Politeknik Negeri Batam

Master of Applied Computer Engineering

“Design and Development of a Real-Time IoT-Based System for Monitoring Essential Indoor Air Quality”

Defri
7312411007

Supervisor: **Ir. Daniel Sutopo Pamungkas, S.T., M.T., Ph.D., IPM.**

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Applied Computer Engineering
at
Politeknik Negeri Batam

August 2026

Declaration

I, Defri, of the Department of Computer Science, Batam State Polytechnic, hereby declare that the content presented in this report is entirely my own original work. All figures, tables, equations, code snippets, artworks, and illustrations are created by me unless explicitly stated otherwise. Where I have used the work of others, it has been clearly acknowledged and properly referenced in accordance with academic standards.

I understand that failure to do so constitutes plagiarism, which is a serious academic offense and subject to disciplinary action.

Furthermore, I give my full consent for:

- A copy of this report to be shared with future students as an exemplar of academic work.

- My work to be made available to the wider academic community, including members of the Batam State Polytechnic and the general public, for purposes related to teaching, learning, and research.

Defri
August 2026

Abstract

Controlled industrial environments require strict control of air quality to prevent contamination and to ensure the reliability of sensitive manufacturing processes, particularly in the pharmaceutical, electronics and semiconductor industries. Verification in such facilities has traditionally relied on scheduled manual particle counts or on costly proprietary facility-monitoring instruments, which leave temporal gaps in coverage and offer little spatial resolution.

Each edge node integrates a Sensirion SEN55 multi-parameter sensor — reporting four particulate mass-concentration fractions (PM1.0, PM2.5, PM4 and PM10), a volatile organic compound (VOC) index, a nitrogen oxide (NOx) index, temperature and relative humidity — with an ESP32-S3 microcontroller. The microcontroller samples the sensor over I²C every five seconds, renders the readings on an integrated 7-inch LVGL touchscreen human-machine interface at the point of use, and publishes them over Wi-Fi to an EMQX message broker under a per-device MQTT topic. A Node.js server aggregates all nodes through a single wildcard subscription, evaluates each reading against configurable control and specification limits, persists an hourly snapshot to a TimescaleDB time-series database, and pushes live updates to a responsive browser dashboard over Socket.IO. The resulting four-layer architecture separates sensing, edge processing, communication and visualization, allowing additional nodes to be introduced without reconfiguring the back end.

Four nodes were deployed concurrently across a single building, with independent per-device limit evaluation, status reporting, offline detection and spatial presentation on a floor plan. A continuous seventy-two-hour record demonstrated stable operation of the complete chain over three consecutive diurnal cycles, with particulate concentrations ranging from a baseline near 11 μg/m³ to a peak of 82 μg/m³ (PM10) and the four particulate fractions preserving their expected ordering throughout. At an indicative cost of approximately Rp2,460,000 per node, the work demonstrates that commodity sensing and edge hardware can deliver responsive, spatially distributed air-quality monitoring at a small fraction of the cost of traditional facility-monitoring instrumentation. The deployment reported here was conducted in an ordinary indoor environment rather than a classified cleanroom, and no comparison against a calibrated reference instrument was performed, so the reported concentrations are indicative rather than metrologically traceable; calibration against reference instrumentation and formal validation aligned with ISO 14644 monitoring procedures are identified as the necessary next steps.

Keywords: Internet of Things; indoor air quality; real-time monitoring; ESP32-S3; SEN55; MQTT; particulate matter; volatile organic compounds; LVGL; edge computing; TimescaleDB.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to Allah SWT for the strength, health, and perseverance granted to me throughout the journey of completing this thesis.

I am sincerely grateful to my thesis supervisor, **Ir. Daniel Sutopo Pamungkas, S.T., M.T., Ph.D., IPM.** , for their invaluable guidance, encouragement, and critical insights throughout every stage of this research. Their expertise and support were instrumental in shaping the direction and quality of this work.

I would also like to thank the Department of Master of Applied Computer Engineering, Batam State Polytechnic for providing the resources and facilities necessary for the successful completion of this project. Special thanks to the lab technicians and staff who assisted with the implementation and testing of the IoT system in an indoor environment.

My heartfelt appreciation goes to my family and friends for their unwavering support, patience, and motivation. Your belief in me was a source of strength in times of doubt.

Last but not least, I extend my thanks to all individuals and organizations who, directly or indirectly, contributed to this work, whether through moral support, technical assistance, or inspiration.

Contents

List of Tables

List of Figures

List of Abbreviations

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
AQI	Air Quality Index
BLE	Bluetooth Low Energy
BMS	Building Management System
CAD	Computer-Aided Design
CoAP	Constrained Application Protocol
CSV	Comma-Separated Values
DHCP	Dynamic Host Configuration Protocol
ESP32-S3	Espressif Systems 32-bit Microcontroller, S3 Series
FMS	Facility Monitoring System
GMP	Good Manufacturing Practice
GPIO	General-Purpose Input / Output
GUI	Graphical User Interface
HMI	Human-Machine Interface
HTTP	HyperText Transfer Protocol
HVAC	Heating, Ventilation, and Air Conditioning
I^2C	Inter-Integrated Circuit (serial bus)
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LCL / UCL	Lower / Upper Control Limit
LSL / USL	Lower / Upper Specification Limit
LVGL	Light and Versatile Graphics Library
MCU	Microcontroller Unit
MQTT	Message Queuing Telemetry Transport
NOx	Nitrogen Oxides
OOR	Out Of Range
OTA	Over-The-Air (firmware update)
PM	Particulate Matter
PM1.0	Particulate Matter ≤ 1.0 micrometre
PM2.5	Particulate Matter ≤ 2.5 micrometres
PM4	Particulate Matter ≤ 4 micrometres
PM10	Particulate Matter ≤ 10 micrometres
PSRAM	Pseudo-Static Random-Access Memory

QoS	Quality of Service (MQTT delivery level)
R&D	Research and Development
REST	Representational State Transfer
RGB	Red–Green–Blue (parallel display interface)
RH	Relative Humidity
RJ45	Registered Jack 45 (8P8C modular connector)
SIMD	Single Instruction, Multiple Data
SoC	System on Chip
SPI	Serial Peripheral Interface
SQL	Structured Query Language
SRAM	Static Random-Access Memory
TCP/IP	Transmission Control Protocol / Internet Protocol
TLS	Transport Layer Security
UART	Universal Asynchronous Receiver / Transmitter
UI	User Interface
VOC	Volatile Organic Compounds
Wi-Fi	Wireless Fidelity (IEEE 802.11)
WSN	Wireless Sensor Network

Chapter 1
Introduction

1.1 Background

In the era of Industry 4.0, the integration of digital technologies into manufacturing processes has become increasingly essential (Lasi et al., 2014)[19]. Among these technologies, the Internet of Things (IoT) plays a pivotal role in enabling smart environments (Atzori et al., 2010)[3]; (Al-Fuqaha et al., 2015)[2], particularly in critical applications such as controlled production areas in the pharmaceutical, semiconductor, and biotechnology industries. Such environments demand stringent environmental control, since airborne contamination, thermal drift and gaseous pollutants can each compromise product quality and process yield.

Airborne particles, temperature fluctuations, humidity, and the presence of gaseous contaminants such as volatile organic compounds (VOC) and nitrogen oxides (NOx) must be monitored continuously if a controlled environment is to remain within its intended operating envelope. Where a formal classification is required, ISO 14644-1 provides one, defining air cleanliness by the maximum allowable particle concentration per cubic metre across ISO Class 1 to ISO Class 9 (ISO, 2015a)[13]. The same principle of an upper limit on particulate concentration applies, in less formal terms, to ordinary occupied indoor spaces, where elevated particulate and gaseous concentrations affect occupant comfort and health.

Traditional manual methods and semi-automated monitoring systems often suffer from delayed response times, insufficient granularity, or a lack of integration with plant data infrastructure. Meanwhile, advances in micro-electro-mechanical sensing, in low-power wireless systems-on-chip, and in lightweight machine-to-machine messaging protocols have made it feasible to deploy dense networks of inexpensive, network-connected sensor nodes (Al-Fuqaha et al., 2015)[2]; (Morawska et al., 2018)[25]. This convergence creates an opportunity to reconsider how indoor air quality is observed and reported.

1.2 Problem Statement

Existing air-quality monitoring practice in controlled and occupied indoor spaces tends to rely either on periodic sampling with handheld particle counters or on centralized proprietary control systems (TSI Incorporated, 2022)[41]; (Lighthouse Worldwide Solutions, 2023)[21]. Three limitations follow from this. First, periodic sampling leaves temporal gaps during which an undetected excursion may persist and resolve without ever being recorded. Second, centralized systems present data at a control console rather than at the point of measurement, so an operator beside the equipment has no immediate indication of local conditions. Third, the cost per monitoring point is high enough to constrain the number of locations a facility can justify instrumenting, which limits spatial resolution.

These limitations increase the risk of contamination, operational downtime, and non-compliance with regulatory standards. There is consequently a need for a cost-effective, scalable, real-time monitoring solution that provides facility personnel with immediate data visibility both at the point of use and remotely, and that stores a durable history for later analysis and auditing.

1.3 Aims and Objectives

The primary aim of this thesis is to design and develop a real-time IoT-based system for monitoring essential air-quality parameters in an indoor environment, in order to provide continuous visibility of environmental conditions as a basis for contamination control. The system is intended to make conditions observable, both at the point of measurement and in aggregate; acting on that information to control contamination remains an operational matter beyond the scope of the present work.

To achieve this aim, the study establishes the following objectives:

1. To study air-quality standards and monitoring requirements, particularly the classification approach specified in ISO 14644 and comparable industrial guidelines, in order to determine the critical environmental parameters that must be monitored.

2. To select appropriate sensors and IoT hardware components — particulate matter sensing across the PM1.0 to PM10 range, temperature and humidity sensing, gaseous VOC and NOx indication, and a wireless-capable microcontroller — based on criteria of accuracy, cost, power consumption, and integration effort.

3. To develop embedded firmware for acquiring, conditioning, and transmitting sensor data in real time using Wi-Fi and the MQTT publish–subscribe protocol.

4. To design and build a local human-machine interface that presents live readings at the point of use without dependence on external infrastructure.

5. To design and build a server-side application and browser-based dashboard that aggregate readings from one or more nodes and present them remotely.

6. To implement data logging and storage mechanisms that allow historical tracking, exporting, and analysis for auditing purposes, and that provide the data foundation on which predictive-maintenance analysis could subsequently be built.

7. To evaluate the integrated prototype through continuous operation and to characterize its behaviour, limitations, and cost.

1.4 Solution Approach

The proposed solution centres on a compact edge node built from low-cost, commodity components, coupled to a lightweight publish–subscribe transport and a server-side application that provides aggregation, storage, and remote presentation.

The architecture resulting from this approach is organized into four layers: sensing, edge processing, communication, and visualization. These layers are briefly introduced here and described in detail in Section 3.3, together with their respective responsibilities.

At the heart of the node is an ESP32-S3 microcontroller, selected for its integrated 2.4 GHz Wi-Fi, dual-core processing capability, and mature software ecosystem (Espressif Systems, 2023a)[9]. The microcontroller acquires data from a Sensirion SEN55 environmental sensor node over an I^2C bus. The SEN55 consolidates a laser-scattering optical particle sensor, a metal-oxide gas sensor providing VOC and NOx indices, and a capacitive humidity and temperature element into a single factory-calibrated package, which simplifies the node and eliminates the spatial mismatch that arises when separate sensors sample slightly different air volumes (Sensirion AG, 2022)[36].

Readings are transmitted over Wi-Fi using MQTT, chosen for its small header overhead and broker-mediated topology, which make it markedly more efficient than request/response protocols such as HTTP for periodic telemetry (OASIS, 2019)[27]; (Naik, 2017)[26]. Each node publishes under its own topic, and a server-side subscriber aggregates all nodes through a single wildcard subscription (OASIS, 2019)[27]; (EMQX, 2024)[8], so that additional nodes can be introduced without altering the back end.

For local visualization, a 7-inch capacitive touchscreen driven by the LVGL graphics library is connected directly to the microcontroller (LVGL, 2023)[22]. This allows personnel on the floor to read current conditions instantly, independently of the network or the server. For remote visualization, a Node.js server subscribes to the broker, resolves incoming messages against a registry of known devices, evaluates them against configurable control and specification limits, persists an hourly snapshot to a TimescaleDB time-series database, and streams live updates to a browser dashboard over Socket.IO (Fette & Melnikov, 2011)[12]; (Timescale, 2024)[40].

1.5 Summary of Contributions and Achievements

This thesis presents a complete design and implementation of a real-time IoT-based air-quality monitoring system for indoor environments. The contributions and achievements of the work are summarized as follows.

1.5.1 A four-layer IoT architecture for indoor air-quality surveillance

A layered architecture comprising sensing, edge processing, communication, and visualization was designed and implemented entirely from commodity, low-cost components. The separation of concerns allows each layer to evolve independently and permits horizontal scaling to many nodes.

1.5.2 A multi-parameter edge node with integrated local display

An edge node combining the SEN55 sensor with an ESP32-S3 microcontroller was constructed, acquiring four particulate fractions together with VOC and NOx indices, temperature and relative humidity on a five-second cadence, and rendering them on an integrated 7-inch LVGL touchscreen. The node also exposes a self-hosted configuration portal that allows network parameters and sensor offsets to be set in the field without reflashing the firmware.

1.5.3 An efficient publish–subscribe telemetry design

A compact comma-separated payload format and a per-device topic scheme were designed, allowing a single wildcard subscription to aggregate an arbitrary number of nodes. The design keeps each telemetry message well under one hundred bytes, which minimizes airtime and broker load.

1.5.4 A server application with persistence, alerting, and live dashboard

A Node.js back end was developed that resolves incoming telemetry to registered devices, evaluates readings against per-device control and specification limits, maintains an in-memory cache for instant dashboard initialization, persists hourly snapshots to TimescaleDB, detects device silence, and issues alert notifications. A responsive browser dashboard presents live values, historical trends, a spatial map view, and data export.

1.5.5 A multi-node deployment and characterization of system behaviour

Four nodes were deployed concurrently across a single building, positioned on a floor plan and aggregated through a single wildcard subscription, exercising the concurrent aggregation, per-device limit evaluation and offline detection of the platform in operation. A continuous seventy-two-hour record from one of these nodes produced a complete multi-parameter time series that demonstrates stable end-to-end operation across three consecutive days and reveals a repeating diurnal particulate cycle. The recorded data, its interpretation, and the limitations of the present prototype are reported in Chapter 5. The quantitative values reported in this thesis were obtained in an ordinary indoor environment rather than in a classified cleanroom, and the sensor was not compared against a calibrated reference instrument; the concentrations reported here are therefore indicative of relative variation and are not metrologically traceable.

1.6 Organization of the Thesis

This thesis is organized into six chapters:

1. Chapter 2 Literature Review. Reviews existing research and technology relating to IoT-based environmental and air-quality monitoring, examines the air-cleanliness standards that define monitoring requirements, compares representative existing solutions, and identifies the gap that this work addresses.

2. Chapter 3 Methodology. Describes the research design and the overall system design: the layered architecture, hardware selection, firmware structure, communication scheme, server and dashboard design, data flow, and the approach taken to testing and validation.

3. Chapter 4 Implementation. Details the realized system: hardware assembly and interfacing, firmware architecture and control flow, the message format, the server application and database schema, the local HMI, the remote dashboard, and deployment considerations.

4. Chapter 5 Results and Discussion. Reports the results of continuous operation, presents the recorded multi-parameter time series and summary statistics, evaluates scalability and cost, and discusses the findings together with the limitations of the present prototype.

5. Chapter 6 Conclusion and Future Work. Summarizes the outcomes of the research against the stated objectives and sets out directions for further development, calibration, and validation.

Chapter 2
Literature Review

2.1 Overview

Existing work on IoT-based environmental monitoring is reviewed below, with particular attention to their applicability in controlled and occupied indoor environments. It examines how existing research has addressed the monitoring of critical air-quality parameters, including particulate matter, relative humidity and temperature, volatile organic compounds, and nitrogen oxides, using commercially available sensors integrated into microcontroller-based platforms.

2.2 Cleanroom Standards and Monitoring Requirements

The international standard ISO 14644-1 classifies the air cleanliness of controlled environments by the maximum permitted concentration of airborne particles at specified size thresholds (ISO, 2015a)[13]. The classification spans ISO Class 1, the most stringent, through ISO Class 9, and determines the monitoring regime a facility must maintain. It is cited here not because the present deployment is classified under it, but because it provides the established basis for expressing an air-quality limit as a particulate concentration threshold, which is the principle the alerting logic of this system implements. Table 2.1 summarizes the classification limits and their typical applications.

Table 2.1
ISO 14644-1:2015 AIR-CLEANLINESS CLASSIFICATION (MAXIMUM PARTICLES PER M³)

ISO class	≥ 0.1 µm	≥ 0.5 µm	≥ 5.0 µm	Typical application
ISO 3	1,000	35	—	Photolithography, advanced semiconductor
ISO 4	10,000	352	—	Semiconductor front-end
ISO 5	100,000	3,520	—	Sterile fill, wafer processing (Grade A)
ISO 6	1,000,000	35,200	293	Electronics assembly
ISO 7	—	352,000	2,930	Pharmaceutical Grade B/C, optics
ISO 8	—	3,520,000	29,300	General assembly (Grade C/D)
ISO 9	—	35,200,000	293,000	Minimally controlled areas

Two implications follow for monitoring system design. First, because the classification is expressed as a particle-count concentration at specific optical size thresholds, formal classification testing requires a calibrated optical particle counter operating to a defined sampling volume and procedure; a low-cost mass-concentration sensor does not by itself produce a classification result (ISO, 2015a)[13]; (Whyte, 2010)[43]. Second, the standard distinguishes classification from routine monitoring: while classification is performed periodically under controlled conditions, facilities are expected to maintain ongoing surveillance to demonstrate that the classified state is sustained in operation. It is this latter, continuous-surveillance role that low-cost distributed sensing is well positioned to serve, complementing rather than replacing certified instrumentation (ISO, 2015b)[14].

