IOT BASED AIR QUALITY MONITORING

PHASE 5

PROGRAM CODE:

ARDUINO CODE:

```
#include <SoftwareSerial.h>
SoftwareSerial esp8266(2, 3); // RX, TX (connect to ESP8266)

const int gasSensorPin = A0; // Analog input for gas sensor

void setup() {
  Serial.begin(9600);
  esp8266.begin(9600);
}

void loop() {
  int sensorValue = analogRead(gasSensorPin);
  float ppm = calculatePPM(sensorValue); // You'll need to implement this function
  sendDataToServer(ppm);
  delay(5000); // Send data every 5 seconds
}

float calculatePPM(int sensorValue) {
  // Implement the conversion from sensor value to PPM here
  // This depends on the sensor model and its datasheet.
  // Refer to the datasheet for your gas sensor.
}
```

```
void sendDataToServer(float ppm) {
  esp8266.print("GET /update?api_key=YOUR_API_KEY&field1=");
  esp8266.print(ppm);
  esp8266.println(" HTTP/1.0");
  esp8266.println("Host: api.thingspeak.com");
  esp8266.println("Content-Type: application/x-www-form-urlencoded");
  esp8266.println("Connection: close");
  esp8266.println();
  delay(500);
}
```

PYTHON CODE:

```
import requests
import matplotlib.pyplot as plt
from datetime import datetime

API_KEY = 'YOUR_THINGSPEAK_API_KEY'
CHANNEL_ID = 'YOUR_THINGSPEAK_CHANNEL_ID'

def read_data_from_thingspeak():
    URL = f'https://api.thingspeak.com/channels/{CHANNEL_ID}/feeds.json'
    params = {'api_key': API_KEY, 'results': 10}
    response = requests.get(URL, params=params)

    if response.status_code == 200:
        data = response.json()
        timestamps = []
        ppm_values = []
```

```
        for entry in data['feeds']:
            timestamp = entry['created_at']
            ppm = entry['field1']
            timestamps.append(timestamp)
            ppm_values.append(float(ppm))
        return timestamps, ppm_values
    else:
        print("Failed to fetch data from ThingSpeak.")
        return [], []

def plot_data(timestamps, ppm_values):
    plt.figure(figsize=(10, 5))
    plt.plot(timestamps, ppm_values, marker='o', linestyle='-')
    plt.title('Air Quality Monitoring')
    plt.xlabel('Timestamp')
    plt.ylabel('PPM')
    plt.xticks(rotation=45)
    plt.grid(True)
    plt.tight_layout()
    plt.show()

if __name__ == '__main__':
    timestamps, ppm_values = read_data_from_thingspeak()
    plot_data(timestamps, ppm_values)
```