applied sciences

MDPI

Article

Data Acquisition, Processing, and Aggregation in a Low-Cost IoT System for Indoor Environmental Quality Monitoring

Alberto Barbaro [1], Pietro Chiavassa [2], Virginia Isabella Fissore [3], Antonio Servetti [2], Erica Raviola [1], Gustavo Ramírez-Espinosa [2,4], Edoardo Giusto [5], Bartolomeo Montrucchio [2], Arianna Astolfi [3] and Franco Fiori [1,*]

[1] Department of Electronics and Telecommunications, Politecnico di Torino, 10126 Turin, Italy; alberto.barbaro@polito.it (A.B.); erica.raviola@polito.it (E.R.)
[2] Department of Control and Computer Engineering, Politecnico di Torino, 10126 Turin, Italy; pietro.chiavassa@polito.it (P.C.); antonio.servetti@polito.it (A.S.); ramirez.g@javeriana.edu.co (G.R.-E.); bartolomeo.montrucchio@polito.it (B.M.)
[3] Department of Energy, Politecnico di Torino, 10126 Turin, Italy; virginia.fissore@polito.it (V.I.F.); arianna.astolfi@polito.it (A.A.)
[4] Department of Electronics, Pontificia Universidad Javeriana, Bogotá 110231, Colombia
[5] Departement of Electrical Engineering and Information Technology, University of Naples, Federico II, 80125 Naples, Italy; edoardo.giusto@unina.it
* Correspondence: franco.fiori@polito.it; Tel.: +39-0110904141

Abstract: The rapid spread of Internet of Things technologies has enabled a continuous monitoring of indoor environmental quality in office environments by integrating monitoring devices equipped with low-cost sensors and cloud platforms for data storage and visualization. Critical aspects in the development of such monitoring systems are effective data acquisition, processing, and visualization strategies, which significantly influence the performance of the system both at monitoring device and at cloud platform level. This paper proposes novel strategies to address the challenges in the design of a complete monitoring system for indoor environmental quality. By adopting the proposed solution, one can reduce the data rate transfer between the monitoring devices and the server without loss of information, as well as achieve efficient data storage and aggregation on the server side to minimize retrieval times. Finally, enhanced flexibility in the dashboard for data visualization is obtained, thus enabling graph modifications without extensive coding efforts. The functionality of the developed system was assessed, with the collected data in good agreement with those from other instruments used as references.

Keywords: indoor environmental quality; indoor air quality; multi-sensor; Internet of Things; low-cost sensors

Citation: Barbaro, A.; Chiavassa, P.; Fissore, V.I.; Servetti, A.; Raviola, E.; Ramírez-Espinosa, G.; Giusto, E.; Montrucchio, B.; Astolfi, A.; Fiori, F. Data Acquisition, Processing, and Aggregation in a Low-Cost IoT System for Indoor Environmental Quality Monitoring. *Appl. Sci.* **2024**, *14*, 4021. https://doi.org/10.3390/app14104021

Academic Editors: Ana Monteiro, Carla Viegas, Sandra Cabo Verde and Marina Almeida-Silva

Received: 19 March 2024
Revised: 30 April 2024
Accepted: 7 May 2024
Published: 9 May 2024

1. Introduction

With European citizens spending 90% of their time indoors, mostly in work environments, there is a growing concern regarding the impact of indoor environmental quality (IEQ) on occupants' overall comfort and well-being [1,2]. Several studies already highlighted the adverse effects of poor IEQ on occupants' health and productivity in offices. For instance, Azuma et al. [3] conducted a cross-sectional study, monitoring temperature, humidity, and IAQ, and collecting survey responses from workers. The study revealed significant correlations between upper respiratory symptoms and higher indoor temperature and particle concentrations. Similarly, Varjo et al. [4] investigated the impact of intelligible speech, room temperature, and air supply on 65 participants in an open-plan office. Results showed reduced performance, particularly in working memory tasks, increased mental workload, cognitive fatigue, and symptoms in adverse conditions.

Assessing IEQ involves a comprehensive evaluation of thermal, visual, acoustic, and indoor air quality (IAQ) domains [5]. Regarding IAQ, the identification and the quan-

tification of hazardous air pollutants, such as benzene, formaldehyde, carbon monoxide, carbon dioxide, and nitrogen oxides, play a key role in assessing human exposure and risk [6,7]. Several studies reveal that indoor environments can be even twice polluted than outdoor [8], with pollutants deriving both from outdoor sources, e.g., vehicular traffic and industrial activities, as well as from indoor sources, e.g., combustion sources, building materials, furnishings, and products for household cleaning [9,10]. A poor IAQ may lead to the onset of the Sick Building Syndrome (SBS), which the World Health Organization (WHO) defined firstly in 1983 as the medical condition in which people in a building suffer from symptoms of illness or feel unwell for no apparent reason [11]. The symptoms, which may include headache, tiredness, and throat irritation, depend on the time spent inside the building and tend to disappear once people leave.

In addition to IAQ, also inadequate levels of temperature, humidity, ventilation, and illuminance contribute to the raising of SBS symptoms [12]. Furthermore, indoor soundscape has recently garnered significant attention, with efforts aimed at creating spaces that not only meet basic standards but also enhance health and well-being [13]. To sum up, a multi-domain approach is required to assess the impact of IEQ on occupants' health, comfort, and work productivity [14–16].

Thanks to the widespread adoption of Internet of Things (IoT) technology, the assessment of environmental conditions has evolved from the use of expensive data loggers for in-field measurements to the use of low-cost wireless-connected monitoring devices [17,18]. Such a trend is evident across various applications. For instance, a low-cost IoT platform was proposed in [19], which can be applied in several acoustic monitoring scenarios, such as monitoring ultrasonic bat calls and audible wildlife vocalizations. Similarly, in the acoustic domain, ref. [20] discussed the GAIA project for monitoring noise levels inside school buildings, showcasing dedicated applications developed to aid students and teachers in accessing sensor data. The feasibility of using open-source hardware was explored in [21], focusing on energy monitoring in school buildings and demonstrating its scalability, responsiveness, and adaptability to various application requirements. Additionally, citizen engagement concerning outdoor air quality through a low-cost IoT platform was examined in [22], enabling pervasive and distributed monitoring systems.

As far as indoor environmental quality is concerned, monitoring devices are typically equipped with a dozen sensors, including, but not limited to, temperature, relative humidity, illuminance, carbon dioxide, and formaldehyde. Recorded data are then transferred to ad hoc cloud software platforms via wireless networks. In such a way, multiple devices can monitor the four IEQ domains simultaneously and continuously in indoor environments, with recorded data accessible to occupants through dedicated dashboards [23–25].

Among the several monitoring devices proposed thus far, a multi-sensor capable of providing a real-time feedback on measured IEQ and displaying it was proposed in [24]. In this study, data from on-board sensors were collected, and averages of each monitored physical quantity updated every 10 min. In [26], a device combining low-cost IEQ sensors with a survey interface to gather feedback regarding the occupants' perceived indoor environmental comfort (IEC) was presented. While it allowed data transmission through a 3G modem, details about the cloud platform were not provided. The SAMBA device proposed in [23] allowed for a continuous IEQ monitoring in indoors. Each monitoring device acquired sensor data during a 270 s window sampling, followed by a 30 s data transmission window, during which averages of sampled data are sent to a cloud server. Another monitoring device, ENVIRA, was proposed in [27], where relevant IEQ parameters are integrated into an overall index through individual weighting coefficients. Finally, a system based on intelligent edge computing was proposed in [28], in which the developed measuring device collects data from 14 types of environmental sensors, and indoor air quality is predicted.

The development of such monitoring devices deals with the addressing of several challenges simultaneously. One primary concern arises from the use of low-cost sensors, as their readings often deviate from those of reference instruments [29]. It is, therefore,

required to conduct a metrological characterization of the low-cost sensors employed into the developed multi-sensor unit, as discussed in [30]. Additionally, attention should be given to the placement of sensors within the monitoring device to prevent mutual interference. Lastly, there is a need to enhanced data processing capability at the monitoring device level.

In addition to monitoring devices, the design of a complete IEQ monitoring system also includes the development of the cloud platform and of the dashboarding application. The design challenges faced in this regard involve efficient data storage and retrieval, as well as the creation of user-friendly and highly customizable user interfaces, aspects that are not typically exhaustively discussed in the architecture of such platforms [31,32].

Regarding the visualization of sensor data graphs, the approaches used in the literature have been essentially twofold: leveraging dashboarding environments or developing a dedicated application/website. In research environments such as [27,31,32], the preferred solution often leans towards dashboarding environments. These environments allow for customization without the need for code development. Conversely, in commercial-like products such as [33] or widely adopted platforms such as [22], the adoption of dedicated applications offers superior usability, service performance, and scalability. However, the limitation of the former approach lies in the limited possibilities for user interface customization. Conversely, the latter case necessitates reliance on IT professionals for both initial development and subsequent modifications.

In summary, several aspects regarding the design of a complete IEQ monitoring system have not been addressed thoroughly in existing works. More precisely, data processing at the monitoring device level typically involves only averaging raw data from sensors over fixed time intervals, which may prevent the detection of rapid variations in monitored physical parameters. On the server side, the actual database organization is neglected, as well as effective solutions for rapid data retrieving. Additionally, proposed solutions for data visualization lack of flexibility and ease of modification for non-IT professionals.

To tackle the aforementioned issues, this paper proposes novel strategies adopted in the design of the PROactive Monitoring for indoor EnvironmenTal quality and cOmfort (PROMET&O) system. Such a system aims to assess IEQ and how it is perceived by occupants in office environments. Indeed, low-cost monitoring devices, placed on workers' desks, monitor IEQ conditions for comparison with their comfort perception feedback. This allows for end-users' engagement and establishes best practices for IEQ enhancement and energy saving through a more conscious management of building systems.

With respect to previous works, this paper proposes a full-stack design of an IEQ monitoring system, in which several strategies have been implemented to overcome the limitations of existing systems in both monitoring devices and cloud platforms. More precisely, this work makes the following contributions, which allow one to step forward with respect to the state of the art.

At the multi-sensor level, the hardware design prioritizes flexibility, while optimizing sensor placement to minimize cross-sensitivity. Data preprocessing is employed to extract significant statistical parameters, thus reducing the amount of data transferred to the server.

At the cloud back-end server level, a reference database design for IEQ parameters collection is introduced, along with a solution for data aggregation challenges over extended time intervals, which were not discussed previously. In addition, preaggregations are introduced to compute the final aggregation faster, allowing the user to receive up-to-date statistics aligned to the instant of the request.

For data visualization on the front-end server, charts from open-source dashboarding software are integrated into a custom web applicatio that allows for extensive customization during development to ensure optimal usability. Moreover, the system's flexibility facilitates easy graph modification to meet evolving requirements without extensive coding.

By incorporating these strategies, the PROMET&O system can evaluate IEQ and occupants' perception in office environments, promoting users' engagement and defining

IEQ improvement and energy-saving practices. Its hardware and software flexibility enable future adaptation to similar applications.

The paper is organized as follows. The overall architecture of the PROMET&O system is discussed in Section 2, with particular focus on the data sampling and aggregation processes. The developed multi-sensor device and the related firmware are discussed in details in Section 3, in which a comprehensive cost analysis is reported as well. Then, the main blocks the cloud server is comprised of are analyzed in Section 4. Preliminary experimental results are reported in Section 5, including a comparison of the proposed system against existing ones. Concluding remarks are drawn in Section 6.

2. System Architecture

In this section, the architecture of the overall PROMET&O system is discussed, with particular focus on the data acquisition process. Part of the code has been released as open source and it is published on GitHub [34].

2.1. The PROMET&O Architecture

The overall system architecture consists of three primary components, as shown in Figure 1. Firstly, the IoT layer encompasses the monitoring devices, networking devices, and connectivity equipment. It is comprised of ad hoc designed multi-sensors devices, which are responsible for measuring physical quantities associated with thermal, lighting, acoustic, and air quality. In such a way, a continuous and real-time indoor monitoring of several physical parameters can be achieved. The acquired data are preprocessed by the multi-sensors devices themselves prior to be sent to the cloud server for further processing and visualization. A modem enables the creation of a WiFi network, to which the monitoring devices are connected, and provides access to the cloud platform via a 3G/4G connection. The data transfer between each multi-sensor unit and the cloud server is accomplished through the message queuing telemetry transport (MQTT) protocol, which is a publish–subscribe protocol popular among IoT applications. Such a protocol allows devices, such as the PROMET&O multi-sensors, to specify topics of interest, serving as communication channels for sharing (publishing) measurements of the various environmental parameters. Devices can also subscribe to these topics to receive the messages containing the published data.