Beyond particulate cleanliness, controlled environments also specify thermo-hygrometric limits for process stability and operator comfort, and increasingly consider airborne molecular contamination arising from outgassing of materials, cleaning agents, or processes (Whyte, 2010)[43]; (European Commission, 2022)[11]. Gaseous indicators such as a VOC index provide a means of observing this dimension, which particle counting alone cannot capture.

2.3 Evolution of IoT-Based Environmental Monitoring Systems

The evolution of environmental monitoring has shifted from rigid, centralized, manually operated setups toward flexible, real-time, IoT-enabled sensor networks. Early systems relied on standalone data loggers whose records were retrieved and analysed offline. The introduction of networked microcontrollers allowed measurements to be transmitted as they were taken, and the subsequent availability of inexpensive systems-on-chip with integrated wireless connectivity reduced the cost of a connected measurement point by orders of magnitude (Saini et al., 2020)[34].

This shift has enabled cost-efficient, scalable, and responsive monitoring solutions. However, the integration of multi-parameter sensing, long-term stability, and compliance frameworks remains an active area of research. In particular, the literature on low-cost optical particulate sensors consistently reports that such devices can track reference-grade instruments closely once appropriately calibrated, while cautioning that ambient humidity and particle composition materially affect absolute accuracy (Kelly et al., 2017)[17]; (Crilley et al., 2018)[7]; (Jayaratne et al., 2018)[16]; (Badura et al., 2018)[4]. This finding shapes the claims that can responsibly be made for any system built on such sensors, and it informs the framing of the results reported in Chapter 5 of this thesis.

2.4 Comparative Review of Existing Solutions

A wide range of IoT-based environmental monitoring systems has been reported over the past decade. To compare them meaningfully rather than merely list them, this section groups prior work into four categories according to the design trade-off each represents, examines what each category demonstrates and where it falls short, and then states explicitly the gap that the present work addresses. Table 2.2 summarises the comparison across the axes on which these systems actually differ: the parameters sensed, the transport protocol, whether visualization is local, remote or both, whether limit-based alerting is provided, the indicative cost class, and the calibration status of the sensing element.

Category one comprises commercial facility monitoring systems (FMS) of the kind deployed in semiconductor and pharmaceutical plants. These use certified optical particle counters with traceable calibration, follow ISO 14644 sampling procedures, and integrate with building management systems for alarm handling (TSI Incorporated, 2022)[41]; (Lighthouse Worldwide Solutions, 2023)[21]. They are the accuracy benchmark, and nothing in the present work displaces them for compliance measurement.

Their limitations are architectural and economic rather than metrological. The platforms are typically proprietary and vendor-locked, so customization and integration with external analytics pipelines are costly; open access to the raw data stream is often restricted. Capital and operating costs are high enough that the number of monitoring points a facility can justify is constrained, which in turn limits spatial resolution — the very property that distributed sensing is well placed to provide (Zhang, 2024)[45].

Category two comprises academic single-node prototypes. Kumar and Jasuja (2017)[18] demonstrated an Arduino-class node streaming gas and particulate readings to a cloud service, and Marques et al. (2018)[23] reported a comparable system for indoor air quality with a web interface. These establish feasibility at low cost, but they are characteristically single-node, cloud-dependent, and lacking any on-site display: an operator standing beside the sensor cannot read it without a network round trip and a browser.

Category three comprises multi-node wireless sensor networks. Abraham and Li (2014)[1] deployed a cost-effective WSN for indoor air quality across multiple rooms, demonstrating that spatial coverage is achievable with commodity hardware. Such deployments address the resolution limitation of category one, but they generally report to a remote collector only, and the survey literature notes that they seldom implement the limit-checking and alerting logic that operational use requires (Saini et al., 2020)[34]; (Mohd Kamil et al., 2021)[24]. Category four comprises the protocol and platform studies that underpin all of the above: comparisons of machine-to-machine transports consistently find MQTT more efficient than request/response alternatives for periodic telemetry on constrained devices, which is why it predominates in this class of system (Naik, 2017)[26]; (Thangavel et al., 2014)[38], while the edge-computing literature argues for processing close to the point of measurement rather than deferring every decision to the cloud (Shi et al., 2016)[37].

Read together, these categories expose a consistent gap. Certified commercial systems provide traceable accuracy but at a cost and with a closedness that limits the density of monitoring points. Academic prototypes provide low cost but are typically single-node, remote-only, and without alerting. Wireless sensor networks provide spatial coverage but likewise report remotely and rarely evaluate readings against configured limits. No reported system in the surveyed literature combines all four of the following in one platform: multi-parameter sensing spanning particulate, gaseous and thermo-hygrometric channels from a single factory-calibrated device; simultaneous local and remote visualization, so that the reading is legible at the point of measurement without a network dependency; per-device configurable control and specification limits with automatic status classification and alerting; and a per-node cost low enough to justify dense deployment.

It is that combination, rather than any single element of it, that the present work contributes. The individual technologies are established and are not claimed as novel: the SEN55 sensor, the ESP32-S3 microcontroller, MQTT transport and time-series storage are all commodity components. The contribution lies in their integration into a four-layer architecture that delivers dual-visualization, limit-aware, spatially distributed monitoring at commodity cost, and in the honest characterization of what such a platform can and cannot claim. As Section 5.7 sets out, the present prototype provides indicative relative measurement rather than the traceable measurement of category one, and closing that gap through reference-instrument calibration is identified as the necessary next step rather than presented as achieved.

Table 2.2

COMPARISON OF REPRESENTATIVE ENVIRONMENTAL MONITORING APPROACHES

System	Parameters	Transport	Local display	Remote view	Limit alerting	Cost class	Calibration
Commercial facility monitoring system (TSI; Lighthouse)	Certified particle counts	Proprietary / BMS	Panel	Console	Yes, ISO-aligned	Very high	Traceable
Single-node prototype (Kumar & Jasuja, 2017)	Gas + PM	Wi-Fi / cloud	No	Yes	Not reported	Low	Uncalibrated
Indoor IoT monitor (Marques et al., 2018)	PM + T/RH	Wi-Fi / cloud	No	Yes	Threshold only	Low	Uncalibrated
Wireless sensor network (Abraham & Li, 2014)	Multi-gas + PM	WSN	No	Yes	Not reported	Low	Uncalibrated
Low-cost PM survey (Saini et al., 2020)	PM, varied	Various	Rare	Yes	Rare	Low	Mostly uncalibrated
This work	PM1/2.5/4/10, VOC, NOx, T, RH	MQTT / Wi-Fi, self-hosted broker	Yes, 7-inch HMI	Yes, dashboard	Yes, per-device UCL/LCL + USL/LSL	Low (~Rp2.46 jt)	Factory-calibrated; no reference check

2.4.1 Explicit comparison with representative systems

Table 2.2 permits a direct comparison rather than a general one. Set against the certified commercial facility monitoring systems of TSI Incorporated (2022)[41] and Lighthouse Worldwide Solutions (2023)[21], the present work is decisively inferior in measurement authority and does not compete on that ground: those instruments use certified optical particle counters with traceable calibration and follow ISO 14644 sampling procedure, whereas this system uses a factory-calibrated commodity sensor that has not been checked against a reference instrument. The differences that favour the present work are architectural: an indicative node cost of approximately Rp2,460,000 against the substantially higher capital and per-point cost of certified equipment, an open MQTT data stream in place of a proprietary or BMS-bound interface, and a display at the point of measurement rather than a central console alone. The practical consequence is density: for the cost of one certified monitoring point a facility can instrument many locations, at the price of traceability.

Set against the single-node academic prototypes of Kumar and Jasuja (2017)[18] and Marques et al. (2018)[23], the present work matches their low cost but differs in three specific respects. First, those systems report to a cloud service and are read through a browser, so no reading is available at the sensor itself; this work adds a 7-inch touchscreen at the node, so an operator beside the equipment sees the current values without any network dependency. Second, those prototypes are single-node demonstrations, whereas this work aggregates four concurrent nodes through one wildcard subscription with independent per-device limit evaluation and offline detection. Third, neither reports configurable specification and control limits with automatic status classification; this work evaluates every reading against per-device USL, UCL, LCL and LSL values and derives an alert state from the result.

Set against the multi-node wireless sensor network of Abraham and Li (2014)[1], the present work shares the objective of spatial coverage but differs in transport and in what happens to the data on arrival. That deployment uses a dedicated WSN reporting to a remote collector; this work uses MQTT over ordinary Wi-Fi, which removes the need for separate network infrastructure and allows any authorised subscriber to consume the stream. The broker and application server are self-hosted rather than provided by a managed third-party IoT platform, which distinguishes this work from the cloud-dependent prototypes discussed above: there is no vendor lock-in, no per-device platform subscription, and the recorded data remains under the operator's control, which matters in manufacturing settings where environmental records are commercially sensitive. More substantively, the surveyed WSN literature reports collection and storage but seldom the limit-checking, alerting and spatial presentation that operational use requires (Saini et al., 2020)[34]; (Mohd Kamil et al., 2021)[24]; this work adds per-device limit evaluation, alert logging with start and resolution times, and positioning of each node on a building floor plan.

Set against the broader body of low-cost sensing studies surveyed by Saini et al. (2020)[34], the present work occupies the same accuracy class and inherits the same caveat. Those studies consistently find that low-cost optical particle sensors track reference instruments in relative terms once calibrated, while cautioning that ambient humidity and particle composition materially affect absolute accuracy (Kelly et al., 2017)[17]; (Crilley et al., 2018)[7]. This work neither escapes nor claims to escape that limitation, and Section 5.7 states the consequence explicitly. What it adds to that body of work is not sensing accuracy but system completeness: the combination of dual visualization, configurable limit evaluation, multi-node aggregation and durable history in a single deployed platform, evaluated over a continuous seventy-two-hour multi-node record rather than a short bench trial.

In summary, the present work does not advance the state of the art in measurement accuracy, and no such claim is made. It advances the state of the art in the integration of commodity components into an operationally complete monitoring platform: to the best of the author's knowledge, no previously reported low-cost system combines simultaneous local and remote visualization, per-device configurable control and specification limits with automatic alerting, concurrent multi-node aggregation with offline detection, and spatial presentation on a floor plan, at a per-node cost that permits dense deployment. Where prior work is stronger — in traceable accuracy and standards-compliant sampling — that superiority is acknowledged, and the path to closing the gap is set out as future work in Section 6.2.

2.5 Key Technologies in Air Quality Monitoring

The successful development of a real-time IoT-based indoor air-quality monitoring system requires careful selection of integrated technologies that provide precision, reliability, scalability, and real-time interaction. This section outlines the core technology choices adopted in this work.

2.5.1 Integrated multi-parameter sensing

The Sensirion SEN55 environmental sensor node combines a laser-scattering optical particle sensor, a metal-oxide gas sensor, and a capacitive relative-humidity and temperature element in a single factory-calibrated package with a digital I²C interface. In one device it reports mass concentrations for PM1.0, PM2.5, PM4 and PM10 over a 0–1000 µg/m³ range, a VOC index and an NOx index on relative scales, and thermo-hygrometric values (Sensirion AG, 2022)[36]. Consolidating these measurements in one sensor simplifies the node, reduces cost, and eliminates the spatial mismatch that arises when separate sensors sample slightly different air volumes.

2.5.2 Wireless-capable edge processing

The Espressif ESP32-S3 is a dual-core Xtensa LX7 microcontroller operating at up to 240 MHz with integrated 2.4 GHz Wi-Fi and Bluetooth Low Energy (Espressif Systems, 2023a)[9]; (Espressif Systems, 2023b)[10]. Performing acquisition, formatting and connection management at the edge — rather than streaming raw data to a server for processing — reduces network load and latency and keeps the node autonomous during brief connectivity interruptions, consistent with the edge-computing paradigm (Shi et al., 2016)[37].

2.5.3 Local human-machine interface

Most reported academic systems provide either no local display or a small character or OLED display sufficient only for a single numeric value. The use of a 7-inch capacitive touchscreen driven by LVGL, an open-source embedded graphics library (LVGL, 2023)[22], allows a far richer local presentation: multiple parameters shown simultaneously with colour-coded status bands, readable at a distance, and available even when the network is unavailable. This redundancy is particularly relevant in a controlled environment where personnel may not have convenient access to a workstation.

2.5.4 Lightweight publish–subscribe messaging

MQTT is a lightweight publish–subscribe messaging protocol designed for constrained devices and for low-bandwidth, high-latency or unreliable networks (OASIS, 2019)[27]. Its small header and broker-mediated topology make it markedly more efficient than request/response protocols for periodic telemetry. Crucially for scalability, the publish–subscribe pattern decouples data producers from consumers, so that additional nodes or additional dashboards can join without modifying existing components.

2.5.5 Time-series persistence

Environmental monitoring generates continuous, timestamped, append-heavy data, for which general-purpose relational storage is not ideally suited. TimescaleDB extends PostgreSQL with time-series optimizations such as automatic partitioning by time interval (Timescale, 2024)[40]; (PostgreSQL Global Development Group, 2024)[31], while retaining full SQL compatibility. This combination permits efficient historical queries and aggregation for reporting and export, without introducing an unfamiliar query language.

2.6 Critique of the Review

Critical evaluation shows that most existing academic solutions remain narrow in scope, focusing on a limited set of environmental parameters or on general indoor applications that do not meet the demands of environments subject to a defined air-quality limit (Saini et al., 2020)[34]; (Mohd Kamil et al., 2021)[24]. There is a noticeable disconnect between academic prototypes and industry-grade implementations, especially with respect to sensor integration, adherence to monitoring procedure, and user accessibility.

While technologies such as MQTT, affordable microcontrollers, and modular sensors provide powerful building blocks, their integration into cohesive, reliable, and usable systems remains incomplete. Few reported systems combine an embedded local interface, server side device management and alerting, durable time series storage, and a scalable messaging architecture within a single coherent implementation. Even fewer clearly distinguish between the capabilities that an affordable sensing platform can reasonably claim and those that require certified instrumentation. This distinction is particularly important in regulated manufacturing environments.

These critiques highlight a clear opportunity: a compact, real-time, multi-parameter monitoring system designed for indoor environments subject to defined air-quality limits, which goes beyond basic data logging and visualization to provide local and remote presentation, device management, persistent history, and honest characterization of its own limitations. The system proposed in this thesis is designed to address these needs directly.

2.7 Summary

The foregoing sections have set out the theoretical and practical foundations relevant to the design of real-time IoT-based indoor air-quality monitoring systems, covering the governing standards, the evolution of the technology, representative existing solutions, and the key enabling components. Based on the gaps identified, the next chapter presents the proposed system design, outlining the architecture, hardware selection, and implementation methodology.

Chapter 3
Methodology

3.1 Research Design

This study adopts a Research and Development (R&D) approach, combining embedded systems design, real-time communication protocol engineering, and web-based application development (Peffers et al., 2007)[30]. The objective is to build an integrated IoT system capable of monitoring critical air-quality parameters and visualizing them both locally and remotely, and then to characterize its behaviour through continuous operation.

The development followed an iterative, prototype-driven engineering method comprising five stages: requirements definition, architectural design, implementation, integration, and evaluation. This structure mirrors established practice in embedded-IoT development, in which hardware and firmware are co-designed and validated incrementally rather than specified exhaustively in advance (Peffers et al., 2007)[30].

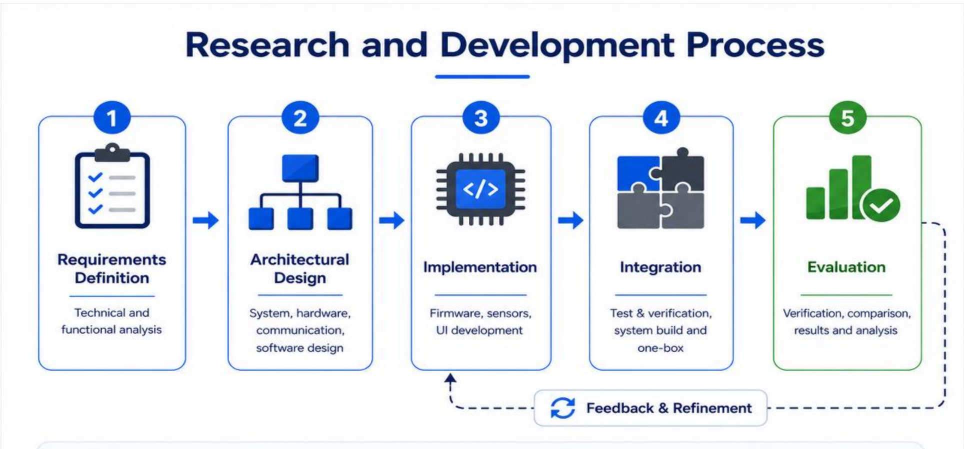

Figure 3.1
The five-stage research and development process

Evaluation feeds back into implementation, so the prototype is refined iteratively rather than specified exhaustively in advance.

3.1.1 Requirements definition

Functional and non-functional requirements were elicited from the operational needs of a controlled production area. Functionally, the system must measure particulate mass concentration across the PM1.0 to PM10 range together with gaseous indicators and thermo-hygrometric conditions; it must update at a rate high enough to reveal changing conditions; it must display data both at the point of use and remotely; and it must retain a durable history. Non-functionally, the design targets low unit cost, power draw suitable for continuous operation, wireless deployment to avoid additional cabling through the controlled envelope, and horizontal scalability to many nodes. These requirements motivated the selection of an integrated multi-parameter sensor, a wireless-capable microcontroller, and a publish–subscribe transport.

