A SCALABLE INTERNET OF THINGS-BASED SYSTEM FOR DISTRIBUTED AIR QUALITY MONITORING AND DATA MANAGEMENT

MARVIC GRIMA, REBECCA CARUANA & ALFRED MICALLEF
Department of Geosciences and Geosciences Observatory (Gozo), Faculty of Science, University of Malta, Malta

ABSTRACT

This paper reports on the design, implementation and evaluation of a distributed air quality monitoring system that addresses key challenges in data acquisition, transmission, storage and accessibility. A literature review guided the selection of air quality parameters, sensor technologies and modern communication and storage strategies, with a focus on recent advancements in Internet of Things and Big Data frameworks. A prototype network was developed for deployment across multiple sites in the Maltese Islands. The system integrates analogue, serial and internet protocol (IP)-based sensors through Espressif Systems Model 32-based gateway devices, enabling standardised data acquisition and user datagram protocol-based transmission over IP networks. A custom flat-file storage engine was implemented to manage high-frequency time-series data efficiently, supporting fast, indexed retrieval without the need for traditional database systems. A web-based interface enables remote data visualisation and management. Initial testing with 1 year data demonstrated robust performance and scalability. Future improvements will focus on enhancing usability, protocol support and system security. This work offers a modular, scalable solution for long-term environmental monitoring, leveraging contemporary technologies to support multi-site air quality data collection and analysis.
Keywords: air quality monitoring system, data management, air pollution networks.

1 INTRODUCTION

Air quality monitoring has emerged as a critical component of environmental management and public health policy, given the well-established link between air pollution and adverse health outcomes, including respiratory and cardiovascular diseases [1]. Accurate and continuous monitoring of air pollutants is essential for assessing environmental quality, informing regulatory frameworks, and guiding mitigation strategies, particularly in densely populated and industrialised regions.

Effective monitoring of air quality necessitates the deployment of a network of sensors capable of measuring multiple atmospheric parameters at high temporal resolution, often on the order of a 1-minute interval or less. Ensuring the accuracy and reliability of these measurements depends not only on the selection of appropriate sensors but also on the design of an integrated data acquisition and management system. Such a system must accommodate the heterogeneous nature of sensor interfaces, support the high frequency and volume of data generated, and provide scalable storage and real-time access to users distributed across multiple locations.

This work discusses a comprehensive, distributed air quality monitoring system that addresses these challenges through a robust architecture emphasising scalability and modularity. The system integrates diverse sensor types via multiple communication protocols, employs a time-series-optimised data ingestion and storage framework, and includes a web-based interface for remote access and analysis. By unifying heterogeneous sensor inputs with real-time, high-resolution data availability, the system delivers a practical solution for urban and industrial air quality monitoring. Its combined strengths in flexible sensor integration, efficient data handling, and user accessibility contribute to the development of next-generation environmental monitoring infrastructure.



WIT Transactions on Ecology and the Environment, Vol 266
© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)
doi:10.2495/AIR260111

2 SENSOR ARCHITECTURES FOR AIR QUALITY MONITORING

Effective air quality monitoring depends on both appropriate sensor technologies and robust data acquisition systems. The choice of sensors must be tailored to detect specific pollutants with precision, while communication protocols, interface electronics and data storage mechanisms must enable seamless data collection, transmission and management. This section begins by reviewing key sensor technologies used to detect some of the major air pollutants and environmental parameters. It then outlines the common hardware and software frameworks used to interface with these sensors, transmit data, and support scalable analysis platforms.

2.1 Sensor technologies for key air pollutants

Air quality monitoring has long focused on pollutants such as carbon monoxide (CO), carbon dioxide (CO_2), methane (CH_4), sulphur dioxide (SO_2), nitrogen oxides (NO and NO_2), ozone (O_3), and particulate matter (PM), along with environmental factors like temperature, humidity and atmospheric pressure [2]–[4]. Monitoring systems often deploy a combination of fixed and mobile sensor nodes that continuously measure pollutant levels and relay data to a central server for processing and analysis.

Different sensing technologies are employed depending on the pollutant of interest. Both low-cost Internet of Things (IoT) sensors and high-precision instruments are used across studies. For instance, CO, a harmful gas linked to respiratory illness [5], has been monitored using a variety of sensors. Liu et al. [6] used the MicroChemical Systems (MiCS) sensor, which detects CO and volatile organic compounds (VOCs) in the 1–1000 ppm range via a resistance-based signal. Postolache et al. [3] employed the Figaro Model 203 Taguchi Gas Sensor (TGS203), which measures CO levels from 50 to 1000 ppm. Kumar and Jasuja [7] used the Hanwei Electronics Model 9 Metal Oxide Gas Sensor (MQ9) capable of detecting CO (10–1000 ppm) and other combustible gases like CH_4. Parmar et al. [5] and Nasution et al. [8] preferred the MQ7, which also uses a resistance-based output but provides enhanced sensitivity to CO compared to the MQ9, while remaining similarly priced. Overall, commercially available CO sensors vary widely in capability, with detection ranges spanning from 0–500 ppm to as wide as 0–10,000 ppm.

