Global Research journal of Natural Science and Technology (Grjnst)
Volume: 03 - Issue 4 (2025)
p.ISSN: 2790-7643 e.ISSN: 2790-7651

Design and Implementation of an IoT-based Air Quality Monitoring System with Mobile App Interface

Muhammad Sobait

Institute of Computer and Software Engineering,

Khwaja Fareed University of Engineering and Information Technology,

Rahim Yar Khan, 64200, Pakistan

Email: *dunkindoc7@gmail.com*

Rauf Ahmad

Swedish College of Engineering and Technology,

Rahim Yar Khan, 64200, Pakistan

Email: *raufahmad39@gmail.com*

Hammad Shahab (Corresponding Author)

Institute of Computer and Software Engineering,

Khwaja Fareed University of Engineering and Information Technology,

Rahim Yar Khan, 64200, Pakistan

Email: *hammad.shahab@kfueit.edu.pk*

ORIC ID: *https://orcid.org/0000-0001-8349-578X*

Abstract:

The climate change and variations in environmental conditions such as temperature, humidity and air quality have increased considerably in recent years. With the advancement of smart technologies and the growing need for real-time, data-driven decision-making, there is a strong demand for automated environmental monitoring systems. Continuous monitoring and representation of environmental data are crucial for safety of humans and environment. This need is addressed by the proposed system which is designed and implemented as an Internet of Things (IoT) based air quality and smoke detection system with mobile application interface. An ESP32 microcontroller is used to monitor the environmental conditions continuously with

the help of DHT11 temperature and humidity sensor, MQ-2 gas sensor and MQ-135 gas sensor. The chosen components are affordable and small, allowing for reliable deployment. Processed data is shown locally on an OLED screen for on-site monitoring and at the same time uploaded to the cloud platform Thing Speak for storage and remote access. A Flask based web application receives the data and uses the threshold-based classification to show safe and hazardous environmental conditions in an interactive dashboard. Moreover, Restful API integration can help to communicate with a mobile application, which means that the remote monitoring can be done in real-time from anywhere. System evaluation shows that manual supervision needs are reduced and real-time visualization of environmental conditions is effective. The proposed system is overall low cost, scalable and efficient for smart environmental monitoring applications.

***Keywords**: Internet of Things (IoT), Environmental Monitoring, Gas Sensors, Flask Web Application, Mobile App*

1. Introduction

In recent years, working and living environmental conditions have gradually worsened, raising safety concerns in areas that undergo work sensitive to air pollutants and smoke [1]. These variations can have ranging consequences from minor discomfort to a dangerous working environment. Thus, undergoing constant automated monitoring in case of a potential hazard is vital for both human life and equipment inefficiencies. As a result of these environmental changes and the need to maintain optimal environmental conditions it is important that a contraption that is both scalable and cost-efficient is developed to ensure on-point and timely response [2]. Conventional monitoring practices largely depend on manual observation and periodic measurements which are always time-consuming and labor intensive [3]. These methods also delay any data transmission, are not viable in areas needing constant supervision and the need for immediate response cannot be addressed. This time delay often leads to inefficient system performance and an increased risk of unfavorable environmental conditions. With the advancement of technology that embeds network connectivity Internet of Thing (IoT) has emerged as a powerful paradigm for enabling real-time data collection all the while making the equipment needed for it affordable and widely deployable across various applications [4]. It allows multiple devices and sensors to be connected at the same time while communicating

