# AI & SUSTAINABILITY IN VET EDUCATION

**Name of the Author/s:** Sucessos Criativos, Lda | P. Porto

**Title of WP:** WP3 – AI-SAVES@ProjectHub

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

European Vocational Training Association

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: 2

2. Project Category: 2

3. Project Description: 2

5. Target Audience: 2

6. Pre-Requisite knowledge: 2

7. Project Objectives: 3

8. Project Timeline: 3

9. Resources Provided: 3

10. Tools and Technologies Used: 4

11. Step-by-Step Implementation Plan: 6

12. Evaluation Criteria: 13

13. Languages Available: 13

14. Contributors: 13

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:**

Fire Detection System with Raspberry Pi and Temperature Sensor

---

**2. Project Category:**

- AI in IoT

---

**3. Project Description:**

This project aims to develop a basic early-warning fire detection system using a Raspberry Pi and a temperature sensor. The system monitors real-time temperature changes and activates LED indicators to reflect risk levels: green (normal), yellow (warning), and red (critical). This cost-effective, scalable solution contributes to environmental protection by detecting potential fire hazards early, whether in forests or indoor spaces.

---

**4. Keywords:**

Fire Detection**,** Raspberry Pi, Iot, Temperature Monitoring, Environmental Protection

---

**5. Target Audience:**

Both Educators and Students

---

**6. Pre-Requisite knowledge:**

- Basic understanding of Ohm's Law and electrical circuits
- Introduction to Python programming (variables, loops, functions)
- Knowledge of Raspberry Pi GPIO control
- Basic experience with PWM (Pulse Width Modulation) and servo motors

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.

**7. Project Objectives:**

- Introduce students to basic IoT and sensor integration.
- Teach real-time data acquisition and interpretation.
- Raise awareness about fire prevention and environmental sustainability.
- Develop hands-on skills in electronics and Python programming.
- Encourage the creation of low-cost, scalable solutions for real-world problems.

**8. Project Timeline:**

**9. Resources Provided:**

- Raspberry Pi 5
- Temperature sensor (e.g., DHT11 or DS18B20)
- 3 LEDs (green, yellow, red) and 3 resistors
- Breadboard and jumper wires
- Python code templates
- Step-by-step instructional guide

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.

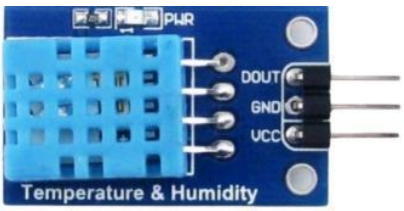
## 10. Tools and Technologies Used:

### 10.1 Hardware:

| 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 |  |
| Diffused 5mm LED Red |  |
| Diffused 5mm LED Yellow |  |

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.

| | |
|---|---|
| Diffused 5mm LED Green |  |
| Resistor 330 Ohm 1/4 W |  |
| DHT11 Temperature-Humidity Sensor (IM14) |   |

**10.2 Software:**

- Raspbian OS, Python 3
- Operating System: Raspberry Pi OS
- Programming Environment: Geany or Python IDE
- Python Libraries: lgpio, time

