PROJECT TITLE: AIR QUALITY MONITORING

Phase 4: Development Part 2

INTRODUCTION:

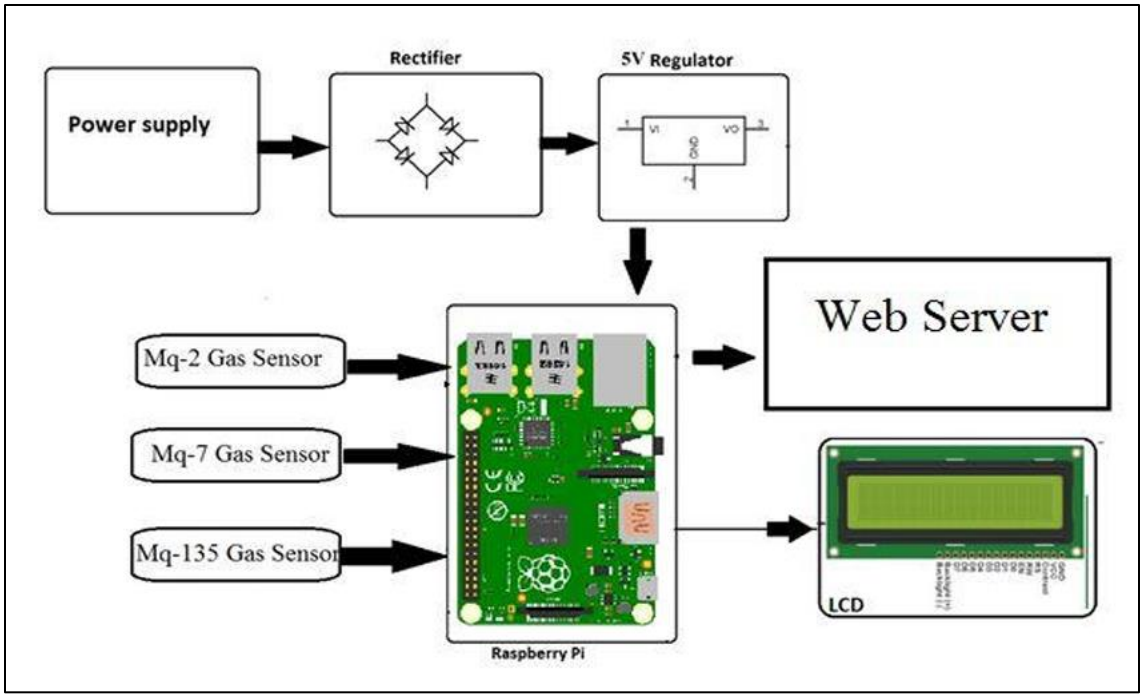
In this day and age, air pollution is certainly an issue of significance. To keep it in control and provide a better quality of life for all, air quality should be monitor and control.

In this project we can measure air quality by using **"Raspberry pi",** temperature and humidity sensor, gas sensor, dust sensor. Sensors have been used to detect the presence of harmful gases/compounds, which are continually transmitted to a controller.

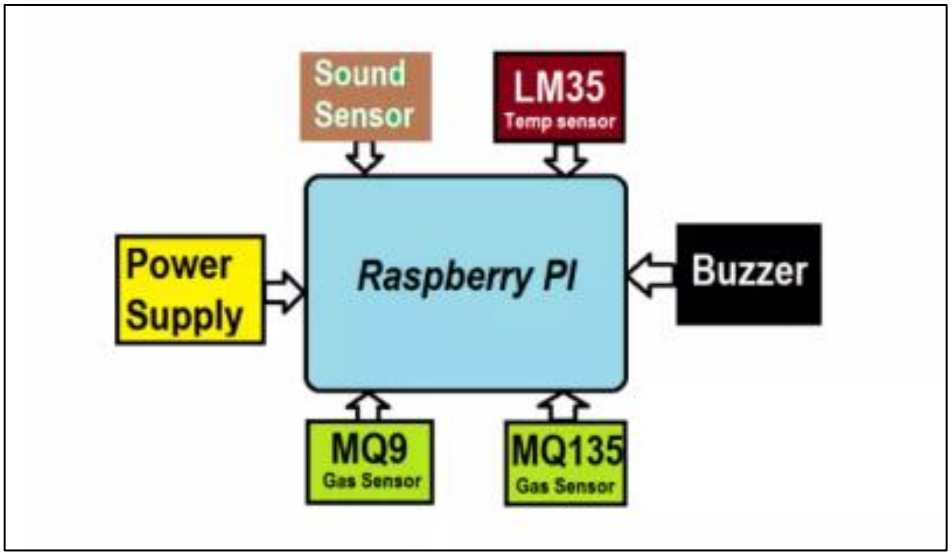
Air quality monitoring and controlling system is proposed in this project, which enable us to monitor and check real time quality or the air temperature, humidity in specific region through IOT. In this project we can also control the quality of air pollution by using air filtering which absorb the carbon in the air and produce a fresh air.

