

**Air Quality Monitoring System in Conflict-Affected Areas: A Case Study of Gaza (April 2025)**



# **Mahmoud Kamal Abu Aisha** Master of Cyber-Physical Systems,

# College of Computer Science and Information Technology, Al-Aqsa University, Gaza Email: mk.abuaisheh@std.alaqsa.edu.ps

**Abstract**

Air pollution in Gaza has escalated due to the ongoing war, which has persisted for over a year and a half. This project proposes a real-time air quality monitoring system tailored to the needs of Gaza’s residents, where environmental risks are increasing, and internet connectivity is unreliable. The proposed system is designed to operate offline, without the need for cloud services, and is capable of measuring key air quality parameters: temperature, humidity, gas concentration, and particulate matter (PM2.5). If any of these parameters exceed predefined health risk thresholds, an alarm is automatically triggered to warn inhabitants. The system uses low-cost sensors integrated with a microcontroller to ensure affordability and portability, making it suitable for homes, shelters, and schools in conflict-affected zones. The measured thresholds are based on health guidelines provided by the WHO and EPA, ensuring that the device can effectively detect critical air conditions. The device is highly relevant for smart health monitoring in war-torn regions, contributing to public health protection through early warning and environmental awareness.

# **Index Terms—** Air quality monitoring, IoT (offline), temperature, humidity, gas detection, PM2.5, Gaza, real-time alarm system.

# **I. Introduction**

Air pollution has become a pressing global concern, posing significant threats to both human health and the environment. In densely populated and conflict-affected regions like the Gaza Strip, these concerns are further amplified due to deteriorating infrastructure, frequent disruptions in public utilities, and limited access to environmental monitoring tools. The prolonged war conditions in Gaza—exceeding one and a half years—have resulted in a surge in hazardous air pollutants arising from debris, fires, fuel combustion, and overcrowded shelters, necessitating an urgent need for localized and continuous air quality monitoring solutions.

Traditional air quality monitoring systems often rely on internet connectivity and cloud-based data services, which makes them impractical for deployment in regions suffering from **severe internet outages** and **unreliable power infrastructure**, as is the case in Gaza. In response to these unique challenges, this project proposes the development of a **standalone air quality monitoring system**, capable of operating **independently of internet services**, and tailored specifically to meet the needs of the local population.

The proposed system utilizes a set of low-cost sensors to measure key environmental parameters, including **temperature, humidity, toxic gas concentrations**, and **particulate matter (PM2.5).** When any of these parameters exceed internationally recognized health thresholds—such as those defined by the WHO or EPA—the system **triggers an audible alert** using a buzzer to immediately warn nearby residents. By eliminating dependence on cloud storage or mobile networks, this solution ensures **real-time local alerting**, affordability, and ease of deployment, particularly in resource-limited and emergency settings.

This paper presents the design, implementation, and testing of the air quality monitoring prototype, emphasizing its potential role in **improving public health awareness** and **protecting vulnerable communities in Gaza** under ongoing conflict conditions.

# **II.** Related Work

Several prior studies have proposed different approaches for air quality monitoring using Internet of Things (IoT) systems, utilizing low-cost and efficient hardware for real-time data collection and cloud integration. In [1], Maharana et al. developed an IoT-based real-time air quality monitoring system that utilized the ESP32 microcontroller alongside MQ-135 and DHT-11 sensors for detecting gases such as CO2, benzene, and ammonia. The system also integrated a cloud platform to transmit the data, ensuring the system's accessibility and providing real-time alerts when gas concentrations exceed safe limits.  
  
Similarly, Borade and Prakasarao in [2] presented a low-cost pollution measurement system employing the ESP32 and multiple sensors, including MQ-9 and DHT22. This system collected data on air quality parameters like CO, CO2, PM10, and other gases, transmitting the measurements via REST API to a cloud-based platform for further analysis and visualization.  
  
In addition, the study by Azman et al. [5] focused on the use of ESP32 for indoor air quality monitoring, integrating various sensors such as MQ-2 and DHT11 to measure gas levels and environmental conditions like temperature and humidity. This system, powered by a solar panel, included a ventilation system that activated automatically based on detected gas levels, highlighting the potential for IoT systems to not only monitor air quality but also take corrective actions autonomously.  
  
Furthermore, a study conducted by Mahetaliya et al. [6] proposed an air quality index monitoring system using ESP32, focusing on parameters such as PM2.5, CO, CO2, and environmental conditions. Their system integrates with ThingSpeak to provide real-time monitoring and data visualization. Alerts were triggered when pollutant levels crossed predefined thresholds, ensuring timely interventions.

# These limitations highlight the need for a **standalone, offline-capable air quality monitoring solution**, which this project aims to fulfill by offering a low-cost, non-cloud-dependent system tailored specifically for Gaza’s wartime conditions

# III. Proposed System

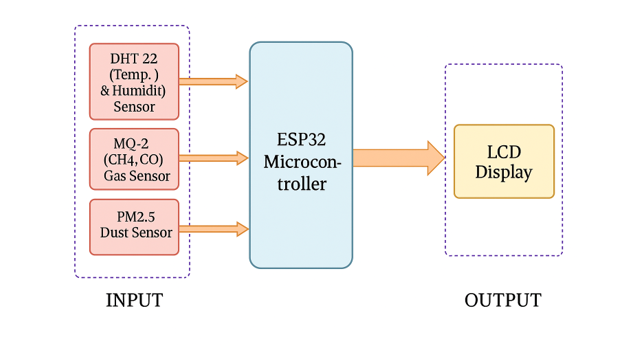
This project proposes a standalone air quality monitoring system specifically designed for Gaza’s population, where ongoing war and infrastructure instability have made cloud-based systems impractical. Unlike conventional IoT solutions, this system operates entirely offline, allowing real-time environmental monitoring without internet connectivity or remote servers.  
  