**Libraries**: Adafruit_DHT, RPi.GPIO, board, time

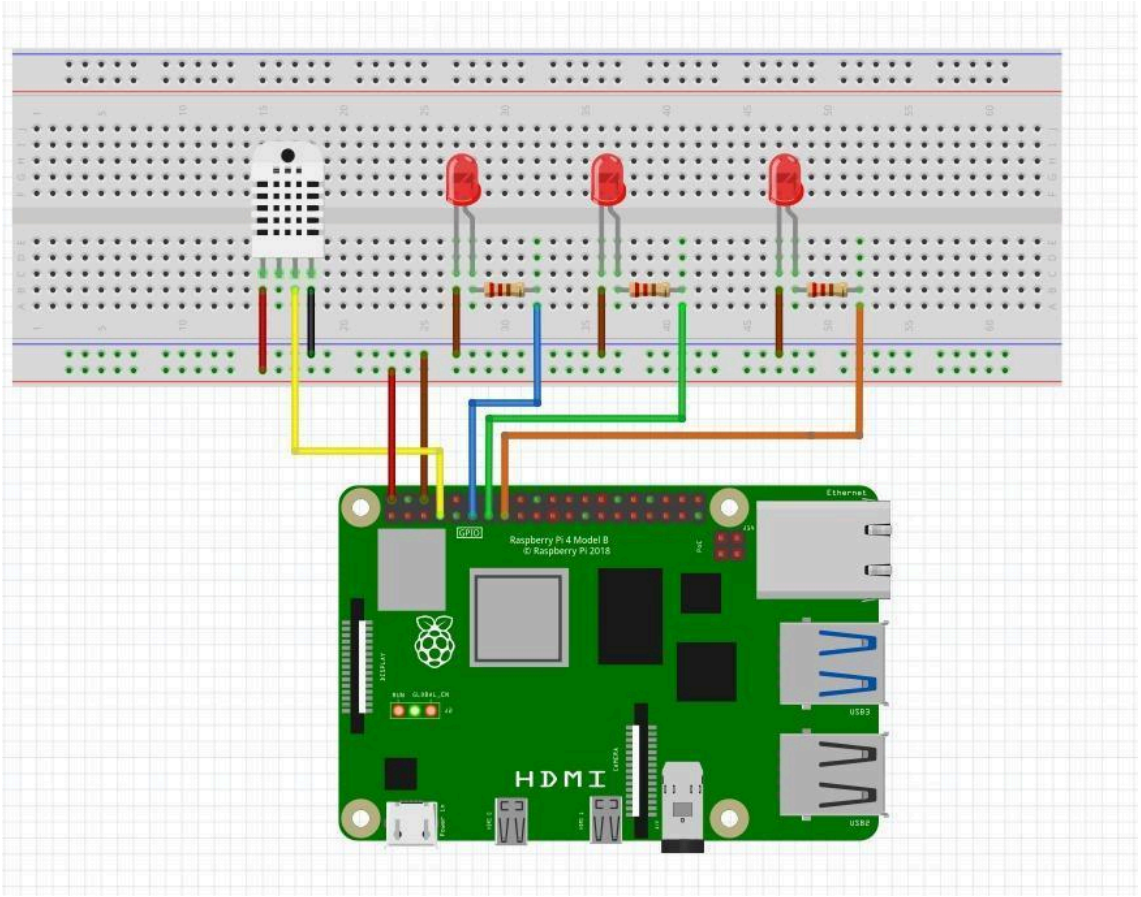
---

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.

## 11. Step-by-Step Implementation Plan:

### 11.1 Hardware Setup




**Circuit Connections**

- DHT11 Sensor:
  - VCC ⟶ 5V (Pin 2)
  - GND ⟶ Ground (Pin 6)
  - DATA ⟶ GPIO 4 (Pin 7)

- LEDs (each with 330Ω resistor):
  - Green LED ⟶ GPIO 17 (Pin 11) ⟶ 330Ω Resistor ⟶ Ground
  - Yellow LED ⟶ GPIO 27 (Pin 13) ⟶ 330Ω Resistor ⟶ Ground
  - Red LED ⟶ GPIO 22 (Pin 15) ⟶ 330Ω Resistor ⟶ Ground

  - Ground Connection: Use Pin 6 (Ground) as common ground for all LEDs and DHT11 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.

**Wiring Steps**

1. Connect DHT11 sensor: 5V, Ground, and GPIO 4
2. Place 3 LEDs on breadboard with 330Ω resistors
3. Connect LED positive legs to GPIO pins 17, 27, 22
4. Connect all LED negative legs (through 330Ω resistors) to common ground rail
5. Connect ground rail to Raspberry Pi Pin 6 (Ground)

⚠️ Important: Once you have assembled the circuit, double-check all connections before proceeding. Make sure each wire is properly connected to the correct pin - loose or incorrectly connected cables can cause unexpected behavior or damage to your components.

### 11.2 Software Setup

Now that the hardware is ready, it's time to set up the code that will bring your temperature monitoring system to life.

**1. Create Virtual Environment**

```
# Create project directory
mkdir temperature_monitor
cd temperature_monitor

# Create virtual environment
python3 -m venv venv

# Activate virtual environment
source venv/bin/actívate
```

**2. Install Dependencies**

```
# Install requirements
pip install -r requirements.txt
```

**3. Add the Code**

- Copy the temperature monitoring Python code to main.py
- Make sure the file is in your project directory

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.

