AIR QUALITY MONITORING USING TINKERCAD

PHASE 5 : PROJECT DOCUMENTATION &SUBMISSION

Creating an air quality monitoring project using Arduino and Tinkercad is a great way to simulate and prototype your system.

PROJECT OBJECTIVE :

1. **Real-time Air Quality Monitoring** :

   Develop a system that can continuously monitor key air quality parameters such as particulate matter (PM2.5 and PM10), carbon dioxide (CO2), carbon monoxide (CO), ozone (O3), nitrogen dioxide (NO2), temperature, humidity, and air pressure.

2. **Data Collection and Logging** :

   Create a data acquisition system to collect and log air quality data at regular intervals, ensuring data integrity and accuracy.

3. **Visualization and User Interface** :

   Develop a user-friendly interface for visualizing the collected data in real-time, allowing users to easily interpret air quality information.

4. **Alerts and Notifications** :

   Implement an alert system that can notify users when air quality levels exceed predefined thresholds. These alerts can be in the form of visual indicators or notifications sent to a user's device.

5. **Historical Data Storage** :

Store historical air quality data for analysis, trend identification, and comparison over time.

6. **Data Analysis and Reporting** :

Utilize data analysis techniques to identify trends, correlations, and potential sources of air quality issues. Generate periodic reports or visualizations to help users understand long-term air quality patterns.

7. **Remote Access** :

Enable remote access to the system, allowing users to monitor air quality from anywhere using a web or mobile application.

8. **Environmental Impact Assessment** :

Use the collected data to assess the environmental impact of air pollution in a specific area and track the effectiveness of any implemented pollution control measures.

9. Integration with IoT and Sensors :

If possible, incorporate Internet of Things (IoT) devices and various sensors to expand the system's functionality and gather additional environmental data.

10. Educational Outreach :

Consider the educational aspect of the project, making it a valuable tool for teaching and raising awareness about air quality and its importance.

COMPONENTS REQUIRED :

- 1. Arduino (e.g., Arduino Uno or Arduino Nano)
- 2. Air quality sensors (e.g., MQ-135 for gases, SDS011 for particulate matter)

- 3. Breadboard and jumper wires
- 4. OLED display (optional)
- 5. Computer with internet access for Tinkercad

PROCEDURE :

1. Log in to Tinkercad.

2. Click "Create new circuit."

3. Drag and drop the components onto the virtual breadboard.

    - Place an Arduino board.
    - Add your air quality sensor(s).
    - If using an OLED display, add it as well.

4. Connect the components with jumper wires.

    - Connect the sensor's signal pins to appropriate Arduino pins.
    - Connect the power (VCC and GND) pins of the sensor(s) to the Arduino's 5V and GND, respectively.

5. Write Arduino code to read data from the air quality sensors and display it on the OLED screen (if using one).

6. Include code for sensor calibration and data conversion.

7. Make sure to include any libraries required for your components (e.g., Adafruit SSD1306 for OLED displays).

8. Upload the code to your virtual Arduino in Tinkercad .

9. Click the "Start Simulation" button in Tinkercad to run the simulation.

10. Observe the virtual components' behavior and data readings in the simulation.

PROGRAM :

```
/*
 Air Monitoring System
```

//
/////////////////////////////
The project's objective is to design a simple system that is used for simulating an Air Monitoring System using Arduino Uno Board, Gas Sensor, Buzzer, LCD display, LEDs.
//
/////////////////////////////

This sketch is used for Air-Monitoring.
The LCD prints "Harmful Gas is Detected", the Buzzer will generate an alarm sound, and the RED LED will be turned on when the Gas Sensor detects a harmful gas in its reading range.
The Buzzer will be turned off and the GREEN LED will be turned on when the Gas Sensor does not detect any harmful gases in its reading range.

