**SMART ENVIRONMENTAL MONITORING SYSTEM**

**INTRODUCTION:**

In Phase 4 of our project, we continue our pursuit of an affordable indoor air quality monitoring system, leveraging the ESP32 microcontroller and the precision of the BME680 sensor. The ESP32 serves as the backbone, enabling real-time data collection and transmission, while the BME680 ensures accuracy in measuring key air quality parameters. To provide users with an intuitive experience, we've integrated the Blynk App as our user interface, allowing easy access to vital information and notifications. With a focus on cost-efficiency and accessibility, our overarching mission remains to democratize indoor air quality monitoring.

The IAQ scale ranges from 0 (clean air) to 500 (heavily polluted air). During operation, the algorithms automatically calibrate and adapt themselves to the typical environments where the sensor is operated (e.g., home, workplace, inside a car, etc.).This automatic background calibration ensures that users experience consistent IAQ performance. The calibration process considers the recent measurement history (typ. up to four days) to ensure that IAQ ~ 25 corresponds to “typical good” air and IAQ ~ 250 indicates “typical polluted” air.

Key parameters for gas sensor

1. Response time (𝜏33−63%) < 1 s (for new sensors)
2. Power consumption < 0.1 mA in ultra-low power mode
3. Output data processing direct indoor air quality (IAQ) index output

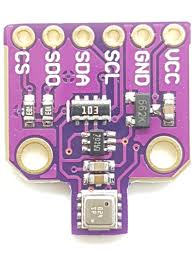
Key parameters for humidity sensor

1. Response time (𝜏0−63%) ~8 s
2. Accuracy tolerance ±3% r.H.
3. Hysteresis ±1.5% r.H.

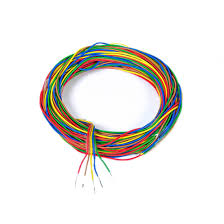
Key parameters for pressure sensor

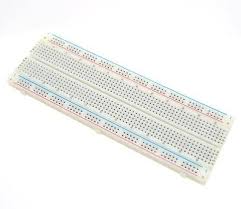
1. RMS Noise 0.12 Pa, equiv. to 1.7 cm
2. Offset temperature coefficient ±1.3 Pa/K, equiv. to ±10.9 cm at 1 °C temperature change

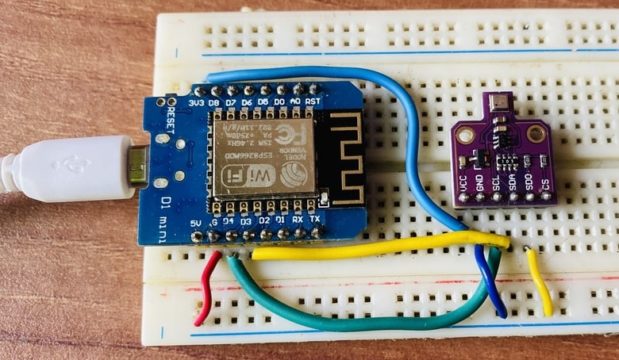
**COMPONENTS AND ARRANGEMENT:**











**WOKWI SOURCE CODE:**

#define BLYNK\_TEMPLATE\_ID "TMPLwToQUqRw"

#define BLYNK\_TEMPLATE\_NAME "Indoor Air Quality Monitoring"

#define BLYNK\_AUTH\_TOKEN "C8Y7TGFr54QF8pdfQ5dZsdfhhSdiQBFLj8mYe"

#define BLYNK\_PRINT Serial

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <DHT.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 16, 2);

byte degree\_symbol[8] =

{

0b00111,

0b00101,

0b00111,

0b00000,

0b00000,

0b00000,

0b00000,

0b00000

};

char auth[] = BLYNK\_AUTH\_TOKEN;

char ssid[] = "WiFi Username"; // type your wifi name

char pass[] = "WiFi Password"; // type your wifi password

BlynkTimer timer;

int gas = 32;

int sensorThreshold = 100;

#define DHTPIN 2 //Connect Out pin to D2 in NODE MCU

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void sendSensor()

{

float h = dht.readHumidity();

float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

if (isnan(h) || isnan(t)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

int analogSensor = analogRead(gas);

Blynk.virtualWrite(V2, analogSensor);

Serial.print("Gas Value: ");

Serial.println(analogSensor);

Blynk.virtualWrite(V0, t);

Blynk.virtualWrite(V1, h);

Serial.print("Temperature : ");

Serial.print(t);

Serial.print(" Humidity : ");

Serial.println(h);

}

void setup()

{

Serial.begin(115200);

//pinMode(gas, INPUT);

Blynk.begin(auth, ssid, pass);

dht.begin();

timer.setInterval(30000L, sendSensor);

//Wire.begin();

lcd.begin();