Other than CO, there are other pollutants of interest such as CO_2 due to the fact that anthropogenic sources increase its natural concentration [6] and it being a greenhouse gas. Neethirajan et al. [9] used the TGS4160, which measures in the range of 300–5000 ppm with ±20% accuracy. The sensor produces a millivolt-level analogue output, requiring a high-impedance buffering circuit.

CH_4, a flammable gas commonly used in domestic settings, has also been widely monitored. Postolache et al. [3] employed the TGS842, which outputs a resistance-based analogue signal over a 0–10,000 ppm range.

SO_2, a colourless gas known to cause respiratory issues, was measured by Rivai et al. [10] using the Alphasense SO_2-BF electrochemical sensor. This sensor operates over a range of 0–100 ppm with a sensitivity of 300–480 nA ppm^{-1}.

O_3 plays a dual role in the environment: protective in the upper atmosphere but harmful at ground level due to its impact on respiratory health. Nasution et al. [8] used the MQ131 sensor, available in low concentration (10–1000 ppb) and high concentration (10–1000 ppm) variants, both with voltage-based analogue outputs.

PM consists of fine airborne particles that can trigger allergies and severe respiratory conditions [7]. Common sensors include the Plantower Technology Model 5005 Particulate Matter Sensor (PMS5005) [11], Samyoung Electronics Model 501A Dust Sensor Module

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

(DSM501A) [7], and Sharp Model 1010 Gas Particle 2 Series Dust Sensor (GP2Y1010) [8]. These sensors are typically capable of detecting particulate concentrations as low as 1 μg m^{-3}. These sensors operate by shining a light – or in the case of the PMS5005, a laser – through air and detecting the amount of light scattered by airborne particles. While cost-effective options like the DSM501A and GP2Y1010 provide analogue outputs, more advanced models such as the PMS5005 feature an on-board microcontroller and support digital serial communication, enhancing data accuracy and handling.

Temperature, humidity and atmospheric pressure are important factors that influence air quality. Combined temperature–humidity sensors are commonly available and are used widely in research. For instance, Zheng et al. [11] employed the Sensirion Model 20 Sensor for Humidity and Temperature (SHT20), which measures temperatures from −40°C to 120°C (±0.3°C) and relative humidity (RH) from 0% to 100% (±3%), providing digital output via the Inter-Integrated Circuit (I^2C) protocol. Banhazi [12] utilised the E+E Elektronik Model EE-16, which is a platinum-thermistor/capacitive sensor covering a temperature range of 0°C to 50°C and RH from 10% to 95% (±3°C for temperature; ±3% for RH), with analogue output. In contrast, Postolache et al. [3] used discrete sensors for each parameter, namely, the Smartec Model SMT160-30 for temperature (−45°C to 130°C; ±0.7°C; analogue) and the Humirel Model HM1500 for humidity (0% to 100% RH; ±3% fast response, analogue). Atmospheric pressure has been monitored with the Bosch Model 180 Barometric Pressure Measurement Sensor (BMP180), a piezo-resistive sensor, with a 0.01 hPa resolution and digital I^2C communication [7].

In summary, sensor technologies differ in cost, sensitivity, output format, and robustness. The optimal sensor choice depends on required precision, environmental conditions, and system integration constraints.

2.2 Sensor interfaces, communication protocols and storage systems

Many air quality sensors lack on-board processing capabilities, so most studies employ a companion microcontroller to collect raw signals, perform basic pre-processing, and transmit the data to centralised processing and storage systems. Sensors generate either analogue or digital outputs. Among the analogue outputs, voltage signals are the most common because these can be read directly by a microcontroller's analogue-to-digital converter (ADC) [8]. Resistance-based sensors are also frequently used and are typically incorporated into a voltage-divider circuit, which converts changes in resistance into measurable voltage variations [3]. A few sensors provide current outputs, which must first be converted to voltages using either a resistor or a transimpedance amplifier [10]. Sensors with digital outputs often incorporate internal microcontrollers and support standard protocols such as Recommended Standard 232 (RS-232) and I^2C [2], [7], [12]. Others rely on proprietary protocols that require custom decoding at the receiving end [6], [13].

The dominant architecture in the existing literature features small microprocessors at the sensor node, responsible for acquiring, pre-processing, and re-transmitting data to a central system for further analysis. Earlier implementations typically employed 8- or 16-bit microcontrollers such as the 8051, STMicroelectronics ARM Cortex-M (STM32) series, and Texas Instruments Mixed Signal Processor (TI-MSP) family [2], [6], [11]. More recent studies have shifted toward IoT-oriented platforms like Arduino, Node Microcontroller Unit (NodeMCU), Espressif Systems Model 8266 (ESP8266), and the Raspberry Python Interpreter (Raspberry PI), which offer low-cost processing, built-in connectivity, and strong community support [4], [5], [7].

