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

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: 5

7. Project Timeline: 5

8. Resources Provided: 5

9. Tools and Technologies Used: 7

10. Step-by-Step Implementation Plan: 7

Step 1: Prepare the Raspberry Pi for Hardware Interaction 7

Step 2: Build the Circuit 8

Step 3: Write the Python Program to Control the Window 11

Step 4: Code Breakdown 14

11. Evaluation Criteria: 16

12. Languages Available: 17

13. Contributors: 17

14. Feedback and Continuous Improvement: 17

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:

First Steps with AI-VET Kit: ENERGY-EFFICIENT DISTANCE MONITORING AND VISUAL FEEDBACK SYSTEM

---

2. Project Category:

Microelectronics Fundamentals

---

3. Project Description:

This project integrates three hardware elements—a distance sensor, an OLED display, and a NeoPixel strip—into a smart energy-efficient system. The ultrasonic sensor measures distance to an object, the OLED screen shows the live reading, and the NeoPixel strip responds with color-coded visual feedback. This real-time feedback system supports sustainability by optimizing power usage through minimal hardware and clear communication. It's ideal for smart rooms, object detection, or visual alert systems that prioritize low power consumption.

---

4. Keywords:

Raspberry Pi, Python, Ultrasonic Sensor, OLED Display, NeoPixel Strip, SPI, I^2C, Distance Monitoring, Visual Feedback, Sustainability

---

5. Target Audience:

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.

Educators, learners, and makers with or without prior experience who are beginning their journey into microelectronics and physical computing.

---

6. Project Objectives:

- Interface ultrasonic sensor, OLED display, and NeoPixel strip with Raspberry Pi 5
- Measure and display distances using real-time feedback
- Use SPI and I^2C communication efficiently
- Promote energy-conscious design with minimal hardware footprint
- Practice integrating multiple sensors into a single Python-controlled system

---

7. Project Timeline:

The estimated duration is approximately **2.5 to 3 hours**, including environment setup and physical assembly.

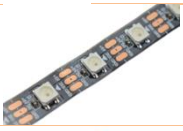
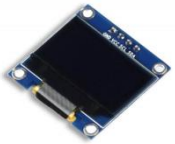
---

8. Resources Provided:

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.

- AI-VET Kit (Raspberry Pi 5, LED, Breadboard, Jumpers, OLED Display, Ultrasonic Sensor, LED Strip)

MATERIAL	
Raspberry Pi 5	
27W USB Type-C Power Supply for Raspberry Pi 5	
Flat Jumper Wires F/F	
Breadboard	
SINGLE-FILE WIRE FOR BREADBOARDS	
WS2812 RGB Programmable LED Strip 15x LED	
0.95inch OLED i2c Display Blue/Yellow SSD1306	
HC-SR04P - ULTRASONIC SENSOR 3V~5V	

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.

9. Tools and Technologies Used:

- Raspberry Pi OS
- Python 3
- Geany IDE
- Terminal (bash)
- Libraries: `lgpio`, `adafruit-circuitpython-ssd1306`, `adafruit-circuitpython-neopixel-spi`, `Pillow`, `board`

---

10. Step-by-Step Implementation Plan:

Step 1: Prepare the Raspberry Pi for Hardware Interaction

This project builds directly on previous AI-VET kit experiences, including:

- **"Getting Started with Raspberry Pi 5: Setup, Configuration, and Hello World in Python"**
- **"Measuring Distance with an Ultrasonic Sensor"**
- **"Working with the OLED Display"**
- **"Controlling NeoPixel Strip"**

Make sure you've completed those for a smoother setup.

In this step, you will:

- Install Raspberry Pi OS via Raspberry Pi Imager

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.