The system is built using an ESP32 microcontroller, chosen for its efficiency and compatibility with multiple environmental sensors. To measure critical air quality parameters, the following hardware components are integrated:

|  |  |
| --- | --- |
| Component | Function |
| ESP32 | Main controller for sensor input and alarm activation. |
| DHT22 Sensor | Measures temperature and humidity with reliable accuracy. |
| MQ-2 Gas Sensor | Detects hazardous gases like carbon monoxide (CO), methane, and LPG. |
| PM2.5 Sensor | Monitors fine particulate matter, a key pollutant in war-affected zones. |
| 20x4 I2C LCD | Displays real-time environmental data and alerts. |
| Piezo Buzzer | Sounds an alert when readings exceed safe limits. |

Sensor values are constantly compared to predefined danger thresholds: temperature above 40°C, humidity over 70%, gas above 750 ppm, and PM2.5 above 150 μg/m³. When any parameter exceeds its safe level, a buzzer is triggered, and a warning message is shown on the screen—ensuring immediate, localized alerting.  
  
By eliminating dependence on mobile apps or internet dashboards, the system ensures maximum reliability in regions facing power and connectivity outages. The design is also low-cost, modular, and portable, making it ideal for deployment in homes, shelters, and schools across Gaza.  
  
This solution addresses a critical need for real-time, offline air quality monitoring under crisis conditions and offers a scalable model for environmental safety in similar vulnerable areas.

# IV. Materials and Method

## 4.1 System Design

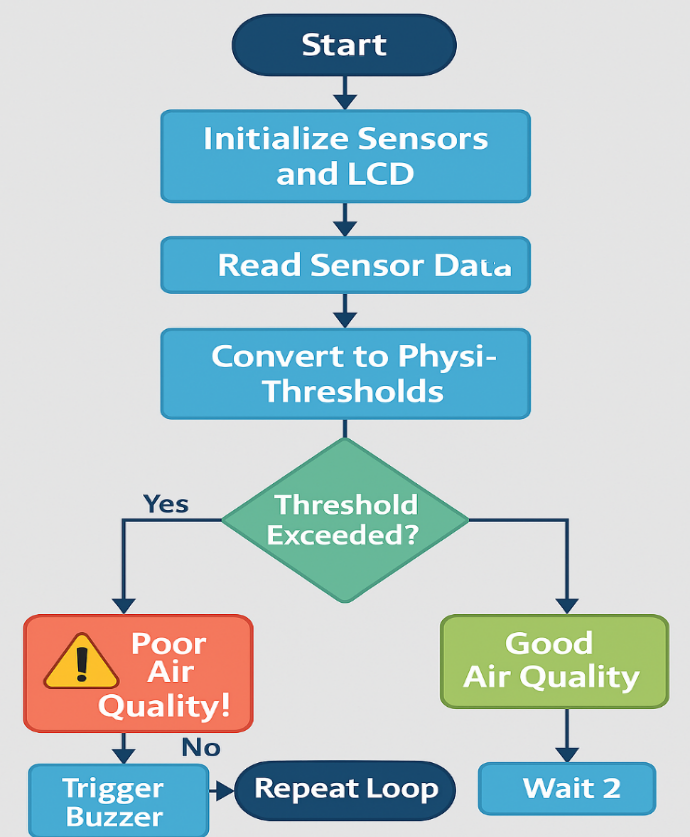


### 4.1.1 System Architecture

The proposed system is designed to monitor air quality parameters in real-time using a modular, low-cost, and offline-capable architecture. Its primary goal is to detect environmental hazards and immediately alert residents without relying on cloud platforms or mobile networks. The design integrates both hardware and software components, forming a self-contained system that can operate under severe infrastructure constraints.

The system follows a layered architecture comprising the following stages:  
1. Data Acquisition Layer:  
 - DHT22 captures temperature and humidity.  
 - MQ2 gas sensor measures combustible gases and carbon monoxide.  
 - PM2.5 sensor monitors fine particulate matter.  
  
2. Processing and Control Layer:  
 - The ESP32 continuously reads analog and digital signals from sensors, converts them into meaningful physical values, and compares them to predefined safety thresholds.  
  
3. Output and Alert Layer:  
 - If any parameter exceeds safe limits, the system:  
 • Triggers a buzzer alarm.  
 • Displays a warning on the 20x4 LCD screen.  
 • Otherwise, it displays a “Good Air Quality” message.

### 4.1.2 System Flowchart Overview



### 4.1.3 Design Justifications

- ESP32 was selected due to its support for analog and digital pins, low cost, and built-in processing capability.  
- The system is power-efficient and can be powered via USB or portable battery packs.  
- All operations are locally embedded within the ESP32 firmware, making the system independent of cloud connectivity, ideal for areas like Gaza with unstable internet.