// lcd.backlight();

// lcd.clear();

lcd.setCursor(3,0);

lcd.print("Air Quality");

lcd.setCursor(3,1);

lcd.print("Monitoring");

delay(2000);

lcd.clear();

}

void loop()

{

Blynk.run();

timer.run();

float h = dht.readHumidity();

float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

int gasValue = analogRead(gas);

lcd.setCursor(0,0);

lcd.print("Temperature ");

lcd.setCursor(0,1);

lcd.print(t);

lcd.setCursor(6,1);

lcd.write(1);

lcd.createChar(1, degree\_symbol);

lcd.setCursor(7,1);

lcd.print("C");

delay(4000);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Humidity ");

lcd.print(h);

lcd.print("%");

delay(4000);

lcd.clear();

//lcd.setCursor(0,0);

// lcd.print(gasValue);

// lcd.clear();

Serial.println("Gas Value");

Serial.println(gasValue);

if(gasValue<1200)

{

lcd.setCursor(0,0);

lcd.print("Gas Value: ");

lcd.print(gasValue);

lcd.setCursor(0, 1);

lcd.print("Fresh Air");

Serial.println("Fresh Air");

delay(4000);

lcd.clear();

}

else if(gasValue>1200)

{

lcd.setCursor(0,0);

lcd.print(gasValue);

lcd.setCursor(0, 1);

lcd.print("Bad Air");

Serial.println("Bad Air");

delay(4000);

lcd.clear();

}

if(gasValue > 1200){

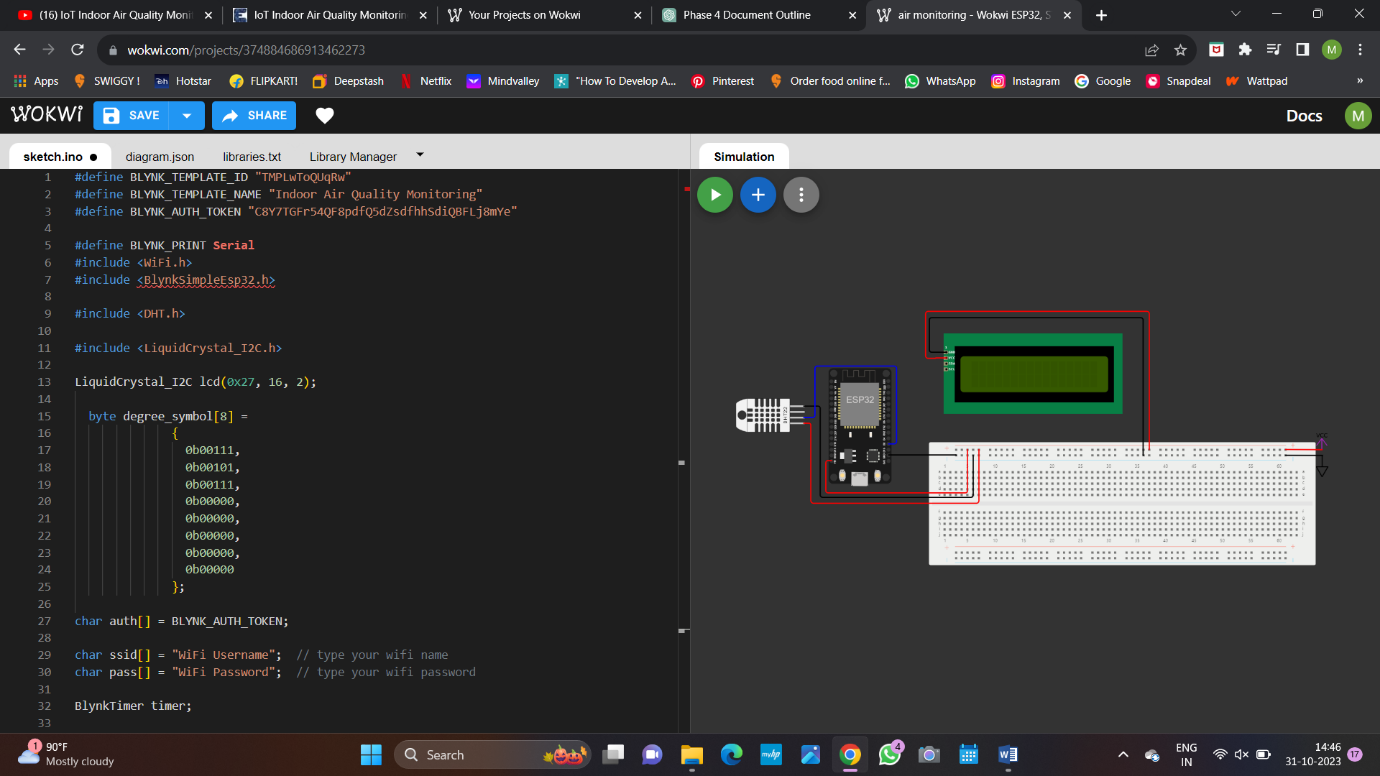
//Blynk.email("shameer50@gmail.com", "Alert", "Bad Air!");