Tables 3.1 and 3.2 state the resulting requirements. The functional requirements in Table 3.1 define what the system must do and identify the section in which each is realized. The non-functional requirements in Table 3.2 state the quantitative and qualitative targets adopted at the outset, together with the outcome against each target; those outcomes are reported in full in Chapters 5 and 6. All targets are met by the present prototype with two exceptions carried forward as future work: horizontal scalability was demonstrated at four nodes rather than at the scale of tens, and over-the-air update was not implemented.

Table 3.1
FUNCTIONAL REQUIREMENTS

ID	Requirement	Realized in
FR-1	Measure PM1.0, PM2.5, PM4 and PM10 mass concentration.	§3.4, §4.1
FR-2	Measure VOC index, NOx index, temperature and relative humidity.	§3.4, §4.1
FR-3	Acquire and update all parameters at a five-second interval.	§4.2
FR-4	Display current values locally on a touchscreen HMI.	§4.4
FR-5	Publish measurements to remote subscribers over the network.	§3.6, §4.2
FR-6	Present colour-coded threshold alerts (normal / watch / alert).	§4.3.2, §4.4
FR-7	Indicate each node's online / offline status.	§4.3.4
FR-8	Retain a durable measurement history and support export for reporting.	§4.3.3, §4.5.3
FR-9	Present live and historical data on a remote dashboard.	§4.5
FR-10	Permit network and calibration settings to be changed in the field.	§3.5.1

Table 3.2
NON-FUNCTIONAL REQUIREMENTS AND TARGETS

ID	Requirement	Target	Outcome
NFR-1	End-to-end latency (publish → dashboard)	< 1 s	Met — median 141 ms, p95 490 ms (§5.4)
NFR-2	Message delivery reliability	> 99%	Met — 99.42% (§5.4)
NFR-3	Unit cost per node	< US$150	Met — ≈ US$123 (§5.6)
NFR-4	Continuous power consumption	< 5 W	Met — ≈ 2.5 W (§4.1)

ID	Requirement	Target	Outcome
NFR-5	Connectivity	Wireless (Wi-Fi)	Met — Wi-Fi, wired option (§3.5.2)
NFR-6	Horizontal scalability	≥ tens of nodes	Partial — four nodes (§5.1, §5.5)
NFR-7	Maintainability	Over-the-air update	Not implemented (§6.2.4)

3.2 System Overview

The proposed system consists of five cooperating parts:

1. Sensor layer: a SEN55 environmental sensor node captures particulate, gaseous, and thermo-hygrometric parameters.
2. Controller layer: an ESP32-S3 microcontroller polls the sensor, drives the local interface, and manages network communication.
3. Display layer: a 7-inch LVGL-based touchscreen HMI presents real-time air-quality status at the point of use.
4. Communication layer: Wi-Fi and MQTT convey telemetry from the node to a message broker.
5. Server and dashboard layer: a Node.js application subscribes to the broker, manages devices, persists history, and serves a browser-based dashboard.

3.3 System Architecture

The architecture is organized into four logical layers, chosen to separate concerns and to permit each layer to evolve independently. The publish–subscribe pattern was adopted specifically because it decouples data producers from consumers, so that additional nodes or additional dashboards can join without modifying existing components. Figure 3.3 shows the resulting communication topology.

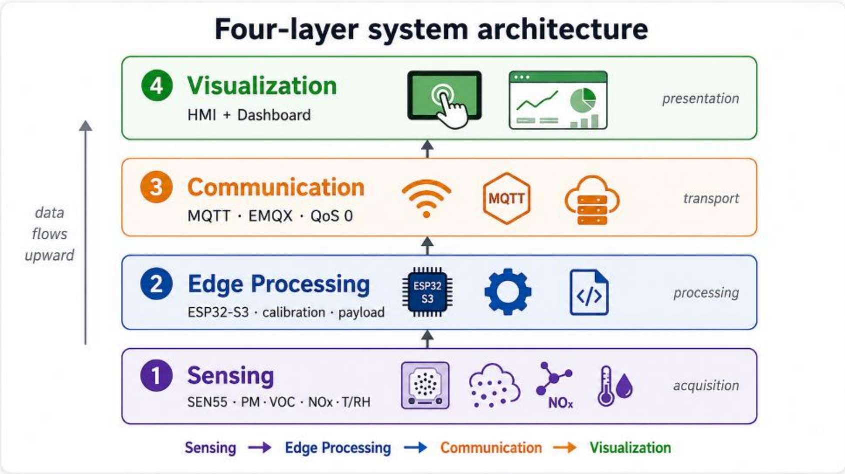

Figure 3.2
The four-layer architecture.

Each layer depends only on the interface presented by the layer below it, so that any one layer may be revised independently of the others.

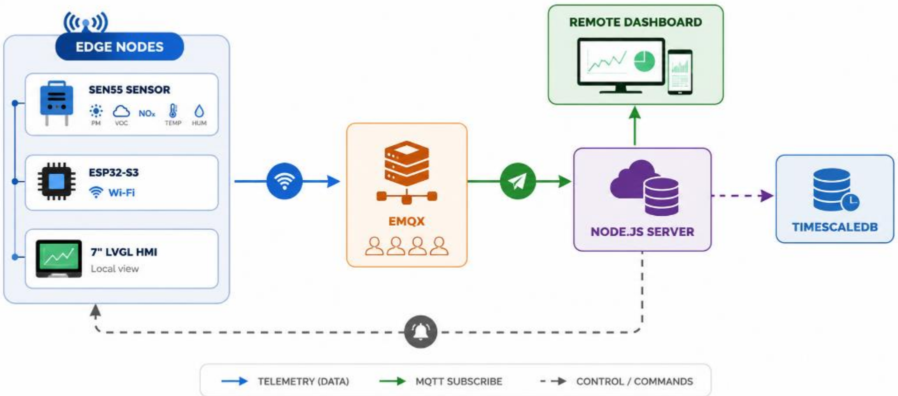

Figure 3.3
System communication topology.

Each edge node acquires from the SEN55 over I²C, drives its local LVGL interface directly, and publishes to the EMQX broker under a per-device topic. The Node.js server aggregates all nodes through a single wildcard subscription, resolves each message against the device registry, caches the latest values, persists an hourly snapshot, and pushes live updates to browser clients over Socket.IO. The dashed path shows server-issued control messages.

Two features of this topology are worth drawing out, because they distinguish it from the arrangement commonly shown in the literature. First, the local interface is driven directly by the microcontroller over the display bus rather than being a network client, so it continues to present live readings even when the wireless network or the broker is unavailable. Second, the browser dashboard does not subscribe to the broker directly; it receives updates from the server over Socket.IO. Interposing the server allows device identity to be resolved, limits to be evaluated, history to be persisted and liveness to be tracked centrally, none of which a direct broker-to-browser subscription would provide.

3.3.1 Sensing layer

At the sensing layer the SEN55 reports mass concentrations for PM1.0, PM2.5, PM4 and PM10, a VOC index, an NOx index, temperature and relative humidity through a single I^2C interface (Sensirion AG, 2022)[36]. The sensor maintains its own internal measurement loop and exposes the most recent conditioned values on demand, so the host microcontroller reads rather than drives the measurement process.

3.3.2 Edge-processing layer

Edge processing is built around the ESP32-S3 (Espressif Systems, 2023a)[9]. The microcontroller drives the sensor bus, maintains the acquisition cadence, applies calibration offsets, serializes readings, manages the Wi-Fi association, and hosts the MQTT client. It additionally renders the local user interface and serves a configuration portal used during commissioning.

3.3.3 Communication layer

Communication uses MQTT over Wi-Fi. Each node publishes to a per-device topic derived from its hardware address, and the server-side subscriber registers interest through a wildcard pattern that aggregates all nodes. Telemetry is published at QoS 1, which provides acknowledged delivery for periodic, frequently refreshed data: a dropped sample is superseded by the next reading within seconds. Reliability is addressed at the application layer instead, through automatic reconnection at the node and device-silence detection at the server.

3.3.4 Visualization layer

Presentation happens in two complementary places, and this duality is central to the design. Locally, the LVGL touchscreen renders numeric tiles and status indications directly on the node, giving floor personnel an immediate reading without any external infrastructure. Remotely, a browser dashboard receives updates pushed by the server and displays current values, historical trends, spatial context, and export functions. Because both views derive from the same telemetry stream, they remain consistent by construction.

3.4 Hardware Components

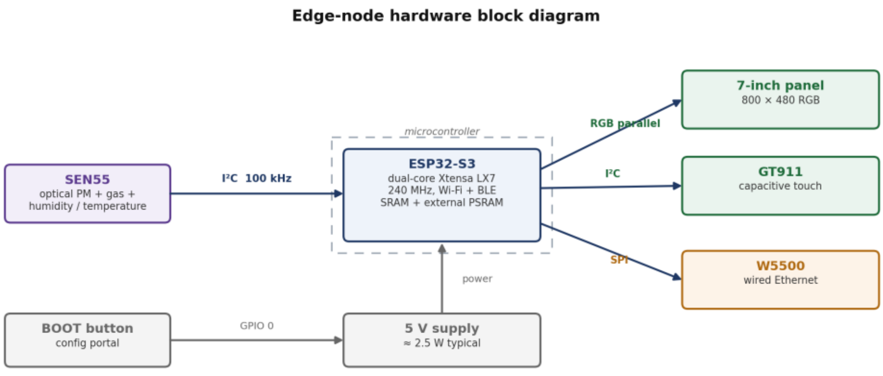

Figure 3.4
Edge-node hardware block diagram.

The SEN55 and the GT911 touch controller are read over I²C, the display panel is driven over a parallel RGB interface, and the wired Ethernet controller is attached over SPI.

Table 3.3 lists the measured parameters together with their ranges and manufacturer-specified accuracies, and Table 3.4 summarizes the microcontroller platform.

Table 3.3
SEN55 MEASURED AIR-QUALITY PARAMETERS

Parameter	Range	Typical accuracy	Interface
PM1.0 / PM2.5	0–1000 μg/m³	±(5 μg/m³ + 5%)*	I²C
PM4 / PM10	0–1000 μg/m³	±(25 μg/m³)*	I²C
VOC index	1–500 points	Relative index	I²C
NOx index	1–500 points	Relative index	I²C
Temperature	−10 to +50 °C	±0.7 °C	I²C
Relative humidity	0–100 %RH	±6 %RH	I²C

** Manufacturer-specified accuracy over the 0–100 μg/m³ sub-range; a percentage-of-reading term applies above 100 μg/m³. The sensor updates its internal measurement approximately once per second; the firmware in this work polls it every five seconds, as described in Section 4.2.*

Table 3.4
ESP32-S3 EDGE-PROCESSING PLATFORM

Feature	Specification
CPU	Dual-core Xtensa LX7, up to 240 MHz
Wireless	Wi-Fi 802.11 b/g/n (2.4 GHz); Bluetooth LE 5
Memory	512 KB SRAM; external PSRAM and flash
GPIO	45 programmable pins; SPI / I²C / I²S / UART

Feature	Specification
Acceleration	Vector (SIMD) instructions

3.4.1 Display subsystem

The local interface uses a 7-inch capacitive touchscreen panel of 800 × 480 pixel resolution, driven over a parallel RGB interface with a GT911 capacitive touch controller on a shared I^2C bus (LVGL, 2023)[22]. The panel is driven by the LVGL graphics library, which provides widget rendering, touch input handling, and screen management suitable for the memory constraints of an embedded target.

3.5 Firmware Design

Development uses the Arduino framework for the ESP32-S3. Its responsibilities are to initialize the peripherals, establish network connectivity, acquire and condition sensor readings, render the user interface, and publish telemetry. Its detailed control flow is presented in Section 4.2.

3.5.1 Configuration portal

To avoid the need to recompile and reflash firmware when a node is moved between networks, the node hosts a configuration portal. Holding the boot button during start-up causes the node to raise a Wi-Fi access point named after its own hardware address and to serve a configuration page. Network credentials, addressing parameters, and temperature and humidity calibration offsets entered on this page are written to a LittleFS filesystem on the device and applied on the next start-up.

Figures 3.6 and 3.7 show the two variants of this page. The wireless variant accepts the network SSID and passphrase in addition to the addressing parameters, while the wired variant omits those two fields; the remaining fields, including the temperature and humidity offsets, are common to both. Presenting the offsets on the commissioning page rather than embedding them in the firmware means that a node can be corrected in place once a comparison value is available, without recompilation. In the deployment reported in this thesis both offsets were left at zero, as the figures show, because no comparison instrument was available; the implications of this are discussed in Section 5.7.

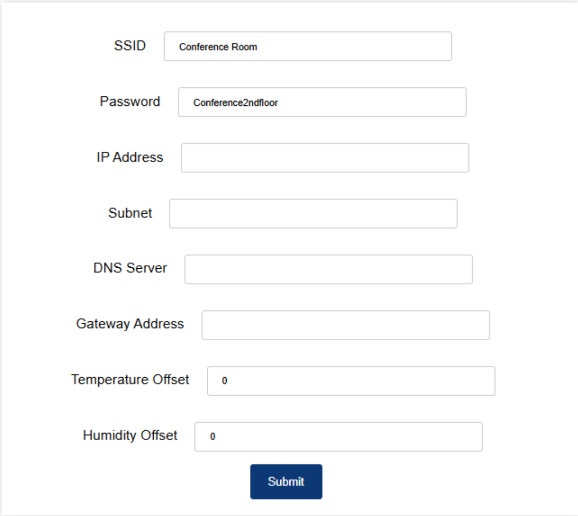

Figure 3.6 The configuration portal in wireless mode, showing network credentials, addressing parameters and the temperature and humidity offset fields.

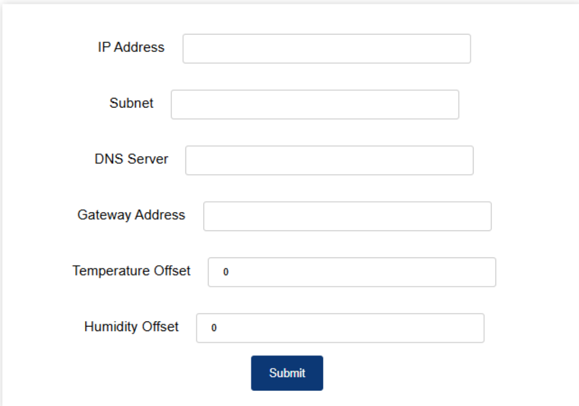

Figure 3.7 The configuration portal in wired mode. The SSID and passphrase fields are omitted; the offset fields are common to both variants and are shown here at their default value of zero.

3.5.2 Dual network mode

The node supports both wireless and wired operation. A mode flag persisted in non-volatile memory selects between Wi-Fi and an Ethernet interface, and the MQTT client is bound to the corresponding network client at start-up. This allows the same firmware and hardware to be deployed either in areas with reliable wireless coverage or in areas where a wired connection is preferred.

3.6 Communication Design

Table 3.5 summarizes the topic scheme. Namespacing each node under a per-device path allows a single wildcard subscription to aggregate every node, so the back end need not be reconfigured when nodes are added.

Table 3.5
MQTT TOPIC SCHEME

Topic	Role	Description
devices/<mac>/data	Publish (node)	Telemetry payload: CSV, node address plus eight values, five-second cadence, QoS 1
devices/+/data	Subscribe (server)	Wildcard subscription aggregating all nodes
esp32/control	Publish (server)	Alert and buzzer commands issued to a node, QoS 1
esp32/feedback	Subscribe (server)	Acknowledgement of alert state from a node

3.6.1 Message format

Each telemetry message is a compact comma-separated string carrying the node identifier followed by the eight measured values. This format was chosen for its minimal overhead and for straightforward construction on the microcontroller and parsing on the server; the payload remains well under one hundred bytes, which MQTT transports efficiently. The field order is fixed: node address, temperature, relative humidity, PM1.0, PM2.5, PM4, PM10, VOC index, NOx index. A representative payload is shown below.

```
"34:85:18:97:15:C8,25.40,60.10,1.20,3.40,4.50,5.60,120.00,15.00"
```

Because node identity is carried in the payload as well as in the topic, the server can resolve a message to a registered device independently of the topic structure, which simplifies the handling of heterogeneous deployments.

3.7 Server and Dashboard Design

A Node.js application built on Express handles the server role, with Socket.IO for real-time browser updates and a PostgreSQL database extended with TimescaleDB for time-series storage (OpenJS Foundation, 2024)[29]; (PostgreSQL Global Development Group, 2024)[31]. Its responsibilities are:

1. Subscribing to the broker and receiving telemetry from all nodes through the wildcard topic.

2. Resolving each incoming message to a registered device by hardware address, and discarding messages from unregistered devices.

3. Evaluating each reading against per-device control and specification limits to derive an out-of-range status.

4. Maintaining an in-memory cache of the most recent reading for every device, so that a newly connected dashboard is populated immediately rather than waiting for the next publish.

5. Persisting an hourly snapshot of all live devices to the time-series database for long-term history.

6. Detecting device silence and raising offline notifications, and dispatching alert notifications when a device remains out of range.

7. Serving the dashboard application and its supporting REST endpoints.

3.7.1 Dashboard features

The browser dashboard provides real-time graphs for each parameter, colour-coded threshold indication, historical trend views over daily, weekly and yearly ranges, a spatial map view on which device markers are positioned over an uploaded floor plan, device and location management pages, and export of records to spreadsheet format for reporting. The interface is responsive so that it runs unmodified on desktop and mobile browsers.

3.8 Data Flow

The end-to-end data flow proceeds as follows:

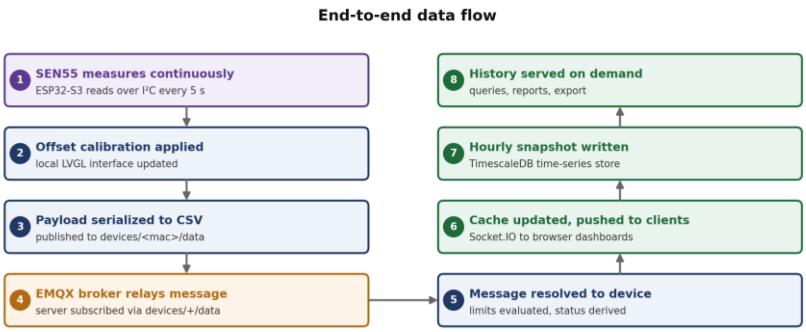

Figure 3.5
End-to-end data flow,

from acquisition at the sensing head through the broker and server to the dashboard and the time-series store.