and relaying data across the internet for storage and remote accessibility [5]. This opens possibilities that were otherwise hardly accessible enabling cost-effective and scalable remote monitoring systems. The following proposed project uses this advantage and creates a system that monitors key environmental factors like temperature, humidity, air quality and CO2 levels to ensure safe living and working environments. This is made possible with the use of sensors DHT11 (for temperature and humidity), MQ-2 (for CO2 levels/ smoke) and MQ-135 (for air quality) [6]. System utilizes microcontroller ESP32 which is a significant upgrade from its previous widely used counterpart ESP8266 making integration seamless and powerful [7]. Selection of these components is based on reliability, affordability and most importantly ease of integration. The data collected is processed inside the microcontroller, displayed offline using an OLED screen for immediate on-site monitoring and transmitted to a cloud-based platform. ThingSpeak (cloud-based platform used) securely stored the data for it to be fetched through its Read Api making data remotely accessible [8]. This enables the user to monitor environmental changes from anywhere enhancing convenience, control and best for the project management [9]. Furthermore, A Flask-based web application that fetches this data and presents it in an interactive and user-friendly dashboard is also developed. This page processes data applying a threshold-based categorization to convert raw sensor readings into meaningful indicators, such as safe, moderate, or dangerous conditions. Users can access history as far as a week of any or all sensor to identify any patterns and potential problems in their work environment. The data is displayed in multiple ways including normal values such as 24°C, a bar graph and a line graph indicating trends. This Web page can be accessed and read without any technical expertise making the web page viable across multiple types of users. Furthermore, seamless integration to a mobile application is made possible via RESTful Api's providing real-time data access directly from user's pocket. The findings extracted from system testing verify that the suggested Iot-based environmental monitoring system functions continuously, is reliable and efficient even in a real-time environment [10]. The sensors were regularly producing the temperature, humidity and air quality data with steady results and the ESP32 microcontroller was able to handle the data and send it to clouds several times without noticeable delay. The system of categorization based on the thresholds turned out to be very useful as it enabled users to quickly grasp the status of their environment through the use of distinctly color-coded signals. Besides, very little latency was observed by the system when it

came to revising the values of the sensors; this way it ensured the capability of the monitoring almost in real time. The whole system is purposely compact, economical and efficient proving that it is now possible to create advanced environmental monitoring systems even with very simple and easily accessible components. Thanks to the automation of data recording and presenting the project relies less on manual work and at the same time, it becomes more capable of reacting to the changes of the environment [11]. The selected solution increases productivity of the operations but at the same time is helpful to the creation of safe and stable environment thereby it is a sensible and scalable method to be used for the contemporary monitoring requirements [12]. The proposed system addressed the limitations of the existing system that made data acquisition, processing and real-time monitoring very efficient. This system employs a layering method, combining hardware elements to sense and gather data with software parts for processing, storing, and visualizing data.

2. Method and Materials

The proposed IoT-based smart smoke and air quality detection system's architecture is aimed at making data acquisition, processing, and real-time monitoring highly efficient. This system employs a layering method, combining hardware elements to sense and gather data with software parts for processing, storing, and visualizing data. This system's architecture allows persistent monitoring of the environment together with the ability to be scaled and the assurance of being dependable through connecting to the cloud and accessing via the web.

2.1 Hardware Architecture

The hardware architecture consists of an ESP32 microcontroller as the central unit of the monitoring system. The ESP32 is selected as the central processing unit because of its ability to perform significant processing while consuming minimal power and providing built-in wireless (WiFi) capabilities to make it easy to use on Internet of Things (IoT)-based systems. It coordinates the operation of all sensors connected to it and controls the acquiring and transmitting of environmental data. The monitoring system consists of various sensors that provide environmental data. The system's temperature/humidity sensor will provide the environment's temperature and humidity to give context in relation to the current air quality; the MQ (metal-oxide) series of sensors (e.g. MQ-2, MQ-135) will be used for detecting gas/smoke in the ambient air. The MQ-2 sensor is used to detect smoke and combustible gas

whereas the MQ-135 sensor can help to identify a larger number of noxious gases such as ammonia, carbon dioxide, carbon monoxide, and others. All of these sensors produce analog voltage outputs that are proportional to the concentration of gases in each area.

With the built-in analog-to-digital converter (ADC), the ESP32 reads these analog signals and converts them into digital values, allowing them to be processed via programming. An important consideration when designing hardware is calibrating the sensors. By calibrating the sensors, their readings will accurately represent the environmental conditions they are measuring. Good sensor calibration will improve the system's reliability and consistency over time. Figure 1 demonstrates the system hardware in action, fully intact and operational.