The data processing layer provides the storage of raw sensor data, performs analytics derived from this data, and exposes an Application Programming Interface (API) for querying purposes. All MQTT messages are conveyed to the broker, a software application that is in charge of managing the MQTT communication. It receives messages published on the different topics and redirects them to all the subscribers. A Python-based software application acts as an MQTT client, subscribing to the topics of all the PROMET&O multi-sensors to receive their measurements. All received data are stored in a database, where a Python-based scheduler is in charge of periodically running software jobs that perform data aggregation and enrichment. A web API allows external access to stored information.

Lastly, the data visualization layer assumes responsibility for data visualization and dashboard creation for end-users, in conjunction with security layers that regulate access exclusively to authorized entities. As far as the the data visualization layer is concerned, a key aspect is the integration of a dashboarding web application, namely Grafana (v10.4.2) [35], and an Identity and Access Management software (IAM), exemplified by Keycloak (v24.0) [36], into a primary web application developed in React (v18.2) [37].

While the predominant approach among low-cost IoT monitoring systems [31,32] involves providing end-users with the same data visualization interface accessible within the backend dashboarding application, it is worth noting that such interfaces are primarily tailored for professionals. In this work, on the contrary, the web application enables the embedding of charts within the organization's website, ensuring consistency in User Interface (UI) style and navigation patterns. In addition, the IAM software offers a fine-

grained authentication system, granting selective access to both internal staff and third-party users.

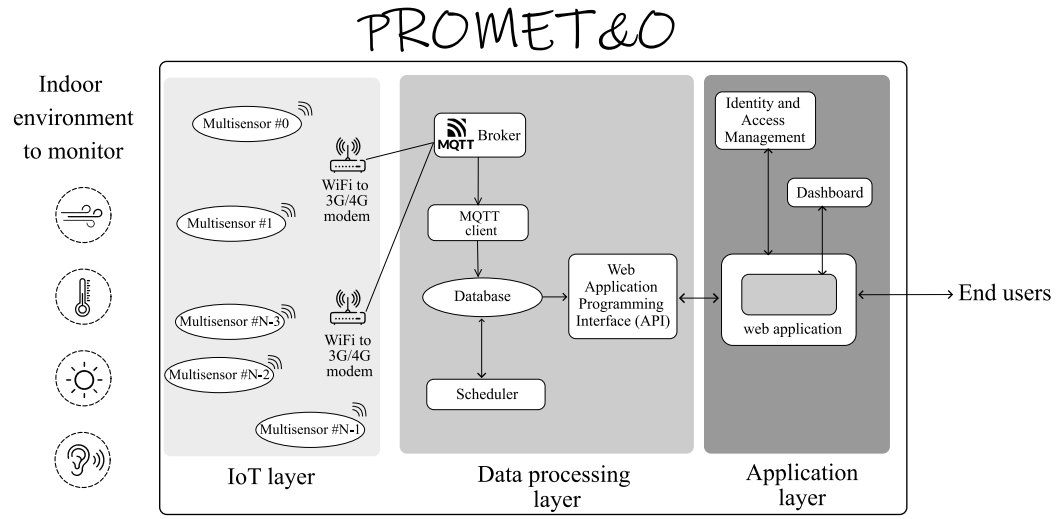

Figure 1. Architecture of the PROMET&O system. The multi-sensor devices measure physical quantities related to the four IEQ domains, performing also preliminary data processing. Data are sent to the MQTT broker through WiFi and 3G/4G wireless networks, and are stored into a database through an MQTT client. Data aggregation and enrichment is performed periodically by a scheduler. End users can access a web application, which reports the data in the database through a dedicated dashboard.

2.2. Data Acquisition Process

A key aspect to consider when developing an IEQ monitoring system as that shown in Figure 1, is the data acquisition process. Data collected from each monitoring device typically undergo processing and aggregation operations to provide representative and immediate feedback on the actual IEQ to the end-user. The timing and locations of these computations, whether at the multi-sensor or cloud platform level, should be addressed during the initial design stages. Trade-offs often arise, such as limited memory and computational capability at the multi-sensor level, the data rate, i.e., the amount of transferred data per unit of time, between multi-sensors and the server, and data compression at the database side.

A possible approach involves the periodic acquisition of the sensors deployed in the monitoring device, and the sending of all raw data to the cloud for processing operations. Under such a scenario, and by assuming that each multi-sensors acquires M_{sens} different physical quantities, the incoming data rate for cloud server, i.e., the number of incoming data from all multi-sensors per unit of time, can be evaluated as:

$$DR_{\text{periodic}} = N_{\text{ms}} \sum_{i=0}^{M_{\text{sens}}} f_{\text{sampl}}(i), \tag{1}$$

where $f_{\text{sampl}}(i)$ is the i-th sampling frequency, i.e., the inverse of the sampling period, associated with the i-th acquired sensor. Although such an acquisition scheme allows the cloud server to have at its disposal the entire set of data related to the physical parameters, it may result in a non-optimal trade-off between the require data rate, which directly affects also the required bandwidth, the database dimension, and the power dissipation of each multi-sensor, and the real-time capability of IEQ monitoring.

The acquisition process exploited in the developed system is shown in Figure 2. At the IoT level, each physical quantity is associated with a given sampling and report time, which are denoted as t_{sampl} and t_{report} in Figure 2. Each multi-sensor is therefore responsible for acquiring data from its M_{sens} sensors according to the specified t_{sampl}, and to temporarily store those value in its memory. Then, as t_{report} for the i-th sensor elapsed, each multi-sensor

aggregates the raw data in a set of K_{param} statistical parameters, and sends them to the cloud server.

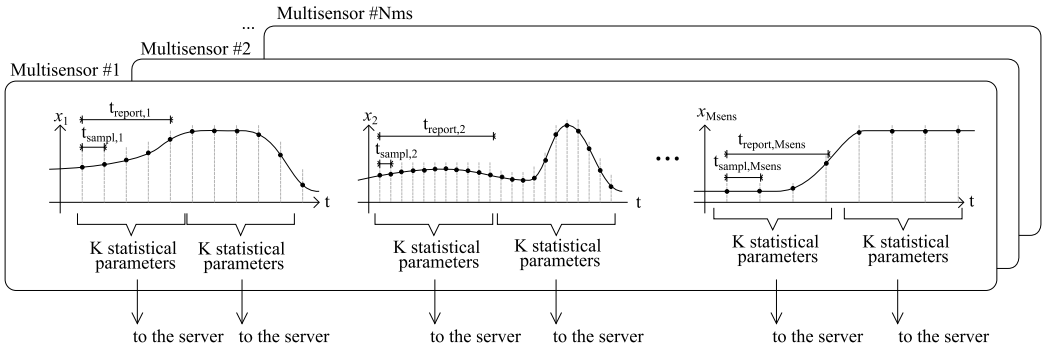

Figure 2. Process of data acquisition employed in the PROMET&O system. Each multi-sensor acquires several physical quantities, which are denoted as $x_1, \ldots, x_{\text{Msens}}$, each one with a given sampling time ($t_{\text{sampl},1}, .., t_{\text{sampl,Msens}}$). Acquired data are then aggregated into K statistical parameters and sent to the server over time intervals of length $t_{\text{report},1}, .., t_{\text{report,Msens}}$.

Under such a scenario, the required data rate for the cloud server can be evaluated as:

$$DR_{\text{report}} = N_{\text{ms}} \sum_{i=0}^{M_{\text{sens}}} K_{\text{param}} f_{\text{report}}(i), \tag{2}$$

where f_{report} is the inverse of the report time. Indeed, by adopting the acquisition scheme shown in Figure 2, the incoming data rate of the server can be reduced without losing information. In the implementation of the system, $K_{\text{param}} = 8$, i.e., the average, minimum, maximum, standard deviation, mode, median, and 10th and 90th percentile. In such a way, rapid variations of physical parameters, e.g., the opening/closing of doors and/or windows, can be detected from the maximum, minimum, and standard deviation parameters. Conversely, the remaining statistical parameters are more suited to be compared with the survey responses of the occupants. It is worth noting that existing works, such as [23,27], exploit $K = 1$, as they only sent the average values to the server. Even though such a solution may further reduced the required data rate, it also results in a loss of information at the server side.

As a consequence, DR_{report} will be lower than DR_{periodic} provided that $f_{\text{report}} > K_{\text{param}} f_{\text{sampl}}$ for each sensor. Indeed, by adopting the acquisition scheme shown in Figure 2, the incoming data rate of the server can be reduced without losing information.

3. Low-Cost Multi-Sensor

In order for the PROMET&O system shown in Figure 1 to estimate the actual IEQ, a real-time distributed in-field monitoring of the physical quantities of interest, e.g., temperature, illuminance, and air contaminants, is required. To this purpose, several low-cost monitoring devices, hereinafter referred to as multi-sensors, were developed and prototyped. In what follows, the hardware components and the operations executed by the firmware running on the multi-sensor are described, with particular focus on the data acquisition and preprocessing operations.

3.1. Hardware Description

The main blocks each multi-sensor is composed of are shown in Figure 3. From such an architecture, three main blocks can be identified, which are the set of sensors (on the left), the power and connectivity board (in the center), and the microcontroller (on the right).

Regarding the sensors, they were selected to monitor all four IEQ domains, i.e., thermal, visual, acoustic, and indoor air quality. Such devices were selected among those commercially available to optimize cost, measurement range, accuracy, dimensions, response time, and power consumption. The main parameters related to the selected

sensors are listed in Table 1, including the default sampling time and report time for each monitored quantity, as discussed in Section 2.2. The only exception is the microphone, whose output is sampled at 22 kHz, but the sound pressure level is evaluated over a 500 ms time window, as reported in Table 1. Most of the selected sensors can be directly connected to the microcontroller through standard digital serial interfaces. Conversely, the CO and NO_2 sensors, which are based on electro-chemical cells, require dedicated circuits to be interfaced with the microcontroller. Such extra circuitry was placed close to the sensing cells in ad hoc designed boards.

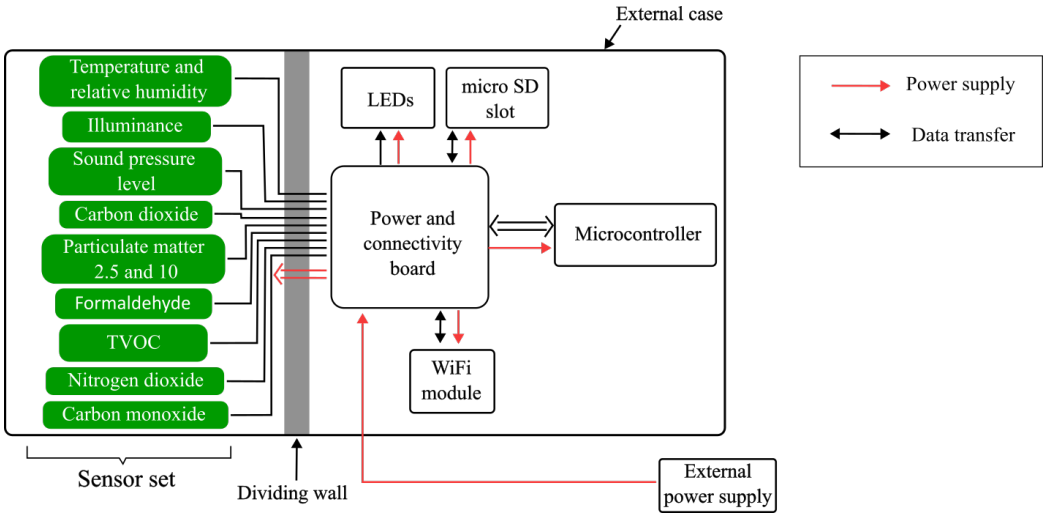

Figure 3. Block scheme of the PROMET&O multi-sensor, which comprises the set of sensors (on the **left**), the connectivity and power board (in the **center**), and the microcontroller (on the **right**). The set of sensors are located in a dedicated volume inside the multi-sensor, while the power and connectivity board is responsible for supplying all subblocks and for routing the interconnections. The operations of the multi-sensor are scheduled by the firmware running in the microcontroller.

Table 1. Parameters of sensors included in the developed multi-sensor.