### 4.1.4 Physical Deployment

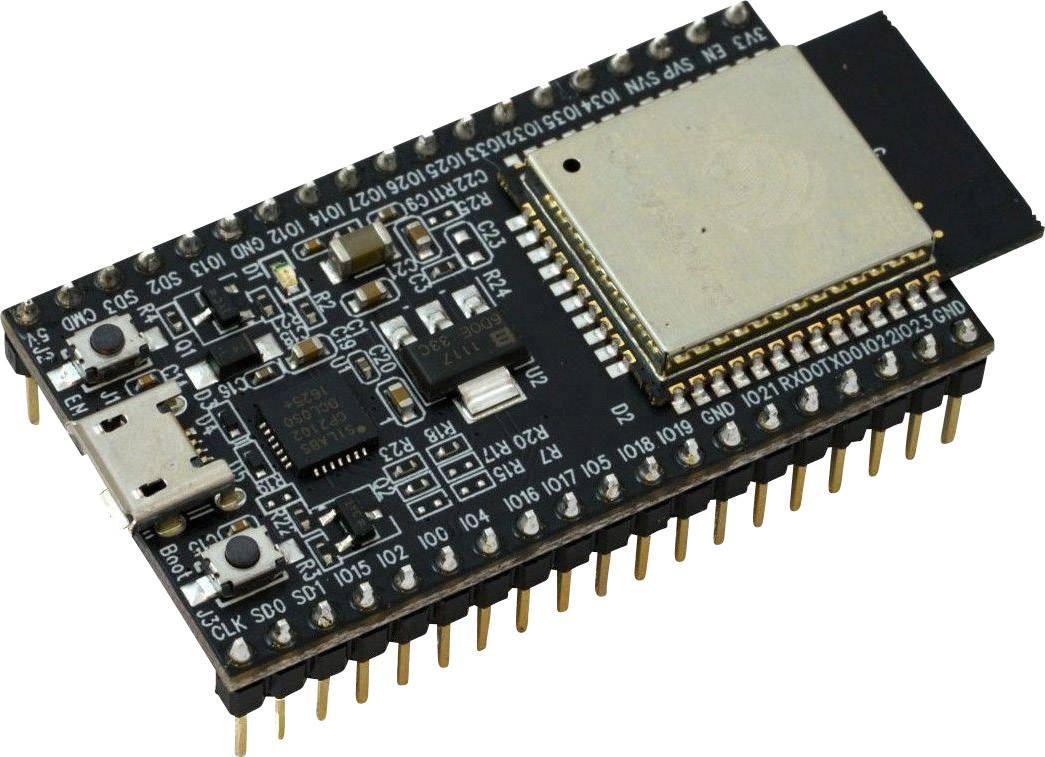
The prototype is built on a breadboard for testing and can later be migrated to a PCB or enclosed in a weather-resistant case for field deployment. The system is scalable and replicable, making it suitable for wide distribution in vulnerable environments.

## **4.2 Hardware Description**

The proposed air quality monitoring system is constructed using the following components:

## ESP32 Microcontroller

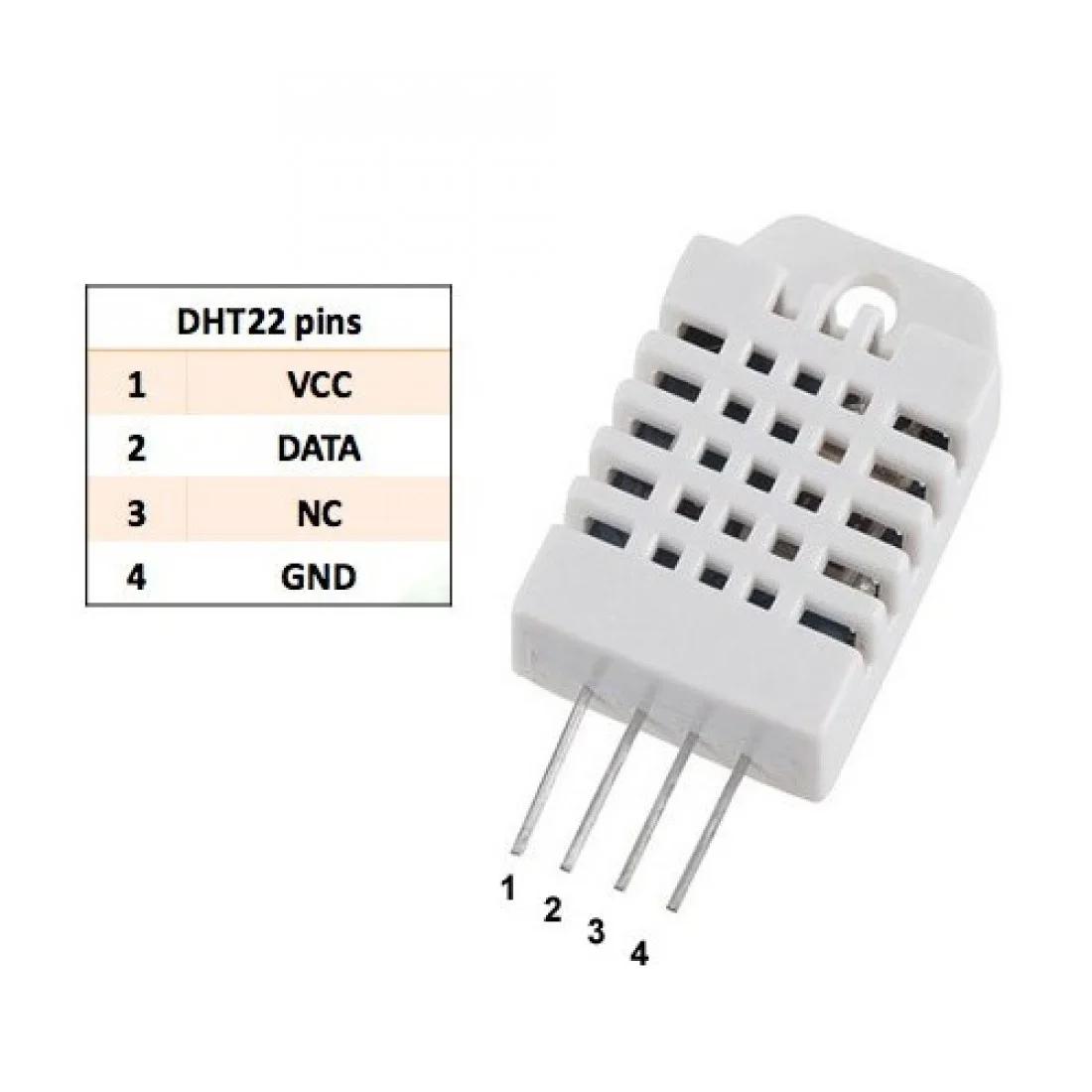
|  |  |
| --- | --- |
| Specifications | ESP32 |
| SRAM | 512 KBytes |
| Bluetooth | Yes |
| GPIO pins | 36 |
| ADC | 12 bits |
| Working Temp.(°C) | -40 to +125 |
| Operating Voltage | 2.3V–3.6V |
| IEEE 802.11b/g/n Wi-Fi | HT40 |
| Clock Speed | 160MHz |
| Price (USD) | $6 |