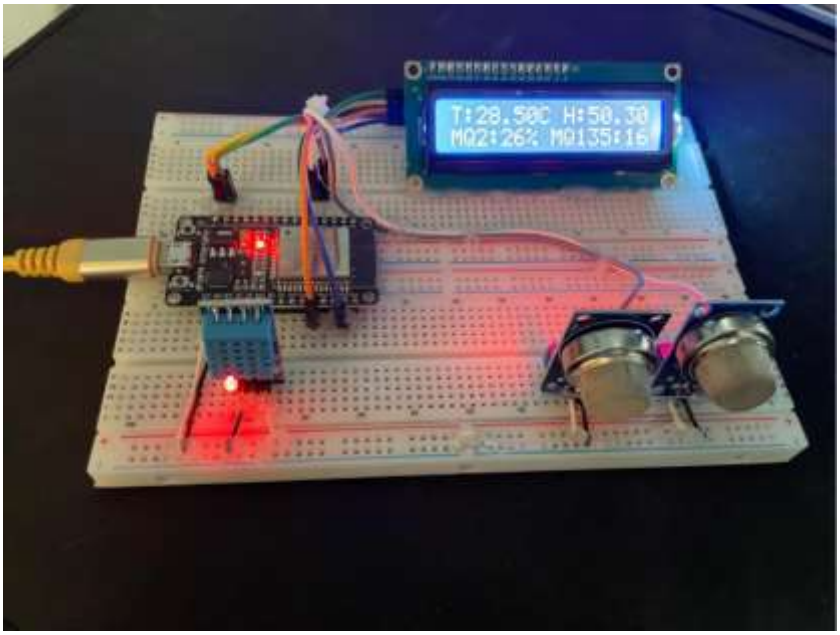

Figure 1: Hardware of the proposed system.

Along with the other devices connected via ESP32's built-in Internet connection by using its onboard Wi-Fi radio module to connect to an external Internet-based computing server or cloud-based service provider, therefore, providing a mechanism for transmitting data wirelessly without the need for any additional networking infrastructure. This data transmission method is designed to be both fast and reliable, allowing very little data to be lost during transmission even in times of poor or intermittent network connectivity. The power supply to the hardware will typically come from either a USB power source or a regulated direct current

(DC) input depending upon the type of installation location. Measures have been put in place (i.e., voltage regulation) to ensure that the electrical power supplied to components is clean and stable. The modular architecture of the hardware also makes it easy to add additional sensors or devices, such as a display module or alarm system, to increase the capabilities and versatility of this entire system.

2.2 Software Architecture

System software architecture is designed to enable efficient data transfer, real-time Processing and smooth user interaction. The system uses a layered approach whereby the embedded device connects to Cloud services and user-facing applications. The core device in the architecture is an ESP32, which is constantly acquiring sensor data and formatting it for transmission. This makes it possible to continually monitor the environment by taking measurements on a scheduled basis. Data is exchanged between the device and external systems via lightweight protocols such as HTTP, which allows for reliable and timely delivery of data. The structured data flow ensures reliable system operation with the least number of delays and the least amount of resources being used. There are several layers of software in which to organize these processes (physical layers), where each layer is responsible for one part of the data flow. The firmware on the ESP32 serves as your lowest physical layer (the point at which your sensor data is received from the sensors, preprocessed, and sent to the cloud). The next physical layer is the configuration of ThingSpeak, which is used for cloud-based storage and real-time visualization of the data. Incoming data from the ESP32 is received and organized by ThingSpeak into various channels so you can easily gain access to it and monitor it in real-time. An intermediate processing layer built on the Flask web application part of this system resides above ThingSpeak. This layer retrieves incoming data from the cloud platform, applies threshold logic to it, and organizes it into a format that could be further processed. The final physical layer under which this system operates is the Android mobile app that serves as the user interface for accessing environmental data remotely, viewing real-time data, and sending alerts to the user via the mobile app. Figure 2 clearly demonstrates these layers in use and their uses.