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

Wi-Fi is the most used transmission method, valued for its internet connectivity and ease of deployment [3], [5]. Earlier systems relied on the Transmission Control Protocol (TCP), while more recent work favours the User Datagram Protocol (UDP) for latency-sensitive, real-time data streams [14]. Additional transmission methods include cellular networks (General Packet Radio Service (GPRS), Global System for Mobile Communications (GSM), and Third Generation Mobile Network (3G)) [4], [6], Bluetooth and proprietary Radio Frequency (RF) protocols [2], [11], and manual retrieval from local storage [12]. Mobile sensing platforms often integrate Global Positioning System (GPS) modules to enable geolocation tagging of sensor data [2].

Relational databases such as My Structured Query Language (MySQL) were commonly used in early systems to store sensor readings [6]. In more recent research, scalable alternatives have been adopted, including NoSQL databases [5], custom-optimised file engines [12] and cloud-based IoT platforms [10]. These modern platforms are designed to manage high-volume, real-time sensor streams while enabling fast access and efficient processing. Big Data architectures have become a foundational component of contemporary air quality monitoring systems [15].

3 SYSTEM REQUIREMENTS

This section outlines the functional and non-functional requirements necessary for the implementation of a distributed air quality monitoring network. The system must integrate seamlessly with a wide range of air quality sensors, transmit high-frequency environmental data without loss, and provide long-term data access for users in multiple locations. To support continuous monitoring and long-term scientific research, the infrastructure must be scalable, resilient, and capable of high-performance.

Sensors typically communicate using one of three methods: analogue signals (voltage or current), digital serial protocols (RS-232 or RS-485), or Internet Protocol (IP)-based connections. The proposed system must support all three interface types. Sensors usually transmit readings at 1-minute intervals, either based on an internal timer or in response to polling, and both triggering modes must be supported.

Given the expected frequency of readings and the number of deployed sensors, the system must be capable of efficiently ingesting, archiving and retrieving large volumes of time-series data.

3.1 Architecture and data ingestion

The system architecture defines the complete data path, starting from each sensor and ending at the central server. In the proposed deployment, two monitoring stations transmit data to a central server hosted at a third location. The system must therefore operate with a distributed topology, ensuring timely and accurate delivery of data from remote nodes.

Digital instruments communicate using either IP-based or serial connections. When IP connectivity is available, the server can interface directly with the sensor using its native protocol. However, for remotely located serial-only devices, direct wiring is impractical. In such cases, a serial-to-IP bridge must be used to relay data to the server.

Analogue sensors require additional signal conditioning. These devices output a continuously varying voltage proportional to the measured parameter. This signal must pass through a conditioning stage before reaching an ADC. Once digitised, the sensor readings are transmitted over IP to the central server for further processing and storage, ensuring a consistent transmission method across all sensor types.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

3.2 Data storage and archiving

Efficient data storage and archiving are essential for preserving the large volume of data produced by the network and making it readily accessible for analysis. The system must support a large number of sensors generating readings at fixed intervals, resulting in a continual and substantial increase in stored data. A preliminary volumetric analysis indicates that each sensor could potentially produce gigabytes to terabytes of data over its operational life.

A scalable, high throughput archiving engine is required, capable of handling concurrent input from multiple sources while enabling fast, indexed retrieval. The storage system must not rely solely on the main memory; instead, it should utilise persistent storage that supports long-term retention and growth.

In addition to storing data efficiently, the system must ensure accessibility. Users across multiple locations may require real-time or near-real-time access to the data. To support this, a web-based, multi-threaded interface is required, enabling low-latency, concurrent access for multiple users.

4 SYSTEM DESIGN

Efficient collection, storage, and retrieval of sensor-generated data are critical for air quality research. To address these needs in a distributed sensor network, the system employs IP-based data communication, enabling seamless integration with modern networking equipment and existing Internet infrastructure.

The system is developed on the C#.NET platform, chosen for its robustness, widespread adoption, and compatibility with advanced development tools and server technologies.

The following subsections describe the system's key components. Section 4.1 outlines how data is acquired and managed from the instruments. Section 4.2 details the storage and retrieval mechanisms used for handling the collected data. Finally, Section 4.3 discusses how users can access and interact with the archived data.

4.1 Sensor communication and gateway design

Air quality instruments vary widely in their communication methods and supported protocols. To support distributed sensor placement, gateway devices are deployed to acquire data periodically, convert it into IP-compatible formats, and transmit it to a central server. This architecture allows for future scalability in terms of both sensor types and communication protocols.

A lightweight and efficient protocol such as UDP is used for data transmission. UDP is routable, incurs minimal resource overhead, and supports payload sizes that exceed the requirements of this application. Since sensor data typically consists of a sensor ID and a few readings, UDP's connectionless nature makes it ideal. In environments with unreliable connectivity, such as remote wireless links, an optional acknowledgment and retransmission mechanism can be enabled to ensure delivery.

The Espressif Systems Model 32 (ESP32) platform was selected to implement these gateways due to its hardware capabilities, low cost, compact size, and ease of development. It features built-in Wi-fi, Bluetooth, an IP stack, and versatile input/output (I/O) support for both analogue and digital interfaces. Programming is done using the widely adopted Arduino Integrated Development Environment (IDE), streamlining development through its simplicity and community support.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

For network connectivity, the built-in Wi-Fi module enables easy and flexible deployment. Where more stable or secure connections are needed, wired networking is available through hardware add-ons. In applications involving IP-capable instruments, an Ethernet module can provide Local Area Network (LAN) connectivity, freeing the Wi-Fi module for upstream server communication. In such configurations, the ESP32 acts as a protocol translator and gateway, supporting up to eight concurrent TCP connections.