BLOCK DIAGRAM:

The Proposed model of the system is as follows. Figure shows how the whole system will work. The block diagram of the system is showing that for a particular area selected how will it work. The device will be set up to take the environmental data and there will be a base standard value. The device will collect data and based on the set values it will show the output.



The system is controlled using Raspberry Pi 3b+ which is interfaced with gas sensors like MQ135, MQ7, MQ4 and also the Dust particle sensor. The DHT11 sensor updates the Temperature and Humidity values on real time basis. The complete monitored values from the sensors are updated on the cloud i.e., Smart Core. The entire data is stored in the cloud and the values will be updated on real time basis. The data updating and the flow its works is as shown in the below flow diagram



PLATFORM REQUIRED:

HARDWARE Components Used:

- **Raspberry Pi 3b+:**

The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and PoE capability via a separate PoE HAT The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market. The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

Specifications:

Processor: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz

Memory: 1GB LPDDR2 SDRAM

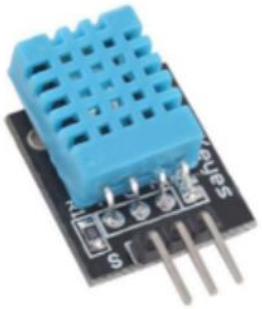
Connectivity:

- 4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps)
- 4 × USB 2.0 portsSpecifications

DHT11:

- **DHT11 Specifications:**
  - Operating Voltage: 3.5V to 5.5V

- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%



- **MQ135 Air quality Sensor:**

Features:

- High Sensitivity
- High sensitivity to Ammonia, Sulfide and Benze
- Stable and Long Life
- Detection Range: 10 – 300 ppm NH3, 10 – 1000 ppm Benzene, 10 – 300 Alcohol
- Heater Voltage: 5.0V
- Dimensions: 18mm Diameter, 17mm High excluding pins, Pins – 6mm High
- Long life and low cost

- **DUST SENSOR**

Dust Sensor is a simple air monitoring module with onboard Sharp GP2Y1010AU0F. It is capable of detecting fine particle larger than 0.8µm in diameter, even like the cigarette smoke. Analog voltage output of the 6sensor is linear with dust density. The module has embedded voltage boost circuit to support wide range of power supply.

- **Air Quality Sensor:** Choose a sensor like the SDS011, CCS811, or Nova PM SDS012 for measuring particulate matter (PM2.5 and PM10) and other pollutants.
- **Temperature and Humidity Sensor:** DHT11 or DHT22 sensors are commonly used for monitoring temperature and humidity levels.

- **Power Supply:** A stable power supply for the Raspberry Pi and the sensors.
- **MicroSD Card:** To install the operating system and software.
- **Wiring and Breadboard:** For connecting sensors to the Raspberry Pi GPIO pins.
- **Enclosure:** To protect your Raspberry Pi and sensors from environmental factors.

Software Requirements:

- **Raspberry Pi OS:** Install a Raspberry Pi-compatible operating system on the microSD card. Raspbian is the most popular choice.
- **Python:** You'll need Python installed on your Raspberry Pi to run the sensor scripts.
- **Sensor Libraries**: Depending on your sensors, you might need specific Python libraries to communicate with them. For example, `sds011_python` for SDS011 sensor.
- **Database**: Choose a database system like MySQL, SQLite, or MongoDB to store the sensor data.

- **Web Server**: Flask or Django can be used to create a web interface for displaying real-time and historical data.
- **Cron Jobs**: If you want to schedule regular data readings, use cron jobs in Linux to automate the process.
- **Data Visualization Tools**: You can use tools like Grafana or Matplotlib (for Python) to create graphs and visualize the collected data.
- **Remote Monitoring (Optional):** If you want to monitor the air quality remotely, you might need services like Ngrok for secure tunneling or setting up a VPN for remote access.

WORKING STEPS:

Creating an Air Quality Monitoring Project with Raspberry Pi involves connecting sensors, reading data, and displaying it. Here's a basic example using a Raspberry Pi, an MQ-135 gas sensor for air quality measurement, and Python programming language. For this project, you'll need:

1. Raspberry Pi (with Raspbian OS installed)

2. MQ-135 Gas Sensor

3. Female-to-Female Jumper Wires

Step 1: Connect the MQ-135 Sensor to Raspberry Pi

- ✓ Connect the VCC pin of the MQ-135 sensor to the 5V pin on the Raspberry Pi.

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V at 100mA	AC or DC
VH	Heating voltage	5V at 100mA	AC or DC
RL	Load resistance	Can adjust	
RH	High resistance	33Ω±5%	Room temperature
PH	Heating compositions	Less than 800mw	
Rs	Sensing resistance	3KΩ-30KΩ	

Table: 1 Standard work conditions for MQ2 gas sensor

- ✓ Connect the GND pin of the MQ-135 sensor to the GND pin on the Raspberry Pi.
- ✓ Connect the AOUT pin of the MQ-135 sensor to any available GPIO pin on the Raspberry Pi (e.g., GPIO17).

