (255, 255, 0),
                    2
                )

        # Display frame
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        cv2.imshow("Person Detection System", annotated_frame)


        # Quit with 'q'
        if cv2.waitKey(1) == ord('q'):
            break


except KeyboardInterrupt:
    print("\n> Interrupted by user")


finally:
    print("> Cleaning up...")
    cv2.destroyAllWindows()
    picam2.stop()
    print("> Program ended")
```

---

11. Evaluation Criteria:

Explain how the success of the project will be measured. This could include qualitative feedback from participants, quantitative performance indicators, or specific project outcomes.

- **System Functionality:** Ability to replicate the project.
- **Creativity:** Suggestions for improvements (new features, new objects detected, etc.). If suggested and tried (with success or failure) should be more rewarded.
- **False Alert Reduction:** Effective use of zone logic and cooldown timers to prevent spam notifications.

---

12. Languages Available:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

English

13. Contributors:

- **Fátima Pais, Luís Soares** – Project Designers and Facilitators

14. Feedback and Continuous Improvement:

Describe how feedback will be collected from participants and how this feedback will be used to improve future projects or iterations of this project.

- **Post-Implementation Survey:** After deploying the project, feedback will be collected through surveys targeting participants involved in the testing and observation process.
- **Team Reflection Sessions:** The project team will organize internal review sessions to discuss challenges faced, what worked well, and areas that need refinement.
- **Technical Evaluation:** Observations regarding detection accuracy, system stability, and hardware durability will be documented for improving future versions.
- **Iterative Updates:** Based on feedback and performance, updates to the software code and system design will be implemented to improve reliability and effectiveness.

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.