The ESP32 microcontroller is the central processing unit of the system. It is a low-power, dual-core microcontroller with built-in Wi-Fi and Bluetooth. It receives data from sensors and triggers outputs accordingly. Its versatility and wide GPIO pin range make it ideal for IoT applications.

## DHT22 Sensor

|  |  |
| --- | --- |
| Name | Specification |
| Interchangeability | Possible |
| Humidity range (%) | 0 – 100 |
| Temperature range (°C) | 40 to 80 |
| Operating voltage (V) | 3.3 – 6 |
| Price (USD) | $3 |



The DHT22 sensor is used to measure temperature and humidity. It features high accuracy and digital output, which makes it easy to interface with microcontrollers. It operates within a wide temperature range and is highly stable over time.

## MQ-2 Gas Sensor

|  |  |
| --- | --- |
| Name of the parameter | Specification |
| Gas Detection | 10 to 1000 ppm |
| Heater Operating Voltage | 5.0V ± 0.2V |
| Operating Current Value | 150mA |
| Digital Output | 0 and 1 (0.1V and 5V) |
| Analog Output Voltage | Clean air (0.1V–0.3V), highest ~4V |
| Load resistance | Variable |
| Price (USD) | $2 |



The MQ-2 gas sensor detects flammable gases and smoke (e.g., CH4, CO, LPG). It outputs analog values corresponding to the gas concentration. The sensor is widely used in air quality systems due to its responsiveness and low cost.

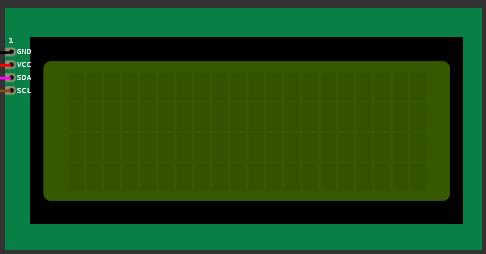
## PM2.5 Dust Sensor

|  |  |
| --- | --- |
| Name of the parameter | Specification |
| LED PIN | Connected to Digital GPIO Pin |
| Pin A0 | Analog Output Interface |
| Capacitor Value | 220µF |
| Resistor Value | 150Ω |
| Price (USD) | $5.5 |