- Configure system settings like Wi-Fi and time zone
- Enable the following interfaces using `raspi-config`:
    - **I²C** for the OLED screen
    - **SPI** for the NeoPixel strip
- Identify the Raspberry Pi's IP using `ifconfig`
- Connect remotely via TigerVNC if needed
- Install required libraries:

```
pip3 install adafruit-circuitpython-ssd1306
pip3 install adafruit-circuitpython-neopixel-spi
sudo apt install python3-rpi-lgpio
```

Step 2: Build the Circuit

In this step, you'll wire three key modules—**Ultrasonic Sensor**, **OLED Display**, and **NeoPixel Strip**—to the Raspberry Pi using different communication protocols: GPIO, I²C, and SPI. Each module plays a unique role in the system and is connected to specific Raspberry Pi pins to ensure proper communication and functionality.

OLED Display (Visual Output – I²C Communication)

The OLED screen receives live text from the Raspberry Pi and displays it in real time. It uses the **I²C (Inter-Integrated Circuit)** protocol, allowing multiple devices to share the same two data lines, which simplifies wiring and supports energy-efficient data transmission.

Connections:

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.

- **VCC** → Raspberry Pi **5V** (pin 2)
  Provides power to the OLED display module

- **GND** → Raspberry Pi **GND** (pin 6)
  Connects the display's ground to the system's ground

- **SDA (Serial Data Line)** → **GPIO 2** (pin 3)
  Transmits data from the Raspberry Pi to the OLED display

- **SCL (Serial Clock Line)** → **GPIO 3** (pin 5)
  Sends timing signals that coordinate I^2C communication

These four connections are all you need to communicate with the OLED screen. Once connected, the Raspberry Pi will send pixel data over I^2C to control the display output.

NeoPixel LED Strip (Visual Feedback – SPI Communication)

The NeoPixel strip delivers color-coded feedback using SPI (Serial Peripheral Interface), which provides reliable and fast communication for lighting effects.

Connections:

- **DIN (Data In)** → GPIO **10** (MOSI) (pin 19)
  Transfers LED color data from the Raspberry Pi

- **VCC** → Raspberry Pi **5V** (pin 4)
  Powers the NeoPixel strip

- **GND** → Raspberry Pi **GND** (pin 14)
  Shares a common ground for accurate signaling

Ultrasonic Sensor (Distance Measurement – GPIO Control)

The ultrasonic sensor measures distance using echo time. It requires two GPIO pins for digital signal transmission.

Connections:

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.