**4. Run the System**

```
bash
# Make sure virtual environment is activated
source venv/bin/activate

# Run the monitoring system
python main.py
```

**5. Stop the System**

- Press Ctrl+C to stop the program safely
- All LEDs will turn off automatically

**System Behavior**

- **Green LED**: Normal operation
- **Yellow LED**: Temperature rising (>1°C in 20 seconds)
- **Red LED**: Critical warning (2 consecutive alerts) - Yellow turns off
- **Green LED (blinking)**: System starting up

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

```
import time
import board
import adafruit_dht
import lgpio

# LED pin definitions
LED_NORMAL = 17      # Green LED - normal operation status
LED_ALERT = 27       # Yellow LED - temperature rise alert
LED_WARNING = 22     # Red LED - consecutive temperature rise warning

# DHT11 temperature and humidity sensor on GPIO 4 (board.D4)
dht = adafruit_dht.DHT11(board.D4)

# Initialize GPIO for LEDs
h = lgpio.gpiochip_open(0)  # Open GPIO chip 0
for pin in [LED_NORMAL, LED_ALERT, LED_WARNING]:
    lgpio.gpio_claim_output(h, pin)  # Set pins as outputs

def led_on(pin):
    """Turn on LED connected to specified pin"""
    lgpio.gpio_write(h, pin, 1)

def led_off(pin):
    """Turn off LED connected to specified pin"""
    lgpio.gpio_write(h, pin, 0)

try:
    # Startup sequence: Turn on NORMAL LED for 1s, then blink twice and stay on
    led_on(LED_NORMAL)
    time.sleep(1)

    # Blink sequence (2 times) to indicate system initialization
    for _ in range(2):
        led_off(LED_NORMAL)
        time.sleep(0.3)
        led_on(LED_NORMAL)
        time.sleep(0.3)

    # Initialize variables for temperature monitoring
    temps = []                      # List to store last 5 temperature readings
```

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.

```
    alertas_consecutivos = 0          # Counter for consecutive temperature rise
alerts

    # Main monitoring loop
    while True:
        try:
            # Read temperature and humidity from DHT11 sensor
            temperatura = dht.temperature
            humedad = dht.humidity
            print(f"Temperatura: {temperatura}°C")
            print(f"Humedad: {humedad}%")
        except RuntimeError as e:
            # Handle sensor reading errors (common with DHT sensors)
            print("Error leyendo del sensor:", e.args[0])
            temperatura = None

        if temperatura is not None:
            # Add current temperature to the list
            temps.append(temperatura)

                # Keep only the last 5 measurements (25 seconds of data at 5s
intervals)
            if len(temps) > 5:
                temps.pop(0)  # Remove oldest measurement

            # Check for temperature rise pattern (need 5 measurements)
            if len(temps) == 5:
                # Calculate temperature delta between newest and oldest reading
                  delta = temps[-1] - temps[0]  # Current temp - temp from 20s
ago

                if delta > 1:  # Temperature rose more than 1°C in 20 seconds
                    alertas_consecutivos += 1

                    # Turn on ALERT LED and turn off NORMAL LED
                    led_on(LED_ALERT)
                    led_off(LED_NORMAL)

                     # If 2 consecutive alerts, turn on WARNING LED and turn off
ALERT LED
                    if alertas_consecutivos >= 2:
                          led_on(LED_WARNING)    # Critical warning - consecutive
temperature rises
```

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.

```
                        led_off(LED_ALERT)     # Turn off yellow LED when red is
active
                    else:
                        led_off(LED_WARNING)

                else:  # Temperature stable or decreasing
                    alertas_consecutivos = 0  # Reset consecutive alert counter

                    # Normal operation: only NORMAL LED on
                    led_on(LED_NORMAL)
                    led_off(LED_ALERT)
                    led_off(LED_WARNING)

            else:
                # Not enough measurements yet - show normal status
                led_on(LED_NORMAL)
                led_off(LED_ALERT)
                led_off(LED_WARNING)

        else:
                # Sensor reading error - keep NORMAL LED on to indicate standby
mode
            led_on(LED_NORMAL)
            led_off(LED_ALERT)
            led_off(LED_WARNING)

        # Wait 5 seconds before next reading
        time.sleep(5)

except KeyboardInterrupt:
    # Handle Ctrl+C gracefully
    print("Saliendo y apagando LEDs...")

finally:
    # Cleanup: turn off all LEDs and close GPIO
    for pin in [LED_NORMAL, LED_ALERT, LED_WARNING]:
        led_off(pin)
    lgpio.gpiochip_close(h)  # Close GPIO chip connection
```