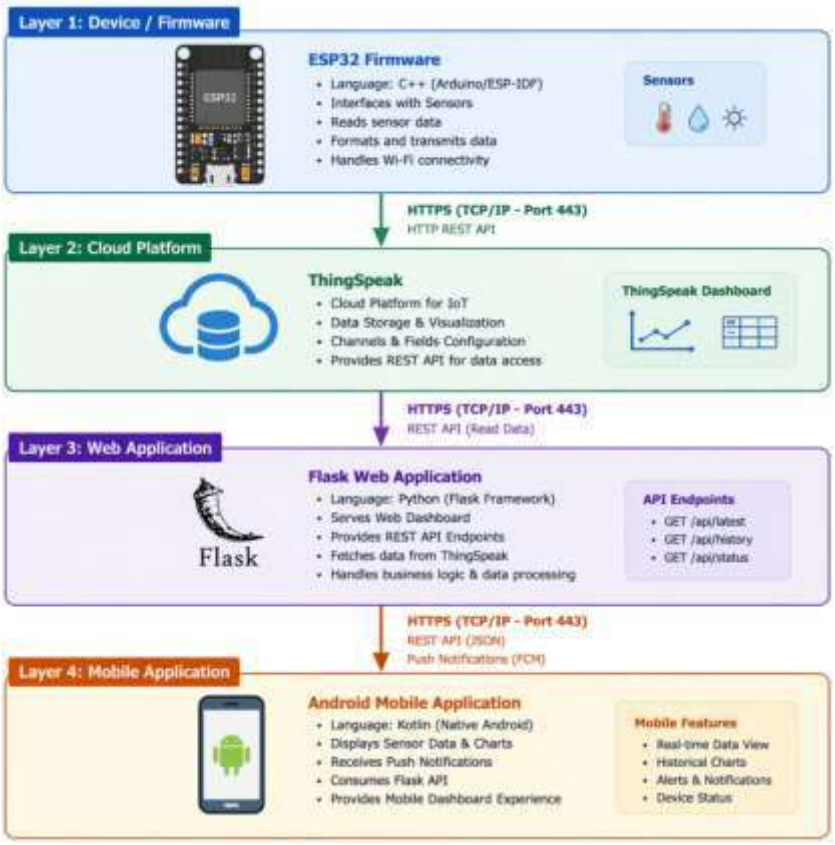

Figure 2: Software Layers

The web application written using Flask serves a crucial purpose as a processing intermediary component of the system, allowing for the retrieval of sensor data from the ThingSpeak API using API requests made by the Flask application, and performing some additional computation on that data prior to returning a response back to the user. This computation may include applying a threshold-based air quality classification to the sensor data, transforming that data into usable formats, and creating returned information that can be utilized by client application software. Furthermore, the Flask application exposes RESTful API endpoints that permit the communication between backend of the system and external platforms, such as the Android application. In essence, the Flask web application provides an interface between the raw sensor data collected from the sensors and the user interface, adding flexibility to the system by providing a common interface for implementing complex logic without modifying the hardware layer. Figure 3 shows the web page dashboard and screenshot of Mobile App.

Figure 3: Website and Mobile App View of proposed system

This design approach is based on being integrated and scalable. All three layers of the software interact with each other through a well-defined interface. Data is passed back and forth between the Android app and the backend using RESTful APIs, allowing for adaptability of the system to multiple client User Interfaces (UIs). Dashboards and Graphical Elements provide a way to visually display environmental conditions to the user. Furthermore, architecture has been designed to allow for expansion (e.g., adding advanced analytics or machine learning models) in the future without requiring significant changes to the infrastructure. The use of layered architecture provides flexibility, maintainability and for the ability of the system to grow to handle more data and more user activity. The summary of component used in the proposed system is given below in Table 1.