The ESP32 platform supports a wide range of sensor and instrument types. Digital instruments may use native IP connectivity, often over TCP, or communicate using serial protocols such as RS-232 or RS-485. This platform includes three on-board Universal Asynchronous Receiver/Transmitters (UARTs), two of which are available for simultaneous use, enabling direct communication with multiple serial-based devices.

For analogue instruments, which typically output voltage signals in the millivolt to volt range, the ESP32 includes multichannel 12-bit ADCs. However, some ADC channels are limited when Wi-Fi is active, and the built-in resolution may be insufficient for high-precision applications. In such cases, an external high-resolution ADC such as ADS1232 is recommended. It supports two channels at 24-bit resolution and provides significantly enhanced performance for capturing fine-grained analogue signals.

4.2 Data storage and management layer

The proposed solution is designed to handle periodic readings from a large number of sensors, potentially collected every minute, which can generate substantial data volumes over time. Traditional relational databases often suffer performance degradation in such scenarios, especially due to frequent updates, indexing overhead, and large-scale retrieval operations. However, the time-sequenced and independent nature of the sensor data streams allows for a more efficient storage approach. Since there are no inter-sensor relationships, the data can be stored in ordered flat files without requiring complex indexing. Each sensor stream is processed independently, leveraging modern multicore processors and enabling fast access through techniques such as binary search.

To implement this, a flat-file-based storage engine is proposed, structured around a DataReading class. Using Object-Oriented Programming (OOP) principles such as inheritance and polymorphism, a base class manages shared attributes (e.g., timestamps) and common methods (e.g., read/write operations), while subclasses represent specific instruments with their unique fields. Each instance serialises its data to a designated file, eliminating the need for traditional indexing while ensuring efficient access through structured storage.

4.3 User access via web interface

Given the physical distribution of both sensors and users, a web-based interface is the most effective method for accessing the system. This approach enables authorised users to view, analyse, or download sensor data from any location via the Internet, providing flexibility and convenience.

By maintaining a centralised back end, the system simplifies maintenance, enhances data security, and ensures consistent access control. Additionally, the web interface can be extended to support real-time data visualisation, user authentication and administrative tools for managing devices and data streams.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

5 IMPLEMENTATION

The proposed system is implemented using modern technologies that support scalability, modularity and efficiency. The core implementation is structured around four main components. First, the network infrastructure enables reliable data transmission across both local and public networks. Second, the instrument interface acts as a bridge between physical sensors and the system, standardising communication and data collection. Third, the data storage engine manages incoming readings, ensuring efficient, time-ordered storage and fast querying capabilities. Finally, the user interface offers a web-based platform for system configuration, monitoring and data management. Each of these components is described in the following subsections.

5.1 Network infrastructure

The system utilises both UDP packets and TCP connections (the latter used by HyperText Transfer Protocol (HTTP) web servers) to enable communication over an IP network. This approach offers flexibility to operate over local networks, the public Internet, or a combination of both.

In this implementation, a hybrid approach is adopted: each location utilises a LAN, while port forwarding enables access to specific services over the Internet. This architecture is illustrated in Fig. 1, which shows the interactions among key system components, including servers, clients, access points, and sensors.

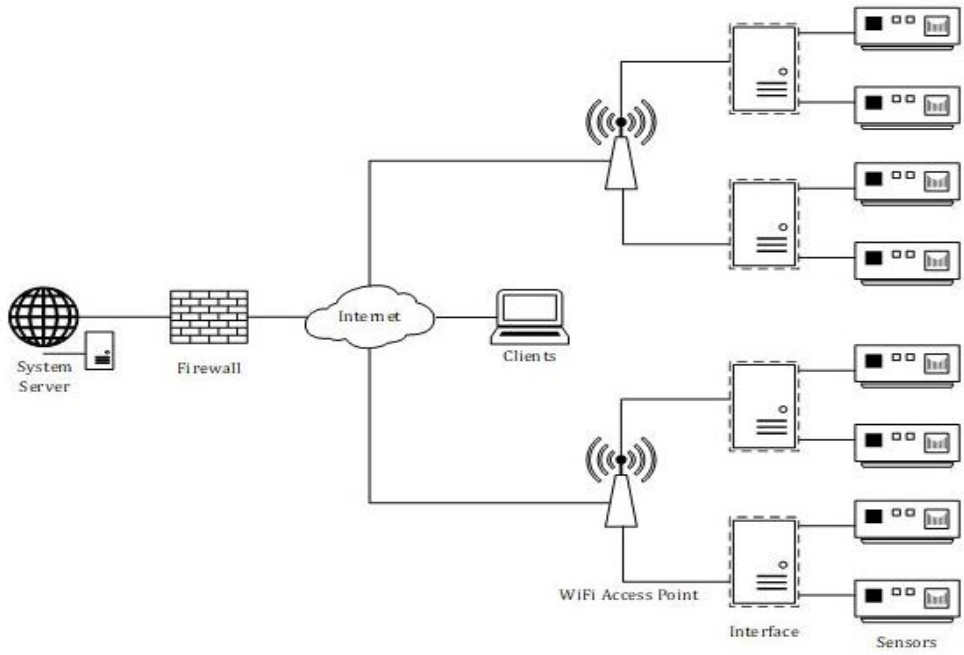


Figure 1: Schematic diagram of the system's network infrastructure, illustrating communication between the server, clients, access points, and sensor interfaces across both local and public networks.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