Physical Quantity	Ref.	Sensor Range	Dimensions	Cost	Sampling Time	Report Time
Temperature	[38]	−40–125 °C	1.5 × 1.5 × 0.54 mm^3	4.04 USD	0.1 s	30 s
Relative humidity	[38]	0–100%	1.5 × 1.5 × 0.54 mm^3	4.04 USD	0.1 s	30 s
Illuminance	[39]	0–120 klx	6.5 × 2.35 × 3 mm^3	4.89 USD	0.1 s	30 s
Sound Pressure Level	[40]	122.5 dB (SPL)	3 × 4 × 1 mm^3	2.13 USD	0.5 s	5 s
Formaldehyde	[41]	0–1 ppm	42 × 24 × 5.5 mm^3	50.23 USD	3 s	1 min
TVOC	[42]	0–10,000 µg/m^3	22.9 × 14 × 3.15 mm^3	21.92 USD	3 s	1 min
PM 2.5	[43]	0–1000 µg/m^3	52.3 × 43.3 × 22.3 mm^3	27.91 USD	3 s	1 min
PM 10	[43]	0–10,000 µg/m^3	52.3 × 43.3 × 22.3 mm^3	27.91 USD	3 s	1 min
Nitrogen dioxide	[44]	0–5 ppm	20 × 20 × 3.8 mm^3	19.65 USD	3 s	1 min
Carbon dioxide	[45]	0–40,000 ppm	35 × 23 × 7 mm^3	51.37 USD	3 s	1 min
Carbon monoxide	[46]	0–1000 ppm	20× 20 × 3.8 mm^3	19.65 USD	3 s	1 min

The second main block is the power and connectivity board, which is responsible to connect all blocks shown in Figure 3, including the sensor array, to the microcontroller, while power supplying all of them. To this purpose, two DC-DC converters were included, to step down the 12 V external power supply to 5 V and 3.3 V required by the subcircuits the multi-sensor is comprised of. Moreover, the WiFi module was included to allow for the wireless connectivity of the multi-sensor. As far as the microntroller is concerned, it acquires data from the sensors, performs preliminary computations and sent data to the

cloud server through the WiFi module. It is based on an ARM Cortex M7 architecture, and it is provided with 1.1 MB RAM and 2 MB flash on-board memories. In addition to the main three blocks, the multi-sensor device includes a microSD slot, allowing to change on-the-fly the functional parameters of the multi-sensor, e.g., the report and sampling time, and two arrays of Light Emitting Diodes (LEDs) to provide users with a visual feedback of the measured IEQ.

The blocks identified in Figure 3 have been disposed of, as shown in Figure 4a, where the interior of the multi-sensor is shown. The structure of the device has been conceived to have the potential sources of heat, i.e., the power and connectivity board and the microcontroller, separated from the sensors. This was achieved by means of a vertical diving wall to provide thermal insulation between the sensors array and the remaining blocks of the multi-sensor. Furthermore, air quality sensors were placed on a vertical V-shaped chassis, which allows their sensitive elements to be as close as possible to the hem of the case. This should improve the response times of the sensors and, hence, the sensor outputs should be significant of the indoor environment to monitor. Finally, the overall assembled device is shown in Figure 4b.

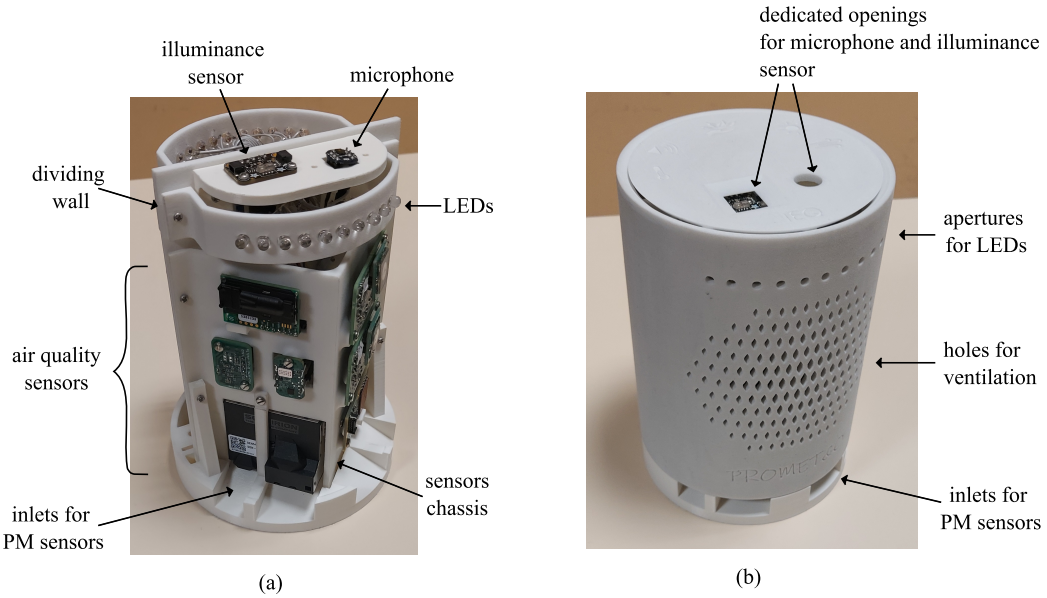

Figure 4. Photograph of (**a**) the internal parts and (**b**) the assembled PROMET&O multi-sensor. The vertical dividing wall in (**a**) allows for separating the IEQ sensors from the microcontroller and power and connectivity board, which are situated on the rear side (not visible in the image). The outer cylindrical case, which is shown in (**b**), features multiple apertures for microphone and illuminance sensors as well as for air ventilation purposes.

After analyzing the main components of the PROMET&O device, a comprehensive cost analysis was carried out, including all the necessary parts for device assembly. The results are reported in Table 2, with the cost of the sensor kit calculated as the sum of the individual prices of each employed commercial sensor, as listed in Table 1. It is worth noting that sensors account for 25% of the overall cost, with the most significant expense related to the 3D printing of the housing.

Table 2. Cost analysis for the PROMET&O multi-sensor.

Component	Price	% of Overall Cost
3D case printing	240 USD	37%
Sensors kit	164 USD	25%
Power and connectivity board	75 USD	11.5%
Custom board for sensors	45 USD	6.5%
Cables, connectors, nuts and bolts	38 USD	5.6%
WiFi module with board	32 USD	5%
Microcontroller board	30 USD	4.5%
LEDs and light pipes	18 USD	2.5%
External power supply	12 USD	1.8%
Overall cost	650 USD	

3.2. Firmware Description and Data Preprocessing

In order for the microcontroller to acquire the sensors data as discussed in Section 2.2, the routine implemented by the firmware running on the microcontroller is outlined in the flowchart reported in Figure 5. The steps reported in the flowchart, which have been numbered from 1 to 7 for the sake of clearness, can be divided between the initialization phase (steps 1–4), the sensor acquisition process (steps 5–6), and the audio processing (step 7).

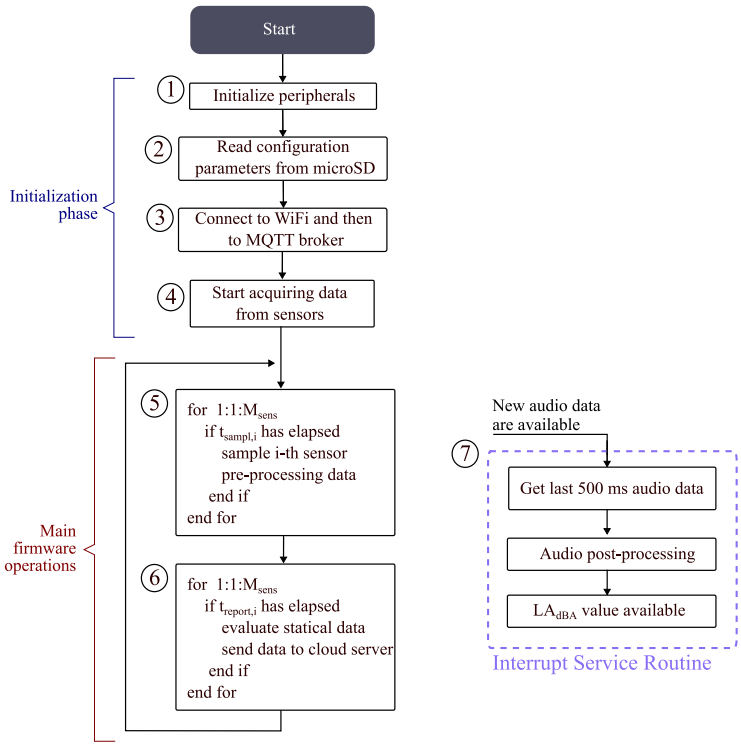

Figure 5. Flowchart of the algorithm of PROMET&O multi-sensor. The initialization steps (1–4) allow for setting hardware and software components, while the main operations are scheduled in steps 5–6. Audio data processing is performed asynchronously in step 7, which is the corresponding interrupt service routine.

Regarding the initialization phase, it is performed once after the multi-sensor is powered up, and it is required to prepare both hardware and software components for the main program execution. More precisely, all the peripherals of the microcontroller, which are required to interface with the different subblocks shown in Figure 3, are initialized in

step 1. Among the configured peripheral blocks, a timer configured with a 10 ms increment is started. Then, in step 2, the microcontroller reads the configuration text file from the microSD card. Such a configuration file allows users to change the parameters related to the acquisition process, i.e., the sampling and report time, as well as those related to the network connectivity, e.g., the WiFi service set identifier (SSID) and password, and to the cloud server, e.g., the server name, port, username, and password, without the need to reprogram the microcontroller. When the microSD is not inserted, the actual multi-sensor configuration is populated with default values. The microcontroller commands the WiFi module to establish a connection to the WiFi network specified in the microSD parameters and then connects to the broker (step 3).

The sensor acquisition process is started in step 4, and from that point on the algorithm continuously check whether sensors have to be sampled. To this purpose, the previously mentioned timebase is exploited to discipline the multi-sensor operations in accordance with the scheme shown in Figure 2. With the sampling time of the *i*-th sensors elapsed (step 5), the output of the sensor is acquired and some preprocessing is performed. More precisely, the new data are inserted in the array related to the ith physical quantity ($x_{\mathrm{i}}[]$) to keep it increasing-ordered, then the maximum and minimum values are updated, if needed, and the last value is added to the sum of the previous ones. Once step 5 is concluded, the firmware checks whether some data must be sent to the cloud server. To this purpose, in step 6, all sensors are scanned to verified whether the corresponding report time has elapsed. In that case, the statistical data discussed in Section 2.2 are evaluated. It is worth mentioning that, as the $x_{\mathrm{i}}[]$ is an ordered array of length $N = (t_{\mathrm{report,i}}/t_{\mathrm{sampl,i}}) + 1$, the median ($M$), i.e., the central value in an ordered distribution, can be directly evaluated as:

$$M = \begin{cases} x\left[\frac{N}{2}\right] & \text{N even,} \\ \frac{1}{2}x\left[\frac{N}{2} - 1\right] + \frac{1}{2}x\left[\frac{N}{2} + 1\right] & \text{N odd.} \end{cases} \tag{3}$$

Similarly, the 10th and 90th percentile can be obtained by evaluating the corresponding index ($0.1N$ or $0.9N$) and considering either the associated value (with the index being an integer number) or the average between the two adjacent values. Finally, the K statistical parameters are sent to the cloud server trough a defined MQTT message. The microcontroller executes step 5 and 6 continuously until some network error occurs. Under such a scenario, the multi-sensor is restarted and the firmware starts again from step 1.

3.3. Audio Data Processing

Regarding the audio processing, the acoustic noise should be continuously sampled by means of the selected microphone, and the equivalent A-weighted acoustic level evaluated. To this purpose, processing of the data stream from the microphone is carried out in step 7, which is performed asynchronously to step 5 and 6 of the flowchart shown in Figure 5. Indeed, the microphone output stream is directly copied in a memory area of the microcontroller, not to the increase the workload of the firmware itself. In accordance with the ANSI S1.42 standard [47], the A-weighting transfer function $H_{\text{A-weight}}$ defined in the Laplace domain is given by:

$$H_{\text{A-weight}}(s) = G_{\mathrm{a}} \frac{(\omega_4^2 s^4)}{(s+\omega_1)^2 (s+\omega_2)(s+\omega_2)(s+\omega_4)^2}, \tag{4}$$

where $\omega = 2\pi f$, and f_1 = 20.6 Hz, f_2 = 107.65 Hz, f_3 = 737.86 Hz, and f_4 = 12,194.2 Hz [48]. The magnitude and phase of $H_{\text{A-weight}}$ are plotted in Figure 6a in solid lines. It is worth noticing that $|H_{\text{A-weigth}}|$ = 0 dB at 1 kHz. To implement such a filter directly by the developed multi-sensor, (4) was time-discretized, meaning transformed in the z^{-1} domain, using Tustin's method [49]. The corresponding magnitude and phase are shown in Figure 6a in dashed lines. As the microphone sampling time was set to f_{s} = 22 kHz due to memory

constraints, the magnitude and phase of the discretized transfer function differ from those of (4) for frequencies higher than 5 kHz.