Table 1: Summary of component used in the proposed system

Component	Type	Function in System	Key Features
ESP32 Microcontroller	Hardware	Central control and processing unit	Low power consumption, built-in Wi-Fi, ADC support,

			real-time data processing
MQ-2 Sensor	Hardware Sensor	Detects smoke and combustible gases	Analog output, smoke and gas detection
MQ-135 Sensor	Hardware Sensor	Detects harmful air pollutants	Detects ammonia, carbon dioxide, carbon monoxide, and toxic gases
Temperature and Humidity Sensor	Hardware Sensor	Measures environmental temperature and humidity	Provides environmental context for air quality monitoring
Analog-to-Digital Converter (ADC)	Embedded ESP32 Feature	Converts analog sensor signals into digital data	Integrated within ESP32 for sensor interfacing
Wi-Fi Communication Module	Embedded ESP32 Feature	Enables wireless data transmission	Internet connectivity without extra networking hardware
Power Supply Unit	Hardware	Provides stable electrical power to system	USB/DC input support with voltage regulation
ThingSpeak Cloud Platform	Software/Cloud	Stores and visualizes sensor data	Real-time cloud monitoring and channel-based data organization
Flask Web Application	Software Backend	Processes and manages sensor data	Threshold analysis, RESTful APIs, data formatting
RESTful API	Software Communication Layer	Enables communication between backend and mobile app	Structured and scalable data exchange
Android Mobile Application	Software Frontend	User interface for monitoring and alerts	Remote access, real-time monitoring, notification support

Dashboard/Web Interface	Software Visualization	Displays environmental data graphically	Graphs, dashboards, and visualization tools
Cloud Server	Network/Cloud Infrastructure	Supports remote data access and storage	Reliable and scalable data handling

3. Results and Discussion

The designed IoT-based smart smoke and air quality detection system has been implemented and tested under a range of environmental conditions for performance, reliability and responsiveness. The system provides valid readings for the temperature, humidity and air quality by using an integrated sensor and showed consistent operation with the collection of data. Additionally, real time data from the ESP32 to the ThingSpeak cloud platform was transmitted with a small number of delays, which demonstrates that the communication protocol and network configuration worked effectively. Data produced by the sensors will be updated on an ongoing basis and can be retrieved remotely through the web application or Android interface, therefore demonstrating that the system will support real-time monitoring of environmental conditions. Sensors like MQ-2 and MQ-135 recorded significant changes to measure the air, showing quality fluctuations. The sensors recorded an increase whenever introducing cannabis or gas to the sensors. As soon as these changes were made to the sensors, the thresholds in the application program 'Flask' triggered action. After registering the increased measurement, the system was able to categorize the quality of the air based upon the defined levels of quality (good, fair, or poor), which are recorded on the visual dashboard. The visual representations of the data available on the web application provide a user-friendly means to analyze the trends or fluctuations in air quality over the defined periods, thereby increasing the user's situational awareness. The sensors for temperature and humidity provided stable and reliable readings for defining the overall environmental conditions. The Flask-based processing layer contributed greatly to the overall processing capability and functionality of the system by applying logical conditions and formatting the incoming data, thereby making it a far more meaningful presentation and interaction than presenting only raw sensor outputs. The RESTful APIs provided for the backend interactions with the Android application allowing for seamless exchanges between the two (2) applications regardless of the platform of the

application. The Android application has provided the user with a portable and readily available means to monitor the overall air and/or environmental conditions in near real-time. The ThingSpeak platform helped visualize and prove out how well the system performed with real time graphical representations of the data collected. While the ESP32 continuously sent temperature, humidity, and air quality readings to the thingSpeak platform, they were then displayed as a time-series plot in the created channel field as a time-series plot on the ThingSpeak platform. The temperature, humidity, and air quality readings were presented as time-series plots in ThingSpeak's created channel fields, illustrating environmental conditions to be very evident by the spikes and dips in the MQ-2 and MQ-135 air quality sensor readings when smoke or gas pollutants were present. Continuous data updates showed constant communication between the hardware and software platform in the cloud confirming that the environmentally measured sensors continuously received communication with the effective and high-performance condition, while the smiley face pattern provided an opportunity to track the behavior of the overall system over time. Additionally, being able to access the graphs from anywhere allows users to have access to the environmental conditions anytime and increases the usability of the overall system. Figure 4 shows graphical representation in ThingSpeak.

