International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Smart Air Quality Monitoring and Management System

Dr. Bandi Das, Assistant Professor
Department of Electronics and Communication Engineering (ECE), CMR Technical Campus, Hyderabad.
bandidas.ece@cmrtc.ac.in

Chatlapalli Anusha,
Department of Electronics and Communication Engineering (ECE), CMR Technical Campus, Hyderabad.
anushachatlapalli35@gmail.com

Vollala Akshaya,
Department of Electronics and Communication Engineering (ECE), CMR Technical Campus, Hyderabad.
vollalaakshaya@gmail.com

Burra Shiva Kumar Goud, Department of Electronics and Communication Engineering (ECE), CMR Technical Campus, Hyderabad.
shivagoud1815@gmail.com

Abstract --

Air quality monitoring is important for identifying changes in indoor environmental conditions and supporting timely responses to potential pollution. This paper presents a Smart Air Quality Monitoring and Management System based on the ESP32 microcontroller. The proposed system integrates an MQ-7 gas sensor for carbon monoxide-related sensing and a DHT11 sensor for measuring temperature and relative humidity. The ESP32 processes the sensor readings and transmits the collected data to Firebase Realtime Database through a Wi-Fi connection, enabling remote monitoring. A Telegram-based notification mechanism provides alerts when the measured gas-sensor value crosses predefined thresholds. In addition, a relay module controls a fan according to the programmed threshold categories. The system combines sensor acquisition, cloud data transmission, remote notifications, and automatic load control in a single low-cost prototype. Its performance should be evaluated through repeated tests of sensor response, data-upload reliability, notification delivery, and relay operation. The current gas-sensor output is mapped to a numerical range and should not be interpreted as calibrated carbon monoxide concentration in parts per million without appropriate calibration and validation. The prototype demonstrates an integrated approach to connected environmental monitoring, while accurate gas measurement and long-term reliability remain areas for further investigation.

Keywords -- Air quality monitoring, Internet of Things (IoT), ESP32, MQ-7, DHT11, Firebase Realtime Database, Telegram notification, automated fan control.

I. INTRODUCTION

Both rural and urban areas are experiencing air pollution, When harmful gases, like carbon monoxide, are present in the atmosphere and the temperature and/or humidity vary widely, it affects human health. When people breathe polluted air for long periods of time they will develop respiratory illnesses, heart problems, and other serious illnesses [1], [2].

Because of the severity of the situation and the fact that most people do not have access to current data on air quality where they live, there is now a demand for improved and convenient air-quality monitoring systems [3].

Conventional methods of measuring air quality tend to be large, costly and located in predetermined geographic areas. They do not operate continuously inside buildings (e.g., in homes, offices, and small businesses) and therefore do not supply immediate/localized/real-time data needed for timeliness of decision making. Therefore, portable, low-cost solutions are required to measure the environment at a micro level [4], [5].

IoT technology has allowed for the creation of smart environmental monitoring systems with integrated sensors, microcontrollers like ESP32 to permanently measure parameters such as gas concentration, temperature and humidity by sending the data collected to cloud platforms via Wi-Fi so that users can see real time data from anywhere with a web-based dashboard [6], [7].

This new air quality monitoring system will not only monitor air quality but will also manage air quality concentration levels as well. An example of this would be if the air quality exceeded safe limits, the system would create an Alert for the user through Telegram via the Telegram App to notify them of the appropriate action to take to lower the air quality levels. If the air quality is too high, the system can also activate exhaust fans to increase airflow to decrease pollutant concentrations in the air [8]. This integration of both monitoring and managing an environment creates an ideal application for this kind of system whether in a commercial, residential or industrial setting.

II. Literature Survey

IoT-based air-quality monitoring has been investigated using low-cost sensors, embedded controllers, wireless connectivity, and cloud-based platforms. Related work discusses monitoring architectures and the use of networked systems to make environmental data more accessible [1], [2], [4]. Such approaches provide a foundation for collecting and presenting measurements, although their performance depends on the selected sensors, calibration procedures, and communication design.

Sensor selection is an important consideration in low-cost monitoring. Gas sensors can provide an electrical response to the presence of target gases, while temperature and humidity sensors provide supporting environmental measurements. The response of a low-cost gas sensor should be interpreted according to its operating conditions and calibration; an analogue reading alone does not provide a validated concentration in ppm. Research on sensor-system validation and instrumentation highlights the importance of characterising measurement performance before drawing quantitative conclusions [6].