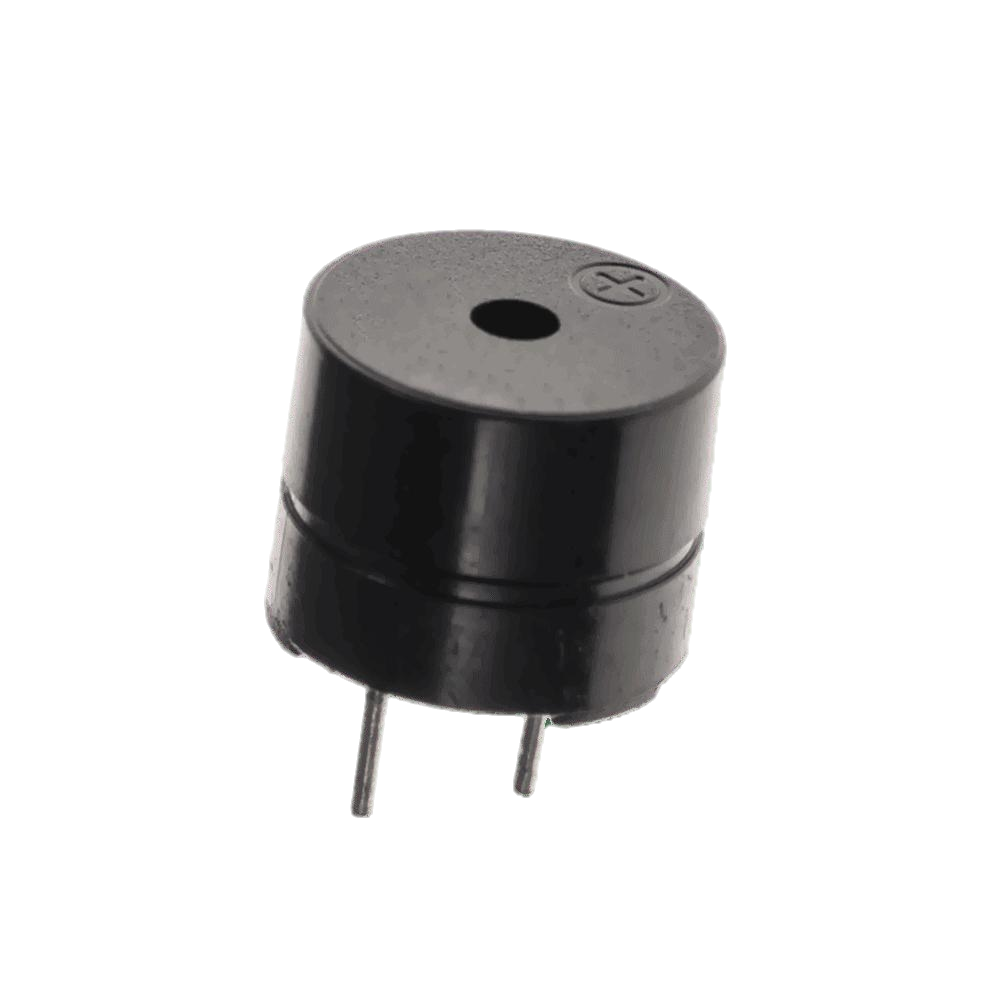
The PM2.5 sensor measures fine particulate matter (2.5 microns or smaller) suspended in the air. These particles are harmful when inhaled. The sensor provides digital/analog output and is essential for air quality monitoring applications.

## 20x4 I2C LCD Display



The 20x4 I2C LCD display is used to show real-time sensor data. It can display four lines of twenty characters each. The I2C interface significantly reduces the number of pins required for communication with the ESP32.

## Piezo Buzzer



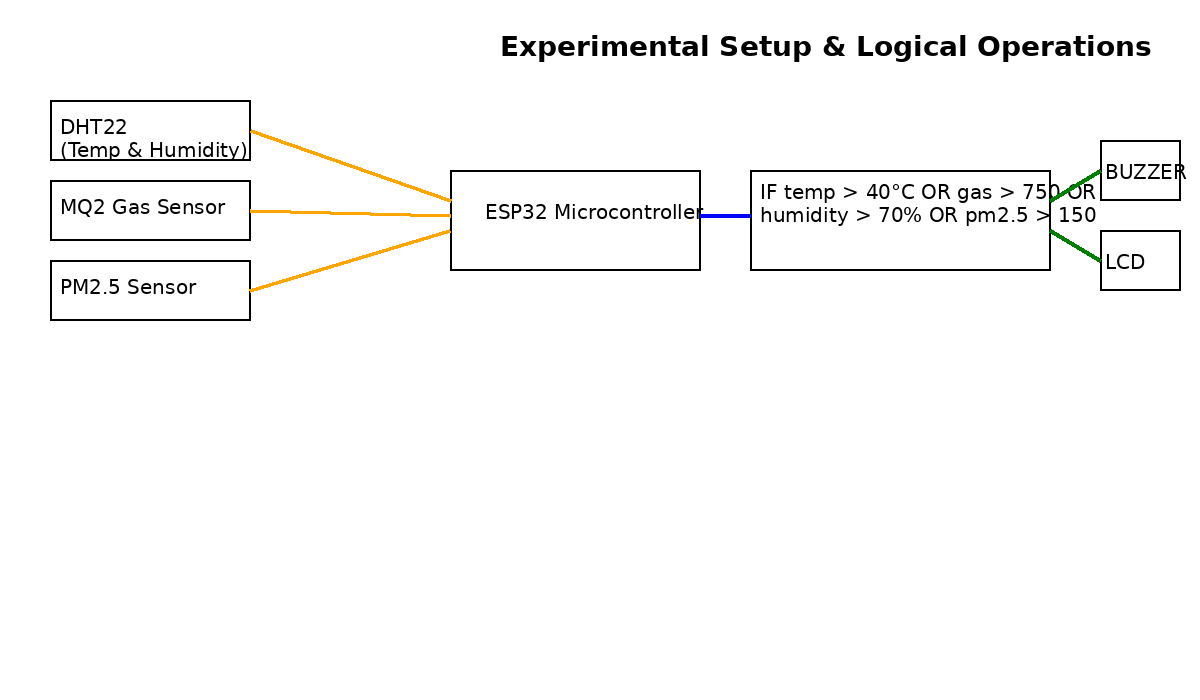
The piezo buzzer is used for alerting users when air quality thresholds are exceeded. It is a simple component that produces sound when activated by a HIGH signal from the ESP32.