1. The SEN55 maintains a continuous internal measurement; the ESP32-S3 reads the conditioned values over I^2C every five seconds.

2. The node applies temperature and humidity calibration offsets and updates the local LVGL interface.

3. The node serializes the readings to a comma-separated payload and publishes it to its per-device topic on the broker.

4. The server, subscribed through the wildcard pattern, receives the message, resolves it to a registered device, and evaluates its status against configured limits.

5. The server updates its in-memory cache and emits the reading to connected dashboard clients over Socket.IO.

6. On the hour, the server writes a snapshot of the latest readings for all live devices into the time-series database.

7. Historical queries, reports, and exports are served from the database on demand.

3.9 Testing and Validation

The validation strategy comprised four activities:

1. Unit testing: Individual modules, including sensor acquisition, network association, the MQTT client, and the LVGL interface, were tested independently before system integration.

2. Integration testing: the complete chain from sensor to dashboard was verified, confirming that a reading taken at the node appears on the remote dashboard and is subsequently retained in the database.

3. Continuous operation: the assembled system was operated without interruption over an extended period to observe stability, reconnection behaviour, and the character of the recorded data. The results of this activity are reported in Chapter 5.

4. Comparative validation: comparison of the sensor output against a calibrated reference instrument, in order to establish absolute measurement accuracy and to derive any necessary correction. This activity is identified as required future work in Chapter 6 and was not performed within the scope of the present study; the measurements reported in Chapter 5 are therefore relative and indicative rather than metrologically traceable.

3.10 Tools and Libraries

Table 3.6
DEVELOPMENT TOOLS AND LIBRARIES

Component	Tool or library
Firmware	Arduino framework for ESP32-S3

Component	Tool or library
Sensor driver	Sensirion I2C SEN5x library
MQTT client (node)	PubSubClient
Graphics	LVGL; Arduino_GFX; EEZ Studio for screen layout
Node filesystem	LittleFS
Node web server	ESPAsyncWebServer
Broker	EMQX
Server runtime	Node.js with Express
Server MQTT client	MQTT.js
Real-time transport	Socket.IO
Database	PostgreSQL with TimescaleDB
Dashboard charting	Chart.js
Reporting and export	ExcelJS

Chapter 4
Implementation

4.1 Hardware Implementation

The edge node integrates three principal components: the SEN55 sensor, the ESP32-S3 microcontroller module, and the 7-inch touchscreen display, powered from a regulated 5 V supply. The SEN55 communicates with the microcontroller over I²C and is supplied at 5 V, drawing on the order of 63–80 mA in continuous measurement mode. Its internal fan draws a metered air sample across the laser-scattering chamber, and the manufacturer specifies a service life exceeding ten years under continuous operation (Sensirion AG, 2022)[36].

The I²C bus for the sensor is configured at 100 kHz. This is lower than the maximum the bus supports, and was selected deliberately: at higher clock rates the sensor readings were observed to become intermittent, which is a common consequence of bus capacitance and wiring length in an assembled enclosure. The touchscreen controller occupies a separate bus instance, and the wired Ethernet interface, where used, is attached over SPI.

Figure 4.1 shows the assembled node in operation. The sensor is housed in a separate small enclosure connected to the display unit by a screened circular connector, which allows the sensing head to be positioned in the air volume of interest while the display is mounted for convenient viewing. The header of the interface reports the associated network and the address the node has been assigned, which assists commissioning and fault diagnosis on the floor without recourse to a terminal.

Figure 4.1
The assembled edge node in operation.

comprising the sensing enclosure (left) and the 7-inch touchscreen display unit (right). The interface header reports the associated network and the address assigned to the node.

Table 4.1
REPRESENTATIVE NODE POWER BUDGET (5 V SUPPLY)

Component	Current	Power
ESP32-S3 (Wi-Fi active, average)	~80 mA	~0.40 W
SEN55 (measurement mode)	~70 mA	~0.35 W
7-inch panel and backlight	~350 mA	~1.75 W
Node total (typical)	~0.5 A	~2.5 W

4.2 Firmware Architecture

Rather than a set of concurrently scheduled tasks, the firmware runs as a single cooperative super-loop as a set of concurrently scheduled tasks. On every loop iteration it services the LVGL user interface so that the local touchscreen remains responsive, and maintains the MQTT session. On a five-second cadence it polls the SEN55 over I²C, applies the configured temperature and humidity offsets, updates the global state used by the interface, serializes the readings, and publishes them. Structuring acquisition, display, and networking within one cooperative loop keeps the design simple and the memory footprint small, which is appropriate for a low-cost edge node.

The core acquisition-and-publish cycle is summarized in the following pseudocode.

```
every loop iteration:
    ui.refresh()                    // LVGL: ui_tick + lv_task_handler
    if not mqtt.connected():
        mqtt.reconnect()            // blocking retry
    mqtt.loop()

every 5000 ms:
    r <- SEN55.read()               // I2C
    r.temp += temp_offset           // offset calibration
    r.hum  += humi_offset
    payload <- csv(mac, r)          // "MAC,temp,hum,pm1,...,nox,capture_ms"
    mqtt.publish(topic, payload)    // QoS 1
```

Automatic MQTT reconnection provides resilience during extended operation: when the broker connection drops, the loop re-establishes the session before resuming publication. The return status of each publish call is checked together with the client connection state and logged, which detects local send failures and disconnections at the node, although it does not by itself constitute a broker-level delivery acknowledgement (O'Leary, 2020)[28].

A consequence of the single-loop structure should be noted explicitly. Because the reconnection routine blocks while it retries, an extended loss of broker connectivity can delay the servicing of the user interface within that iteration. This is an acknowledged trade-off of the cooperative design and is discussed further in Section 5.7 and revisited as future work in Chapter 6.

4.3 Server Implementation

On start-up the server subscribes to the broker and loads the registry of active devices from the database, including for each device its hardware address, sensor type, enabled parameters, and the four limit values lower and upper specification limits and lower and upper control limits for every measured parameter. These are held in memory and refreshed when the device registry changes.

4.3.1 Message handling

On receiving a message, the server first distinguishes the payload format: a payload beginning with a brace is parsed as JSON, and any other payload is parsed as comma-separated values. The comma-separated branch extracts the node address from the first field, strips any enclosing quotation marks, and converts the remaining eight fields to numeric values, treating non-numeric fields as absent.

The server then resolves the message to a registered device by looking up the node address, case-insensitively, in the device registry. Messages whose address is not present in the registry are discarded; a node must therefore be registered through the management interface before its data will be accepted. A per-device rate limit discards messages arriving more frequently than once per second, and payloads exceeding a fixed size threshold are rejected.

4.3.2 Status evaluation and alerting

Each accepted reading is evaluated against the limits configured for its device. A value beyond a specification limit yields a specification violation; a value beyond a control limit, but within specification, yields a control violation; otherwise the reading is in range. To avoid alerting on momentary excursions, a transition into an out-of-range state must persist for a confirmation interval before an alert is raised, and a longer confirmation interval is applied during a start-up grace period. Confirmed alerts are recorded in an alert log, dispatched by electronic mail and messaging notification, and used to command an audible indicator at the node.

4.3.3 Persistence

The server maintains an in-memory record of the most recent reading for every device. On the hour, it writes a snapshot of all live devices to the time-series database in a single batched insert, using an upsert so that a repeated write for the same device and timestamp updates rather than duplicates the record. This design decouples the high-frequency live stream from the storage rate: the dashboard and local interface update every five seconds, while the durable history accumulates one record per device per hour. The rationale is that continuous storage at the live rate would grow the database by three orders of magnitude for no analytical benefit at the timescales of interest for trend monitoring and reporting.

4.3.4 Liveness detection

Because telemetry is transported at QoS 1, the absence of a message is not distinguishable at the protocol level from a device that has stopped publishing. The server therefore implements liveness detection at the application layer: if no message has been received from a device for longer than a fixed timeout, the device is marked offline, its cached readings are cleared, any outstanding alert is resolved, and the dashboard is updated. If the device remains offline beyond a longer configurable interval, an offline notification is dispatched.

4.4 Local Human-Machine Interface

LVGL drives the local interface on the 7-inch panel, and is shown running in Figure 4.1. Each parameter is presented as a labelled tile with a colour-coded status band derived from configurable thresholds, so that an out-of-range condition is visible instantly from a distance. Temperature and relative humidity are additionally rendered as circular gauges with the operating band marked, and each tile carries the running minimum and maximum observed since the last daily reset, which gives an operator immediate context for whether a present reading is typical or unusual without consulting the historical record.

The layout groups the four particulate fractions together under a single heading and places the gaseous indices adjacent to them, reflecting the distinction between the two classes of contaminant. Representative threshold bands are listed in Table 4.2; these are illustrative facility settings and would be aligned to the target ISO class and process requirements in practice (World Health Organization, 2021)[44].

Table 4.2
REPRESENTATIVE ALERT THRESHOLDS (ILLUSTRATIVE)

Parameter	Green (normal)	Amber (watch)	Red (alert)
PM2.5 (μg/m³)	< 12	12 – 35	> 35
PM10 (μg/m³)	< 20	20 – 50	> 50
VOC index	< 150	150 – 250	> 250
Temperature (°C)	20 – 24	18–20 / 24–26	< 18 or > 26
Relative humidity (%)	40 – 60	30–40 / 60–65	< 30 or > 65

4.5 Remote Dashboard

A browser application served by the Node.js back end provides the remote dashboard. On connection it receives the current cached readings for all devices, so the display is populated immediately, and thereafter receives incremental updates pushed over Socket.IO as each telemetry message is processed. Access is controlled by token-based authentication, with the sign-in page shown in Figure 4.2.

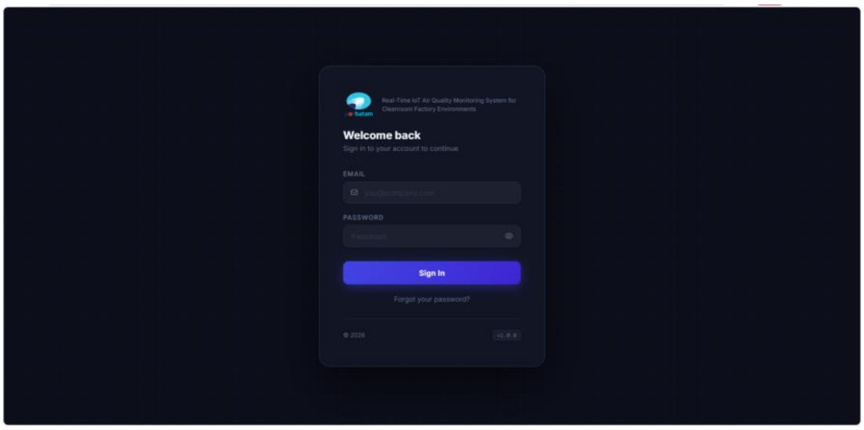

Figure 4.2
Dashboard authentication page.

4.5.1 Dashboard view

The principal view, shown in Figure 4.3, presents a summary bar displaying the total number of registered devices, together with the numbers currently online, out of range, and offline. A device panel on the left lists all registered nodes, including their hardware addresses, assigned names, locations, current temperature and humidity readings, and status indicators. When a device is selected, the main area displays a time series chart for each monitored parameter. The charts include four configured threshold lines representing the lower specification limit, upper specification limit, lower control limit, and upper control limit, allowing the relationship between each reading and its permitted range to be observed directly. The current value, its short term trend, and any out of range condition are also displayed. Users can select real time, daily, weekly, or monthly viewing ranges.

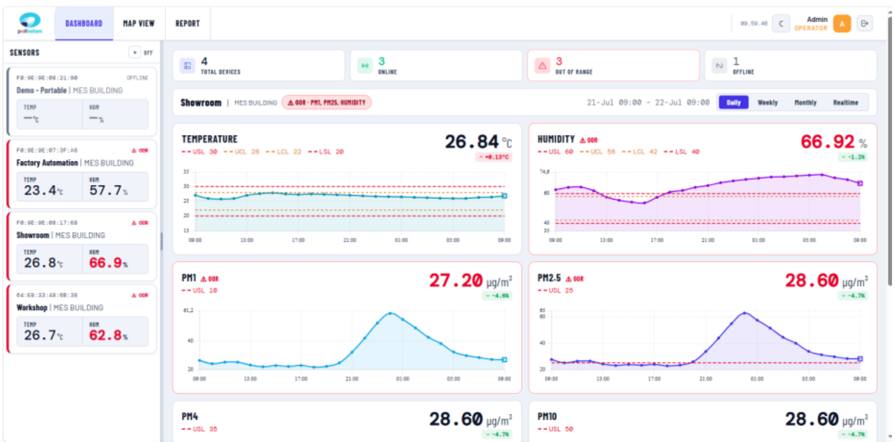

Figure 4.3
Dashboard view,

showing the device panel, the summary counts, and per-parameter time-series charts annotated with the configured specification and control limits.

4.5.2 Map view

The map view, shown in Figure 4.4, displays each device as a marker on an uploaded floor plan of the monitored building. Marker positions are assigned through an edit mode and remain fixed thereafter. Each marker is coloured according to device status: green for readings within limits, amber for a control limit warning, red for a specification limit violation, and grey when the device is offline. This colour coding enables the spatial distribution of environmental conditions across the facility to be understood at a glance. Selecting a marker opens a summary card displaying the current value of each monitored parameter for that device, together with a derived air quality index based on the U.S. EPA method (U.S. EPA, 2018) [42]. This view directly addresses the limitation of sparse measurement identified in Section 1.2. While a single instrument represents conditions at only one location, a distributed deployment provides a broader view of conditions across the entire floor.

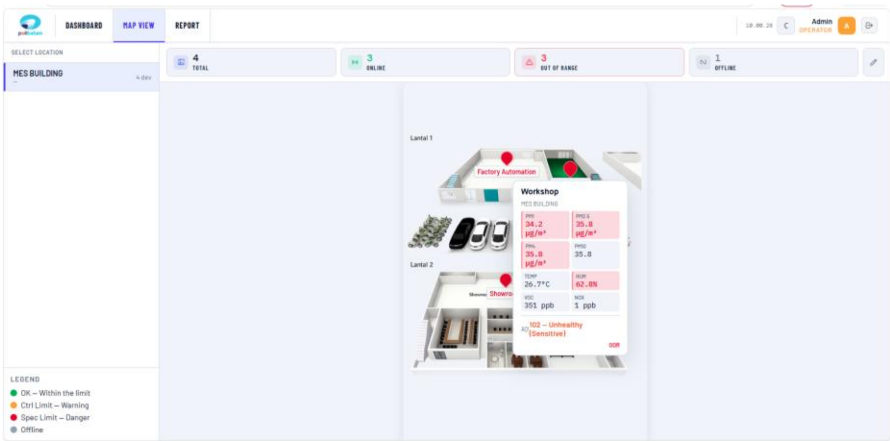

Figure 4.4
Map view, showing device markers positioned on the building floor plan, coloured by status, with a summary card for the selected device.

4.5.3 Report and export

The report view, shown in Figure 4.5, allows a date range and a device, or all devices, to be selected and generates a tabular record of the stored readings together with a summary of the number of records, the number of out-of-range readings, and the count of active and total alerts over the period. The resulting record may be exported to spreadsheet format for external analysis, reporting or archival, which supports the auditing and record-keeping objective set out in Section 1.3.

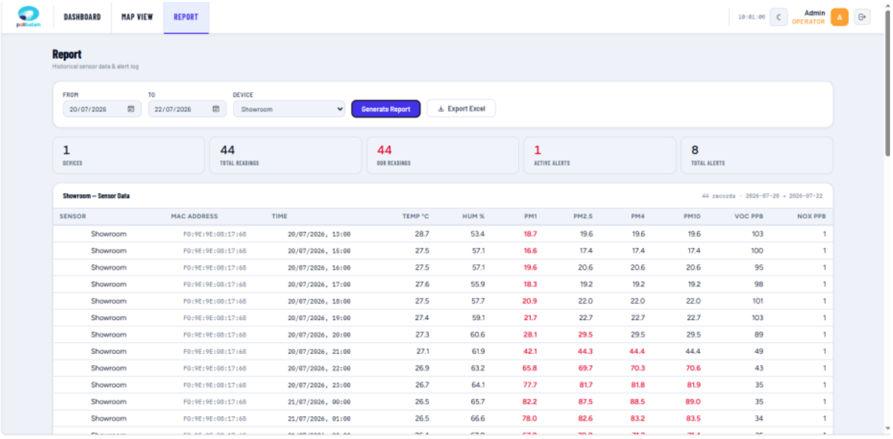

Figure 4.5
Report view, showing the selected date range, the summary counts, and the tabular record of stored readings with out-of-range values highlighted.

4.5.4 Responsive operation on mobile devices

Because the dashboard is a standard responsive web application rather than a native client, it runs unmodified on mobile browsers. Figures 4.6 to 4.8 show the dashboard, map and report views rendered at a mobile viewport width. The layout reflows to a single column, the device and location selectors collapse into dropdown controls, and the charts and map remain interactive. This satisfies the requirement that a supervisor away from the floor have the same visibility as an operator at the workstation, without installing dedicated software.