Blynk.logEvent("pollution\_alert","Bad Air");

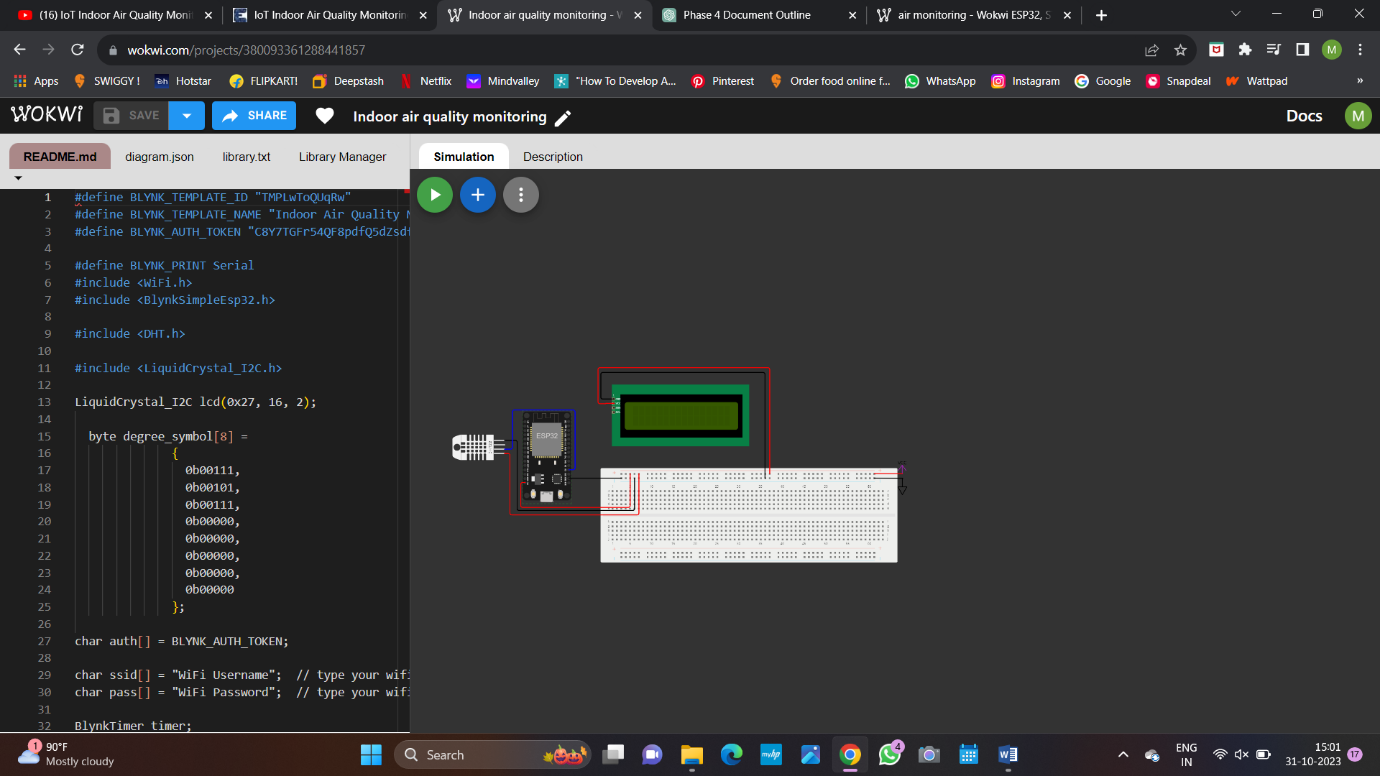
}

}

**WOKWI SIMULATION:**



WOKWI PROJECT LINK: <https://wokwi.com/projects/380093361288441857>



**CONCLUSION:**

This document highlights the significant progress made in Phase 4 of our indoor air quality monitoring project. Leveraging the ESP32 microcontroller and the precision of the BME680 sensor, we've successfully developed a monitoring system aimed at providing accurate and cost-effective indoor air quality data. In addition to these hardware components, the Wokwi simulation played a crucial role in testing and fine-tuning our system in a virtual environment before transitioning to real hardware. The careful arrangement of these components ensures optimal performance and reliable data collection. This phase marks a significant step toward achieving our primary goal of delivering an affordable and precise indoor air quality monitoring solution, empowering individuals and organizations to make informed decisions for healthier indoor environments.