- **VCC** → Raspberry Pi **5V** (pin 1)
  Supplies power to the sensor

- **GND** → Raspberry Pi **GND** (pin 9)
  Establishes common ground with the Pi

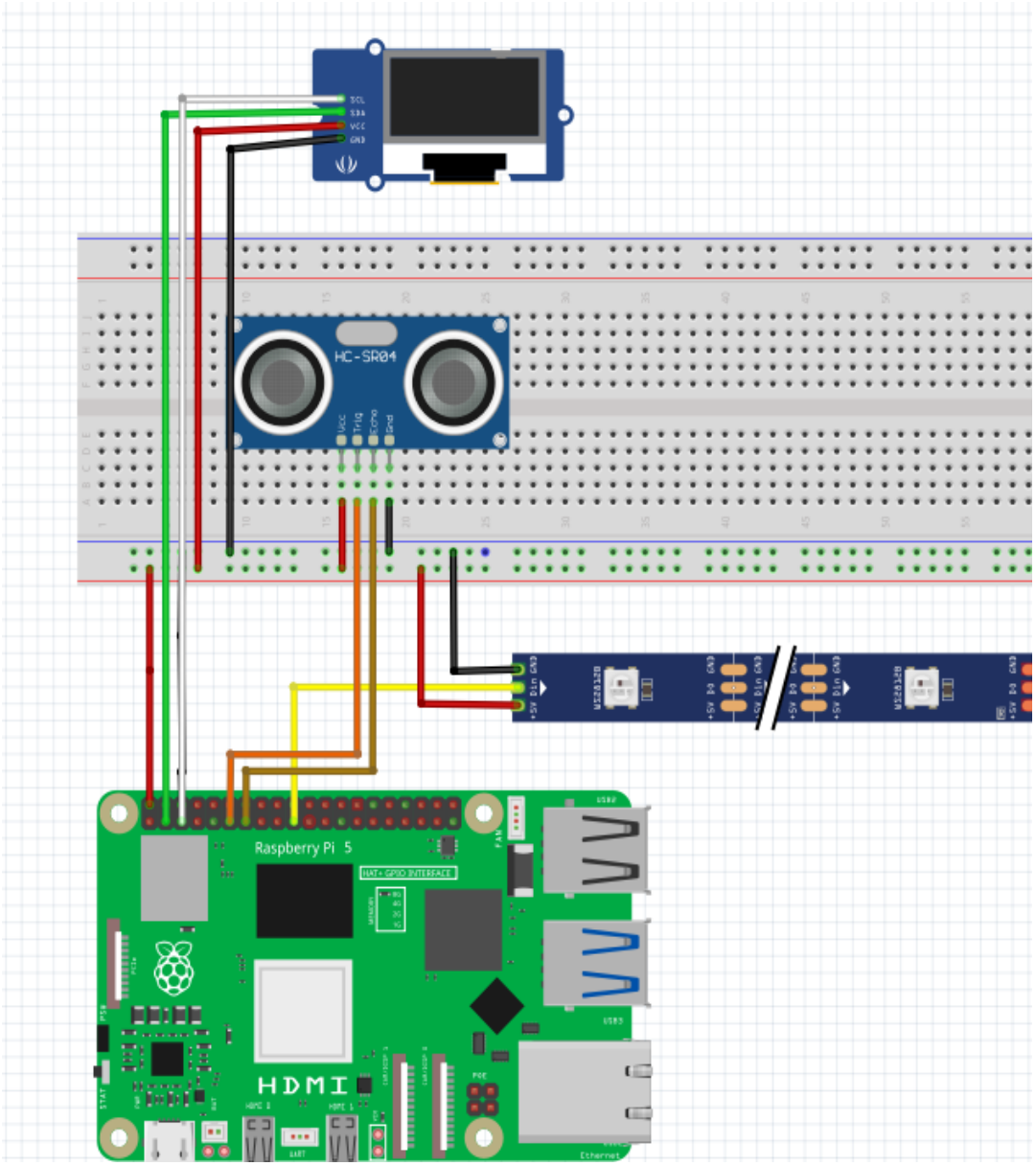
- **TRIG** → GPIO **17** (pin 11)
  Sends the ultrasonic pulse

- **ECHO** → GPIO **27** (pin 13)
  Receives the returning echo to calculate distance

Together, these connections enable the Raspberry Pi to collect distance data, display values on the OLED screen, and trigger color changes on the NeoPixel strip—forming a complete, energy-efficient visual feedback system.

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.



Step 3: Write the Python Program to Control the Window

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.

Open Geany and enter the following script. This code initializes all three components and links them through synchronized behavior:

```
import time
import board
import digitalio
from PIL import Image, ImageDraw, ImageFont
import adafruit_ssd1306
import lgpio
import neopixel_spi as neopixel
# Define GPIO pins for the ultrasonic sensor
TRIGGER_PIN = 17  # GPIO pin for the trigger of the ultrasonic sensor
ECHO_PIN = 27     # GPIO pin for the echo of the ultrasonic sensor
# Define the OLED display dimensions
WIDTH = 128   # OLED width in pixels
HEIGHT = 64   # OLED height in pixels
# Number of LEDs in the NeoPixel strip
NUM_PIXELS = 12  # Number of NeoPixel LEDs in the strip
# Define the color order for the NeoPixels (Green, Red, Blue)
PIXEL_ORDER = neopixel.GRB  # GRB color order for NeoPixel
# Define three colors (Red, Green, Blue) in hexadecimal format
COLORS = (0xFF0000, 0x00FF00, 0x0000FF)  # Red, Green, Blue colors
# Initialize GPIO chip (the default GPIO chip is "/dev/gpiochip0")
h = lgpio.gpiochip_open(0)
lgpio.gpio_claim_output(h, TRIGGER_PIN)  # Set TRIGGER_PIN as output
lgpio.gpio_claim_input(h, ECHO_PIN)      # Set ECHO_PIN as input
# Initialize OLED display using I2C communication
i2c = board.I2C()
oled = adafruit_ssd1306.SSD1306_I2C(WIDTH, HEIGHT, i2c, addr=0x3C)
# Initialize NeoPixel strip with SPI communication
spi = board.SPI()
pixels   =   neopixel.NeoPixel_SPI(spi,   NUM_PIXELS,   pixel_order=PIXEL_ORDER,
auto_write=False)
# Function to measure distance using the ultrasonic sensor
def measure_distance():
    # Send a 10-microsecond pulse to the trigger pin
    lgpio.gpio_write(h, TRIGGER_PIN, 1)
    time.sleep(0.00001)  # Wait for 10 microseconds
    lgpio.gpio_write(h, TRIGGER_PIN, 0)
    # Measure the time when the echo signal starts (goes high)
    start_time = time.time()
    while lgpio.gpio_read(h, ECHO_PIN) == 0:
        start_time = time.time()
    # Measure the time when the echo signal ends (goes low)
    end_time = time.time()
    while lgpio.gpio_read(h, ECHO_PIN) == 1:
        end_time = time.time()
    # Calculate the duration of the pulse and convert it to distance
    pulse_duration = end_time - start_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.