5.2 Instrument interface

The instrument interface is responsible for periodically collecting readings from various instruments and transmitting them to the central server. It serves as an abstraction layer between the diverse hardware and communication protocols used by the sensors and the standardised UDP protocol expected by the server.

To maximise installation flexibility, Wi-Fi is employed for server connectivity, reducing the need for physical cabling and enabling the use of public Wi-Fi networks. Sensor data is transmitted to the server using unicast UDP at regular intervals. For simplified initial setup, the device operates as a wireless access point upon first use, allowing users to access a configuration screen directly. Fig. 2 illustrates this configuration interface, which provides options for setting up network parameters, ADC settings, and sensor calibration. Available configuration options vary depending on the device's hardware.

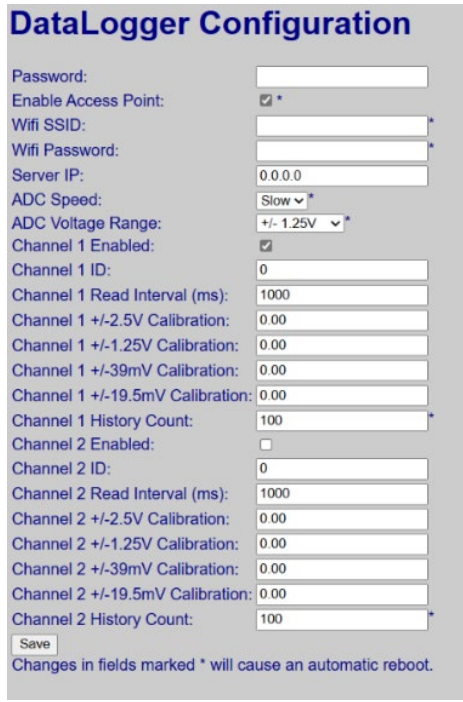


Figure 2: Configuration interface for the data logger, showing options for Wi-Fi setup, ADC settings, and multi-range sensor calibration parameters to ensure accurate data acquisition and transmission.

The interface features a modular hardware and software design, allowing scalable expansion and integration of additional connectivity mechanisms without significant modifications to the core system. A basic four-LED status display provides visual feedback: red for system power, blue for Wi-Fi connectivity, yellow for instrument communication, and green for UDP communication – facilitating quick diagnostics and system monitoring.

For digital instruments, connectivity is typically established via Ethernet or Serial communication. If Ethernet is used, a Serial Peripheral Interface (SPI) Ethernet module is

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

integrated into the device hardware. For Serial communication, one of the on-board UARTs of the ESP32 microcontroller is used.

For analogue instruments, although the ESP32 includes a built-in 12-bit multichannel ADC, a more robust ADS1232 ADC chip was chosen based on the typical voltage ranges of connected sensors. The ADS1232 provides 24-bit resolution and superior noise tolerance, which is crucial for instruments producing outputs in the millivolt range. It supports two analogue channels; for applications requiring more, the four-channel ADS1234 variant, may be used.

A simple graphing interface is also provided to visualise recent sensor readings (with configurable history window) and assist in calibrating ADC values against a known precision reference instrument. Fig. 3 shows an example of this interface. In the displayed output, the system captures a stable signal of approximately 0.02921 V on Channel 1, demonstrating both the resolution and reliability of the ADC process. This signal is measured using a custom-built ADC circuit, shown in Fig. 4, designed to accurately acquire low-voltage analogue signals for digital processing. This circuit interfaces directly with the data logging software to record real-time voltage values.

5.3 Data storage engine

The data storage engine is implemented in C# as a separate class library integrated into the main server. This modular design enhances maintainability and encourages code reuse where applicable. The primary interface to the storage engine is the SensorDB class, which manages individual DataFile instances. Each DataFile corresponds to a specific Sensor.