## **4.3 Software Description**

The firmware is written in Arduino C++, using libraries such as DHT.h for the DHT22 sensor, Wire.h and LiquidCrystal\_I2C.h for the LCD display. Sensor values are read, processed, and compared to predefined thresholds. If dangerous levels are detected, a buzzer is activated and a warning is displayed.

The entire system was developed and tested using the Wokwi simulation platform, enabling virtual prototyping of ESP32-based components without physical hardware deployment .

## **4.4 Experimental Setup and Logical Operations**



The system operates offline, continuously monitoring environmental parameters. Below are the key logic conditions:

### 4.4.1 Mathematical and Logical Operations

This document summarizes the key mathematical and decision-making operations that occur in the background of the Air Quality Monitoring System using ESP32 and simulated sensors (MQ2, DHT22, PM2.5) in the Wokwi environment.

## 1. MQ2 Gas Sensor

- The MQ2 outputs an analog value which is read by ESP32 via analogRead ().  
- Value range (in simulation): approximately 300–900.  
- The voltage from this value is calculated as:  
 Vout = (analogValue / 4095.0) × 5.0 (since ESP32 ADC is 12-bit and uses 5V reference)  
- Example:  
 If analogValue = 820 ⇒ Vout ≈ (820 / 4095) × 5 ≈ 1.0V  
- If gas percentage > 70% ⇒ Vout > 3.5V ⇒ Alarm is triggered.

## 2. DHT22 Sensor (Temperature and Humidity)

- The DHT22 sensor provides temperature and humidity directly.  
- No mathematical conversions are needed.  
- Thresholds:  
 Temperature > 35°C ⇒ trigger alert  
 Humidity > 70% ⇒ trigger alert

## 3. PM2.5 Sensor (Simulated)

- The PM2.5 value is simulated as a raw µg/m³ value.  
- Threshold used in decision:  
 If PM2.5 > 100 µg/m³ ⇒ air considered polluted

## 4. Decision Logic

Combined conditional logic used in loop ():  
if (temperature > 35 || humidity > 70 || gasValue > 716 || pm25 > 100):  
 - Activate buzzer  
 - Display 'Poor Air Quality' on LCD  
else:  
 - Turn off buzzer

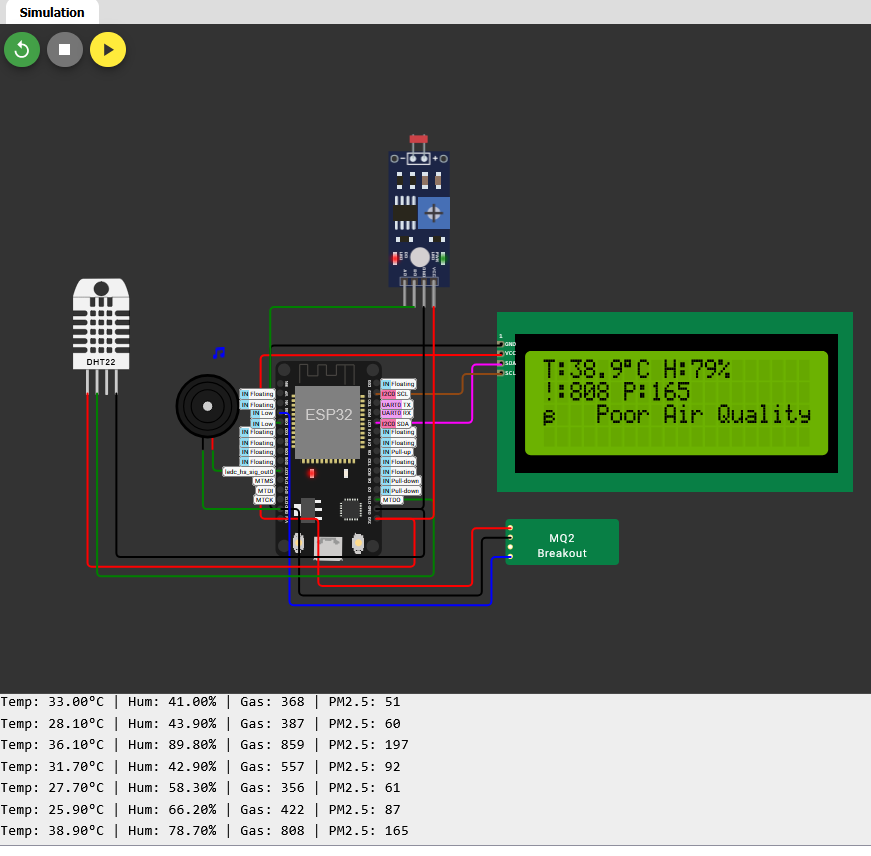
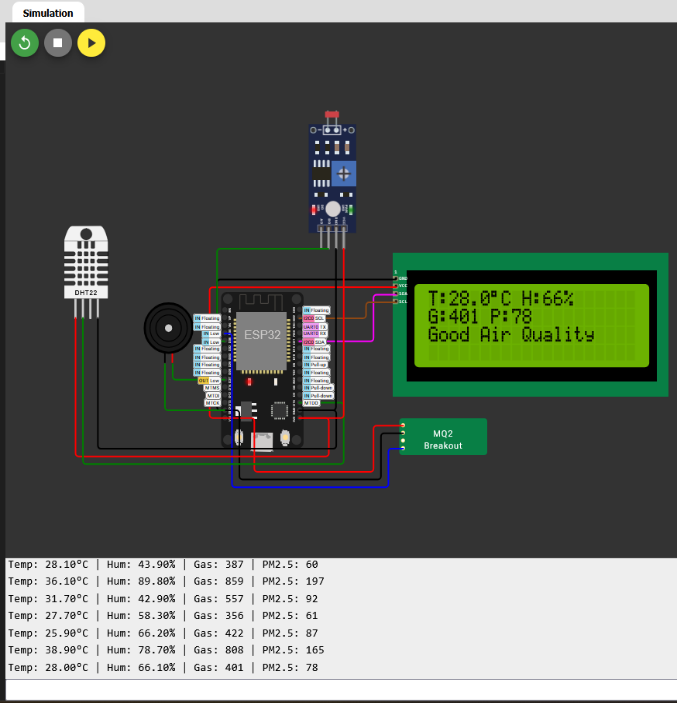
## 5. Summary Table