The Circuit Connections:
-LCD:
* LCD RS pin to digital pin 12
* LCD Enable pin to digital pin 11
* LCD D4 pin to digital pin 5
* LCD D5 pin to digital pin 4
* LCD D6 pin to digital pin 3
* LCD D7 pin to digital pin 2
* LCD R/W pin to ground
* LCD VSS pin to ground
* LCD VCC pin to 5V
* wiper to LCD VO pin (pin 3)

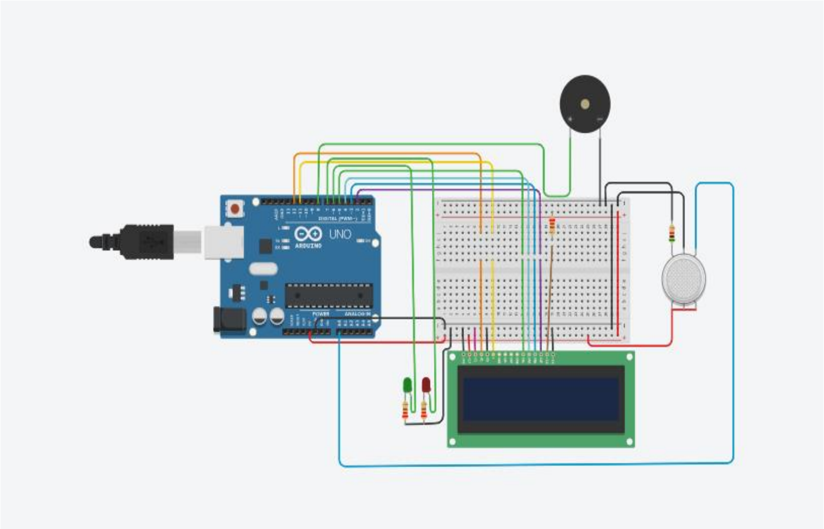
-Gas Sensor:
* Gas Sensor Analog read pin to analog pin A0

-Buzzer:
* Buzzer pin to digital pin 8

-LEDs:
* Green LED pin to digital pin 6
* Red LED pin to digital pin 7

```
 */

// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int sensor = A0;
int val = 0;
int limit = 40;

void setup() {
  Serial.begin(9600);
  // declaring LEDs pins as OUTPUTS Pins and turned them off at the beginning.
  pinMode(7, OUTPUT);
  pinMode(6, OUTPUT);
  digitalWrite(6, LOW);
  digitalWrite(7, LOW);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("Air Monitoring");
  lcd.setCursor(0, 1);
  lcd.print("System");
  delay(500);
  lcd.clear();
}

void loop() {

  val = analogRead(sensor);         // Initialize the value of (val) Variable with the Gas
Sensor Reading.
  val = map(val, 306, 750, 0, 100); //Mapping the output voltage values from the Gas
Sensor to values in the range from 0 to 100.
  Serial.println(val);              //Printing the Gas Sensor Reading in the Serial Monitor.
//If a harmful gas is detected.
    if (val > limit) {
      lcd.setCursor(0, 0);
      lcd.print("Harmful Gas is");  //Print a message to the LCD.
      lcd.setCursor(0, 1);
      lcd.print("Detected");
      digitalWrite(7, HIGH);        //The Red LED wil be turned ON.
      digitalWrite(6, LOW);         //The Green LED wil be turned OFF.
      tone(8, 1000);                //The Buzzer will generate an alarm tone.
      delay(100);                   //Sound ON for 100 msec.
      noTone(8);
      delay(100);                   //Sound OFF for 100 msec.
    }

    else {
      lcd.clear();                  //Clear the warning message from the LCD.
      noTone(8);                    //Sound OFF.
      digitalWrite(7, LOW);         //The Red LED wil be turned OFF.
      digitalWrite(6, HIGH);        //The Green LED wil be turned ON.
    }
}
```

CIRCUIT DIAGRAM :



CONCLUSION :

This project in Tinkercad provides a great starting point for understanding the basics of air quality monitoring using Arduino. Remember that for a real-world implementation, you would need to adapt the project to physical components and consider factors like power supply, data transmission, and enclosure design.