Step 2: Install Required Python Libraries

Open a terminal window on your Raspberry Pi and install the required libraries:

```
bash
pip install RPi.GPIO
pip install Adafruit_GPIO
```

Step 3: Write Python Code for Air Quality Monitoring

Create a Python script (for example, `air_quality_monitor.py`) and add the following code:

```python
import RPi.GPIO as GPIO
import time

# Set GPIO mode to BCM
GPIO.setmode(GPIO.BCM)

# Define the GPIO pin connected to the AOUT pin of the MQ-135
    sensor
MQ135_PIN = 17

def read_gas_sensor(pin):
    # Read analog value from the MQ-135 sensor
    GPIO.setup(pin, GPIO.IN)
    total_value = 0
    for _ in range(100):
        total_value += GPIO.input(pin)
    return total_value

try:
    while True:
        gas_value = read_gas_sensor(MQ135_PIN)
        print("Air Quality Value:", gas_value)
        time.sleep(1)  # Delay for 1 second before the next
            reading

except KeyboardInterrupt:
    print("Monitoring Stopped by the User")
    GPIO.cleanup()

finally:
    GPIO.cleanup()
```

Step 4: Run the Python Script

In the terminal, navigate to the directory where your `air_quality_monitor.py` script is located and run the script:

```
python air_quality_monitor.py
```

This script will continuously read the analog value from the MQ-135 sensor connected to GPIO pin 17 and print it to the console.

Python Code for Air Quality Monitoring System:

This system utilizes the MQ-135 air quality sensor to measure the air quality index (AQI) based on the concentration of harmful gases.

Components Needed:

1. Raspberry Pi (with Raspbian OS installed)

2. MQ-135 Air Quality Sensor

3. Analog-to-Digital Converter (ADC) (e.g., MCP3008)

4. Jumper wires

```
import RPi.GPIO as GPIO
import time
import spidev

# Set up GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# Define MQ-135 sensor pin connected to ADC
MQ135_PIN = 0  # Analog pin 0

# Create SPI object for ADC communication
spi = spidev.SpiDev()
spi.open(0, 0)
spi.max_speed_hz = 1000000  # Set SPI speed

# Function to read sensor data from ADC
def read_adc(channel):
    adc_data = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_value = ((adc_data[1] & 3) << 8) + adc_data[2]
```

```
    return adc_value
# Function to calculate air quality index (AQI)
def calculate_aqi(adc_value):
    voltage = adc_value * (3.3 / 1023)  # Convert ADC value to
        voltage (3.3V)
    ppm = (voltage - 0.1) * 200  # Convert voltage to ppm
        (parts per million)
    return ppm
def measure_air_quality():
    try:
        while True:
            adc_value = read_adc(MQ135_PIN)  # Read ADC value
                from sensor
            aqi = calculate_aqi(adc_value)  # Calculate AQI
            print(f"Air Quality Index (AQI): {aqi}")
            time.sleep(2)
    except KeyboardInterrupt:
        GPIO.cleanup()
measure_air_quality()
```

Explanation:

- **GPIO Setup**: The code starts by importing necessary libraries and setting up GPIO using the Broadcom SOC channel numbering (BCM).
- **SPI Setup:** It creates an SPI object using the `spidev` library to communicate with the MCP3008 ADC.
- **Reading ADC Data**: The `read_adc(channel)` function is defined to read analog data from the specified channel (in this case, channel 0) of the ADC. It uses SPI communication to retrieve the ADC value.
- **Calculating Air Quality Index (AQI):** The `calculate_aqi(adc_value)` function takes the ADC value as input and converts it into voltage. Then, it calculates the AQI based on the voltage. The specific conversion formula may

vary based on the characteristics of your MQ-135 sensor. Adjust the formula as needed for accurate results.

- **Main Function**: The `measure_air_quality()` function is the main part of the code. It runs in an infinite loop, reading the ADC value, calculating the AQI using the `calculate_aqi()` function, and printing the result. The program waits for 2 seconds between readings.
- **Exception Handling**: The code includes a KeyboardInterrupt exception to handle the user interrupting the program (Ctrl + C). It cleans up the GPIO pins using `GPIO.cleanup()` before exiting.

Make sure to connect the MQ-135 sensor and the ADC (MCP3008 or similar) correctly to the Raspberry Pi's GPIO pins. Additionally, calibrate the AQI calculation based on your specific sensor's characteristics for accurate air quality measurements.

CONCLUSION:

Using different sensors connected to a Raspberry Pi module, environmental parameters like gas density, humidity, temperature and soil moisture are observed and recorded at frequent intervals. We can also control the quality of air pollution by using air filter, which absorb the carbon in the air and produce a fresh air.

A system which can monitor the leakage of toxic gases and hence the level of pollution using Raspberry-Pi and IoT is proposed which can prevent fatal accidents. By the use of MQ135/6/7 gas sensors the poisonous gases can be sensed and alert can be given to save the life of people. Raspberry-Pi serves as the heart of this module which controls the entire process. It supports the new technology and effectively supports the healthy life concept.