With the transfer function in the z^{-1} domain, an infinite impulse response filter (IIR) in transposed direct form II was implemented in step 7. The effect of applying the IIR filter to broadband 100 Hz–10 kHz noise, which was generated by a sound source [50] and sampled by the selected microphone, is shown in Figure 6b.

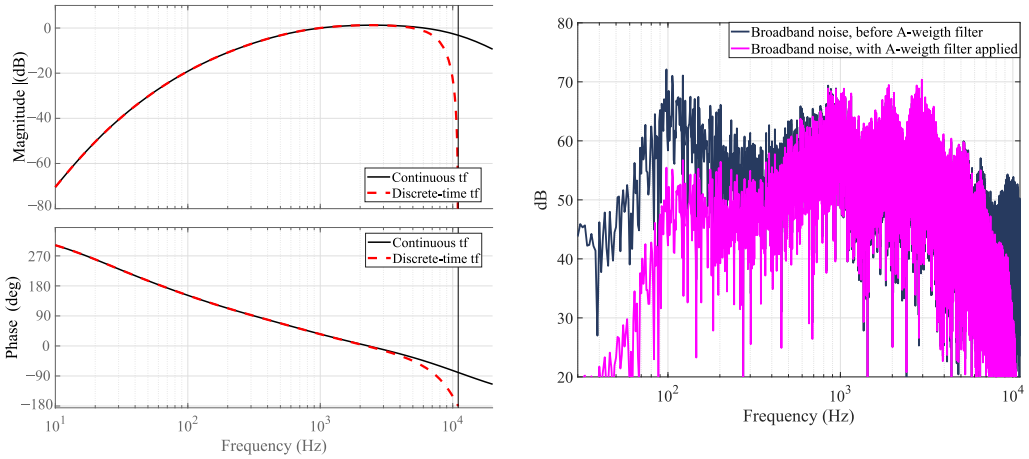

Figure 6. In (**a**), the magnitude (on the top) and phase (on the bottom) of the time-continuous (solid) and time-discrete (dashed line) transfer function associated with the A-weighted filter. In (**b**), the magnitude of the spectra of broadband 100 Hz–10 kHz noise acquired by the multi-sensor before and after applying the A-weighted filter.

4. Cloud Server

Recalling the PROMET&O architecture shown in Figure 1, the cloud server hosts two layers of the system, i.e., data processing and data visualization layers. The cloud server consists of a set of containerized applications managed by Docker [51]. Such a tool allows developers to deploy their applications consistently and portably across various environments and to replicate them with minimal changes for different clients or scenarios.

The applications in the *data processing layer,* shown in Figure 7, collect data from the multi-sensor devices via a messaging client that stores them in the database in real time. These data are then further processed by a scheduler that, at regular intervals, computes several derived measures to help reduce the computation time for future real-time analysis.

The *data visualization layer*, shown in Figure 8, relies on a web application that provides the graphical front-end through which end-users can navigate and explore the data. Most of the user interface is implemented in the application itself except for the rendering of the graphs that are embedded from a dashboarding software, Grafana [35], largely used for time-series representation.

The *data visualization layer* also benefits from an *API gateway*, implemented with Nginx [52], which is positioned in front of all the applications and that acts as a single entry point for the backend services. It provides centralized handling of all the communications with end-users and delivers security, policy enforcement, routing, and service visibility. Consequently, it streamlines the development and management of internal applications that can offload non-functional requirements to a central entity.

For instance, authentication is fully managed by the gateway through the OpenID Connect protocol. This protocol enables the gateway to retrieve user information from an identity and access manager (IAM), such as Keycloak [36]. The gateway intercepts all unauthenticated requests and redirects clients to the IAM's login page. Upon successful authentication, the gateway transmits user information, encapsulated within the HTTP headers, to the upstream service. By means of the gateway, which also works as a reverse proxy, all applications are exposed through the same web domain, e.g., prometeo.polito.it,

so that they can easily share information through HTTP cookies. Various paths are allocated to distinct applications within the system architecture: the primary root (/) is dedicated to the web application, wherein "/dashboard/" denotes the chart services. URLs commencing with "/chart/" are directed to Grafana, for plot embedding. Lastly, the path "/auth" redirects to the identity and access management (IAM) system. Thus, there is no direct communication between the user browser and the data processing layer.

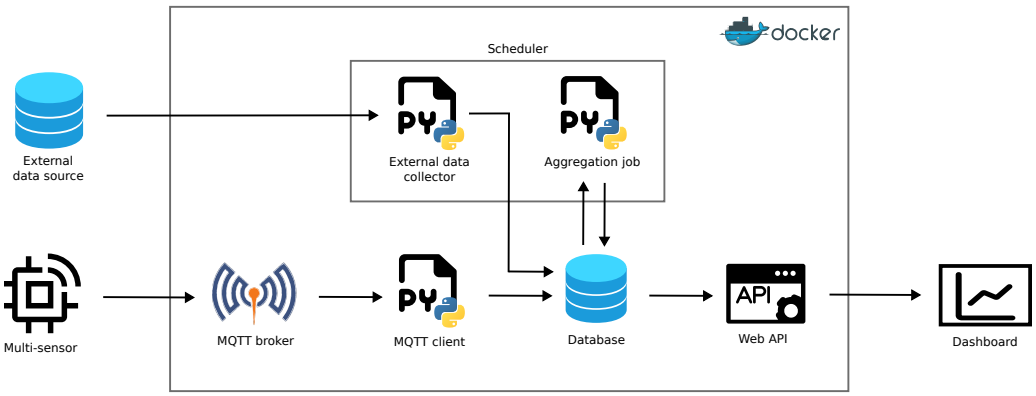

Figure 7. Data processing architecture providing data collection, storage, and retrieval.

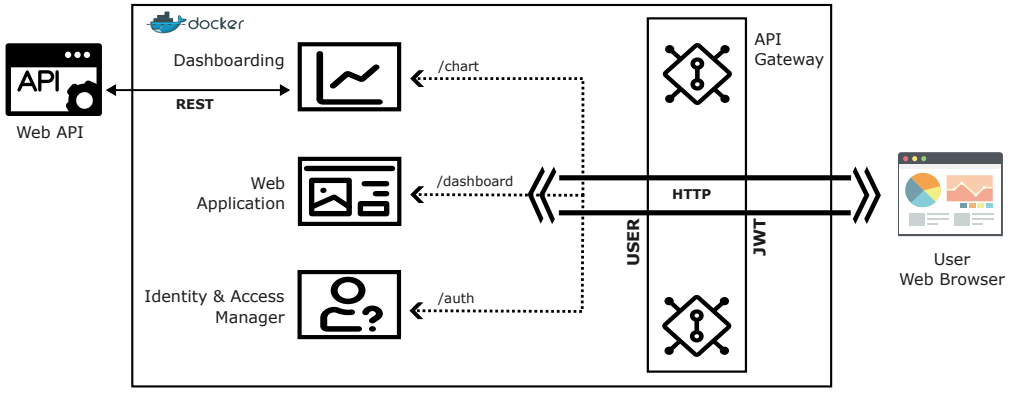

Figure 8. Data visualization architecture providing API gateway, identity and access manager, dashboarding application, and web application.

4.1. Data Processing Architecture

The software infrastructure of the *data processing layer* must provide efficient data collection, storage, and retrieval for visualization and data analysis. As shown in Figure 7, it is composed of a message queuing telemetry transport (MQTT) broker (Mosquitto), a client, a database management system (MySQL), a python job scheduler, and a web API. More precisely, the broker receives the messages from the PROMET&O multi-sensors, and it forwards them to the MQTT client for saving the contained measurements in the MySQL database. The Python scheduler is used to run aggregation jobs and fetch data from external sources, to enrich the stored information. The data stored in the database are made available to the data visualization layer via a web API.

4.1.1. Broker

The first component in the software architecture is the MQTT broker, which handles the messages coming from the PROMET&O multi-sensors. It operates on a publish–subscribe pattern, allowing clients to define communication channels via topics. Subscribing clients receive messages from others clients publishing on these topics. The broker handles topic subscription, message reception, and delivery to subscribers, offering different quality of service (QoS) levels (zero, one, or two) for messages and subscriptions. Mosquitto [53] was

selected in this work for its simplicity and ease of configuration, despite some limitations in user management and authorization control encountered during development.

4.1.2. MQTT Client

A Python MQTT client was developed using the paho-mqtt library [54]. During the initialization, it retrieves information about the installed multi-sensors from the database and subscribes to their respective topics on the broker. Upon reception of messages, it parses their content and saves the contained data in the database. Additionally, the MQTT client computes the indoor environmental indexes, for which data obtained from external sources are required, and stores them in the database.

4.1.3. Database

The database management system exploited in PROMET&O was MySQL [55], being open source, well documented, and established in the market. Indeed, the project scope did not require extensive scalability and strict performance requirements. With such constraints, timeseries-oriented databases in both SQL and NoSQL flavours, such as Timescale [56] and InfluxDB [57], could be investigated.

The database schema was designed to support general uses cases of IoT sensor deployment. The Entity Relationship (ER) diagram of the developed database is shown in Figure 9. The *BOARD* table represents a general IoT device, identified by a serial number and a vendor model linked via a foreign key to the *VENDOR_MODEL* table. Each IoT device generates multiple timeseries of measurements, each of them identified by a logical sensor from the *LOGICAL_SENSOR* table. Logical sensors have properties such as acquisition frequency and unit of measure, described in the *UNIT_OF_MEASURE* table.

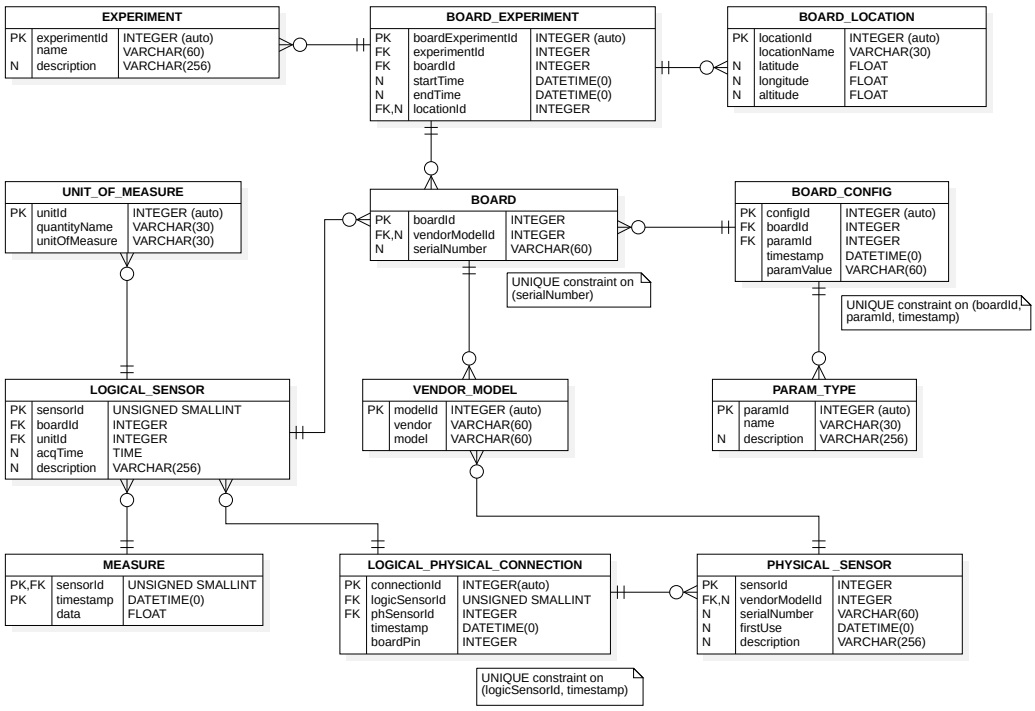

Figure 9. Database ER diagram.

