AIR POLLUTION MONITORING

PHASE 2: INNOVATION

HARDWARE COMPONENTS:

Air Quality Sensors:

- Air quality sensors are the core components for measuring different air pollutants. Common sensors include:
  - MQ series sensors (e.g., MQ-7 for CO, MQ-135 for CO2)
  - Particulate matter (PM) sensors (e.g., SDS011, PMS5003)
  - Ozone (O3) sensors
  - Nitrogen dioxide (NO2) sensors

Microcontroller:

- NodeMCU (ESP8266) or Arduino boards (e.g., Arduino Uno, Arduino Nano) to interface with sensors, process data, and send it to a central server or cloud.

Communication Module:

- WiFi module (e.g., ESP8266, ESP32) to connect the microcontroller to the internet, enabling data transmission to the server or cloud.

Power Supply:

- Power source (e.g., battery, power adapter) to power the sensors, microcontroller, and communication modules.

Breadboard and Jumper Wires:

- Breadboards for prototyping and jumper wires to connect components on the breadboard.

LCD Display:

- LCD module (e.g., 16x2 character LCD) to display real-time air quality data.

Enclosure:

- A protective housing to house the components and protect them from environmental conditions (optional, especially for outdoor installations).

Other Components:

- Resistors, capacitors, and other electronic components for interfacing sensors with the microcontroller.

Optional Components:

- GPS module: To record location information along with air quality data.
- Temperature and humidity sensors: To measure environmental conditions.
- Solar panels and related components: For powering the system using solar energy (useful for remote or outdoor installations).

SOFTWARE COMPONENTS:

Arduino IDE:

- The Arduino Integrated Development Environment (IDE) is commonly used for programming the microcontroller (e.g., NodeMCU, Arduino boards). It allows writing, compiling, and uploading firmware code to the microcontroller.

CODE:

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

char auth[] = "YOUR_BLYNK_AUTH_TOKEN”; // Replace with your Blynk auth token
char ssid [] = "YOUR_WIFI_SSID”;
char pass [] = "YOUR_WIFI_PASSWORD";

const int coSensorPin = A0;   // Analog pin for CO sensor (MQ-7)
const int co2SensorPin = A1;  // Analog pin for CO2 sensor (MQ-135)

LiquidCrystal_I2C lcd(0x27, 16, 2);  // Address for 16x2 LCD
```

```
void setup() {
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
  lcd.begin();
  lcd.backlight();
}

void loop() {
  Blynk.run();

  // Read sensor values
  int coSensorValue = analogRead(coSensorPin);
  int co2SensorValue = analogRead(co2SensorPin);

  // Convert sensor values to air quality levels
  float coLevel = map(coSensorValue, 0, 1023, 0, 100);  // Assuming a linear mapping
  float co2Level = map(co2SensorValue, 0, 1023, 0, 100);  // Assuming a linear mapping

  // Display pollution levels on the Blynk app
  Blynk.virtualWrite(V0, coLevel);   // Virtual pin V0 for CO level
  Blynk.virtualWrite(V1, co2Level);  // Virtual pin V1 for CO2 level

  // Display pollution levels on the LCD
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("CO: ");
  lcd.print(coLevel);
  lcd.print("%");

  lcd.setCursor(0, 1);
  lcd.print("CO2: ");
  lcd.print(co2Level);
  lcd.print("%");
```

```
  // Check if pollution level is below 60% and send a notification
  if (coLevel < 60 || co2Level < 60) {
    Blynk.notify("Air pollution below 60%! Action needed.");
  }

  delay(5000);  // Adjust delay based on your sampling frequency
}
```