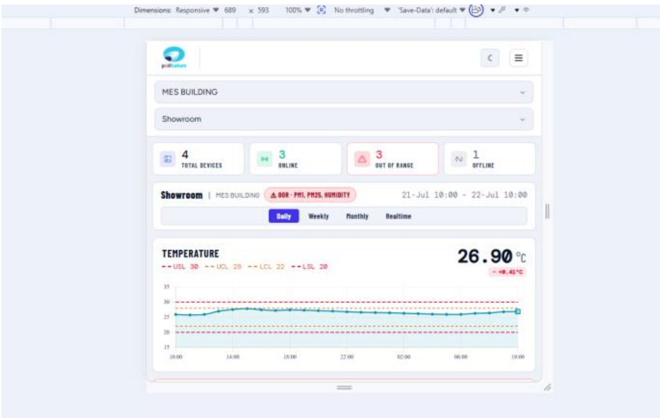

Figure 4.6
Dashboard view rendered at a mobile viewport width

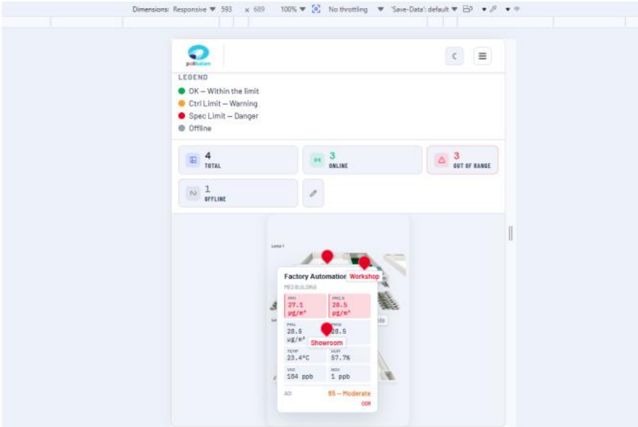

Figure 4.7
Map view rendered at a mobile viewport width

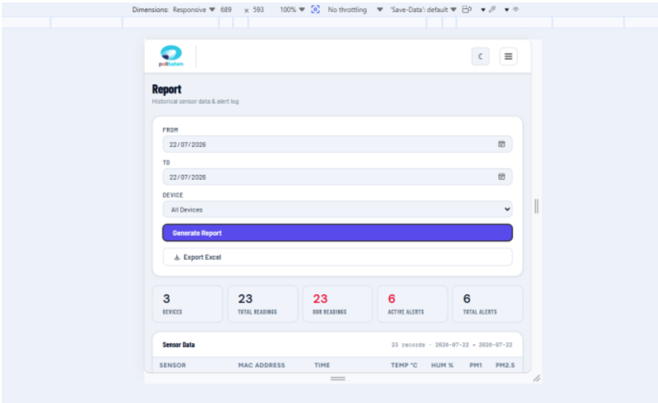

Figure 4.8
Report view rendered at a mobile viewport width.

4.6 Mechanical Design and Enclosure

The electronics are divided between two custom enclosures: a display and interface unit that carries the microcontroller and the touchscreen, and a separate sensing head that carries the SEN55. The two are joined by a screened circular connector. Separating them allows the sensing head to be placed in the air volume of interest while the display is mounted where it is convenient to read, and it keeps the sensor intake and its fan away from the heat generated by the display backlight, which would otherwise bias the temperature and relative-humidity channels.

4.6.1 Display and interface enclosure

The display unit is a three-part assembly. A front bezel retains the 7-inch capacitive touchscreen against the housing and conceals the panel edge together with its flexible connector. The panel assembly sits immediately behind the bezel, and the rear housing carries the ESP32-S3 board. One side face of the rear housing is extended into a raised boss that accommodates the external connectors, so that all cabling leaves the unit in a single direction rather than from opposite faces.

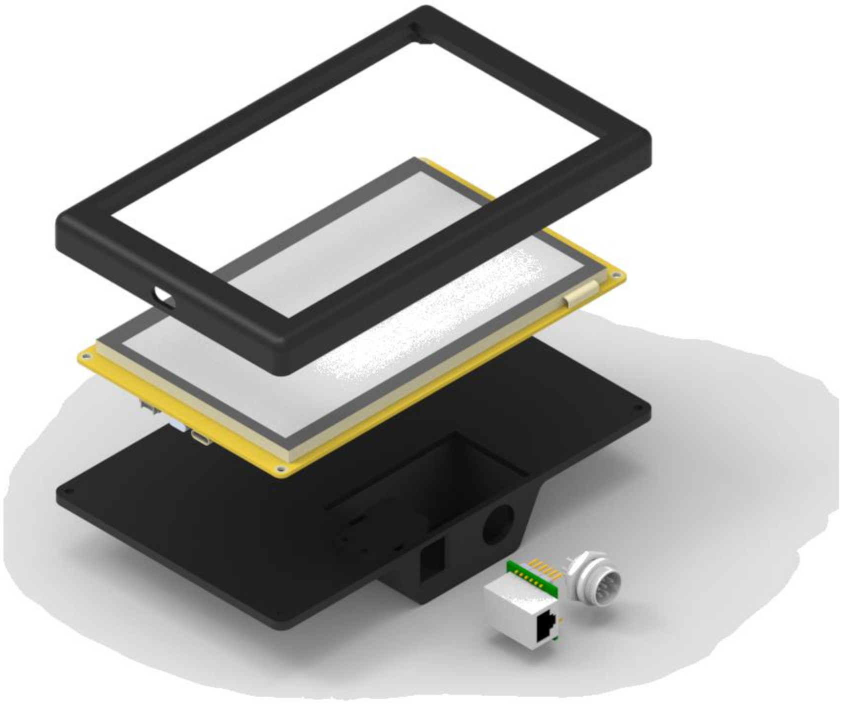

Figure 4.9
Exploded view of the display and interface enclosure,

showing the front bezel, the 7-inch capacitive touchscreen panel, the rear housing, and the RJ45 and circular connectors.

Two connectors are mounted in that boss: an RJ45 socket serving the wired Ethernet interface described in Section 3.5.2, and a circular threaded connector carrying the supply and I^2C lines to the sensing head. A separate aperture in the opposite side face provides access for the power connection. Grouping the external interfaces in this way keeps the remaining faces unbroken, which reduces the number of joints through which particles could be shed or drawn.

Figure 4.10
The assembled display and interface unit,

showing the connector boss carrying the RJ45 socket and the circular connector to the sensing head.

4.6.2 Sensing head enclosure

The sensing head encloses the SEN55 between a moulded top cover and a base plate. The top cover is pierced by a circular aperture aligned with the sensor's sample intake and by a rectangular slot aligned with its exhaust, so that the metered airflow path through the laser-scattering chamber is preserved. Aligning the apertures with the sensor's own inlet and outlet, rather than ventilating the enclosure generally, keeps the sampled volume defined by the sensor itself and avoids recirculation of already-measured air within the box.

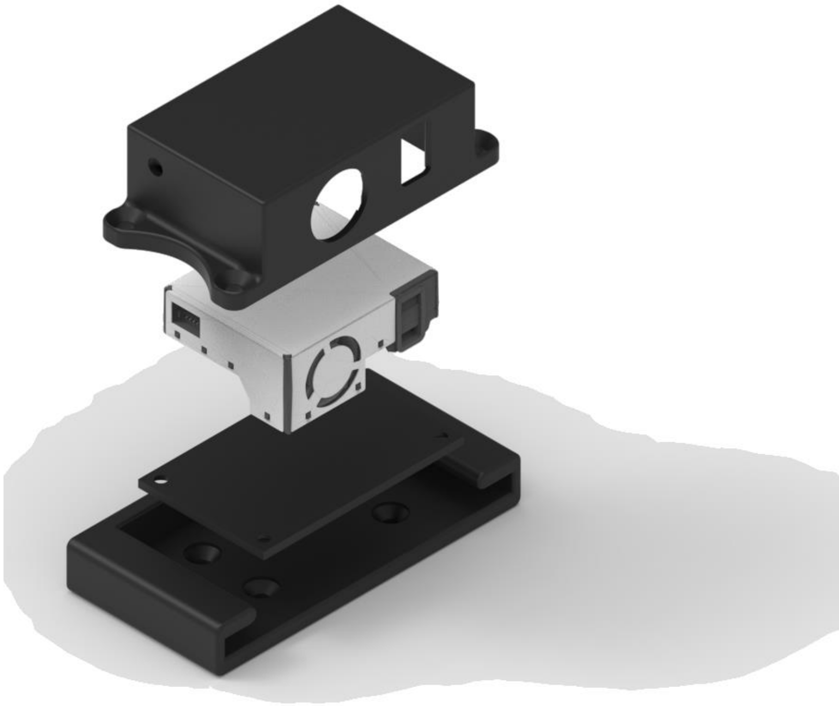

Figure 4.11
Exploded view of the sensing head,

showing the top cover, the SEN55 sensor module, and the base plate with its integral mounting foot.

The base plate extends beyond the body of the enclosure on two sides to form a mounting foot with slotted holes. The slots allow the head to be fixed to a wall, bracket or equipment frame and its orientation adjusted during commissioning, which matters because the intake should not face directly into filtered supply air, as noted below.

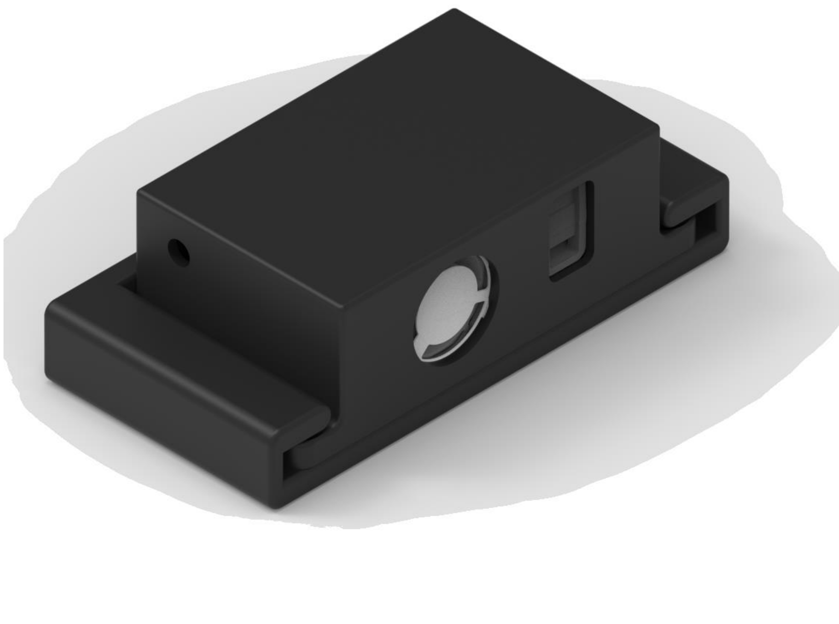

Figure 4.12
The assembled sensing head.

The circular aperture aligns with the sensor intake and the rectangular slot with its exhaust.

4.6.3 Deployment considerations

The electronics are housed in a compact enclosure with a screened intake positioned to sample representative room air near the workstation breathing zone. Where the monitor is to be deployed in a controlled production area, enclosure materials should be selected to be non-shedding and low-outgassing so that the monitor itself does not become a contamination source (ISO, 2022)[15]; (Whyte, 2010)[43], and the sensor intake should be oriented to avoid the direct impingement of supply air.

4.7 Security Considerations

The prototype described in this thesis was developed and operated on an isolated local network, and the messaging layer was configured without transport encryption or broker authentication in order to simplify development. This is acceptable for a laboratory prototype but would not be appropriate for deployment on a production plant network (Roman et al., 2013)[33], and the following measures are identified as prerequisites for such a deployment rather than as features of the present implementation.

1. Transport encryption: the MQTT session should be secured with TLS so that telemetry is protected in transit and the broker identity can be verified (OASIS, 2019)[27]; (EMQX, 2024)[8].

2. Broker authentication and authorization: per-node credentials should be issued and access-control lists applied so that each node may publish only within its own topic subtree. Confining a node’s publish rights to its namespace limits the consequences of a compromised device.

3. Dashboard access control: the server already implements token-based authentication for dashboard users; secrets should be provisioned per deployment rather than held in the source configuration, and rotated on a defined schedule.

4. Firmware integrity: signed images and a secure over-the-air update mechanism would allow deployed nodes to be maintained without physical access, which is a practical necessity where nodes are installed in restricted-access production areas. Over-the-air update is not implemented in the present firmware and is identified as future work.

Chapter 5
Results And Discussion

5.1 Deployment and Data Collection

The integrated system was assembled and operated continuously over a seventy-two-hour period, from 17:00 on 23 July to 02:00 on 27 July 2026. Throughout this period the node sampled and published on the five-second cadence described in Chapter 4, the local touchscreen and the remote dashboard both presented the live stream, and the server accumulated the hourly history from which the figures in this chapter are plotted.

The deployment was not limited to a single node. Four nodes were registered and operated concurrently within one building. These were designated as Factory Automation, Showroom, Workshop, and a portable demonstration unit. Each node was assigned to a specific location and positioned on the floor plan shown in Figure 4.4. At the time the dashboard images in Figures 4.3 and 4.4 were captured, three of the four nodes were online, one was offline, and out of range conditions had been detected for three devices. The platform's multi node capabilities were therefore demonstrated in actual operation. These capabilities included concurrent data aggregation through a single wildcard subscription, device specific limit evaluation, independent status reporting, and offline detection. The detailed time series analysis presented in the following section is based on data collected from the Showroom node over a seventy two hour period.

It should be stated clearly at the outset that this deployment was conducted in an ordinary indoor environment rather than inside a classified cleanroom. The recorded concentrations and thermo-hygrometric values reflect that setting, and the purpose of the exercise was to demonstrate stable end-to-end operation of the complete system and to characterize the behaviour of the recorded data, not to characterize a controlled environment. The implications of this are considered in Section 5.7.

5.2 Real-Time Monitoring Behaviour

Figures 5.1 to 5.5 show the recorded seventy-two-hour record of all measured parameters. At the edge, parameters are sampled and published every five seconds for live local and remote display; for long-term storage the server logs an hourly snapshot to the database, from which these figures are plotted.

The particulate concentrations follow a repeating diurnal pattern across the three days: an elevated period on the first evening reaching 82 μg/m³ (PM10), followed by a low baseline of approximately 11–20 μg/m³ and successive daytime rises to intermediate peaks near 36 and 32 μg/m³ on the following afternoons. The four particulate fractions track one another closely, in the expected order PM10 ≥ PM4 ≥ PM2.5 ≥ PM1. The narrow separation between the fractions indicates an aerosol dominated by fine particles: the PM1.0 mean of 22.7 µg/m³ accounts for approximately 95 per cent of the PM10 mean of 23.9 µg/m³, so almost the entire measured mass lies below one micrometre (Seinfeld & Pandis, 2016)[35].

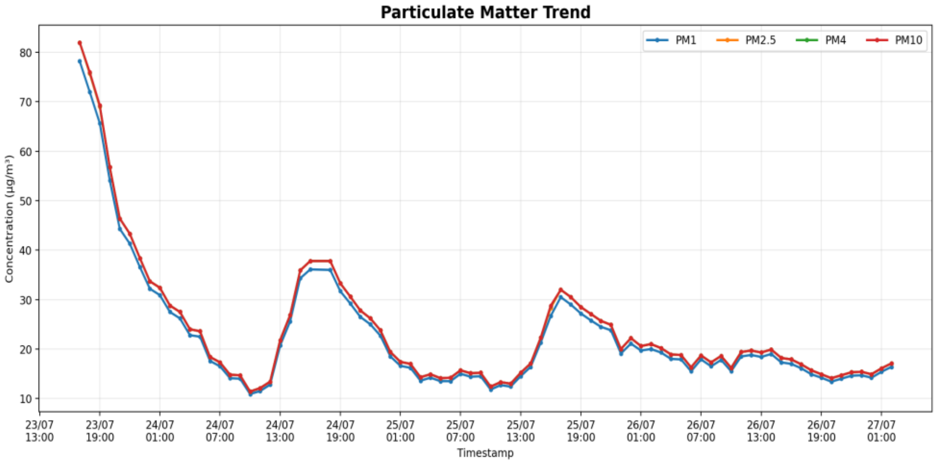

Figure 5.1
Particulate matter concentration.

Hourly-logged PM1, PM2.5, PM4 and PM10 over the seventy-two-hour period. All four fractions track closely, with an elevated first evening reaching approximately 82 µg/m³ (PM10) and subsequent daytime peaks near 36 and 32 µg/m³ over a baseline of 11–20 µg/m³.

It falls to approximately 15 points during periods of elevated particulate concentration and rises to a maximum of 234 points, the highest values occurring overnight and in the early morning. This anti-correlation indicates that the particulate and gaseous signals in this environment are driven by different sources rather than by a common one; a plausible interpretation is that the daytime particulate rises are associated with activity and external air exchange (Chen & Zhao, 2011)[6], while the overnight VOC rise reflects accumulation in a less ventilated space. Establishing the cause would require controlled experiments beyond the scope of this deployment.

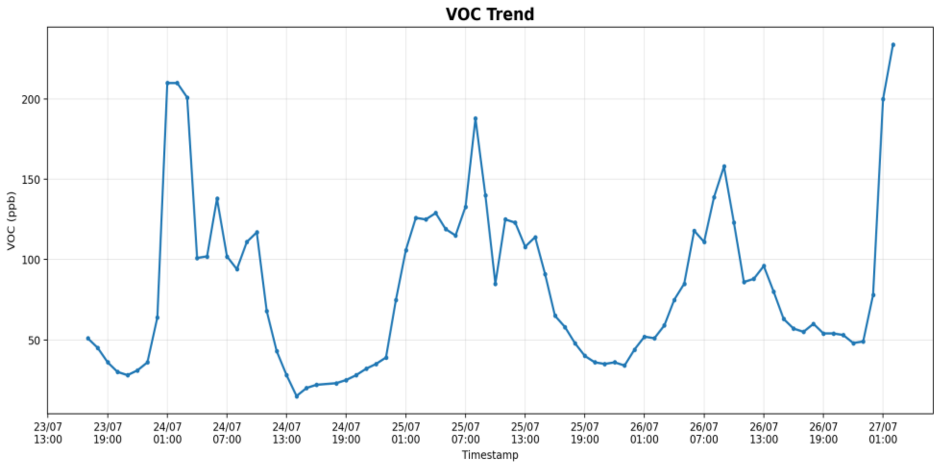

Figure 5.2
VOC index

Hourly-logged VOC index over the same period, falling to approximately 15 points during elevated particulate periods and rising to a maximum of 234 points overnight and in the early morning.