```
    distance = (pulse_duration * 34300) / 2  # Speed of sound is 34300 cm/s
    return distance
# Function to update the OLED display with the distance measurement
def update_display(distance):
    oled.fill(0)  # Clear the OLED display
    oled.show()

    # Create a new image for the OLED display
    image = Image.new("1", (oled.width, oled.height))
    draw = ImageDraw.Draw(image)
    # Draw a border rectangle for aesthetics
    BORDER = 5
    draw.rectangle(
        (BORDER, BORDER, oled.width - BORDER - 1, oled.height - BORDER - 1),
        outline=0,
        fill=0,
    )
    # Load the default font and prepare the text
    font = ImageFont.load_default()
    text = f"Distance: {distance:.2f} cm"
    # Calculate the size of the text to center it on the screen
    font_width, font_height = font.getbbox(text)[2:4]
    # Draw the text centered on the OLED screen
    draw.text(
        (oled.width // 2 - font_width // 2, oled.height // 2 - font_height //
2),
        text,
        font=font,
        fill=255,  # White text color
    )
    # Send the image to the OLED display
    oled.image(image)
    oled.show()
# Function to update the NeoPixel strip based on the distance
def update_neopixels(distance):
    if distance < 10:
        color = COLORS[0]  # Set to Red if distance is less than 10 cm
    elif distance < 20:
        color = COLORS[1]  # Set to Green if distance is between 10 cm and 20
cm
    else:
        color = COLORS[2]  # Set to Blue if distance is greater than 20 cm
    # Apply the selected color to all LEDs in the strip
    for i in range(NUM_PIXELS):
        pixels[i] = color
    pixels.show()
# Main loop to continuously measure distance, update OLED display, and NeoPixel
strip
try:
    while True:
        # Measure the distance from the ultrasonic sensor
```

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.