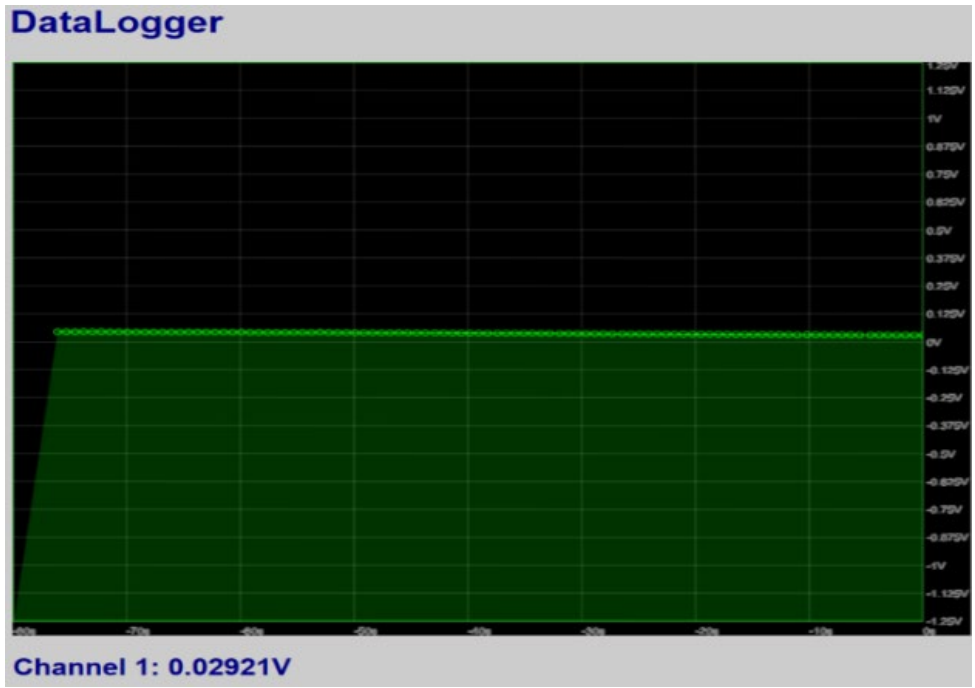


Figure 3: Real-time voltage reading (0.02921 V) from Channel 1 as displayed by the data logging software.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

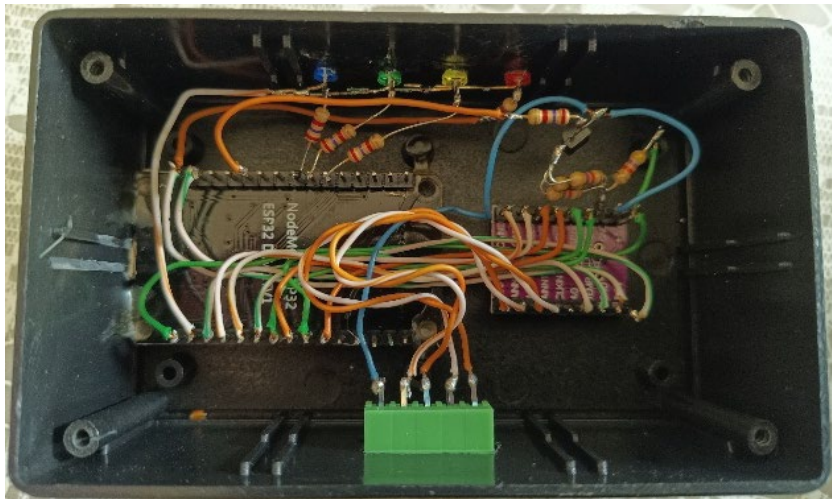


Figure 4: Custom-built ADC circuit setup used to interface analogue input signals with the system.

Sensor data is stored as a sequence of fixed-length records, written in the order received. This creates a naturally time-ordered dataset, removing the need for index files or extra sorting logic. Since queries are primarily time-based, this structure provides both simplicity and strong performance. Each DataFile functions independently, with no relational links to other files, as each handles readings from a single sensor.

The system uses OOP principles, including inheritance and polymorphism, to support the creation of specialised subclasses tailored to different sensor types and their associated data fields. This architecture is illustrated in the class diagram, shown in Fig. 5.

Efficient time-based retrieval is enabled by the Seek function in the DataFile class, which uses a binary search algorithm. The fixed record size and chronological storage order allow the system to scan only the timestamp portion of each record when locating a specific time, significantly improving search performance.

The system supports virtually unlimited file sizes, constrained only by the available physical storage and the file addressing limits of the operating system, which on Windows allows up to 2^{64} bytes.

5.4 User interface

The user interface, shown in Fig. 6, is implemented as a web-based platform using ASP.Net. It features a simple and user-friendly design, with a sidebar navigation menu on the left side of the screen providing access to core functionalities. When a function is selected, the corresponding screen or content is displayed on the right.

Key functions include the management of stations and sensors. Sensors are organised by stations, with each station representing a specific geographic location that groups multiple sensors. Each sensor within a station has its own record, is assigned a unique ID, and can be configured with various settings. As data is received, sensor readings are automatically logged to their respective files, enabling the generation of reports, graphical plots, and other analytical outputs.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)



Figure 5: Class diagram of the data storage engine, illustrating the core components (SensorDB, DataFile and Sensor) and their relationships. Specialised subclasses represent different sensor types, demonstrating the use of inheritance and polymorphism to enable extensible, type-specific data handling.

The navigation menu also includes access to the inventory and scheduled tasks. The inventory displays information about available instruments, including make, model, serial number, and inventory number. The scheduled tasks function shows upcoming calibrations and other required maintenance. A data import feature allows users to upload historical data from tab-separated files, while a data export function enables extraction of system data. Additionally, a notification feature alerts users to important events, such as sensor failures or disconnections.