Cloud integration allows sensor readings to be stored and accessed remotely. Related studies also consider prediction, privacy, and monitoring across different locations or operating conditions [3]–[5], [7]. These capabilities are relevant to larger monitoring deployments, but prediction or federated learning is not implemented in the prototype described in this paper.

Alert mechanisms can help notify users when measured values cross configured thresholds. Local actuation can complement remote monitoring by enabling a connected controller to operate a load in response to the programmed condition. In the proposed implementation, Telegram is used for notifications and a relay is used for fan control. The alert categories used by the software are application-specific labels and are not presented as official air-quality categories.

Based on these considerations, this work focuses on integrating sensor acquisition, cloud upload, Telegram notification, and relay-based fan control in a single ESP32 protoBased on these considerations, this work focuses on integrating sensor acquisition, cloud upload, Telegram notification, and relay-based fan control in a single ESP32 prototype. The implementation is evaluated by checking whether sensor readings are acquired, data is uploaded, notifications are received, and relay states match the programmed logic. This paper does not claim a new sensing principle, a machine-learning prediction method, or validated CO measurement; these remain outside the current implementation.

III. Proposed Methodology

The proposed system consists of an MQ-7 gas-sensing unit, a DHT11 temperature and humidity sensor, an ESP32 controller, a Wi-Fi connection, Firebase Realtime Database, Telegram notifications, and a relay-controlled fan. The ESP32 reads the sensors, processes the readings, assigns a software status using predefined thresholds, uploads the data to the cloud, and controls the relay according to the programmed logic.

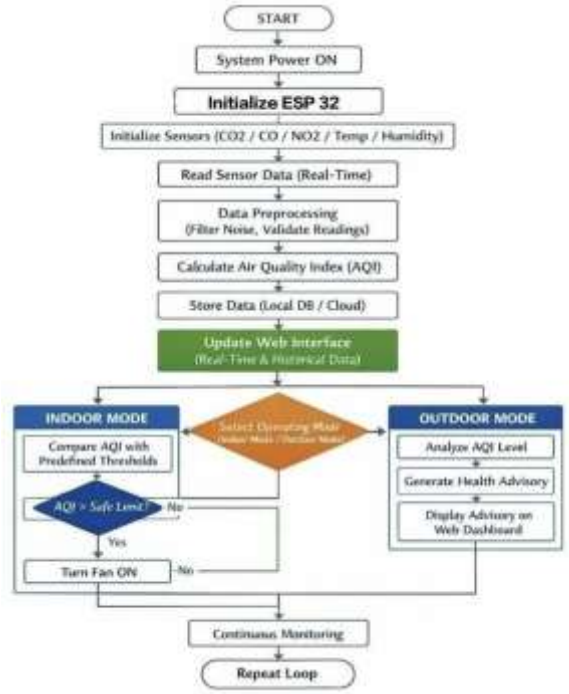


Fig. 1. System Flow Chart

Fig. 1 illustrates the system workflow. After initialising the sensors and communication services, the ESP32 reads the MQ-7 analogue value and the DHT11 measurements. The gas-sensor reading is processed by the configured threshold logic. The system then attempts to upload the sensor data to Firebase, handles Telegram notification conditions, and sets the relay output according to the software status. These operations repeat during normal execution.

The implementation averages multiple analogue samples from the MQ-7 sensor to reduce short-term variation in the raw reading. It then maps the averaged ADC value to a numerical range used by the threshold logic. This mapping is a software scaling operation and is not a calibrated conversion to CO concentration in ppm. A valid concentration estimate requires the sensor to be operated according to its specified heating and measurement procedure and calibrated against a suitable reference instrument.

The current software uses four labels based on the scaled gas-sensor value: GOOD for values up to and including 50, MODERATE for values above 50 and up to 100, UNHEALTHY for values above 100 and up to 200, and DANGEROUS for values above 200. These are thresholds defined by the implementation for demonstration and must not be interpreted as official AQI categories or validated CO exposure limits.

The system repeats the sensing, processing, communication, notification, and relay-control steps during operation. The current implementation depends on Wi-Fi and cloud availability; reliable deployment should include explicit reconnection handling, timeouts, and clear reporting of sensor or communication failures.

The entire operation runs in a continuous loop, enabling real-time monitoring and dynamic response to changing environmental conditions.

A. ESP32 Microcontroller

The central controller used in the proposed system is the ESP32, as shown in Fig. 2.