```
        distance_cm = measure_distance()
        print(f'Distance: {distance_cm:.2f} cm')    # Print distance to the
console
        # Update the OLED display with the measured distance
        update_display(distance_cm)
        # Update the NeoPixel strip based on the measured distance
        update_neopixels(distance_cm)
        # Delay between each loop iteration
        time.sleep(0.3)
# Catch Ctrl+C to exit the program gracefully
except KeyboardInterrupt:
    print("\nExiting program.")
# Cleanup GPIO resources and close the program
finally:
    lgpio.gpiochip_close(h)
    print("GPIO cleaned up!")
```

Save the file as: NeoDistOled.py

Step 4: Code Breakdown

Setup and Imports:

- `board`, `digitalio`, and `lgpio` for hardware interaction
- `Pillow` and `adafruit_ssd1306` for OLED control
- `neopixel_spi` for LED strip lighting
- Time for loop delays

Sensor and Display Initialization:

- GPIO pins 17 and 27 are claimed for ultrasonic reading
- OLED and NeoPixel objects are created using I^2C and SPI protocols

Core Functions:

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.

- `measure_distance()` uses TRIG and ECHO to calculate distance in cm
- `update_display()` draws the distance on the OLED
- `update_neopixels()` sets LED colors based on distance thresholds:
    - <10 cm → Red
    - 10–20 cm → Green
    - 20 cm → Blue

Main Loop:

- Continuously measures distance, displays it on screen, and updates LED strip
- Exits cleanly with Ctrl + C, freeing GPIO resources

Step 5: Run the Program and Observe the Output

1. **Save** the file as `NeoDistOled.py`
2. **Open** the **terminal** and **execute**:

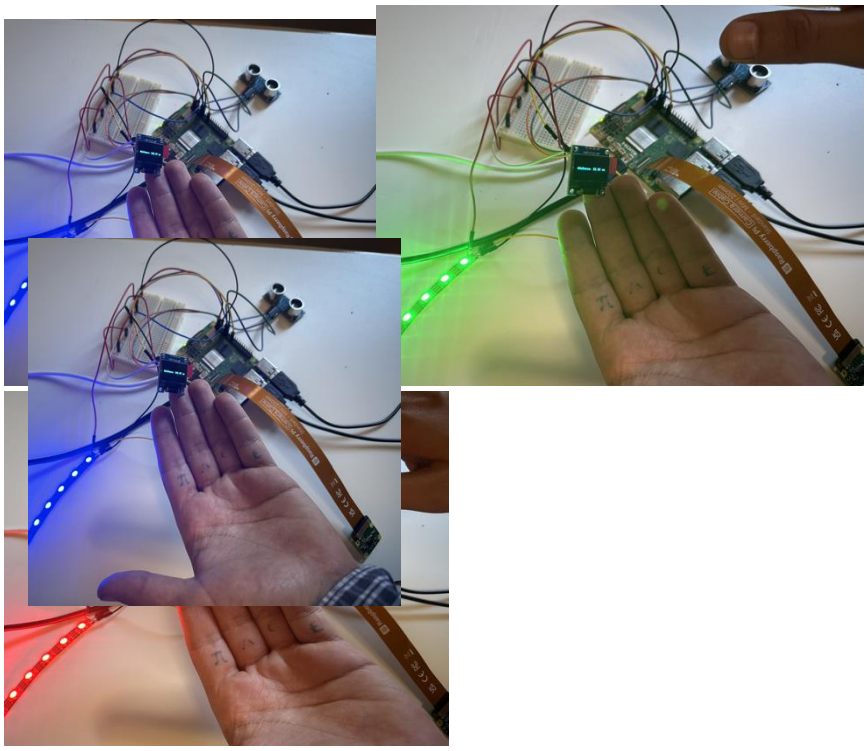
```
python3 NeoDistOled.py
```

You should observe:

- The OLED screen displaying real-time distance
- The NeoPixel strip lighting up in different colors based on proximity
- Terminal printout of measured distances

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.



11. Evaluation Criteria:

- **Sensor Accuracy:** The ultrasonic sensor correctly measures distances within a ±1 cm margin
- **Visual Feedback:** NeoPixel colors change accurately depending on the distance threshold
- **OLED Display:** Live readings are visible, legible, and updated in real 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.

- **Code Stability:** Script runs smoothly without errors, handles interruption gracefully

- **Integration Skills:** All three components work together synchronously to provide responsive feedback

---

12. Languages Available:

English, **Portuguese** (translations in progress)

---

13. Contributors:

Fátima Pais, Luís Soares – Project Designers and Facilitators

---

14. Feedback and Continuous Improvement:

- **Surveys**: At the end of this milestone, students and educators will complete feedback surveys to assess the effectiveness of the project.
- **Project Reviews**: After project completion, a final review meeting will be held to discuss potential improvements for future AI-based projects.
- **Data Analysis**: Feedback data will be analyzed to identify strengths and areas for improvement, informing future projects.

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.