6 CONCLUSION

This study presents a centralised, scalable, and efficient system for the acquisition, storage, and analysis of air quality data collected from a distributed sensor network across the Maltese Islands. By integrating diverse sensor types through standardised communication protocols and utilising modern technologies such as IoT devices and high-performance data storage, the system ensures reliable, high-frequency data collection and ensures long-term accessibility. Its modular architecture supports future scalability and adaptation, while the web-based user interface enables real-time monitoring and management by multiple users across various locations. Altogether, this system addresses key challenges in distributed environmental monitoring and offers a robust platform to support both scientific research and public health initiatives.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

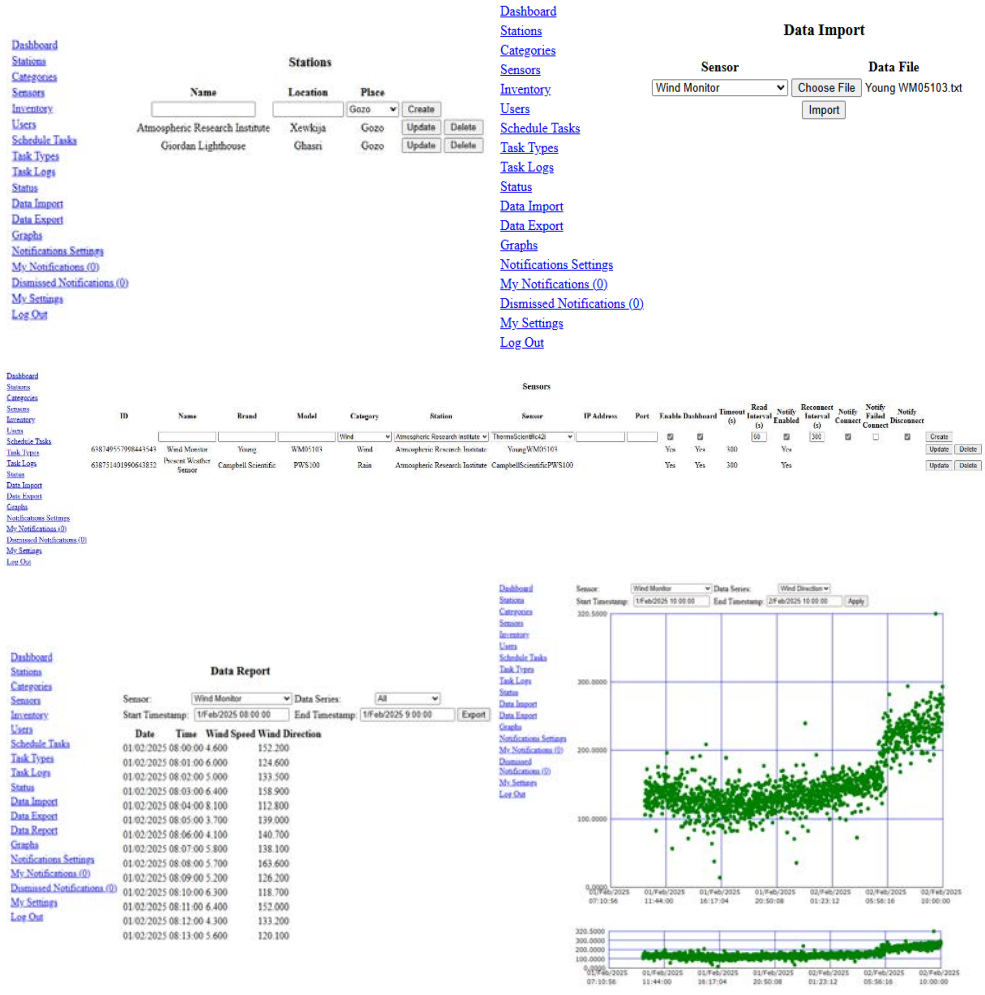


Figure 6: Screenshots of the web-based user interface, showcasing key features including station and sensor management, data import, numerical data display, and graphical data visualisation.

7 FURTHER WORK

While the current user interface is functional, it remains minimalistic and would benefit from enhancements such as graphical buttons, dropdown menus, and improved navigation elements to create a more intuitive and engaging user experience. Future development should prioritise refining the interface to better support user interaction and accessibility.

Expanding the implementation and evaluation of additional communication protocols and interfaces will improve compatibility with a broader range of sensor types, thereby extending the system's applicability and sensor network coverage.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

To strengthen data security, it is recommended to implement encryption for data in transit. This would safeguard sensor readings from potential interception or tampering during transmission between gateways and the central server.

Although the storage engine has demonstrated robust performance over a 1-year dataset, further testing with significantly larger datasets (e.g., spanning 20 years or more) is necessary to evaluate long-term scalability and performance. Similarly, the graph rendering engine should undergo stress testing and potential optimisation to ensure responsiveness when visualising large volumes of historical data.