The *MEASURE* table, which is central in this design, records data acquisition by each logical sensor using tuples in the format *(sensorId, timestamp, data)*, with the foreign key being composed of the tuple *(sensorId, timestamp)*. This arrangement prioritizes *sensorId* for its lower cardinality than *timestamp*, speeding up the filtering. To reduce the space disk, the space allocated for storing each measurement should be minimized. Therefore, *sensorId* is an *UNSIGNED_SMALL_INT* and *data* is in single precision, using two and four

bytes, respectively. The *timestamp* is stored as a *DATETIME* without fractional seconds, occupying five bytes. As the date are always expressed in coordinated universal time (UTC); no timezone conversion occurs during storage or retrieval. To sum up, eleven bytes are required to store each measurement. This optimization, however, imposes constraints on measurement precision, number of logical sensors, and sampling frequency, which can be adjusted by modifying data type sizes.

The *PHYSICAL_SENSOR* and *LOGICAL_PHYSICAL_CONNECTION* tables are used to express the connection between an hardware sensor installed on the device and the produced timeseries (logical sensor). This allows to represent hardware sensors measuring different parameters and the possibility of an hardware sensor being replaced due to a failure, with each new one acting as the same logical sensor.

For tracking device configurations, the *BOARD_CONFIG* and *PARAM_TYPE* tables are used, while the *EXPERIMENT*, *BOARD_EXPERIMENT*, and *BOARD_LOCATION* tables monitor device usage across measurement campaigns and deployment locations.

Additional tables, shown in Figure 10, are used to store measurement aggregations computed by the scheduler. The *FIVE_MIN_AVG_MEASURE* table contains data aggregations for each logical sensor over five minute intervals. The mean, maximum, minimum, and standard deviation of the measurements are computed over the five minute interval and stored inside the table. The *HOUR_AVG_MEASURE* stores the same information but for hour averages. The *VIEW_UPDATE* table contains, for each aggregation table, the table name and the timestamp of the last update.

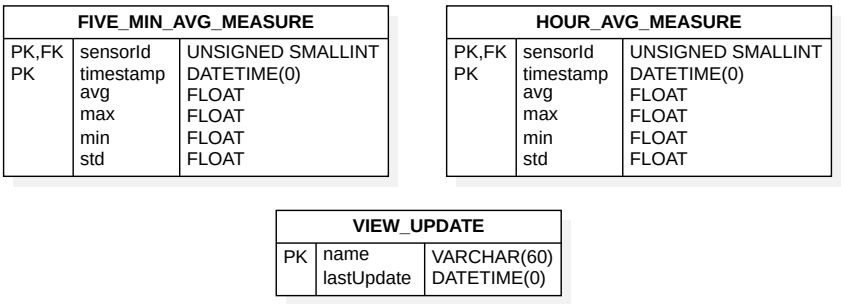

Figure 10. Database ER diagram—aggregation tables.

4.1.4. Scheduler

The role of the scheduler is to preprocess the data for further analysis and visualization. To this purpose, it was implemented using the APScheduler Python library [58], which allows to run periodic jobs at specific times. It performs three main tasks: data aggregation, external information retrieval, and comfort index computation.

Data aggregations, including minimum, maximum, and standard deviation evaluations are carried out every five minutes and every hour, and stored in the *HOUR_AVG_MEASURE* and *FIVE_MIN_AVG_MEASURE* tables previously mentioned.

Secondly, some scheduler jobs were configured to access external data, such as the outdoor temperature, via web APIs and store it in the database.

Finally, remainder jobs compute indoor comfort indexes, which are stored and aggregated similarly to standard measurements, using both multi-sensor data and external information.

The scheduler also allows to recompute past aggregation via the *VIEW_UPDATE* table. By changing the time of the last update for a table to an older timestamp, the scheduler recomputes past aggregations and updates the aggregation tables accordingly.

4.1.5. Web API

The web application programming interface (API) enables external services to access data stored in the database, with a format suitable for the required task. Such APIs were developed using the Flask [59] python library.

As the Grafana [35] tool is the main client of the APIs, the API endpoints must follow a format supported by Grafana. In the developed system, the selected format is as follows:

base_url/{board_id}/{measure_name}?avg={ }&ts={ }

where "board_id" is the multi-sensor device identifier, as stored in the *BOARD* database table, and "measure_name" identifies which measurement should be retried, e.g., temperature and TVOCs. The "avg" query parameter is the time aggregation of the retrieved data points, which can be selected amongst real time (RT), three hours, twenty-four hours, three days, one week, and one month. The "ts" query parameter is the length of the time interval, starting from the instant of the request, for which the data must be retrieved. It can be specified via a number followed by a time descriptor, e.g., 3 s, 4 m, 2 h, 5 d, 1 w, 6 M, and 1 y. Table 3 reports the database pre-aggregation used by the API to compute each available time aggregation, distinguishing between measurements and comfort indexes. The 24-h aggregation alignes with midnight, whereas others aggregations align with the request instant. This allows the user to always receive up-to-date information, without the need to wait for the next aggregation to be computed. Real-time data remain unaggregated, as they are retrieved directly from the database table.

An important aspect is that averages of indoor parameters should not be calculated during periods when the indoor space is not in use, e.g., at night, during holidays, or weekends. Doing so could distort comfort evaluations, as the monitored environment may not be maintained under habitable conditions, e.g., no heating, lighting, or air circulation. However, it can still be valuable to track how conditions evolve during these unused periods. To address this point, real-time and 3-h aggregations are calculated using all available data, while other aggregations only consider measurements collected during the actual utilization periods. This distinction is achieved by configuring a calendar, which the API consults before computing the aggregations.

Table 3. Table selection according to measurement type and avg parameter.

Data Type	Aggregations					
	RT	**3 h**	**24 h**	**3 d**	**1 w**	**1 M**
Measures	Measure table/5 min	5 min	5 min	5 min	1 h	1 h
Comfort indexes	5 min	5 min	5 min	5 min	1 h	1 h

4.2. Data Visualization

The main objective of developing the data visualization application was to provide users with the ability to explore and monitor collected data graphically. Additionally, the aim was to enable users to customize the visualization of the data without the need for coding. To achieve this, the integration of a dashboarding application within the data visualization layer was implemented. The detailed architecture is depicted in Figure 8.

From a practical standpoint, upon receiving a user request to visualize sensor data, the web application forwards an HTTP request to the dashboarding system. This request contains all the necessary information to identify the type of chart to create, the sensor, the measurement of interest, the temporal interval of the data, and their aggregation time. Subsequently, the dashboarding system retrieves the data through the web API, which serves as an intermediary with the database. It then generates the HTML code to be returned to the web application for displaying the chart on the page.

4.2.1. Web Application

The web interface has been implemented as a single page application (SPA) based on the React web development framework [37]. A screenshot of the main view is shown in Figure 11.

The page layout consists of three main sections: the sidebar on the left, a navigation menu on the top, and a main body. The sidebar is used to inform viewers about the current measure, while buttons on the top allow to navigate across different aggregations.

On the web page, two additional views can be accessed via the buttons "Hide the graph" and "Compare the graphs". The former is used to alternate the chart with a dashboard that shows all the available measures, affording users the ability to switch between them. The latter, when activated, presents a view where users can visualize and compare four distinct charts, offering two options: (i) comparing four charts of the same measure but with differing aggregation periods and (ii) comparing four chart with the same aggregation period but featuring different measures.

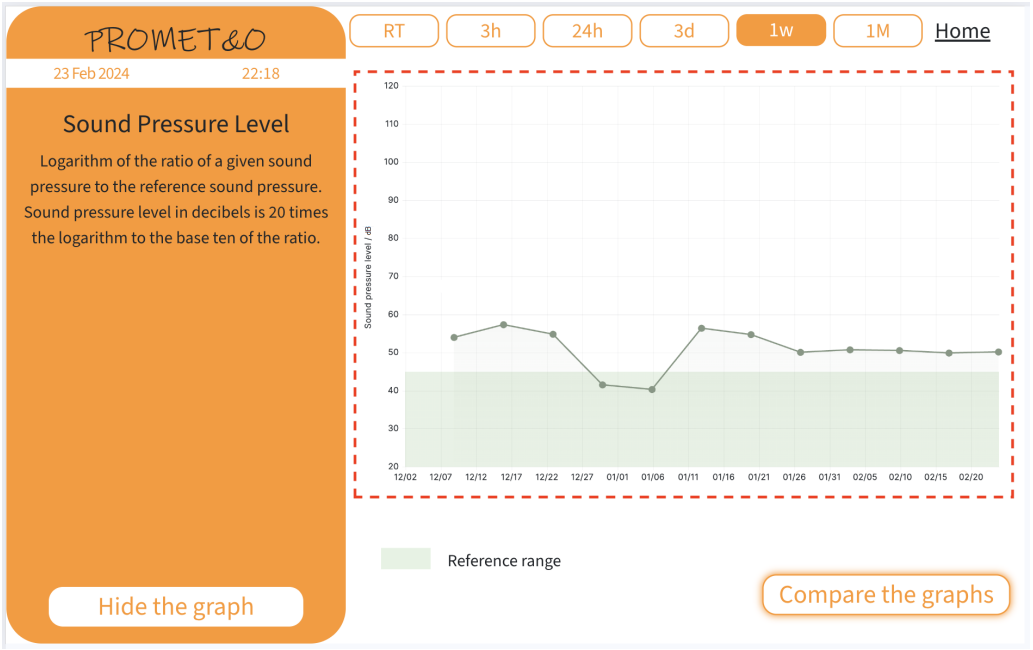

Figure 11. Web application screenshoot with the three main components: the sidebar on the left, the navigation bar on the top, and the chart in the center (the portion in the dashed red frame).

4.2.2. Dashboarding Integration

The dashboarding application integrated into the system is Grafana [35], an open source visualization and analytics software that provides several tools to turn time-series data into insightful graphs and visualizations.

Graphs are encapsulated inside panels that define how data are retrieved, potentially transformed, and ultimately displayed. Each of these configuration parameters is editable via a user-friendly graphical interface obviating the need for any programming expertise. However, the project entails monitoring a dozen of measurements with six different aggregations across several multi-sensors, making it impractical to manually create hundreds of panels for each visualization. The identified solution was to confine customization to one panel per measurement, allowing the definition of axis labels, value ranges, optimal intervals (denoted by a green band), line and point colors and formats, etc. Variables are utilized for other customizations, including the multi-sensor id to query, aggregation interval, and temporal interval of the chart. All this information can be encapsulated in the URL used to request Grafana to render the panel.

An illustrative example of a URL is as follows:

https://ns01-prometeo.polito.it/chart/d-solo/bdb8ca18-df79-4743-818a-ebc8123908e5/prometeo?panelId=1&var-boardId=1001&var-avg=3h&var-ts=48h

The URL comprises four components: the scheme (https), the host (ns01-prometeo.polito.it), the path, and the query (the characters following the question mark symbol), which is a sequence of name and value pairs separated by an ampersand character.

In the path the string "chart/d-solo" indicates to Grafana the rendering of the panel alone, excluding other components of the Grafana dashboard. The identifiers "bdb8ca18-df79-4743-818a-ebc8123908e5" and "prometeo", respectively, denote the id and name of the dashboard to which the panel belongs.

Within the query, the "panelId" parameter selects the panel (e.g., panel with id 1 corresponds to the temperature graph), while other parameters prefixed with "var-" designate different Grafana variables: "boardId"signifies the multi-sensor's identifier, "ts" specifies the time span of the data extending from the current time back 48 h, and "avg" denotes the aggregation time (3 h). These parameters are also utilized to perform the query to the web API of the database in order to retrieve the data.

Additionally, Grafana offers the capability to embed panels and graphs into external web applications through an *iframe* element. An *iframe* is a standard HTML tag and it serves as a container for HTML content provided by a third-party application. It requires the specification of three attributes: the "src" attribute, denoting the URL of the resource to be embedded in the web page, the "width" and "height" attributes, which specify the desired dimensions of the container. In Figure 11, the panel embedding by means of an *iframe* is marked by a dashed red frame.

By using different URLs, it is possible to customize which chart and data to display in the *iframe*. Whenever a new page is opened or a user interacts by selecting a different visualization mode, the URL of the *iframe* is changed, and a new panel is displayed.