Fig. 2. Esp32 Microcontroller

The center of our system is the ESP32 microcontroller, which collects data from each of the various sensors we have connected, including the MQ-7 gas sensor and the DHT11 temperature and humidity sensor. The ESP32 processes the data it gets from the sensors and determines whether or not the conditions for air quality are acceptable according to a set of predefined parameters. Also, the ESP32 uses its built-in Wi-Fi connectivity to transmit all sensor data to our cloud platform in real-time. Furthermore, the ESP32 can control devices connected to it (e.g., fan) through the use of a relay module if the air quality does not meet defined established safety limits.

Because of the built-in features and energy-efficiency of the ESP32, it is ideal for use in any IoT-based environmental monitoring system.

B. MQ-7 Gas Sensor

The gas sensing component used in the proposed system is the MQ-7 sensor, as shown in Fig. 3.



Fig. 3. MQ-7 Gas Sensor

The MQ-7 is used in this prototype as the gas-sensing component associated with carbon monoxide detection. The sensor provides an electrical response that is read by the ESP32 analogue input. The firmware averages 20 analogue samples and maps the result from the ADC range to a value between 0 and 300 for the implemented threshold logic. This mapped value is a scaled indicator, not a calibrated CO concentration in ppm. Quantitative CO reporting requires appropriate sensor conditioning, the specified heater cycle, calibration, and comparison with a reference instrument.

C. DHT11 Sensor

The environmental sensing component used for temperature and humidity measurement is the DHT11 sensor, as shown in Fig. 4.

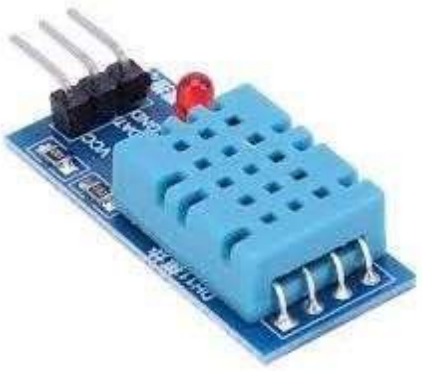


Fig. 4. DHT11 Sensor

The ambient temperature and relative humidity can be measured with the DHT11 sensor. It provides the digital output data directly to the ESP32 microcontroller. These parameters can assist in assessing and interpreting weather conditions that impact quality of air. The data collected from the DHT11 along with gas sensor readings will determine overall assessment of air quality for the area that gas measurements were taken. The DHT11's accuracy and ease of use make it a very effective sensor for real-time usage in this system.

D. Relay Module

The switching component used in the proposed system is the relay module, as shown in Fig. 5.

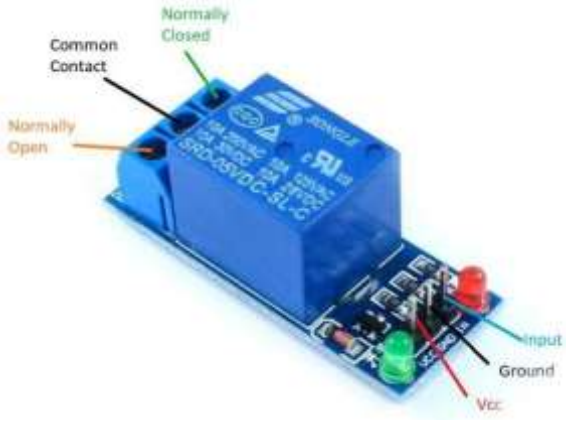


Fig. 5. Relay Module

The relay module allows a low-voltage ESP32 output to control a separate fan circuit. In the current firmware, the relay output is set low for the GOOD and MODERATE software categories and high for the UNHEALTHY and DANGEROUS categories. The active-high or active-low behaviour of the physical relay module must be verified during testing. Fan operation alone does not establish that the surrounding air has returned to a safe condition.

IV. Results and Discussion

The prototype is evaluated by examining four functions: sensor-data acquisition, cloud data transmission, Telegram notification delivery, and relay-based fan control. The available demonstration and interface screenshots can illustrate operation, but quantitative claims about accuracy, reliability, or response time should be made only when supported by recorded test data.

A. Sensor Data Acquisition

The real-time sensor readings obtained from the ESP32 are presented in Fig. 6.



Fig. 6. Fig. Real-Time Sensor Data Output

The ESP32 reads the MQ-7 analogue signal and obtains temperature and relative-humidity values from the DHT11. The MQ-7 samples are averaged before threshold classification. The readings can be inspected through the serial monitor and are prepared for transmission to Firebase. Since the MQ-7 output is scaled in software, the displayed gas-sensor value should not be described as a measured CO concentration in ppm unless calibration has been completed.

B. Cloud-Based Monitoring

The cloud-based visualization of environmental data is shown in Fig. 7.



