# REAL TIME ENVIRONMENT MONITORING AND AIR QUALITY SENSING

**TEAM MEMBERS:**

**DHARSHINI S S**

**ANGELIN PRESENCIA P**

**GAYATHRI D**

**ANANYA C V**

**AIM:**

To develop a system that monitors environmental conditions and air quality in real-time for better health and safety.

It measures key parameters like temperature, humidity, gases, and particulate matter.

The system provides instant data to create awareness and support timely actions.

**COMPONENTS REQUIRED:**

- Raspberry Pi Pico W(Microcontroller)
- DHT11 sensor
- MQ-135 Gas sensor
- Dust Sensor (SDS011)
- OLED display
- Buzzer
- LED

**RASPBERRY PI PICO W:**

- A low-cost microcontroller board with built-in Wi-Fi (2.4 GHz, IEEE 802.11n).
- Operates at 133 MHz dual-core ARM Cortex-M0+ processor with 264 KB RAM.
- Wi-Fi range is typically up to 30–50 meters indoors and up to 100 meters outdoors.

**DHT11 SENSOR:**

- Measures temperature (0°C to 50°C for DHT11).
- Measures humidity (20% to 90% for DHT11).
- Provides calibrated digital output via a single data pin.

**MQ-135 GAS SENSOR:**

- Detects gases like $CO_2$, $NH_3$, NOx, benzene, and alcohol.
- Operating range: 10 ppm to 1000 ppm depending on gas type.
- Analog output changes with gas concentration, can be read via Raspberry Pi ADC.

**DUST SENSOR(SDS011):**

- Measures PM2.5 and PM10 particles in the air.
- Detection range: 0.3 to 10 microns particle size.
- Provides digital UART output for accurate real-time monitoring.

**OLED DISPLAY:**

- Displays sensor readings in real-time.
- Low power consumption with 128x64 pixel resolution.
- Works on 3.3V–5V and communicates via I2C pins.

**BUZZER:**

- Provides audio alert when air quality crosses safe limit.
- Works in 3.3V–5V range, can be controlled via GPIO.
- Produces sound frequency typically between 2 kHz – 4 kHz.

**LED:**

- Visual indicator for air quality status (Green = Good, Red = Bad).
- Forward voltage: ~2V (red), ~3V (green/blue).
- Current rating: 10–20 mA.

**AVAILABLE COMPONENTS:**

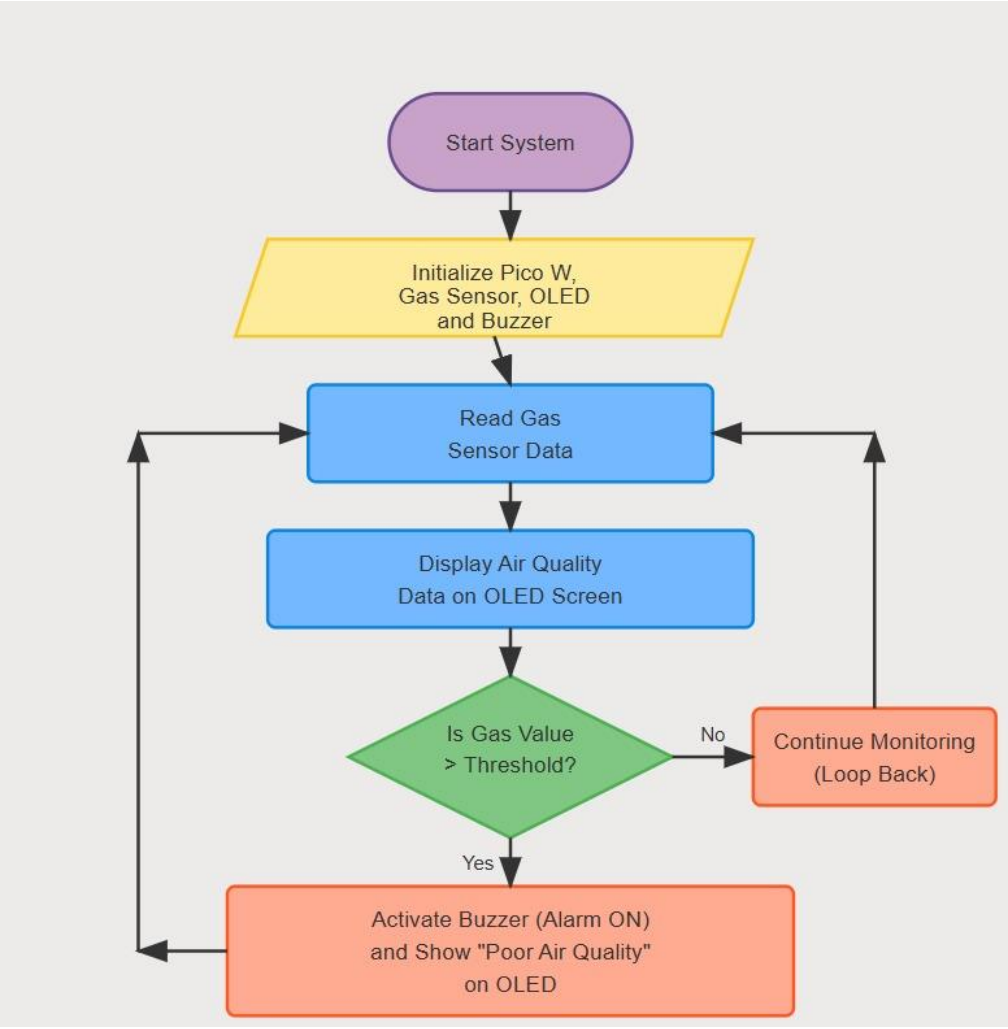
- Raspberry Pi Pico W
- Gas sensor
- OLED display
- Buzzer