Introducing comparison graphs that allow users to visualise multiple sensor data streams simultaneously would enhance analytical capabilities, enabling more effective comparison across stations and sensor types.

Finally, given the prevalence of two dominant sensor categories – professional-grade units, which are typically larger and more costly, and compact, cost-effective IoT-oriented sensors – empirical performance comparisons are warranted. Evaluating real-world measurements against manufacturer specifications would provide valuable insights into the accuracy and trade-offs of each sensor type, supporting informed selection for various monitoring applications.

REFERENCES

[1] Granieri, D., Vita, F. & Inguaggiato, S., Volcanogenic SO_2, a natural pollutant: Measurements, modeling and hazard assessment at Vulcano Island (Aeolian Archipelago, Italy). *Environmental Pollution*, **231**(1), pp. 219–228, 2017. https://doi.org/10.1016/j.envpol.2017.07.101.

[2] Völgyesi, P., Nádas, A., Koutsoukos, X. & Lédeczi, Á., Air quality monitoring with SensorMap. *2008 International Conference on Information Processing in Sensor Networks (IPSN 2008)*, St. Louis, Missouri, USA, 22–24 Apr., pp. 529–530, 2008. https://doi.org/10.1109/IPSN.2008.50.

[3] Postolache, O.A., Pereira, J.D. & Girão, P.S., Smart sensors network for air quality monitoring applications. *IEEE Transactions on Instrumentation and Measurement*, **58**(9), pp. 3253–3262, 2009. https://doi.org/10.1109/TIM.2009.2022372.

[4] Devarakonda, S., Sevusu, P., Liu, H., Liu, R., Iftode, L. & Nath, B., Real-time air quality monitoring through mobile sensing in metropolitan areas. *Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing (UrbComp 2013)*, Chicago, Illinois, USA, 11 Aug., pp. 1–8, 2013. https://doi.org/10.1145/2505821.2505834.

[5] Parmar, G., Lakhani, S. & Chattopadhyay, M.K., An IoT based low cost air pollution monitoring system. *2017 International Conference on Recent Innovations in Signal Processing and Embedded Systems (RISE-2017)*, Bhopal, India, 27–29 Oct., pp. 524–528, 2017. https://doi.org/10.1109/RISE.2017.8378212.

[6] Liu, J.H. et al., Developed urban air quality monitoring system based on wireless sensor networks. *2011 Fifth International Conference on Sensing Technology*, Palmerston North, New Zealand, 28 Nov.–1 Dec., pp. 549–554, 2011. https://doi.org/10.1109/ICSensT.2011.6137040.

[7] Kumar, S. & Jasuja, A., Air quality monitoring system based on IoT using Raspberry Pi. *2017 International Conference on Computing, Communication and Automation (ICCCA)*, Greater Noida, India, 5–6 May, pp. 1341–1346, 2017. https://doi.org/10.1109/CCAA.2017.8230005.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)

[8] Nasution, T.H., Muchtar, M.A. & Simon, A., Designing an IoT-based air quality monitoring system. *The 2019 International Conference on Information Technology and Engineering Management*, Belitung, Indonesia, 27–29 Jun., *IOP Conference Series: Materials, Science and Engineering*, **648**(1), 012037, 2019. https://doi.org/10.1088/1757-899X/648/1/012037.

[9] Neethirajan, S., Jayas, D.S. & Sadistap, S., Carbon dioxide (CO_2) sensors for the agri-food industry: A review. *Food and Bioprocess Technology*, **2**, pp. 115–121, 2009. https://doi.org/10.1007/s11947-008-0154-y.

[10] Rivai, M., Rahmannuri, H., Rohfadli, M. & Pirngadi, H., Monitoring of carbon monoxide and sulfur dioxide using electrochemical gas sensors based on IoT. *2020 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Virtual Conference, 22–23 Jul., pp. 61–65, 2020. https://doi.org/10.1109/ISITIA49792.2020.9163731.

[11] Zheng, K., Zhao, S., Yang, Z., Xiong, X. & Xiang, W., Design and implementation of LPWA-based air quality monitoring system. *IEEE Access*, **4**, pp. 3238–3245, 2016. https://doi.org/10.1109/ACCESS.2016.2582153.

[12] Banhazi, T.M., User-friendly air quality monitoring system. *Applied Engineering in Agriculture*, **25**(2), pp. 281–290, 2009. https://doi.org/10.13031/2013.26331.

[13] Balasubramaniyan, C. & Manivannan, D., IoT enabled air quality monitoring system (AQMS) using Raspberry Pi. *Indian Journal of Science and Technology*, **9**(39), pp. 1–6, 2016. https://doi.org/10.17485/ijst/2016/v9i39/90414.

[14] Rana, G., Parmar, A., Parikh, H. & Patel, P., Comparative study of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). *International Journal of Engineering Research and Technology (IJERT)*, **13**(10), 2024.

[15] Khare, S. & Totaro, M., Big data in IoT. *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kanpur, India, 6–8 Jul., pp. 1–7, 2019. https://doi.org/10.1109/ICCCNT45670.2019.8944495.

© 2026 WIT Press
www.witpress.com, ISSN 1743-3541 (on-line)