5. System Performance

With the system shown in Figure 1 set up, some preliminary measurements were carried out to assess the functioning and the performance of the developed system. Firstly, the multi-sensor was tested, and collected data were compared to those from other measurement instruments. An analysis of such measurement is reported in what follows. Then, performance results regarding the user interface are provided, and enhancements to decrease the related refresh time are discussed.

5.1. Multi-Sensor Validation

To validate the multi-sensor operations, a one-day measurement campaign was conducted in an office space with the setup shown in Figure 12. Alongside the developed multi-sensor, a commercially available IEQ monitoring device [33] was positioned as closely as possible to the multi-sensor to ensure collection of comparable data. Additionally, measurements from reference instruments situated in the same room were obtained for comparison purposes. More precisely, the reference for relative humidity was [60], ref. [61] for illuminance, ref. [62] for sound pressure level, and ref. [63] for carbon dioxide.

The results of the measurement campaign are shown in Figure 13 for temperature, relative humidity, illuminance, and sound pressure level. Solid lines refer to data collected by the developed multi-sensor, dashed lines correspond to data from the commercial device, and dashed-dotted lines refer to measurements from the reference instruments. As far as data collected by PROMET&O are concernened, they refer to the averages at report time for temperature, relative humidity, and illuminance. Conversely, sound pressure level collected by the multi-sensor refers to equivalent sound pressure level evaluated over a 500 ms time window. Regarding temperature, the measurements from the PROMET&O and the commercial device exhibit good agreement, whereas thermocouple used as reference resulted in a measurement 2 °C lower. Similarly, illuminance showed consistent agreement among the various devices. Concerning relative humidity, the PROMET&O device overestimated the measurement by 5% compared to the other two devices. Lastly, the sound pressure level, weighted in accordance with the A-curve as reported in (4), was

compared between the developed multi-sensor and the sound level meter. These results are in good agreement, with a minor error of a few decibels.

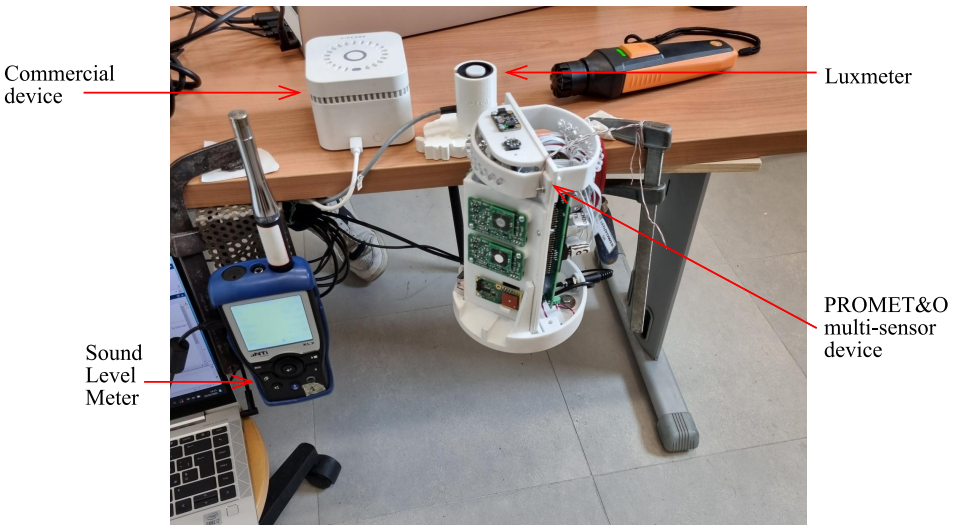

Figure 12. Experimental setup with the PROMET&O multi-sensor, a commercial device, and some reference instruments in an office.

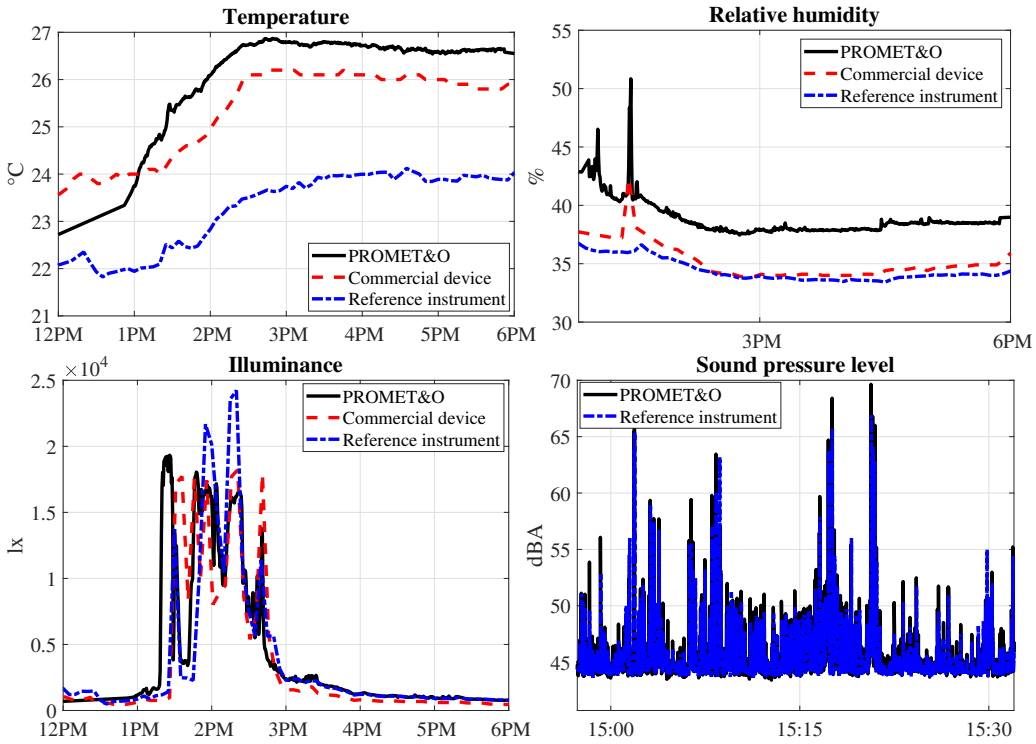

Figure 13. Temperature, relative humidity, illuminance, and A-weighted sound pressure level data acquired by the multi-sensor (solid), the commercial device (dashed), and the reference instruments (dashed-dotted lines) during a one-day measurement campaign.

Regarding IAQ, the measurements of carbon dioxide, TVOCs, PM 2.5, and PM 10 for the setup shown in Figure 12 are reported in Figure 14, where solid, dashed, and dashed-dotted lines represent data collected by PROMET&O, the commercial device, and the reference instruments, respectively. More precisely, solid lines refer to the average value of data collected by the respective sensor over a report time window. For the other air quality sensors included in the developed multi-sensor device, their respective gas concentrations

were either too low, such as NO_2, or lacked comparison data, such as formaldehyde, and thus, were not included in Figure 14.

Concerning carbon dioxide, the measurements obtained by the developed multi-sensor are in good agreement with those from the reference instrument, with some peaks observed. Conversely, the commercial device tends to overestimate the actual concentration. As for TVOCs and PM 2.5/10, the measurements from the developed multi-sensor do not seem to capture certain transient events observed by the commercial device.

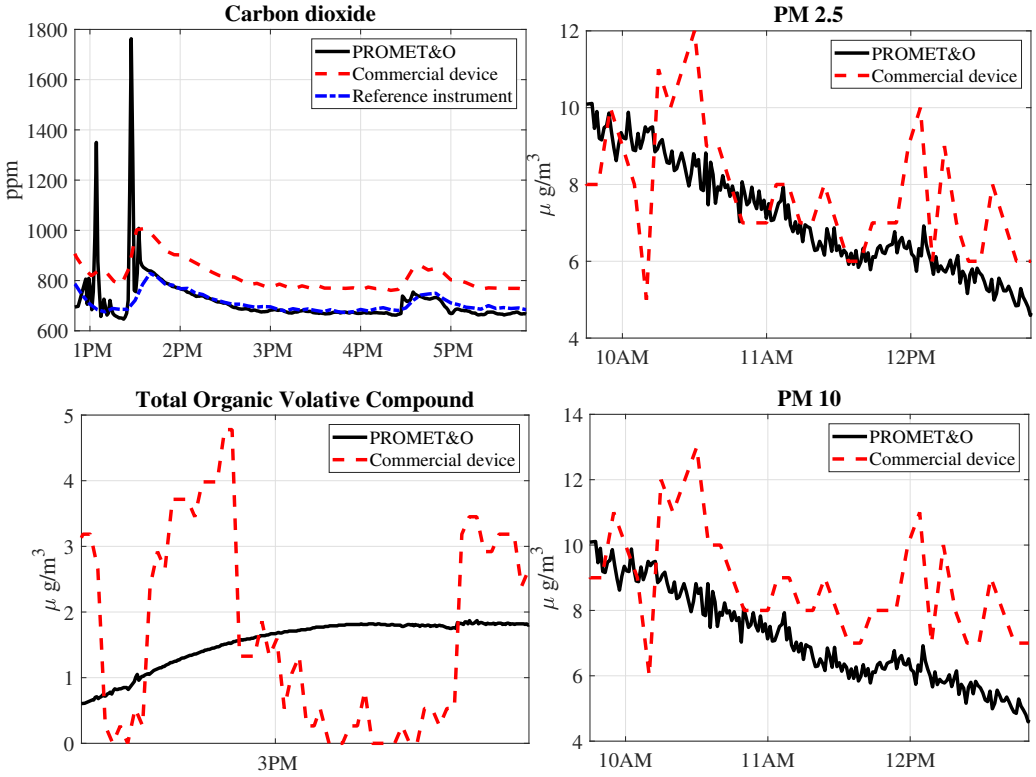

Figure 14. Carbon dioxide, TVOCs, PM 2.5, and PM 10 data acquired by the multi-sensor (solid), the commercial device (dashed), and the reference instruments (dashed-dotted lines) during a one-day measurement campaign. Data collected by the PROMET&O device refer to averages values evaluated at every report time.

Some differences among the measurements can be due to the experimental setup, as the devices were located in different position in the room, meaning that they experienced different IEQ parameters. More precisely, the increase of illuminance values detected by the multi-sensor, reported in solid line at the lower left of Figure 13, precedes those recorded by both the commercial device and the reference instrument. Such an early detection is attributed to the direct sunlight reaching the location of the multi-sensor before reaching those of the other devices. At the same time, the preliminary results reported in Figures 13 and 14 assessed the necessity for a metrological characterization of the sensors exploited by the multi-sensor.

5.2. Impact of Sampling Time

The data collected from the multi-sensor device were exploited to assess the impact of sampling time. To this purpose, the original series of data, which were acquired with the sampling times reported in Table 1 and denoted as x_m hereafter, were decimated using various decimation factors. The decimated series were then linearly interpolated (x_i) to reconstruct series of equal length to the original ones. Subsequently, the maximum absolute error between the original and the interpolated series was evaluated and it is

shown in Figure 15 for temperature, relative humidity, illuminance, TVOCs, PM 2.5, and carbon dioxide.

As the sampling time increases, the maximum error also increases for all the considered physical quantities. Such a result is related to the down-sampling effect, meaning that a too-low sampling time may result in inaccurate measurement of transient events.

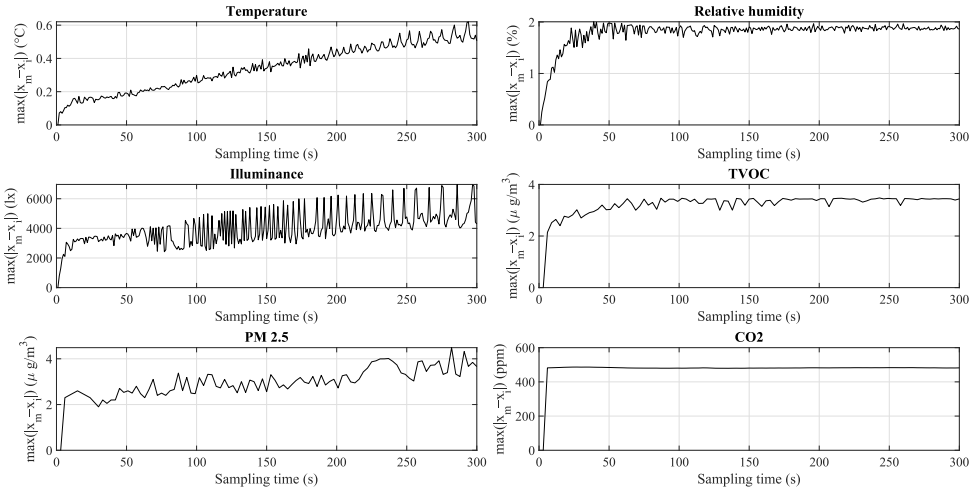

Figure 15. Maximum error between the decimated and the original series for sampling times up to 5 min.