**PIN CONFIGURATION:**

| **COMPONENTS** | **RASPBERRY PI PICO W** |
|---|---|
| GAS SENSOR(MQ-135):<br>VCC<br>GND<br>AO | <br>3.3V<br>GND<br>GP26(ADC0) |
| OLED DISPLAY<br>VCC<br>GND | <br>3.3V<br>GND |

| | |
|---|---|
| SDA<br>SCL | GP0<br>GP1 |
| BUZZER:<br>VCC<br>GND | <br>GP15<br>GND |

**FLOWCHART:**




**PROCEDURE:**

- **Setup Components** → Connect Raspberry Pi Pico W with Gas Sensor (ADC pin), OLED Display (I2C pins), and Buzzer (GPIO).
- **Initialize System** → Load MicroPython firmware on Pico W and set up OLED, sensor, and buzzer in the code.
- **Read Sensor Data** → Continuously read air quality values from the gas sensor (ADC readings).

- **Display & Check Threshold** → Show real-time values on OLED and compare with a preset threshold.
- **Trigger Alarm** → If value exceeds threshold, activate buzzer and display "Poor Air Quality" warning on OLED.

**PROGRAM:**

```
from machine import Pin, ADC, I2C
import ssd1306
import time
# MQ-135 Sensor setup (Analog input on GP26/ADC0)
gas_sensor = ADC(Pin(26))
# I2C setup for OLED (try I2C0 with SDA=GP0, SCL=GP1)
i2c = I2C(0, scl=Pin(1), sda=Pin(0), freq=400000)
# Scan I2C devices
devices = i2c.scan()
if not devices:
    print("❌ No I2C device found. Check wiring (SDA/SCL).")
    while True:
        time.sleep(1)
print("✅ I2C devices found:", [hex(d) for d in devices])
# Use first detected device as OLED
oled_addr = devices[0]
oled = ssd1306.SSD1306_I2C(128, 64, i2c, addr=oled_addr)
# Buzzer setup (GP15)
buzzer = Pin(15, Pin.OUT)
# Threshold for gas detection (adjust as per calibration)
GAS_THRESHOLD = 28000
while True:
```

```
# Read gas sensor value (0 - 65535 in 16-bit scale)
gas_value = gas_sensor.read_u16()
# Clear display
oled.fill(0)
oled.text("Gas Detector", 0, 0)
oled.text("Value: {}".format(gas_value), 0, 20)
# Check gas level
if gas_value > GAS_THRESHOLD:
    oled.text("ALERT!", 0, 40)
    buzzer.value(1)  # Turn buzzer ON
else:
    oled.text("Safe", 0, 40)
    buzzer.value(0)  # Turn buzzer OFF
# Show on OLED
oled.show()
# Small delay
time.sleep(0.5)
```

**EXECUTION:**

**RESULT:**

The system successfully monitored air quality in real time using a gas sensor with Raspberry Pi Pico W. Live readings were displayed on the OLED screen for user awareness. When pollutant levels crossed the safe threshold, the buzzer alerted with a warning.