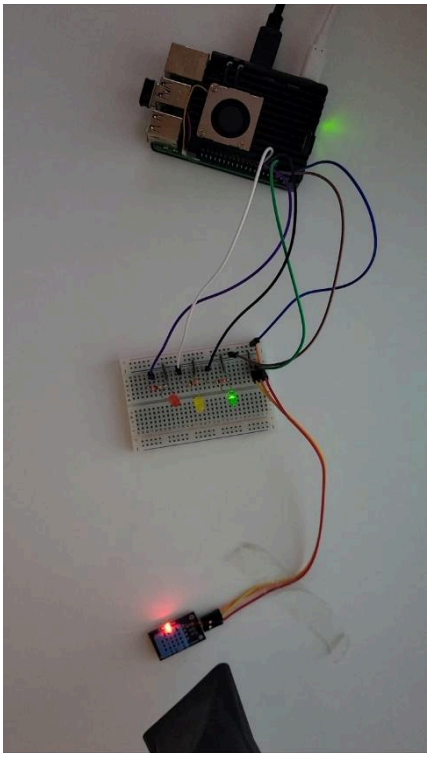
Co-funded by the European Union

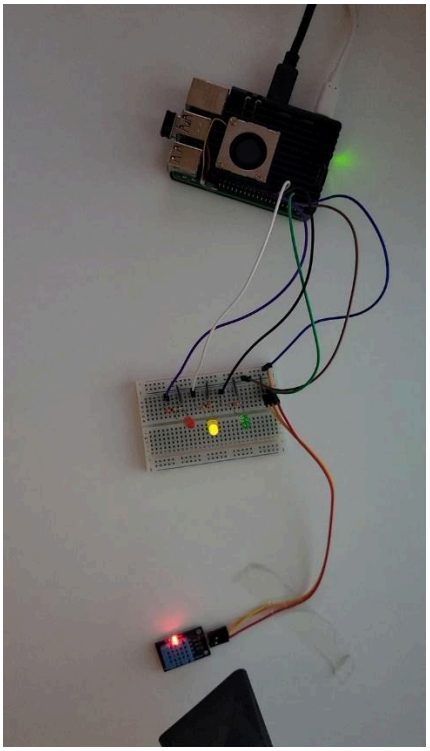
This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

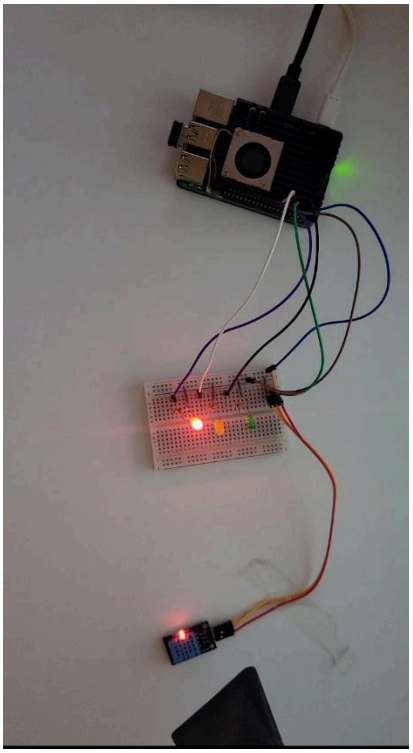
Now, you can run the program and see the result:

```
Temperature: 29.2°C
Humidity: 33%
Temperature: 29.2°C
Humidity: 33%
Temperature: 29.2°C
Humidity: 33%
Temperature: 36.6°C
Humidity: 30%
Temperature: 46.0°C
Humidity: 21%
Temperature: 47.0°C
Humidity: 16%
Temperature: 48.0°C
Humidity: 12%
Temperature: 48.0°C
Humidity: 10%
Temperature: 47.0°C
Humidity: 11%
Temperature: 46.0°C
Humidity: 11%
Temperature: 46.0°C
Humidity: 10%a
Temperature: 45.0°C
Humidity: 10%
```

```
Temperature: 29.1°C
Humidity: 29%
Temperature: 29.2°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.2°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.2°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.2°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
Temperature: 29.1°C
Humidity: 30%
```







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.

**12. Evaluation Criteria:**

- **Hardware Setup (35%)**: Correct wiring of sensor and LEDs; clean, stable connections; functional circuit.
- **Code Functionality (30%)**: Code runs without errors; temperature changes trigger correct LED responses.
- **System Understanding (20%)**: Clear explanation of how the system detects fire risk and how each component contributes.
- **Innovation & Improvement (15%)**: Creative suggestions (e.g., buzzer alert, data logging, cloud integration).

---

**13. Languages Available:**

---

**14. Contributors:**

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.