|  |  |  |  |
| --- | --- | --- | --- |
| Sensor | Raw Value | Computation | Threshold / Effect |
| MQ2 | Analog (300–900) | Vout = (value / 4095) × 5V | Vout > 3.5V or value > 716 ⇒ Trigger |
| DHT22 Temp | Direct (°C) | None | Temp > 35°C ⇒ Trigger |
| DHT22 Humid | Direct (%) | None | Humidity > 70% ⇒ Trigger |
| PM2.5 | Direct (µg/m³) | None / Simulated | PM2.5 > 100 ⇒ Trigger |

### 4.4.2 implementation -Simulated Firmware for ESP32 Using Wokwi Platform

The following is the full code used to operate the air quality monitoring system:

|  |  |  |
| --- | --- | --- |
| #include <Wire.h>  #include <LiquidCrystal\_I2C.h>  #include <DHT.h>  #Define BUZZER 27  const float TEMP\_THRESHOLD = 35.0;  const float HUM\_THRESHOLD = 70.0;  const int GAS\_THRESHOLD = 716;  lcd. backlight ();  lcd. setCursor (0, 0);  lcd.print (" Air Quality System");  delay (2000);  lcd. clear ();  pinMode (BUZZER, OUTPUT);  noTone (BUZZER);  }  void loop () {  float temperature, humidity;  int gasValue, pm25;  if (counter % 4! = 3) {  temperature = random (250, 340) / 10.0;  humidity = random (400, 690) / 10.0; | gasValue = random (300, 710);  bool alert = (  gasValue > GAS\_THRESHOLD ||  temperature > TEMP\_THRESHOLD ||  humidity > HUM\_THRESHOLD ||  pm25 > PM\_THRESHOLD);  Serial. Print ("Temp: ");  Serial. Print(temperature);  Serial. Print ("°C | Hum: ");  Serial. Print(humidity);  Serial. Print ("% | Gas: ");  Serial. Print(gasValue);  Serial. Print (" | PM2.5: ");  Serial.println(pm25);  lcd. clear ();  lcd. setCursor (0, 0);  lcd.print("T:");  lcd.print (temperature, 1);  lcd.print((char)223);  lcd.print ("C H:"); | lcd.print (humidity, 0);  lcd.print ("%");  lcd. setCursor (0, 1);  lcd.print("G:");  lcd.print(gasValue);  lcd.print (" P:");  lcd.print(pm25);  lcd. setCursor (0, 2);  if (alert) {  tone (BUZZER, 1000);  lcd.print ("⚠ Poor Air Quality!");  } else {  noTone (BUZZER);  lcd.print ("Good Air Quality ");  }  counter++;  delay (2000);  } |



# V. **Results**

The proposed air quality monitoring system was tested using the Wokwi simulation environment to evaluate its real-time performance under varying environmental conditions. Multiple simulation scenarios were designed to assess the system’s response to changes in temperature, humidity, gas concentration, and PM2.5 levels. The test results demonstrated the effectiveness of the system’s alert logic and sensor integration.  
  
Three key testing scenarios were implemented:  
  
1. Normal Air Quality:  
In the baseline condition, all sensor readings remained within the predefined safe thresholds (temperature < 35°C, humidity < 70%, gas < 716, PM2.5 < 100 µg/m³). The system displayed a “Good Air Quality” message on the LCD and the buzzer remained inactive.  
  
2. Polluted Environment Simulation:  
In this scenario, all sensor readings exceeded the critical limits (e.g., temperature > 40°C, gas value > 800, PM2.5 > 120 µg/m³). The system responded by activating the buzzer and displaying a warning message “⚠ Poor Air Quality!” on the LCD, confirming proper alert functionality.  
  
3. Dynamic Variation:  
A cyclic testing mode was implemented where the system experienced good air quality for three cycles followed by a polluted condition in the fourth. This confirmed the reliability of the decision logic and the responsiveness of the alert mechanism across changing environments.  
  