Figure 4: ThingSpeak cloud dashboard view

Although there was successful implementation, limitations were identified during testing. The MQ series sensors were generally effective; however, they are sensitive to external environmental factors, including humidity and temperature, which can affect sensor accuracy and necessitate adjusting the calibration from time to time. In addition, network instability sometimes resulted in minor delays in data transmission, which could limit the ability to respond immediately in critical situations. Another important consideration for long-term deployment of sensors is the stability of the power supply, since power supply fluctuations could impact performance and consistency of data retrieved from the sensors. The table 2 below shows the proposed IoT based smoke and air quality monitoring system with the existing solutions in terms of monitoring capability, connectivity and mobile application support. It demonstrates that the proposed system offers more sophisticated real-time monitoring, cloud-based and mobile application integration than traditional and basic systems.

Table 2: Comparison of proposed system with existing systems

System	Monitoring Type	Connectivity	Mobile App/ Website	
Proposed IoT System	Real-time smoke + air quality monitoring	Wi-Fi + Cloud (ESP32, ThingSpeak)	Yes	[13]
Traditional Smoke Alarm	Smoke detection only	None	No	[14]
Standalone Gas Sensor System	Single gas detection	None	No	[15]
Basic Air Quality Monitor	Limited air quality monitoring	Limited / local only	Rare	[16]
Smart Home Air Monitor	Air quality monitoring	Wi-Fi / Bluetooth	Yes	[17]

4. Conclusion

The developed IoT based smart smoke and air quality detection system has shown to be an efficient method for real-time monitoring of the environment by utilizing embedded hardware, cloud computing, and contemporary software technologies. Accurate data collection and transmission of temperature, humidity and air quality data was achieved and feedback was given through web and mobile applications. The system demonstrated from its detection of harmful gases and ability to change to environmental variables is indicative of its capabilities to promote safety and awareness in numerous areas. The development results showed that the overall performance of the system is reliable, extensible and applicable to real-world use, although there were some limitations in terms of sensitivity and network dependency. With additional enhancements to the system proposed here, it can lay the groundwork for even more sophisticated and intelligent environmental monitoring systems.

Conflict of interest

The authors declare no conflict of interest.

Data availability

Data will be made available on request.

Author contributions

Muhammad Sobait*:* Conceptualization, Investigation, Data curation, Methodology, Formal analysis, Writing – original draft, review & editing. Rauf Ahmad*:* Resources, Formal analysis, review & editing. Hammad Shahab*:* Supervision, Formal analysis, Methodology, Resources, Visualization, Writing – review & editing.

References

[1] I. Manisalidis, E. Stavropoulou, A. Stavropoulos, and E. Bezirtzoglou, "Environmental and Health Impacts of Air Pollution: A Review," *Front. Public Health*, vol. 8, no. February, pp. 1–13, 2020, doi: 10.3389/fpubh.2020.00014.

[2] W. Paper, "Transforming the Hybrid Cloud for Emerging AI Workloads," pp. 1–70, 2024.

[3] M. A. Musarat, A. M. Khan, W. S. Alaloul, N. Blas, and S. Ayub, "Automated monitoring innovations for efficient and safe construction practices," *Results in Engineering*, vol. 22, p. 102057, Jun. 2024, doi: 10.1016/J.RINENG.2024.102057.

[4] O. Vermesan *et al.*, "Internet of Things beyond the Hype: Research, innovation and deployment," *Building the Hyperconnected Society: Internet of Things Research and Innovation Value Chains, Ecosystems and Markets*, pp. 15–118, 2015, doi: 10.1201/9781003337454-3.