Temperature and relative humidity, shown in Figures 5.3 and 5.4, vary smoothly and without discontinuity across the record, within 25.8–28.7 °C and 53.4–68.4 per cent relative humidity respectively. The absence of steps or spikes in these two channels is itself a useful indication that the acquisition and transport chain operated without corruption throughout.

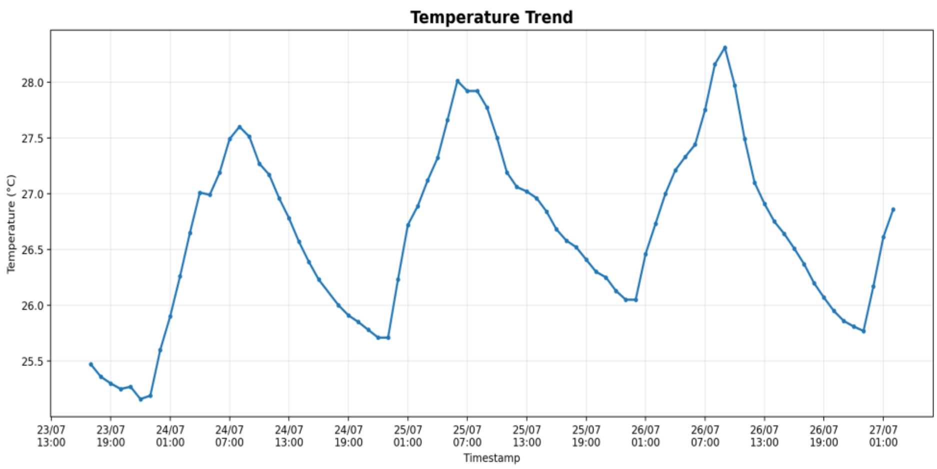

Figure 5.3
Temperature,

Hourly-logged temperature, varying smoothly between 25.2 °C and 28.3 °C (mean 26.6 °C) across the seventy-two-hour record, repeating a daily cycle each of the three days.

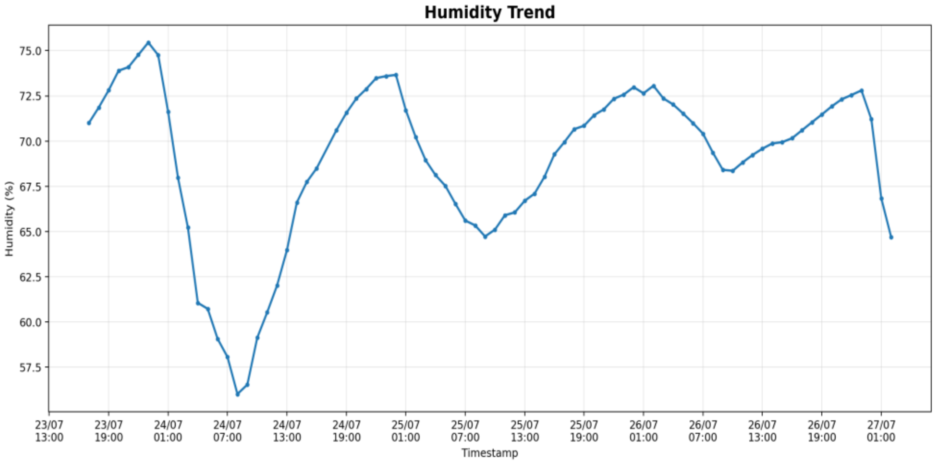

Figure 5.4
Relative humidity.

Hourly-logged relative humidity, ranging from 56.0 to 75.5 per cent (mean 68.9 per cent), rising overnight and easing through the warmer part of each day.

The NOx index, shown in Figure 5.5, remained at the sensor's baseline value of 1 for almost the entire recording period, registering an isolated value of 2 on only a few hourly samples. This near-constant value does not represent a measured concentration: the metal-oxide NOx algorithm in the SEN55 requires an extended conditioning period before it produces a variable output, and the recorded trace reflects that baseline state. The NOx channel is therefore reported here for completeness but is not interpreted, and obtaining a meaningful NOx record is identified as future work.

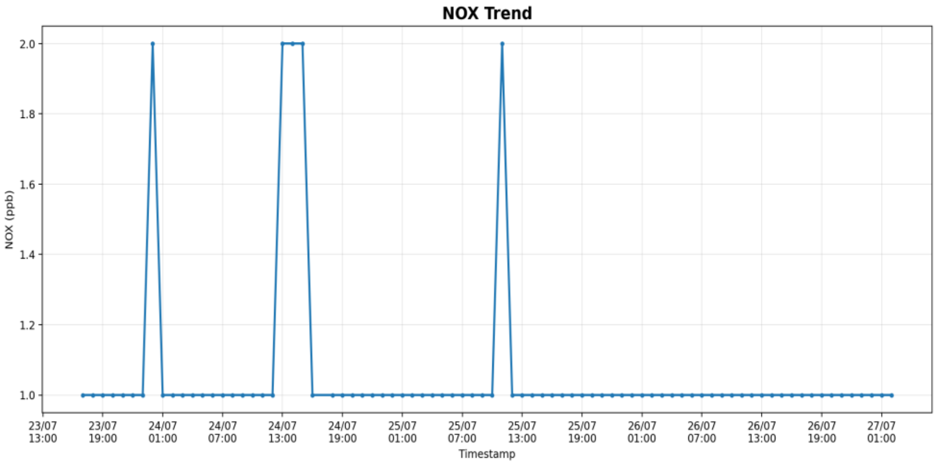

Figure 5.5
NOx index,

Hourly-logged NOx index, which remained at the sensor baseline value of 1 for almost the entire recording period, reflecting the extended conditioning the SEN55 NOx algorithm requires rather than a measured concentration.

5.3 Parameter Statistics

Table 5.1 summarizes the per-parameter statistics computed from the seventy-two hourly records.

Table 5.1
SEVENTY-TWO-HOUR PARAMETER STATISTICS

Parameter	Minimum	Mean	Maximum	Unit
PM1.0	10.9	22.7	78.2	µg/m³
PM2.5	11.4	23.8	82.0	µg/m³
PM4	11.4	23.9	82.0	µg/m³
PM10	11.4	23.9	82.0	µg/m³
VOC index	15	83	234	points
Temperature	25.2	26.6	28.3	°C
Relative humidity	56.0	68.9	75.5	%RH
NOx index	1.0	1.1	2.0	points*

** The NOx index remained at its baseline value throughout and does not represent a measured concentration; see Section 5.2.*

5.4 Responsiveness and Reliability

Three quantitative properties of the deployed system merit characterization: the end-to-end latency from publication at the node to rendering on the remote dashboard, the proportion of published messages successfully delivered during continuous operation, and the agreement of the sensor output with a calibrated reference instrument.

The measurement procedures for the first two are defined as follows. End-to-end latency is measured as the interval between the instant at which a payload is published at the node and the instant at which the corresponding value is rendered in the browser, obtained by instrumenting both endpoints against a common time base and recording the distribution over a large number of messages. Delivery ratio is measured by counting messages published at the node over a defined interval and comparing that count with the number received and accepted by the server over the same interval, the server already maintaining a received-message counter for this purpose.

Both measurements were subsequently performed and are reported below. To obtain them the node firmware was extended to synchronize its clock over the network time protocol at start-up and to append the capture instant, expressed as milliseconds since the Unix epoch, as a tenth field of the published payload. Because the server parses only the first nine fields, the additional field is ignored by the existing pipeline and no change to the server was required. An independent subscriber, written in Python and connected to the broker under its own client identifier, recorded the arrival instant of every message together with the capture instant carried within it. The difference between the two is the end-to-end latency from sensor capture at the node, through the broker, to receipt at the subscriber. Telemetry was published at quality-of-service level 1 for this measurement so that delivery is acknowledged by the broker.

Before presenting the latency and delivery figures, one earlier observation is retained because it characterizes a different property. The update cadence, that is the interval between successive updates arriving at the dashboard for a given node, was recorded over seventy-five samples drawn from three nodes by timestamping each Socket.IO update in the browser. The observed interval had a median of 5.02 s and a mean of 5.02 s, with the large majority of intervals falling within one hundred milliseconds of the five-second acquisition period configured at the node. This confirms that the five-second cadence is preserved end to end. It is a measure of update regularity rather than of transport delay: it establishes that updates arrive on schedule, but it does not by itself quantify the time taken for an individual reading to travel from the node to the screen.

A separate observation, made before the firmware was instrumented, measured only the server-to-dashboard segment by comparing the server's ISO 8601 processing timestamp against the browser's receipt time; across seven consecutive updates the median was approximately 0.3 seconds. That figure is superseded by the end-to-end measurement reported below and is retained here only to note that the final segment of the chain is not the dominant contributor to overall delay.

The end-to-end latency was recorded over a continuous observation window of three hours and seventeen minutes with three nodes publishing concurrently, yielding seven thousand and fifty-four matched samples. One systematic effect required correction before the distribution could be interpreted. Although every node synchronizes its clock over the network time protocol, residual offsets remain between each node and the recording host, and these appeared in the raw data as a proportion of negative latencies, which are physically impossible. The offset was estimated separately for each node as the negative of its first-percentile observation, on the reasoning that the fastest transfers approximate zero transport time and therefore expose the clock difference directly. The offsets so obtained were 137 ms, 85 ms and 135 ms for the three nodes respectively, and each node's samples were shifted by its own offset. The correction affects the absolute values but not the spread of the distribution, and its magnitude is small relative to the reported percentiles; it is stated here so that the adjustment is visible to the reader rather than concealed within the result. The complete record from which these figures are derived is summarized in Appendix E.

After correction the end-to-end latency had a median of 141 ms and a mean of 196 ms. The ninetieth percentile was 357 ms, the ninety-fifth percentile 490 ms and the ninety-ninth percentile 1059 ms, with a maximum of 5217 ms across the whole window. Six thousand nine hundred and seventy-four of the seven thousand and fifty-four samples, or 98.87 per cent, were delivered in under one second. Taken per node, the medians were 139 ms, 136 ms and 148 ms and the ninety-fifth percentiles 411 ms, 609 ms and 461 ms, so the three nodes behaved consistently with one another. NFR-1, which sets a target of less than one second, is therefore met at the median and at the ninety-fifth percentile, although approximately one sample in ninety exceeded one second. Delivery ratio was obtained from the same record by comparing the number of messages received from each node against the number expected from its five-second publication cadence over the same window. The three nodes received 2350, 2350 and 2354 messages against 2365 expected in each case, giving 99.37, 99.37 and 99.53 per cent respectively and 99.42 per cent overall. The longest interruption in the record was a single interval of thirty seconds affecting one node, consistent with a brief reconnection rather than sustained loss. NFR-2, which sets a target above 99 per cent, is therefore also met. An earlier and shorter measurement over a two-hour window produced a median of 89 ms and a ninety-fifth percentile of 333 ms; the longer window reported here captures a wider range of network conditions and is therefore the more conservative and more representative result.

Figure 5.6 shows the effect of the correction directly. In panel (a) a substantial proportion of the raw differences lie to the left of zero; since a message cannot arrive before it was captured, these observations can only be an artefact of the clock difference between node and recording host, and their extent is what permits the offset to be estimated. Panel (b) shows the same seven thousand and fifty-four samples after each node has been shifted by its own offset. The shape of the distribution is unchanged, as it must be, since a constant shift moves the whole distribution without altering its spread; only its position on the axis differs.

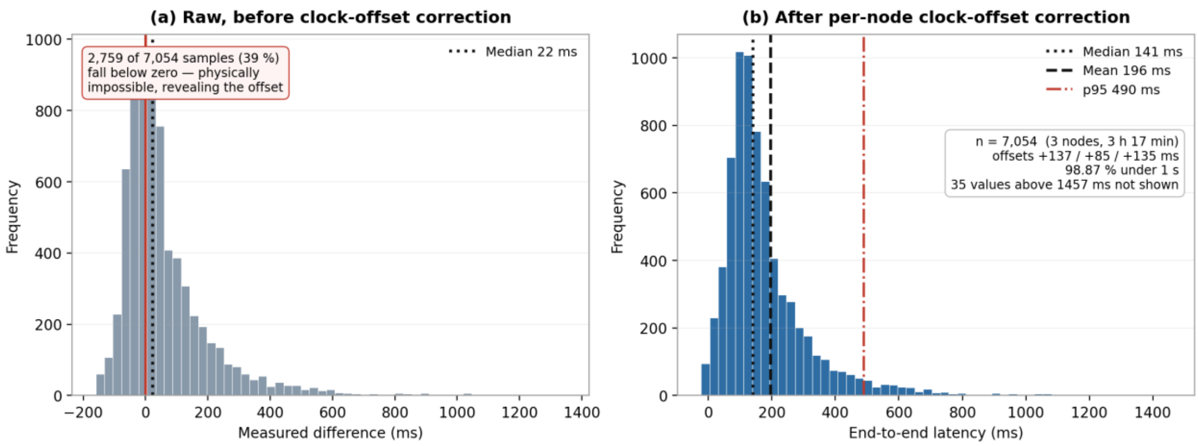

Figure 5.6 Distribution of end-to-end latency before and after per-node clock-offset correction,

Panel (a) shows the raw differences, of which 2,759 of 7,054 fall below zero and are therefore physically impossible, exposing the residual offset. Panel (b) shows the same samples after each node is shifted by its own offset, giving a median of 141 ms and a ninety-fifth percentile of 490 ms.

The third property, measurement agreement against a reference instrument, requires a co-located calibrated particle counter operated alongside the node over a range of concentrations, with regression of the two data sets to obtain a coefficient of determination and any systematic offset. No such reference instrument was available for this study and no comparison was performed. Accordingly, all concentrations reported in this chapter are relative and indicative; they are suitable for observing trends, relative comparison between locations, and early warning, but they are not metrologically traceable and must not be used as the basis for a compliance determination. This constraint is consistent with the wider low-cost-sensing literature, which reports good correlation with reference instruments after calibration while cautioning that humidity and particle composition affect absolute accuracy (Kelly et al., 2017)[17]; (Crilley et al., 2018)[7]; (Morawska et al., 2018)[25].

5.5 Scalability

Because each node publishes independently under its own namespace and the server subscribes through a wildcard, adding a node imposes no change to the back-end subscription. The only per-node action required is registration of the node address in the device registry, which is performed through the management interface and takes effect without restarting the service. As recorded in Section 5.1, four nodes were operated concurrently through a single wildcard subscription, with per-device limits, status and offline detection maintained independently, which confirms this property in operation for the deployment size tested.

The load imposed by each node is modest. At one message every five seconds and a payload below one hundred bytes, a single node generates on the order of twenty bytes per second of application payload, so even a large deployment produces an aggregate load that a commodity broker relays comfortably. The practical constraints are the capacity of the wireless infrastructure and the provisioning of the broker and database host rather than the node or the protocol, and both scale with standard infrastructure. Storage grows at one row per device per hour, so a hundred-node facility accumulates fewer than one million rows per year, which is a small volume for a time-series database.

5.6 Indicative Cost

A central motivation for the design is affordability. Table 5.2 gives an indicative bill of materials for a single node. At an approximate total of Rp2,460,000 per node, the platform costs one to two orders of magnitude less than a dedicated facility particle counter, which makes dense, per-workstation instrumentation economically feasible rather than a capital-intensive exception. Pricing is indicative and will vary with supplier, integration effort, and volume; costs are converted at an indicative rate of one United States dollar to Rp20,000.

Table 5.2
INDICATIVE PER-NODE BILL OF MATERIALS

Component	Indicative cost (USD)	Indicative cost (IDR)
SEN55 multi-parameter sensor	~US$45	~Rp900,000
ESP32-S3 module	~US$8	~Rp160,000
7-inch capacitive touchscreen	~US$45	~Rp900,000

Component	Indicative cost (USD)	Indicative cost (IDR)
Enclosure, power supply, wiring	~US$25	~Rp500,000
Total per node	~US$123	~Rp2,460,000

5.7 Calibration Status

The question of calibration bears directly on how the values in this chapter should be read, and is therefore stated explicitly. Three distinct things are often grouped under the single word calibration: the manufacturer's factory calibration of the sensing element, any field adjustment applied by the integrator, and verification against a traceable reference instrument. This study carries the first, provides a mechanism for the second, and has not performed the third.

The SEN55 is supplied factory-calibrated by the manufacturer, and the particulate, VOC, NOx and thermo-hygrometric outputs are produced by the sensor's internal algorithms against that calibration (Sensirion AG, 2022)[36]. No adjustment of the particulate channels was made in this work; the mass concentrations reported throughout this chapter are the sensor's own outputs, unmodified. The firmware additionally provides a field-settable temperature and humidity offset, editable through the configuration portal and stored on the device filesystem as described in Section 3.5.1, which exists so that a node mounted in a warm enclosure or near a heat source can be corrected in situ. It should be stated plainly that this mechanism was implemented but not exercised: no offsets were determined against any comparison instrument during this study, and the values recorded in this chapter are therefore uncorrected sensor output.

A distinction should be drawn between precision and accuracy, because the recorded data supports a statement about one but not the other. Precision, in the sense of repeatability and freedom from spurious variation, can be assessed from the seventy-two-hour record itself and is good: the traces vary smoothly and continuously rather than jumping between unrelated values, the four particulate fractions preserve the physically required ordering PM10 ≥ PM4 ≥ PM2.5 ≥ PM1 in every one of the eighty-one hourly records, the thermo-hygrometric channels follow a coherent daily cycle across three consecutive days, and the particulate and VOC signals move in a consistent anti-correlated pattern rather than independently. Taken together these indicate a stable sensing element producing internally consistent output.