5.3. Enhanced Interactivity in Data Visualization

With the functioning of the multi-sensor device, some amendments were made to the web application to enhance the user experience. Embedding a chart inside an *iframe* allows seamless integration of Grafana with the web application, ensuring visual consistency across web pages. However, interaction between the web application and the *iframe* is essential for users to navigate through different visualizations and update the chart accordingly.

Grafana offers only a rudimentary mechanism for interacting with the *iframe*, i.e., altering its URL. This action prompts the *iframe* to automatically clear its content and request another panel from the server in accordance with the specified parameters, as outlined in Section 4.2.2. However, as shown in Figure 16, this operation is not instantaneous and, by requiring the *iframe* to clear before displaying a new chart, the previously occupied area remains empty for a significant period, causing a visually unpleasant effect.

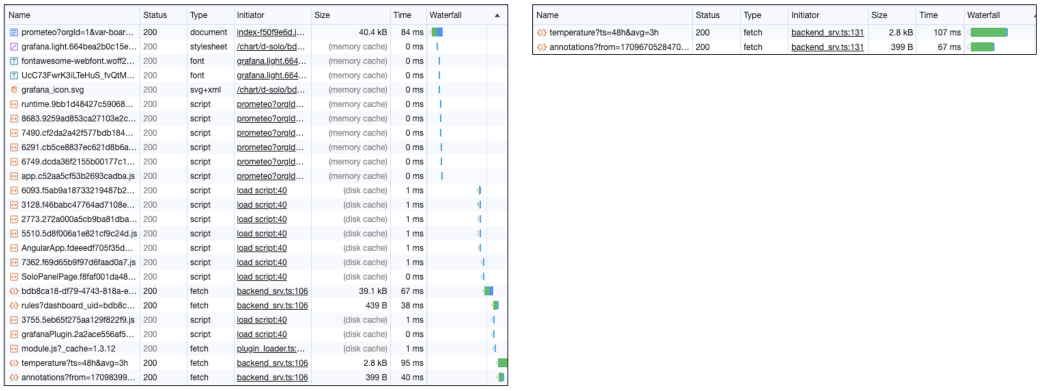

Figure 16. Detail of the network panel of the Chrome DevTools application showing the number of resources requested by the original (**left**) and the proposed (**right**) approach together with the time spent.

An alternative mechanism has been implemented to prevent the *iframe* from clearing its content when a chart needs updating. Instead of changing the URL, a message with the new parameters is passed from the application to the *iframe* (which is itself a React web application) to communicate the need for an update. As a result, upon successfully retrieving the new data, the chart is redrawn accordingly without any visual interruption for the user. This solution is based on the JavaScript method *window.postMessage()* that safely enables communication between window objects (i.e., web pages), such as the main page and the embedded *iframe*. The sender window may obtain a reference to the destination window and then dispatch a *MessageEvent* on it, with the data passed to the method (i.e., the "message" containing the required data) being exposed to the receiving window through the event object; this window can then listen for dispatched messages by executing the *addEventListener* method.

The implementation of the mechanism on the sender side, the project web application, is relatively straightforward, because all the necessary data are readily available. Conversely, implementing this mechanism on the receiver side necessitates understanding and carefully modifying the Grafana code. Fortunately, Grafana is open source, and its codebase is accessible on GitHub, facilitating the necessary modifications.

Two distinct message handlers are necessary in two separate components of the Grafana React Application, which is contained within the *iframe*. To modify the panel id, a callback must be registered via the *addEventListener* method in the *GrafanaRoute* component. Then, the change of the other parameters is managed by an a*ddEventListener* callback in the *PanelStateWrapper* component, where the panel variables can be adjusted, and a refresh of the panel can be triggered to display the new data. By implementing such modifications, the user is not presented with a blank page during the process.

As shown in Figure 17, the proposed solution reduces the loading time, thereby offering a more immediate feedback to the user. The amount of resources that need to be retrieved from the server is significantly reduced, as detailed Figure 16, and although most of them are cached by the browser, this difference results in a refresh time reduction of almost an order of magnitude. In the original implementation, the time required to load and display the new chart is approximately 700 ms, whereas with the new implementation, its median is about 100 ms.

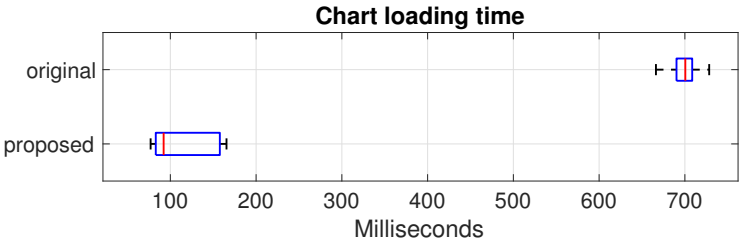

Figure 17. Loading time comparison between the proposed implementation and the original one, provided as a default by Grafana, for updating a chart embedded in an *iframe*. On each box, the central mark indicates the median, the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively, and the whiskers extend to the most extreme data points, the minimum and the maximum values.

A fork of the original Grafana codebase with the modifications has been published on Github [34] and an issue with a pull request (https://github.com/grafana/grafana/pull/86012 (accessed on 6 May 2024)) has been opened on the original repository to discuss and potentially incorporate the changes.

5.4. Comparison with Existing IEQ Monitoring Systems

With the functionality of the PROMET&O system assessed, a comparative analysis was performed to evaluate the performance of the proposed system against those of existing systems proposed in the literature, such as SAMBA [23], ENVIRA [27], and commercial products such as [33,64]. The results of such an analysis, which considers parameters and

indicators both at the monitoring device and at the cloud platform levels, are reported in Table 4.

At the monitoring device level, factors such as the type of sensors employed, overall volume, cost, and data processing capabilities were considered. While most references evaluated all four IEQ domains with 5 to 10 different types of sensors, limited data processing was performed, typically restricted to average calculations on raw sensor data. The cost of such units ranged from hundreds to almost a thousand dollars, significantly lower than reference instruments, with a compact volume in the order of a few cubic decimeters. These research works provided valuable insights into the development of multi-sensor devices.

Table 4. Comparison of existing IEQ monitoring systems.

	Device Level					Platform Level		
Ref.	**Measured Parameters**	**Data Processing**	**Cost**	**Volume**	**Design Insights**	**Server Platform**	**Dashboard**	**Design Insights**
[23]	Illuminance, SPL, CH_2O, CO, PM, CO_2, temp., RH, global temp.	Averages	210 USD sensors only	2.3 dm^3	yes	based on Amazon web services	Commercial products	no
[27]	dry bulb temp, black globe temp, RH, air velocity CO_2, TVOC, PM 2.5, SPL, illuminance	Averages	180 USD	5 dm^3	yes	Blynk commercial platform	Smartphone app	no
[28]	Temp., RH, illuinance, CO_2 pressure, PM Alcohol, ozone motion, nro. people	Prediction models based on edge computing	≈850 USD	n/a	yes	Open-source platform	Open-source dashboard	no
[22]	Temp., RH, SPL, illuminance, barometric pressure, CO_2, CO, NO_2, PM	Raw data to server	850 USD	n/a	yes	Open-source custom platform	Custom web app	no
[64]	Temp., RH, CO_2, VOC PM, CH_2O	n/a	730 USD	1.5 dm^3	n/a	Custom platform	Custom web app	n/a
[33]	Temp., RH, atmos. pressure, TVOC, PM, CO_2, SPL, illuminance	n/a	600–800 USD	1.5 dm^3	n/a	Custom platform	Custom web app	n/a
This work	Temp., RH, SPL, illuminance TVOC, CH_2O, PM, CO_2, CO, NO_2	Statistical parameters from raw data	650 USD	2 dm^3	yes	Containerized architecture based on open-source software	Custom web app based on open-source dashboarding software	yes

Regarding the cloud platform, fewer details are typically available in the literature. Ready-made software platforms are more common in case of research work, probably due to their fastest set-up time, while architectural details for commercial IEQ monitoring systems are not available. The typical exploited dashboard is either a smartphone application or a web application. In all cases, little information regarding the development and optimization of the front-end and of the back-end are reported.

In comparison, the proposed system aligns well with monitoring device indicators, featuring more sensors for indoor air quality than existing systems. Its volume and cost are also within the range of existing solutions, indicating a low-cost design. As reported in Section 3, the most expensive component is the 3D printing of the case, which could be significantly reduced through large-volume production. The intermediate architecture of the cloud platform is a novel aspect, which offers higher flexibility and customization, achievable even by non-IT professionals, compared to existing solutions.

6. Conclusions

A low-cost IoT system for the monitoring of indoor environmental quality is presented, with a focus on the process of data acquisition, processing, and aggregation. Multi-sensor devices were specifically designed to measure various physical quantities related to IEQ. Rather than sending to the cloud server only averages for each monitored parameter, the on-board microcontroller preprocesses the acquired data to extract statistical features from the acquired data series. Regarding the developed data processing layer, it receives MQTT messages from the PROMET&O multi-sensor and stores the contained data inside a MySQL database. The stored information is processed via a Python scheduler, which computes various time aggregations, and can be accessed from extern via web API. Lastly, the data visualization layer serves as the primary access point for users to engage with the information collected by the system. Its design principles prioritize cost-effectiveness, harnessing open-source software and modular architecture to facilitate seamless integration of diverse software components. As a result, the web application responsible for dashboard visualization and user navigation effectively delegates authentication, authorization, and chart creation to third-party components, preserving its functional specificity. Moreover, enhancements to the dashboarding software have refined the chart update mechanism, resulting in notable improvements in user interaction and a significant reduction in waiting times.

Author Contributions: Conceptualization, A.B., V.I.F., P.C., A.S., E.R., G.R.-E., E.G., B.M., A.A. and F.F.; software, A.B., E.R., P.C., A.S. and G.R.-E.; validation, V.I.F., E.R. and A.S.; writing—original draft preparation, A.B., P.C., V.I.F., A.S. and E.R.; writing—review, G.R.-E., E.G., B.M., A.A. and F.F.; supervision, A.S., B.M., A.A. and F.F.; funding acquisition, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the national action program in the framework of the REACT EU Initiative: Programma Operativo Nazionale (PON) Ricerca e Innovazione 2014–2020 REACT EU Percorsi di dottorato su tematiche green e sui temi dell'innovazione D.M. 1061 del 10 August 2021. The research was funded also by Politecnico di Torino, CALOS Department, and the companies Italgas Reti S.p.A. and C2R Energy Consulting S.r.l.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Acknowledgments: The authors would like to thank: (i) Manuela Baracani and Fabio Favoino for their support in the experimental validation of the developed multi-sensor; (ii) Sara Bellatorre, Michele Masiello, Vittorio Arpino, Luca Errani, and Martina Saugo for supporting the development of the data visualization layer; (iii) Thomas Jacques Francisco Osorio for the support in the development of the backend services and Giuseppina Emma Puglisi and Louena Shtrepi for their support during the development of the PROMET&O project; and (v) Alessio Carullo for their support in the metrological characterization of the developed multi-sensor.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Bluyssen, P.M. Towards an integrated analysis of the indoor environmental factors and its effects on occupants. *Intell. Build. Int.* **2020**, *12*, 199–207. [CrossRef]
2. Felgueiras, F.; Mourão, Z.; Moreira, A.; Gabriel, M.F. Indoor environmental quality in offices and risk of health and productivity complaints at work: A literature review. *J. Hazard. Mater. Adv.* **2023**, *10*, 100314. [CrossRef]
3. Azuma, K.; Ikeda, K.; Kagi, N.; Yanagi, U.; Osawa, H. Physicochemical risk factors for building-related symptoms in air-conditioned office buildings: Ambient particles and combined exposure to indoor air pollutants. *Sci. Total Environ.* **2018**, *616–617*, 1649–1655. [CrossRef] [PubMed]
4. Varjo, J.; Hongisto, V.; Haapakangas, A.; Maula, H.; Koskela, H.; Hyönä, J. Simultaneous effects of irrelevant speech, temperature and ventilation rate on performance and satisfaction in open-plan offices. *J. Environ. Psychol.* **2015**, *44*, 16–33. [CrossRef]