[5] R. Piyare and S. R. Lee, "T Owards I Nternet of T Hings (I O T S): I Ntegration of W Ireless S Ensor N Etwork To C Loud S Ervices for D Ata C Ollection and," vol. 5, no. 5, pp. 59–72, 2013.

[6] L. M. Easterline, A. A. Z. R. Putri, P. S. Atmaja, A. L. Dewi, and A. Prasetyo, "Smart Air Monitoring with IoT-based MQ-2, MQ-7, MQ-8, and MQ-135 Sensors using NodeMCU ESP32," *Procedia Comput. Sci.*, vol. 245, pp. 815–824, Jan. 2024, doi: 10.1016/J.PROCS.2024.10.308.

[7] F. Serepas, I. Papias, K. Christakis, N. Dimitropoulos, and V. Marinakis, "Lightweight Embedded IoT Gateway for Smart Homes Based on an ESP32 Microcontroller," *Computers*, vol. 14, no. 9, pp. 1–19, 2025, doi: 10.3390/computers14090391.

[8] P. Jebane, P. Anusuya, M. Suganya, S. Meena, and M. D. A. Priya, "Iot Based Health Monitoring and Analysing System Using Thingspeak Cloud & Arduino," *International Journal of Trendy Research in Engineering and Technology*, vol. 5, no. 4, pp. 1–6, 2021.

[9] J. Nawaz, H. Shahab, M. Ziaullah, H. Raza, and M. U. Sardar, "The Influence of project manager's motivation on project success through developing trust and knowledge sharing," *Information Management and Computer Science (IMCS)*, vol. 3, no. 2, pp. 22–24, 2020.

[10] M. Alam, M. M. Islam, N. M. Nayan, and J. Uddin, "An IoT Based Real-Time Environmental Monitoring System for Developing Areas," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 52, no. 1, pp. 106–121, 2025, doi: 10.37934/araset.52.1.106121.

[11] S. El-Omari and O. Moselhi, "Integrating automated data acquisition technologies for progress reporting of construction projects," *Autom. Constr.*, vol. 20, no. 6, pp. 699–705, Oct. 2011, doi: 10.1016/J.AUTCON.2010.12.001.

[12] A. Hassanat Adepoju, B. Austin-gabriel, O. Hamza, and A. Collins, "Advancing Monitoring and Alert Systems: A Proactive Approach to Improving Reliability in Complex Data Ecosystems," *IRE Journals*, vol. 5, no. 11, pp. 281–294, 2022.

[13] L. M. Easterline, A. A.-Z. R. Putri, P. S. Atmaja, A. L. Dewi, and A. Prasetyo, "Smart air monitoring with IoT-based MQ-2, MQ-7, MQ-8, and MQ-135 sensors using NodeMCU ESP32," *Procedia Comput. Sci.*, vol. 245, pp. 815–824, 2024.

[14] O. Alsamrai, M. D. Redel-Macias, and M. P. Dorado, "Real-time intelligent monitoring of outdoor air quality in an urban environment using IoT and machine learning algorithms," *Applied Sciences*, vol. 15, no. 16, p. 9088, 2025.

[15] P. Saraswathi, M. Prabha, B. Dhiyanesh, R. A. Kumar, J. B. Ahamed, and M. M. Ismail, "IoT-Powered Solution for Proactive Air Quality Management," in *2024 5th International Conference on Image Processing and Capsule Networks (ICIPCN)*, IEEE, 2024, pp. 754–758.

[16] M. Sinambela, M. R. Nugraha, A. Widodo, J. S. P. Kadir, A. M. Yasir, and T. W. Aji, "Iot-based air quality monitoring system design and development using esp32," in *2024 Ninth International Conference on Informatics and Computing (ICIC)*, IEEE, 2024, pp. 1–6.

[17] M. V. Guvvala and D. Ch, "ThingSpeak based air pollution monitoring system using raspberry Pi," in *2023 3rd Asian Conference on Innovation in Technology (ASIANCON)*, IEEE, 2023, pp. 1–6.