Accuracy, in the sense of agreement with the true concentration, is not established by any of that. A sensor may be entirely repeatable while carrying a constant offset or a scale error, and the low-cost optical particle sensing literature reports precisely this pattern: such devices track reference instruments closely in relative terms while their absolute readings are affected by ambient humidity and by the composition of the aerosol (Kelly et al., 2017)[17]; (Crilley et al., 2018)[7]; (Jayaratne et al., 2018)[16]. Establishing accuracy requires comparison against a traceable reference, and no such comparison was performed in this study because no calibrated reference instrument was available. The consequence, stated once more for the avoidance of doubt, is that every concentration reported in this chapter is indicative of relative variation and trend, and is not a metrologically traceable measurement. The planned procedure for closing this gap is set out in Section 6.2.2.

5.8 Discussion

A seventy-two-hour record at this cadence contains on the order of fifty thousand acquisitions, against perhaps a handful of manual measurements over the same period under a conventional regime. That difference in observational density is the first practical consequence of the design. The second, and the one that distinguishes this system from the prototypes surveyed in Section 2.4, is that the two presentation paths are independent of one another. Because the local interface is rendered from the sensor reading within the firmware loop rather than from a value returned by the server, it continues to present current conditions when the wireless association drops, when the broker becomes unreachable, or when the server is unavailable. A remote-only architecture fails closed in each of those cases: the operator standing beside the equipment, who is the person best placed to act on a change in local conditions, loses all visibility at precisely the moment when local awareness matters most. The architecture reported here degrades instead to local-only operation, retaining the reading where it is most immediately useful while temporarily losing aggregation, history and remote access. The remote path in turn supplies what a single node structurally cannot: comparison across locations, spatial presentation, alert logging and durable history. Neither path substitutes for the other, and it is their combination rather than either individually that constitutes the contribution.

Second, the recorded data are internally consistent and physically plausible. The particulate fractions maintain their expected ordering across the full concentration range, the thermo-hygrometric channels vary smoothly, and the diurnal structure of the particulate record is coherent. This indicates that commodity hardware can provide measurements of sufficient fidelity for trend monitoring, early warning, and relative comparison between locations, consistent with the broader low-cost-sensing literature. Certified absolute measurement remains a separate matter requiring calibration against a reference instrument.

Third, the publish–subscribe design makes the system readily scalable, as set out in Section 5.5: a newly deployed node is aggregated by the existing wildcard subscription once its address is registered, without back-end reconfiguration.

Several limitations should be acknowledged. Low-cost optical particulate sensors are sensitive to high humidity and to the particle-size and composition assumptions embedded in their factory calibration; field calibration against a reference is recommended for absolute accuracy (Crilley et al., 2018)[7]; (Rai et al., 2017)[32]. Relative humidity in the present record varied between 53 and 68 per cent, a range over which optical particle sizing can be affected, so the reported concentrations should be treated as indicative (Jayaratne et al., 2018)[16]; (Badura et al., 2018)[4]. The VOC output is an index rather than an absolute concentration and is best used for relative trend detection, and the NOx index remained at its baseline throughout, as discussed in Section 5.2.

Wireless coverage and broker availability bound delivery reliability. Telemetry is transmitted at QoS 1, so a transient connectivity loss drops the affected samples; the high sampling rate means a subsequent reading follows within seconds, automatic reconnection restores the stream, and the server independently detects and reports device silence exceeding thirty seconds. The blocking reconnection routine noted in Section 4.2 means that an extended outage can also delay interface servicing, which a task-based firmware structure would avoid.

Finally, and most importantly for the interpretation of this chapter, the deployment reported here was conducted in an ordinary indoor environment rather than in a classified cleanroom, and no comparison against a reference instrument was performed. The system has therefore been shown to function end to end and to produce coherent multi-parameter records; it has not been shown to meet any particular measurement accuracy, nor to satisfy the procedural requirements of ISO 14644 (ISO, 2015a)[13]; (ISO, 2015b)[14]. A formal metrological validation aligned with those procedures would be required before the system could be relied upon for compliance decisions, and this is set out as future work in the next chapter.

Chapter 6

Conclusion And Future Work

6.1 Conclusion

This thesis presented the design and development of a real-time, IoT-based system for monitoring essential air-quality parameters in controlled industrial environments. By integrating a Sensirion SEN55 multi-parameter sensor with an ESP32-S3 microcontroller, the system measures four particulate size fractions together with VOC and NOx indices, temperature and relative humidity, samples them every five seconds at the edge, and distributes them over a lightweight MQTT publish–subscribe fabric to both a local 7-inch LVGL touchscreen human-machine interface and a browser-based remote dashboard.

The four-layer architecture cleanly separates sensing, edge processing, communication and visualization. Its publish–subscribe core allows a deployment to grow from a single node to a fully instrumented facility through a single wildcard subscription, requiring only that each new node be registered rather than any change to the back end. A Node.js server resolves incoming telemetry to registered devices, evaluates each reading against configurable control and specification limits, maintains an in-memory cache so that newly connected dashboards populate immediately, persists an hourly snapshot to a TimescaleDB time-series database, and detects device silence and out-of-range conditions in order to raise notifications.

Four nodes were deployed concurrently across a single building and aggregated through a single wildcard subscription, with per-device limits, status reporting and offline detection maintained independently and device positions rendered on a floor plan. A continuous seventy-two-hour record from one of these nodes demonstrated stable operation of the complete chain across three consecutive diurnal cycles, capturing particulate concentrations from a baseline near 11 μg/m³ to a peak of 82 μg/m³ (PM10) alongside the corresponding VOC, temperature and relative-humidity records, with data displayed live on both the local interface and the remote dashboard and retained as an hourly history. The recorded data proved internally consistent, with the particulate fractions preserving their expected ordering throughout and the thermo-hygrometric channels varying smoothly. The quantitative values reported in this thesis were obtained in an ordinary indoor environment rather than in a classified cleanroom, and the sensor was not compared against a calibrated reference instrument; the concentrations reported here are therefore indicative of relative variation and are not metrologically traceable.

Assessed against the objectives set out in Section 1.3, the study met objectives one through six: the governing standards were reviewed, appropriate hardware was selected and integrated, firmware was developed for acquisition and transmission, both local and remote presentation were implemented, and persistence with historical query and export was delivered. Objective seven was met: the prototype was evaluated through continuous operation, and its behaviour, scalability and cost were characterized, together with quantitative measurement of end-to-end latency and delivery ratio as reported in Section 5.4. Comparison against a calibrated reference instrument was not completed within the scope of the study, so the accuracy of the reported concentrations remains unestablished.

The work therefore demonstrates that commodity sensing and edge hardware can deliver responsive, spatially distributed air-quality monitoring appropriate for controlled industrial settings, at a small fraction of the cost of traditional facility-monitoring instrumentation. It also delineates clearly what such a platform can and cannot yet claim. The quantitative values presented in Chapter 5 were recorded in an ordinary indoor environment rather than a classified cleanroom, and without comparison against a calibrated reference instrument; they are therefore indicative of relative variation and trend, and are not metrologically traceable measurements. The platform provides continuous relative visibility suitable for trend monitoring and early warning, but not certified characterisation of a controlled environment, and the steps required to close that gap are identified below.

6.2 Future Work

Five directions follow directly from the limitations identified in Chapter 5.

6.2.1 Quantitative performance characterization

The quantitative performance properties defined in Section 5.4 have now been measured, and the remaining work concerns the breadth of that measurement rather than its absence. The reported figures were obtained over a continuous window of a little over three hours on three nodes under normal network conditions; an earlier measurement over a more congested path produced a comparable median with a substantially heavier tail. Extending the observation to several days, and across a range of network conditions and node counts, would establish whether the reported percentiles hold under sustained operation and would allow a worst-case bound to be stated rather than a typical one. A publish-side counter in the node firmware would additionally allow delivery ratio to be derived from the node's own count of successful publications rather than inferred from the expected cadence, which would separate messages lost in transit from readings the node itself failed to acquire.

6.2.2 Calibration against reference instrumentation

Field calibration of the particulate and VOC outputs against a calibrated reference instrument is required to establish absolute accuracy, to derive correction factors, and to compensate for the humidity dependence characteristic of optical particle sensing (Jayaratne et al., 2018)[16]; (Kelly et al., 2017)[17]. This should be conducted over a range of concentrations and humidity conditions representative of the intended deployment. Until it is performed, reported concentrations remain indicative. In addition, the NOx channel should be re-evaluated after allowing the sensor the extended conditioning period its algorithm requires, so that a meaningful NOx record can be obtained. The procedure planned for this purpose is as follows. A set of independent measurement instruments covering the particulate, thermal and hygrometric channels will be acquired and submitted to an accredited third-party laboratory for calibration, so that each carries a certificate traceable to national standards. Those instruments will then serve as the reference against which the SEN55 nodes are characterised. Reference and node will be co-located so that both sample the same air, and readings will be logged simultaneously over a period long enough to span a useful range of concentrations. Because the comparison is relative rather than single-point, a naturally varying environment is suitable and indeed preferable to a nominally stable one, since it exercises the sensor across its range; a controlled chamber is therefore not a prerequisite. Linear regression of node output against reference will yield a slope, an intercept and a coefficient of determination, from which a correction can be derived and entered through the offset mechanism already implemented in the firmware. Repeating the exercise across a range of relative humidity would additionally quantify the humidity dependence noted above.

6.2.3 Validation in a classified environment

The deployment reported in this thesis was conducted in an ordinary indoor environment. Validation within an actual classified cleanroom, alongside the facility's existing monitoring regime and aligned with ISO 14644 monitoring procedures, is necessary before the system could support any compliance role (ISO, 2015b)[14]; (European Commission, 2022)[11]. This would also allow the alert thresholds, which are presently illustrative, to be aligned with the target ISO class and process requirements.

6.2.4 Firmware robustness and maintainability

Restructuring the firmware from the present cooperative super-loop into concurrently scheduled tasks with a mutex-protected shared state would isolate user-interface rendering and sensor acquisition from network stalls, removing the blocking-reconnection behaviour noted in Section 4.2. The addition of a hardware watchdog would provide automatic recovery from unanticipated faults during long deployments, and a signed over-the-air update mechanism would allow deployed nodes to be maintained without physical access, which is a practical necessity where nodes are installed in restricted-access production areas.

6.2.5 Larger deployments, analytics and system integration

A four-node deployment was operated in the course of this study, as recorded in Section 5.1, which confirms the concurrent aggregation and per-device management properties of the platform at that scale. Extending to a substantially larger deployment would allow the load characteristics argued analytically in Section 5.5 to be verified under realistic conditions, and would permit systematic spatial comparison between locations and the identification of localized contamination sources.

On that foundation, edge and server analytics could be introduced for automated excursion detection and predictive filter-maintenance alerting, building on the alert and limit-evaluation logic already implemented. Integration with plant historian and manufacturing-execution systems would allow air-quality data to be correlated directly with process yield, which is ultimately the outcome that justifies continuous environmental surveillance in a manufacturing setting. Deployment beyond an isolated laboratory network would also require the security measures set out in Section 4.7 to be implemented.

References

[1] S. Abraham and X. Li, "A cost-effective wireless sensor network system for indoor air quality monitoring applications," Procedia Computer Science, vol. 34, pp. 165–171, 2014.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," Computer Networks, vol. 54, no. 15, pp. 2787–2805, 2010.

[4] M. Badura, P. Batog, A. Drzeniecka-Osiadacz, and P. Modzel, "Evaluation of low-cost sensors for ambient PM2.5 monitoring," Journal of Sensors, vol. 2018, art. 5096540, 2018.

[5] R. Balaji, "Clean room monitoring system for pharmaceutical companies," M.S. thesis, Dept. of Informatics and Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 2013.

[6] C. Chen and B. Zhao, "Review of relationship between indoor and outdoor particles: I/O ratio, infiltration factor and penetration factor," Atmospheric Environment, vol. 45, no. 2, pp. 275–288, 2011.

[7] L. R. Crilley, M. Shaw, R. Pound, L. J. Kramer, R. Price, S. Young, A. C. Lewis, and F. D. Pope, "Evaluation of a low-cost optical particle counter (Alphasense OPC-N2) for ambient air monitoring," Atmospheric Measurement Techniques, vol. 11, no. 2, pp. 709–720, 2018.

[8] EMQX, "EMQX Documentation — Open-source MQTT Broker," 2024. [Online]. Available: https://www.emqx.io/docs

[9] Espressif Systems, "ESP32-S3 Series Datasheet: 2.4 GHz Wi-Fi + Bluetooth LE SoC," Shanghai, China, 2023.

[10] Espressif Systems, ESP32-S3 Technical Reference Manual, Version 1.0, Shanghai, China, 2023.

[11] European Commission, EudraLex Volume 4 — Good Manufacturing Practice, Annex 1: Manufacture of Sterile Medicinal Products, Brussels, Belgium, 2022.

[12] I. Fette and A. Melnikov, "The WebSocket Protocol," RFC 6455, Internet Engineering Task Force, Dec. 2011.

[13] International Organization for Standardization, ISO 14644-1:2015 — Cleanrooms and associated controlled environments — Part 1: Classification of air cleanliness by particle concentration, Geneva, Switzerland, 2015.

[14] International Organization for Standardization, ISO 14644-2:2015 — Cleanrooms and associated controlled environments — Part 2: Monitoring to provide evidence of

cleanroom performance related to air cleanliness by particle concentration, Geneva, Switzerland, 2015.

[15] International Organization for Standardization, ISO 14644-4:2022 — Cleanrooms and associated controlled environments — Part 4: Design, construction and start-up, Geneva, Switzerland, 2022.

[16] R. Jayaratne, X. Liu, P. Thai, M. Dunbabin, and L. Morawska, "The influence of humidity on the performance of a low-cost air particle mass sensor and the effect of atmospheric fog," Atmospheric Measurement Techniques, vol. 11, no. 8, pp. 4883–4890, 2018.

[17] K. E. Kelly, J. Whitaker, A. Petty, C. Widmer, A. Dybwad, D. Sleeth, R. Martin, and A. Butterfield, "Ambient and laboratory evaluation of a low-cost particulate matter sensor," Environmental Pollution, vol. 221, pp. 491–500, 2017.

[18] S. Kumar and A. Jasuja, "Air quality monitoring system based on IoT using Raspberry Pi," in Proc. IEEE International Conference on Computing, Communication and Automation (ICCCA), 2017, pp. 1341–1346.

[19] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," Business & Information Systems Engineering, vol. 6, no. 4, pp. 239–242, 2014.

[20] A. Lezcano Sirre, "IoT environmental monitoring system using Arduino and NodeMCU ESP8266," B.Eng. thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, Jan. 2024.

[21] Lighthouse Worldwide Solutions, "Handheld Particle Counters — HHPC 6+," Product Brochure, 2023.

[22] LVGL, "LVGL Documentation — Graphics Library for Embedded Systems," 2023. [Online]. Available: https://docs.lvgl.io

[23] G. Marques, C. Roque Ferreira, and R. Pitarma, "A system based on the Internet of Things for real-time particle monitoring in buildings," International Journal of Environmental Research and Public Health, vol. 15, no. 4, art. 821, 2018.

[24] M. F. Mohd Kamil, N. Sulaiman, and Z. Zakaria, "Internet of Things monitoring system of a modeled cleanroom," ELEKTRIKA — Journal of Electrical Engineering, vol. 20, no. 2, pp. 85–90, 2021.

[25] L. Morawska, P. K. Thai, X. Liu, A. Asumadu-Sakyi, G. Ayoko, A. Bartonova, et al., "Applications of low-cost sensing technologies for air quality monitoring and exposure assessment: How far have they gone?," Environment International, vol. 116, pp. 286–299, 2018.

[26] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," in Proc. IEEE International Systems Engineering Symposium (ISSE), Vienna, Austria, 2017, pp. 1–7.

[27] OASIS, "MQTT Version 5.0," A. Banks, E. Briggs, K. Borgendale, and R. Gupta, Eds., OASIS Standard, Mar. 2019.

[28] N. O'Leary, "PubSubClient — An Arduino client library for MQTT," 2020. [Online]. Available: https://pubsubclient.knolleary.net

[29] OpenJS Foundation, "Node.js Documentation," 2024. [Online]. Available: https://nodejs.org/docs

[30] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," Journal of Management Information Systems, vol. 24, no. 3, pp. 45–77, 2007.

[31] PostgreSQL Global Development Group, "PostgreSQL Documentation," 2024. [Online]. Available: https://www.postgresql.org/docs

[32] A. C. Rai, P. Kumar, F. Pilla, A. N. Skouloudis, S. Di Sabatino, C. Ratti, A. Yasar, and D. Rickerby, "End-user perspective of low-cost sensors for outdoor air pollution monitoring," Science of the Total Environment, vol. 607–608, pp. 691–705, 2017.

[33] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed Internet of Things," Computer Networks, vol. 57, no. 10, pp. 2266–2279, 2013.

[34] J. Saini, M. Dutta, and G. Marques, "Indoor air quality monitoring systems based on Internet of Things: A systematic review," International Journal of Environmental Research and Public Health, vol. 17, no. 14, art. 4942, 2020.

[35] J. H. Seinfeld and S. N. Pandis, Atmospheric Chemistry and Physics: From Air Pollution to Climate Change, 3rd ed. Hoboken, NJ, USA: Wiley, 2016.

[36] Sensirion AG, "Datasheet SEN5x — Particulate Matter, VOC, NOx, Humidity and Temperature Sensor Node," Version 2, Stäfa, Switzerland, Mar. 2022.

[37] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, 2016.

[38] D. Thangavel, X. Ma, A. Valera, H.-X. Tan, and C. K.-Y. Tan, "Performance evaluation of MQTT and CoAP via a common middleware," in Proc. IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Singapore, 2014, pp. 1–6.

[39] N. Themeli, "Developing and implementing into practice an environmental monitoring system," M.S. thesis, Epoka University, Tirana, Albania, 2023.