5. Schweiker, M.; Ampatzi, E.; Andargie, M.S.; Andersen, R.K.; Azar, E.; Barthelmes, V.M.; Berger, C.; Bourikas, L.; Carlucci, S.; Chinazzo, G.; et al. Review of multi-domain approaches to indoor environmental perception and behaviour. *Build. Environ.* **2020**, *176*, 106804. [CrossRef]
6. Kotzias, D. Built environment and indoor air quality: The case of volatile organic compounds. *AIMS Environ. Sci.* **2021**, *8*, 135–147. [CrossRef]
7. Settimo, G.; Manigrasso, M.; Avino, P. Indoor Air Quality: A Focus on the European Legislation and State-of-the-Art Research in Italy. *Atmosphere* **2020**, *11*, 370. [CrossRef]
8. European Commission. *Indoor Air Pollution: New EU Research Reveals Higher Risks than Previously Thought*; Technical report; European Commission: Brussels, Belgium, 2003.
9. Cincinelli, A.; Martellini, T. Indoor Air Quality and Health. *Int. J. Environ. Res. Public Health* **2017**, *14*, 1286. [CrossRef]
10. García, M.R.; Spinazzé, A.; Branco, P.T.; Borghi, F.; Villena, G.; Cattaneo, A.; Gilio, A.D.; Mihucz, V.G.; Álvarez, E.G.; Lopes, S.I.; et al. Review of low-cost sensors for indoor air quality: Features and applications. *Appl. Spectrosc. Rev.* **2022**, *57*, 747–779. [CrossRef]
11. Dhungana, P.; Chalise, M. Prevalence of sick building syndrome symptoms and its associated factors among bank employees in Pokhara Metropolitan, Nepal. *Indoor Air* **2020**, *30*, 244–250. [CrossRef]
12. Ghaffarianhoseini, A.; AlWaer, H.; Omrany, H.; Ghaffarianhoseini, A.; Alalouch, C.; Clements-Croome, D.; Tookey, J. Sick building syndrome: Are we doing enough? *Archit. Sci. Rev.* **2018**, *61*, 99–121. [CrossRef]
13. Torresin, S.; Aletta, F.; Babich, F.; Bourdeau, E.; Harvie-Clark, J.; Kang, J.; Lavia, L.; Radicchi, A.; Albatici, R. Acoustics for Supportive and Healthy Buildings: Emerging Themes on Indoor Soundscape Research. *Sustainability* **2020**, *12*, 6054. [CrossRef]
14. Al horr, Y.; Arif, M.; Katafygiotou, M.; Mazroei, A.; Kaushik, A.; Elsarrag, E. Impact of indoor environmental quality on occupant well-being and comfort: A review of the literature. *Int. J. Sustain. Built Environ.* **2016**, *5*, 1–11. [CrossRef]
15. González-Martín, J.; Kraakman, N.J.R.; Pérez, C.; Lebrero, R.; Muñoz, R. A state-of-the-art review on indoor air pollution and strategies for indoor air pollution control. *Chemosphere* **2021**, *262*, 128376. [CrossRef] [PubMed]
16. Fissore, V.I.; Fasano, S.; Puglisi, G.E.; Shtrepi, L.; Astolfi, A. Indoor Environmental Quality and Comfort in Offices: A Review. *Buildings* **2023**, *13*, 2490. [CrossRef]
17. Roskams, M.J.; Haynes, B.P. Testing the relationship between objective indoor environment quality and subjective experiences of comfort. *Build. Res. Inf.* **2021**, *49*, 387–398. [CrossRef]
18. Saini, J.; Dutta, M.; Marques, G. Sensors for indoor air quality monitoring and assessment through Internet of Things: A systematic review. *Environ. Monit. Assess.* **2021**, *193*, 66. [CrossRef] [PubMed]
19. Hill, A.P.; Prince, P.; Snaddon, J.L.; Doncaster, C.P.; Rogers, A. AudioMoth: A Low-Cost Acoustic Device for Monitoring Biodiversity and the Environment. *HardwareX* **2019**, *6*, e00073. [CrossRef]
20. Mylonas, G.; Fraile, L.P.; Tsampas, S.; Kalogeras, A. A Study on Indoor Noise Levels in a Set of School Buildings in Greece Utilizing an IoT Infrastructure. *arXiv* **2023**, arXiv:2309.02797. [CrossRef]
21. Amaxilatis, D.; Akrivopoulos, O.; Mylonas, G.; Chatzigiannakis, I. An IoT-Based Solution for Monitoring a Fleet of Educational Buildings Focusing on Energy Efficiency. *Sensors* **2017**, *17*, 2296. [CrossRef]
22. Camprodon, G.; González, O.; Barberán, V.; Pérez, M.; Smári, V.; Heras, M.A.d.; Bizzotto, A. Smart Citizen Kit and Station: An open environmental monitoring system for citizen participation and scientific experimentation. *HardwareX* **2019**, *6*, e00070. [CrossRef]
23. Parkinson, T.; Parkinson, A.; de Dear, R. Continuous IEQ monitoring system: Context and development. *Build. Environ.* **2019**, *149*, 15–25. [CrossRef]
24. Tiele, A.; Esfahani, S.; Covington, J. Design and development of a low-cost, portable monitoring device for indoor environment quality. *J. Sens.* **2018**, *2018*. [CrossRef]
25. Geng, Y.; Zhang, Z.; Yu, J.; Chen, H.; Zhou, H.; Lin, B.; Zhuang, W. An Intelligent IEQ Monitoring and Feedback System: Development and Applications. *Engineering* **2022**, *18*, 218–231. [CrossRef]
26. Carre, A.; Williamson, T. Design and validation of a low cost indoor environment quality data logger. *Energy Build.* **2018**, *158*, 1751–1761. [CrossRef]
27. Mujan, I.; Licina, D.; Kljajić, M.; Čulić, A.; Anđelković, A.S. Development of indoor environmental quality index using a low-cost monitoring platform. *J. Clean. Prod.* **2021**, *312*, 127846. [CrossRef]
28. Kim, M.; Kim, T.; Park, S.; Lee, K. An Indoor Multi-Environment Sensor System Based on Intelligent Edge Computing. *Electronics* **2023**, *12*, 137. [CrossRef]
29. Kosmopoulos, G.; Salamalikis, V.; Wilbert, S.; Zarzalejo, L.F.; Hanrieder, N.; Karatzas, S.; Kazantzidis, A. Investigating the Sensitivity of Low-Cost Sensors in Measuring Particle Number Concentrations across Diverse Atmospheric Conditions in Greece and Spain. *Sensors* **2023**, *23*, 6541. [CrossRef] [PubMed]
30. Astolfi, A.; Carullo, A.; Fissore, V.; Puglisi, G.E.; Arcamone, G.; Shtrepi, L.; Raviola, E.; Barbaro, A.; Espinosa, G.R.; Chiavassa, P.; et al. Development and Metrological Characterization of a Multi-sensor Device for Indoor Environmental Quality (IEQ) Monitoring. In Proceedings of the 2023 IEEE International Workshop on Metrology for Living Environment (MetroLivEnv), Milano, Italy, 29–31 May 2023; pp. 179–184. [CrossRef]
31. Yasin, A.; Delaney, J.; Cheng, C.T.; Pang, T.Y. The Design and Implementation of an IoT Sensor-Based Indoor Air Quality Monitoring System Using Off-the-Shelf Devices. *Appl. Sci.* **2022**, *12*, 9450. [CrossRef]

32. Chiesa, G.; Avignone, A.; Carluccio, T. A Low-Cost Monitoring Platform and Visual Interface to Analyse Thermal Comfort in Smart Building Applications Using a Citizen-Scientist Strategy. *Energies* **2022**, *15*, 564. [CrossRef]
33. Air Care 2.0. Available online: https://www.aircare.it/en/aircare-2-0/ (accessed on 30 April 2024).
34. Promet&o Software Repository. Available online: https://github.com/servetti-polito/prometeo-iot-monitoring-platform (accessed on 22 April 2024).
35. Grafana—The Open Platform for Analytics and Monitoring. Available online: https://grafana.com (accessed on 2 February 2024).
36. Keycloak—Open Source Identity and Access Management. Available online: https://www.keycloak.org/ (accessed on 2 February 2024).
37. React—The Library for Web and Native User Interfaces. Available online: https://react.dev/ (accessed on 2 February 2024).
38. SHT41-Sensirion. Available online: https://sensirion.com/products/catalog/SHT41/ (accessed on 18 April 2024).
39. VEML7700-Vishay. Available online: https://www.vishay.com/docs/84286/veml7700.pdf (accessed on 18 April 2024).
40. IMP34DT05-STMicroelectronics. Available online: https://www.st.com/en/mems-and-sensors/imp34dt05.html (accessed on 18 April 2024).
41. SFA30-Sensirion. Available online: https://sensirion.com/products/catalog/SFA30/ (accessed on 18 April 2024).
42. MICS_VZ_89TE-Amphenol. Available online: https://www.amphenol-sensors.com/en/telaire/524-voc/3285-mics-vz-89te (accessed on 18 April 2024).
43. SEN54-Sensirion. Available online: https://www.sensirion.com/products/catalog/SEN54/ (accessed on 18 April 2024).
44. 3SP_NO2_5F-Spec Sensors. Available online: https://www.spec-sensors.com/product/no2-nitrogen-dioxide/ (accessed on 18 April 2024).
45. SCD30-Sensirion. Available online: https://www.sensirion.com/products/catalog/SCD30/ (accessed on 18 April 2024).
46. 3SP_CO_1000-Spec Sensors. Available online: https://www.spec-sensors.com/product/co-carbon-monoxide/ (accessed on 18 April 2024).
47. *ANSI/ASA S1.42*; Design Response of Weighting Networks for Acoustical Measurements. Acoustical Society of America: Melville, NY, USA, 2020.
48. Rimell, A.N.; Mansfield, N.J.; Paddan, G.S. Design of Digital Filters for Frequency Weightings (A and C) Required for Risk Assessments of Workers Exposed to Noise. *Ind. Health* **2015**, *53*, 21. [CrossRef]
49. Parks, T.W.; Burrus, C.S. *Digital Filter Design*; John Wiley & Sons: Hoboken, NJ, USA, 1987.
50. Bruek & Kjaer. *Sound Source HP 1001*. Available online: https://www.bksv.com/media/doc/bp0264.pdf (accessed on 30 April 2024).
51. Docker: Accelerated Container Application Development. Available online: https://www.docker.com/ (accessed on 2 February 2024).
52. NGINX: Advanced Load Balancer, Web Server, & Reverse Proxy. Available online: https://www.nginx.com/ (accessed on 2 February 2024).
53. Eclipse Mosquitto: An Open Source MQTT Broker. Available online: https://mosquitto.org/ (accessed on 7 March 2024).
54. Eclipse Paho: MQTT Python Client Library. Available online: https://pypi.org/project/paho-mqtt/ (accessed on 7 March 2024).
55. MySQL. Available online: https://www.mysql.com/ (accessed on 8 March 2024).
56. Timescale: PostgreSQL ++ for Time Series and Events. Available online: https://www.timescale.com/ (accessed on 8 March 2024).
57. InfluxDB. It's about Time. Available online: https://www.influxdata.com/ (accessed on 8 March 2024).
58. APScheduler: Advanced Python Scheduler. Available online: https://apscheduler.readthedocs.io (accessed on 11 March 2024).
59. Flask. Available online: https://flask.palletsprojects.com (accessed on 11 March 2024).
60. E+E. EE160 Relative Humidity Sensor. Available online: https://www.epluse.com/products/humidity-instruments/humidity-transmitters-for-hvac/ee160/ (accessed on 30 April 2024).
61. Senseca. Illuminance Probe LPPHOT01. Available online: https://environmental.senseca.com/product/lpphot01-illuminance-lux-probe/ (accessed on 30 April 2024).
62. NTI Audio. XL2 Audio and Acoustic Analyzer. Available online: https://www.nti-audio.com/en/products/sound-level-meters/xl2-audio-acoustic-analyzer1 (accessed on 30 April 2024).
63. Senseca. HD37VBT Carbon Dioxide Logger. Available online: https://environmental.senseca.com/product/hd37-co2-co2-and-temperature-transmitters/ (accessed on 30 April 2024).
64. Air Coach Pro. Available online: https://www.nateosante.com/en/indoor-air-quality-monitor/air-coach/ (accessed on 30 April 2024).