Fig. 7. Web-Based Air Quality Monitoring Interface

Firebase Realtime Database is configured to store the processed gas-sensor value, temperature, humidity, and software status. The firmware attempts to upload these values at approximately five-second intervals during normal execution. Cloud availability and upload reliability should be assessed by recording successful and failed upload attempts over a defined test period. The current software does not calculate a validated Air Quality Index (AQI), so the mapped gas-sensor value should not be labelled as an official AQI.

C. *Alert Notification System*

The alert notification mechanism of the system is illustrated in Fig. 8.



Fig. 8. Alert Notification via Telegram

Whenever an established limit of CO has been hit, notifications get sent via Telegram. For example, a CO concentration level of 73 ppm results in a "Moderate Air Quality" notification. Thus, users will be notified immediately so they can respond accordingly.

D. *Overall System Performance*

The system shown is an effective way of monitoring and controlling air quality using a combination of sensors, ESP32 microcontrollers, internet cloud platforms and alert systems to provide consistent and ongoing operation with real-time data. Additionally, this method is less expensive than typical monitoring/controlling solutions; thus, it has the potential to be used practically as a portable system and may also be available commercially to businesses/organizations involved in environmental protection.

V. CONCLUSION

This paper presents a Smart Air Quality Monitoring and Management System implemented using an ESP32 microcontroller, MQ-7 gas sensor, DHT11 temperature and humidity sensor, Firebase Realtime Database, Telegram notifications, and relay-based fan control. The prototype integrates sensor acquisition, cloud transmission, software threshold classification, remote notification, and local actuation in a single platform.

The implementation provides a foundation for remote observation of environmental measurements and demonstrates how programmed threshold conditions can be used to trigger notifications and control a fan. Its operational reliability must be established through repeated tests and quantitative measurements of data transmission, alert delivery, and relay response. The current gas-sensor scaling method does not provide a validated CO concentration in ppm.

Future work will include calibration against a reference gas-measurement instrument, improved handling of sensor and network failures, reliable notification generation during category transitions, and secure communication with cloud and notification services. Further experiments should assess long-term stability and the effectiveness of fan-based ventilation. The prototype is a starting point for connected environmental monitoring and is not a replacement for certified gas-detection or air-quality measurement equipment

REFERENCES

[1] L. R. Kumar, S. S. Akanksha, S. Keerthana, S. Singh, et al., "Smart Air Quality Monitoring Using IoT," MATEC Web of Conferences, vol. 392, p. 01084, 2024. doi: 10.1051/matecconf/202439201084.

[2] T. Kumar, A. Doss, "AIRO: Development of an Intelligent IoT-based Air Quality Monitoring Solution for Urban Areas," Procedia Computer Science, vol. 218, pp. 262–273, 2023. doi: 10.1016/j.procs.2023.01.008.

[3] V. Bongale, K. S. Harish Kumar, "IoT-Enabled Air Quality Monitoring and Prediction Using Machine Learning: A Cost-Effective Approach," Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 16, no. 3, pp. 514–531, 2025. doi: 10.58346/JOWUA.2025.I3.031.

[4] J. Kustija, D. Fahrizal, M. Nasir, et al., "Revitalizing IoT-based air quality monitoring system for major cities in Indonesia," SINERGI, vol. 28, no. 3, pp. 605–616, 2024. doi: 10.22441/sinergi.2024.3.016.

[5] A. Alwabli, "Federated Learning for Privacy-Preserving Air Quality Forecasting using IoT Sensors," Engineering, Technology & Applied Science Research, vol. 14, no. 4, pp. 16069–16076, 2024. doi: 10.48084/etasr.7820.

[6] S. I. Lopes, C. Orłowski, P. T. B. S. Branco, K. Karatzas, G. Villena, J. Saffell, et al., "Low-Cost Sensor Systems and IoT Technologies for Indoor Air Quality Monitoring: Instrumentation, Models, Implementation, and Perspectives for Validation," Sensors, vol. 25, no. 24, p.

7567, 2025. doi: 10.3390/s25247567.

[7] S. Wawale, C. R. Vijay, S. Suneel, D. B. Darshan, et al., "IoT-Based Real-Time Air Quality Index Monitoring and Predictive Modelling," International Journal of Environmental Sciences, vol. 11, pp. 1050–1059, 2025. doi: 10.64252/jcgecm90.

[8] V. Bongale, K. S. Harish Kumar, "IoT-Enabled Air Quality Monitoring and Prediction Using Machine Learning: A Cost-Effective Approach," Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 16, no. 3, pp. 514–531, 2025. doi: 10.58346/JOWUA.2025.I3.031.