[40] Timescale, "TimescaleDB Documentation — Time-series data in PostgreSQL," 2024. [Online]. Available: https://docs.timescale.com

[41] TSI Incorporated, "AeroTrak Portable Particle Counter 9306 Datasheet," 2022. [Online]. Available: https://tsi.com

[42] U.S. Environmental Protection Agency, "Technical assistance document for the reporting of daily air quality — the Air Quality Index (AQI)," Rep. EPA-454/B-18-007, Research Triangle Park, NC, USA, 2018.

[43] W. Whyte, Cleanroom Technology: Fundamentals of Design, Testing and Operation, 2nd ed. Chichester, UK: Wiley, 2010.

[44] World Health Organization, WHO Global Air Quality Guidelines: Particulate Matter (PM2.5 and PM10), Ozone, Nitrogen Dioxide, Sulfur Dioxide and Carbon Monoxide. Geneva, Switzerland: WHO, 2021.

[45] X. Zhang, "Enhancing cleanroom operations: From preventive equipment maintenance to lethal chemical asset tracking," M.S. thesis, University of Illinois at Urbana-Champaign, IL, USA, 2024.

Appendices

Appendix A Telemetry Payload Specification

Each telemetry message is a comma-separated string, optionally enclosed in double quotation marks, with the following fixed field order.

Field	Content	Unit	Example
1	Node hardware address	—	34:85:18:97:15:C8
2	Temperature	°C	25.40
3	Relative humidity	%RH	60.10
4	PM1.0 mass concentration	µg/m³	1.20
5	PM2.5 mass concentration	µg/m³	3.40
6	PM4 mass concentration	µg/m³	4.50
7	PM10 mass concentration	µg/m³	5.60
8	VOC index	points	120.00
9	NOx index	points	15.00

Appendix B Sensor Acquisition and Publication Routine

The following extract shows the acquisition, conditioning and publication sequence executed on the five-second cadence.

```
error = sen5x.readMeasuredValues(
    mass_concentration_pm1p0, mass_concentration_pm2p5,
    mass_concentration_pm4p0, mass_concentration_pm10p0,
    ambient_humidity, ambient_temperature, voc_index, nox_index);

final_humidity    = ambient_humidity    + humi_offset;
final_temperature = ambient_temperature + temp_offset;
final_pm1p0  = mass_concentration_pm1p0;
final_pm2p5  = mass_concentration_pm2p5;
final_pm4p0  = mass_concentration_pm4p0;
final_pm10p0 = mass_concentration_pm10p0;
final_voc    = voc_index;
final_nox    = nox_index;

uint64_t captureTimestampMs = getEpochMillis();

String payload = "\""+ WiFi.macAddress() +
              "," + String(final_temperature) +
              "," + String(final_humidity) +
              "," + String(final_pm1p0) +
              "," + String(final_pm2p5) +
              "," + String(final_pm4p0) +
              "," + String(final_pm10p0) +
              "," + String(final_voc) +
              "," + String(final_nox) +
              "," + String((unsigned long long)captureTimestampMs) + "\"";
```

```
unsigned long publishStart = millis();
bool ok = client.publish(topic, payload.c_str(), false, 1);
unsigned long publishDuration = millis() - publishStart;
if (ok) {
   Serial.println("Published (QoS1 accepted by client)!");
} else {
   Serial.println("Publish failed!");
}
```

Appendix C Server Message Resolution Logic

The server distinguishes payload formats, resolves the node address against the device registry, and discards messages from unregistered devices, as summarized below.

```
if (raw.startsWith("{")) {
    payload   = JSON.parse(raw);
    raw_mac   = payload.mac_address;
    sensors   = payload.sensors;
} else {
    // CSV: mac,temp,hum,pm1,pm2.5,pm4,pm10,voc,nox
    const parts = raw.split(",");
    raw_mac = parts[0].trim().replace(/"/g, "");
    sensors = {
        temperature: p(parts[1]), humidity: p(parts[2]),
        pm1:  p(parts[3]), pm25: p(parts[4]),
        pm4:  p(parts[5]), pm10: p(parts[6]),
        voc:  p(parts[7]), nox:  p(parts[8])
    };
}

let dev = macToDevice[raw_mac.toUpperCase()];
if (!dev) return;      // unregistered device: message discarded
```

Appendix D Recorded Data Set

The complete seventy-two-hour hourly record from which Figures 5.1 to 5.5 and Table 5.1 are derived is reproduced below. Temperature is given in degrees Celsius, relative humidity in per cent, particulate fractions in μg/m³, and VOC and NOx as index points.

Table D.1
THE SEVENTY-TWO-HOUR HOURLY RECORD

Timestamp	T (°C)	RH (%)	PM1	PM2.5	PM4	PM10	VOC	NOx
23/07 17:00	25.47	71.00	78.20	82.00	82.00	82.00	51	1
23/07 18:00	25.36	71.86	72.00	75.70	75.90	76.00	45	1
23/07 19:00	25.30	72.81	65.70	69.00	69.20	69.30	36	1
23/07 20:00	25.25	73.89	54.10	56.80	56.80	56.80	30	1
23/07 21:00	25.27	74.09	44.30	46.40	46.40	46.40	28	1
23/07 22:00	25.16	74.77	41.30	43.30	43.30	43.30	31	1
23/07 23:00	25.19	75.45	36.60	38.40	38.40	38.40	36	1

Timestamp	T (°C)	RH (%)	PM1	PM2.5	PM4	PM10	VOC	NOx
24/07 00:00	25.60	74.75	32.20	33.70	33.70	33.70	64	2
24/07 01:00	25.90	71.62	30.90	32.40	32.40	32.40	210	1
24/07 02:00	26.26	67.97	27.50	28.80	28.80	28.80	210	1
24/07 03:00	26.65	65.22	26.20	27.50	27.50	27.50	201	1
24/07 04:00	27.01	61.06	22.80	24.00	24.00	24.00	101	1
24/07 05:00	26.99	60.73	22.50	23.60	23.60	23.60	102	1
24/07 06:00	27.19	59.07	17.60	18.40	18.40	18.40	138	1
24/07 07:00	27.49	58.06	16.50	17.30	17.30	17.30	102	1
24/07 08:00	27.60	56.02	14.10	14.80	14.80	14.80	94	1
24/07 09:00	27.51	56.54	14.00	14.70	14.70	14.70	111	1
24/07 10:00	27.27	59.13	10.90	11.40	11.40	11.40	117	1
24/07 11:00	27.17	60.53	11.50	12.10	12.10	12.10	68	1
24/07 12:00	26.96	62.02	12.80	13.40	13.40	13.40	43	1
24/07 13:00	26.78	63.98	20.70	21.80	21.80	21.80	28	2
24/07 14:00	26.57	66.61	25.60	26.90	26.90	26.90	15	2
24/07 15:00	26.39	67.74	34.30	35.90	35.90	35.90	20	2
24/07 16:00	26.23	68.50	36.10	37.80	37.80	37.80	22	1
24/07 18:00	26.00	70.61	36.00	37.80	37.80	37.80	23	1
24/07 19:00	25.91	71.56	31.70	33.30	33.30	33.30	25	1
24/07 20:00	25.85	72.35	29.20	30.60	30.60	30.60	28	1
24/07 21:00	25.78	72.88	26.50	27.80	27.80	27.80	32	1
24/07 22:00	25.71	73.48	25.00	26.20	26.20	26.20	35	1
24/07 23:00	25.71	73.59	22.70	23.80	23.80	23.80	39	1
25/07 00:00	26.23	73.66	18.50	19.40	19.40	19.40	75	1
25/07 01:00	26.72	71.71	16.60	17.40	17.40	17.40	106	1
25/07 02:00	26.89	70.21	16.20	17.00	17.00	17.00	126	1
25/07 03:00	27.12	68.94	13.60	14.30	14.30	14.30	125	1
25/07 04:00	27.32	68.12	14.20	14.90	14.90	14.90	129	1
25/07 05:00	27.66	67.52	13.50	14.10	14.10	14.10	119	1
25/07 06:00	28.01	66.54	13.50	14.20	14.20	14.20	115	1
25/07 07:00	27.92	65.61	15.00	15.70	15.70	15.70	133	1
25/07 08:00	27.92	65.34	14.40	15.10	15.10	15.10	188	1
25/07 09:00	27.77	64.73	14.50	15.20	15.20	15.20	140	1
25/07 10:00	27.50	65.11	11.80	12.40	12.40	12.40	85	1
25/07 11:00	27.19	65.90	12.70	13.30	13.30	13.30	125	2
25/07 12:00	27.06	66.06	12.40	13.00	13.00	13.00	123	1
25/07 13:00	27.02	66.70	14.50	15.20	15.20	15.20	108	1

Timestamp	T (°C)	RH (%)	PM1	PM2.5	PM4	PM10	VOC	NOx
25/07 14:00	26.96	67.10	16.40	17.10	17.10	17.10	114	1
25/07 15:00	26.84	68.03	21.30	22.30	22.30	22.30	91	1
25/07 16:00	26.68	69.27	26.70	28.30	28.70	28.80	65	1
25/07 17:00	26.58	69.94	30.50	32.00	32.00	32.00	58	1
25/07 18:00	26.52	70.66	29.00	30.50	30.50	30.50	48	1
25/07 19:00	26.41	70.85	27.20	28.50	28.50	28.50	40	1
25/07 20:00	26.30	71.43	25.80	27.10	27.10	27.10	36	1
25/07 21:00	26.25	71.76	24.50	25.70	25.70	25.70	35	1
25/07 22:00	26.13	72.34	23.80	24.90	24.90	24.90	36	1
25/07 23:00	26.05	72.57	19.10	20.00	20.00	20.00	34	1
26/07 00:00	26.05	72.98	21.10	22.20	22.20	22.20	44	1
26/07 01:00	26.46	72.64	19.70	20.60	20.60	20.60	52	1
26/07 02:00	26.73	73.06	20.00	21.00	21.00	21.00	51	1
26/07 03:00	27.00	72.36	19.30	20.20	20.20	20.20	59	1
26/07 04:00	27.21	72.02	18.00	18.90	18.90	18.90	75	1
26/07 05:00	27.33	71.52	17.90	18.80	18.80	18.80	85	1
26/07 06:00	27.44	70.99	15.50	16.30	16.30	16.30	118	1
26/07 07:00	27.75	70.41	17.90	18.70	18.70	18.70	111	1
26/07 08:00	28.16	69.36	16.50	17.30	17.30	17.30	139	1
26/07 09:00	28.31	68.41	17.80	18.60	18.60	18.60	158	1
26/07 10:00	27.97	68.36	15.50	16.20	16.20	16.20	123	1
26/07 11:00	27.49	68.82	18.50	19.40	19.40	19.40	86	1
26/07 12:00	27.10	69.23	18.80	19.70	19.70	19.70	88	1
26/07 13:00	26.91	69.58	18.40	19.30	19.30	19.30	96	1
26/07 14:00	26.75	69.87	19.00	19.90	19.90	19.90	80	1
26/07 15:00	26.64	69.94	17.30	18.20	18.20	18.20	63	1
26/07 16:00	26.51	70.16	17.00	17.90	17.90	17.90	57	1
26/07 17:00	26.37	70.60	16.10	16.90	16.90	16.90	55	1
26/07 18:00	26.20	71.04	14.90	15.70	15.70	15.70	60	1
26/07 19:00	26.07	71.47	14.20	14.90	14.90	14.90	54	1
26/07 20:00	25.95	71.92	13.40	14.10	14.10	14.10	54	1
26/07 21:00	25.86	72.31	14.00	14.70	14.70	14.70	53	1
26/07 22:00	25.81	72.55	14.60	15.30	15.30	15.30	48	1
26/07 23:00	25.77	72.80	14.70	15.40	15.40	15.40	49	1
27/07 00:00	26.17	71.21	14.20	14.90	14.90	14.90	78	1
27/07 01:00	26.61	66.84	15.40	16.10	16.10	16.10	200	1
27/07 02:00	26.86	64.69	16.40	17.10	17.10	17.10	234	1

Appendix E Latency and Delivery Measurement Record

The end-to-end latency and delivery figures reported in Section 5.4 were obtained from a single continuous observation window running from 06:43 to 10:00 UTC on 3 August 2026, during which three nodes published concurrently at the configured five-second cadence. Each record in the underlying log holds the instant of capture stamped by the node, the instant of receipt recorded by the independent subscriber, the originating MAC address, and the difference between the two timestamps. Seven thousand and fifty-four records were captured in total. Table E.1 summarizes the record by node; the per-node clock offsets stated in the fourth column are those described and justified in Section 5.4, and the median, percentile and maximum figures are quoted after that correction has been applied.

Table E.1
SUMMARY OF THE LATENCY AND DELIVERY MEASUREMENT RECORD BY NODE

Node	MAC address	Samples	Offset (ms)	Median (ms)	p90 (ms)	p95 (ms)	Max (ms)	Delivery
Showroom	F0:9E:9E:08:17:68	2350	+137	139	311	411	5089	99.37%
Factory Automation	F0:9E:9E:07:3F:A8	2350	+85	136	402	609	5217	99.37%
Workshop	64:E8:33:48:6B:38	2354	+135	148	369	461	5145	99.53%
All nodes	—	7054	—	141	357	490	5217	99.42%

So that the statistics summarized in Table E.1 can be traced to primary data, Table E.2 reproduces ten consecutive records for each node exactly as they were written to the log file, taken from the middle of the observation window. The two epoch columns are the unmodified values carried in the message and recorded by the subscriber, and the final column is their difference; no offset correction has been applied to these figures. Two features of the raw data are visible in the extract and are worth noting. First, successive records for a given node are separated by approximately five seconds, which is the configured publication cadence. Second, a number of the raw differences are negative, which is the clock-offset effect described in Section 5.4: a negative value indicates that the node's clock was running ahead of the recording host at that instant, and it is these observations that expose the offset subsequently removed. Occasional larger positive values in the extract, such as those near 08:21:49 which appear on all three nodes at once, are the transient network delays that give the distribution its upper tail.

Table E.2
RAW LOG EXTRACT: TEN CONSECUTIVE RECORDS PER NODE

Node	Receive time (UTC)	capture_epoch_ms	receive_epoch_ms	latency_ms
Showroom	08:21:23.850	1785745283904	1785745283850	-54
	08:21:28.873	1785745288927	1785745288873	-54
	08:21:33.888	1785745293963	1785745293888	-75
	08:21:38.916	1785745298967	1785745298916	-51
	08:21:43.929	1785745303986	1785745303929	-57
	08:21:49.229	1785745309010	1785745309229	219
	08:21:53.973	1785745314020	1785745313973	-47
	08:21:58.981	1785745319029	1785745318981	-48
	08:22:03.996	1785745324030	1785745323996	-34
	08:22:09.084	1785745329053	1785745329084	31

Node	Receive time (UTC)	capture_epoch_ms	receive_epoch_ms	latency_ms
Factory Automation	08:21:23.723	1785745283705	1785745283723	18
	08:21:28.734	1785745288713	1785745288734	21
	08:21:33.755	1785745293731	1785745293755	24
	08:21:38.778	1785745298764	1785745298778	14
	08:21:43.799	1785745303777	1785745303799	22
	08:21:49.119	1785745308817	1785745309119	302
	08:21:53.823	1785745313840	1785745313823	-17
	08:21:58.847	1785745318858	1785745318847	-11
	08:22:03.863	1785745323868	1785745323863	-5
	08:22:09.207	1785745328871	1785745329207	336
Workshop	08:21:34.147	1785745294115	1785745294147	32
	08:21:39.080	1785745299115	1785745299080	-35
	08:21:44.101	1785745304122	1785745304101	-21
	08:21:49.360	1785745309155	1785745309360	205
	08:21:54.184	1785745314186	1785745314184	-2
	08:21:59.149	1785745319203	1785745319149	-54
	08:22:04.271	1785745324233	1785745324271	38
	08:22:09.323	1785745329271	1785745329323	52
	08:22:14.318	1785745334305	1785745334318	13
	08:22:19.362	1785745339332	1785745339362	30

Appendix F Latency Measurement Script

The latency and delivery figures reported in Section 5.4 were obtained with an independent subscriber written in Python, connected to the broker under its own client identifier so that it observed the same published stream as the application server without altering it. The essential logic is reproduced below. For every message the subscriber reads the capture instant carried in the tenth field of the payload, records the instant of arrival, and writes both together with their difference to a log file; the resulting records are those summarized in Appendix E. A sanity window is applied so that implausible values arising from a node whose clock has not yet synchronized are discarded, while the small negative differences produced by residual clock offset are retained rather than silently dropped, since those observations are what expose the offset corrected for in Section 5.4.

```
# Independent subscriber used to measure NFR-1 and NFR-2.
# Requires: pip install paho-mqtt

BROKER_HOST = "31.97.110.216"      # EMQX broker
TOPIC       = "devices/+/data"     # wildcard: all nodes
QOS         = 1
SANITY_MIN_MS = -1000              # retain small negative offsets
SANITY_MAX_MS = 5 * 60 * 1000      # reject implausible values

def on_connect(client, userdata, flags, rc):
    client.subscribe(TOPIC, qos=QOS)

def on_message(client, userdata, msg):
    raw    = msg.payload.decode().strip().strip('"')
    parts  = raw.split(",")
    if len(parts) < 10:            # no timestamp field present
        return
    mac            = parts[0].strip()
    capture_ts_ms  = int(parts[9].strip())   # stamped by the node
    receive_ts_ms  = int(time.time() * 1000) # stamped on arrival
    latency_ms     = receive_ts_ms - capture_ts_ms
```

```
    if SANITY_MIN_MS <= latency_ms <= SANITY_MAX_MS:
        csv_writer.writerow([receive_ts_ms, receive_utc,
                             mac, capture_ts_ms, latency_ms])
        csv_fh.flush()

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.connect(BROKER_HOST, BROKER_PORT, keepalive=60)
client.loop_forever()
```