Throughout all tests, the sensor readings were accurately displayed on the 20x4 I2C LCD, and alerts were consistently triggered when necessary. The buzzer activation was immediate upon threshold violation and deactivated automatically when conditions returned to safe levels.  
  
These results verify that the system operates reliably in an offline mode and can provide real-time localized environmental alerts, making it highly suitable for use in conflict zones or low-connectivity environments like Gaza.

# VI. **Conclusion**

This project presents the successful development and simulation of a low-cost, real-time air quality monitoring system tailored for conflict-affected environments such as Gaza. Utilizing an ESP32 microcontroller and sensors including DHT22, MQ-2, and PM2.5, the system effectively monitors temperature, humidity, gas levels, and particulate matter concentrations without relying on internet connectivity or cloud-based platforms.  
  
The system's ability to trigger immediate alerts via an LCD display and a buzzer when air quality thresholds are exceeded demonstrates its suitability for emergency scenarios where residents require instant, localized warnings. Through Wokwi simulation, the system proved responsive and reliable across various test conditions, validating the decision logic and real-time alert functionality.  
  
By eliminating the need for cloud services and emphasizing offline operation, portability, and affordability, this solution addresses critical gaps in environmental monitoring within low-resource and high-risk areas. The project offers a scalable, replicable model that can be deployed in shelters, schools, and homes to promote public health awareness and mitigate environmental hazards.

# VII. **Future** **Scope**

The proposed air quality monitoring system lays a solid foundation for future enhancements aimed at improving its functionality, scalability, and long-term usability in real-world conditions.  
  
1. Integration with Renewable Energy:  
The system can be upgraded to support solar-powered operation, making it ideal for deployment in off-grid and war-affected regions where electricity is unreliable or unavailable.  
  
2. Data Logging and Analysis:  
Incorporating an SD card module would enable offline storage of historical air quality data, allowing researchers and authorities to analyze trends and take proactive environmental actions.  
  
3. Wireless Alert Systems:  
Future iterations may include wireless alert modules such as RF or LoRa-based communication, allowing alerts to be transmitted to remote locations, such as nearby hospitals or community centers.  
  
4. Mobile Application Support (Optional):  
While the current design is offline, future versions could include optional Bluetooth or Wi-Fi-based synchronization with mobile apps for data visualization and remote monitoring in areas with intermittent connectivity.  
  
5. Additional Sensors:  
The system could be extended to include sensors for CO2, NO2, or VOCs to provide more comprehensive environmental profiling, especially in industrial or high-risk urban areas.  
  
6. Real-World Field Deployment:  
The next step involves transitioning from simulation to real hardware implementation, testing the system under actual environmental conditions in Gaza or similar contexts.

7. Smart Notification Integration:  
Future versions of the system can support smart notifications via SMS, push notifications, or Telegram bots. By integrating with GSM modules or cloud-based notification services (when connectivity is available), the system can automatically notify users or emergency services  
  
These enhancements will further solidify the system’s role as a robust, adaptable solution for real-time environmental monitoring, particularly in vulnerable and infrastructure-limited settings.

# VIII. **References**

[1] B. K. Moharana, P. Anand, S. Kumar, and P. Kodali, “Development of an IoT-based Real-Time Air Quality Monitoring Device,” in 2020 IEEE International Conference on Communication and Signal Processing (ICCSP), July 2020, doi: 10.1109/ICCSP48568.2020.9182330.  
  
[2] S. S. Borade and A. Prakasarao, “A Low-Cost Smart Pollution Measurement System Using REST API and ESP32,” in 2018 3rd International Conference for Convergence in Technology (I2CT), Pune, India, Apr. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8356531.  
  
[3] A. I. Azman and L. Bakar, “IoT Based Gas Leakage Detection and Ventilation for Home Using Solar,” Progress in Engineering Application and Technology, vol. 5, no. 2, pp. 276–287, 2024. doi: 10.30880/peat.2024.05.02.028.  
  
[4] N. Omkar, S. Rahul, R. Vijaykrishna, N. Swapna, and M. R. Sura, “Smart Environmental Monitoring Using ESP32 Microcontroller,” IOSR Journal of Electronics and Communication Engineering, vol. 19, no. 3, pp. 07–12, May–June 2024. doi: 10.9790/2834-1903010712.  
  
[5] S. Md Rozali, N. S. Mohd Shari, E. Ruslan, R. Ramli, and M. F. Rahmat, “Development of Particulate Matter Detection and Air Monitoring System by Using Arduino,” International Journal of Academic Research in Business and Social Sciences, vol. 14, no. 12, 2024. doi: 10.6007/IJARBSS/v14-i12/24398.  
  
[6] P. Megantoro, S. A. Aldhama, G. S. Prihandana, and P. Vigneshwaran, “IoT-Based Weather Station with Air Quality Measurement Using ESP32 for Environmental Aerial Condition Study,” TELKOMNIKA Telecommunication, Computing, Electronics and Control, vol. 19, no. 4, pp. 1316–1325, Aug. 2021. doi: 10.12928/TELKOMNIKA.v19i4.18990.  
  
[7] S. Mahetaliya, D. Makwana, A. Pujara, and S. Hanumante, “IoT Based Air Quality Index Monitoring using ESP32,” International Research Journal of Engineering and Technology (IRJET), vol. 8, no. 4, pp. 5186–5190, Apr. 2021.