SYSTEM FOR MONITORING OF ENVIRONMENTAL CONDITIONS IN LABORATORIES

Miroslav Behul[1], Marian Vojs[1], Štefan Čerba[1]

[1]Slovak University of Technology in Bratislava, Faculty of Electrical Engineering and Information Technology, Ilkovičova 3, 812 19 Bratislava, Slovak Republic, E-mail: miroslav.behul@stuba.sk

Abstract
This paper presents the design, implementation and verification of a compact Internet of Things (IoT) system oriented for monitoring of environmental conditions in laboratories. The developed device enables continuous measurement of temperature, relative humidity, atmospheric pressure, volatile organic compounds (VOC), equivalent carbon dioxide concentration (CO_2eq), and indoor air quality index (IAQ). The system is based on the ESP32 microcontroller, includes an integrated web server and provides MQTT and HTTP communication for remote monitoring and controll. The proposed solution is suitable for educational purposes, laboratories, research facilities, and indoor environmental monitoring applications where low-cost and real-time monitoring are required.

Introduction

Environmental conditions in laboratories significantly influence the reliability of experiments, safety of personnel, and operation of sensitive equipment. Temperature, humidity, and air quality directly affect the accuracy of the measurement and stability of electronic systems, chemical processes and biological samples. Continuous monitoring of environmental conditions has therefore become an important requirement in modern laboratories. Commercial monitoring systems are often expensive and difficult to integrate into educational or small-scale laboratory environments. Recent advances in IoT technologies enable the development of low-cost monitoring devices with wireless communication and cloud connectivity [1][2]. Microcontroller platforms such as ESP32 provide sufficient computational performance, integrated WiFi communication, and compatibility with a wide range of environmental sensors [3]. Modern environmental monitoring systems are increasingly based on distributed IoT architectures that combine embedded sensing devices, wireless communication, cloud services, and web-based visualization platforms. Such systems enable continuous acquisition, storage, and remote analysis of environmental data while maintaining low hardware complexity and low deployment cost. Open-source technologies and embedded web servers additionally simplify system configuration, remote accessibility, and integration into educational and research environments[1][2]. This paper presents the design and implementation of a laboratory environmental monitoring system based on the ESP32 microcontroller and the BME680 environmental sensor. The proposed system focuses on reliable environmental data acquisition, simultaneous HTTP and MQTT communication, local and remote visualization, and low-power autonomous operation suitable for long-term laboratory deployment.

System Architecture

The developed environmental monitoring system is based on the wireless sensing node (GaSens) and on the cloud platform for storage and visualization of the measured data. The GaSens node integrates sensing, processing, communication, configuration storage, and embedded web-based user interface available via WiFi (Fig. 1). The GaSens node is based on the ESP32 microcontroller platform with integrated serial and wireless communication. The hardware architecture of the GaSens node illustrated in Fig. 2 combines the ESP32 microcontroller, BME680 environmental sensor, EEPROM embedded memory, local user interface, and wireless communication. The device includes status LEDs and control buttons for user interaction. A red LED indicates power connection and normal operation, while a blue LED indicates active operation of the local user interface and WiFi hotspot. During low-power mode in which the local user interface and hotspot are disabled and the data are measured and transmitted to the remote server only during timer interruptions, the blue LED is switched off. The button marked by letter A is intended for toggling between low power mode and the active mode when the embedded web-based user interface is accessible via WiFi. The button marked by letter B is intended for resetting the device. The device is powered through a standard micro-USB interface with a supply voltage of 5 V. Micro-USB connector serves also for serial communication. The BME 680 environmental sensor from Bosh company continuously measures environmental parameters and communicates with the microcontroller via I_2C serial bus [4]. After first startup, the device automatically runs the calibration, and the functionalities are not available until the device is fully calibrated. After calibration, the system creates the wireless access point identified by the SSID “GaSensID”. Users can connect directly to the device and configure operational parameters through a standard web browser. The configuration interface is accessible through the IP address 192.168.1.1 or via the hostname gasens.local. After entering local WiFi credentials, the device connects to the selected network and receives an IP address from the router. The local user interface is accessible simultaneously in both Access Point and Station WiFi mode. This approach simplifies deployment because no additional software installation is required. The monitoring node stores WiFi credentials, MQTT settings, calibration states, temperature correction values, and measurement intervals inside EEPROM memory. Measured environmental data are transmitted wirelessly through WiFi communication using both HTTP REST requests and MQTT telemetry transfer [6]. The hybrid communication architecture increases interoperability and enables integration with dedicated cloud services as well as universal IoT platforms.

Fig. 1 Prototype of the GaSens device for monitoring of environmental parameters.

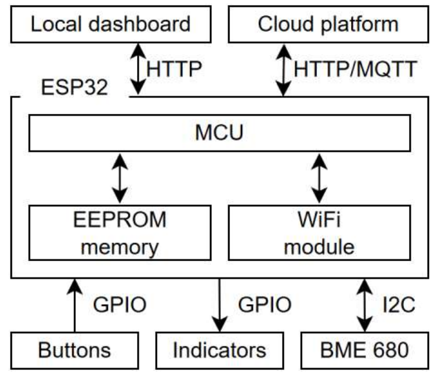
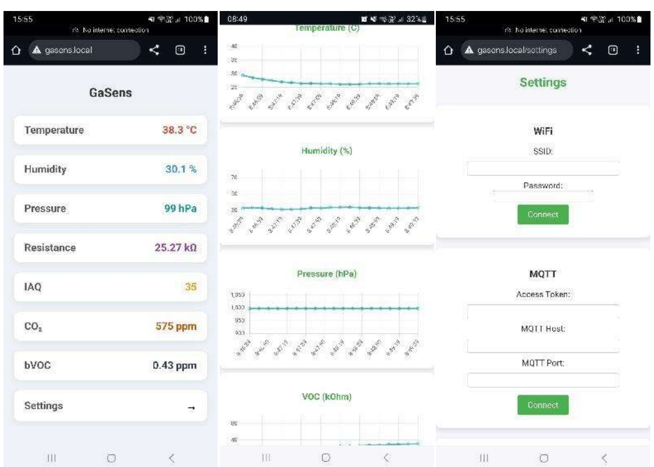


Fig. 2 Architecture of the environmental monitoring system.

Firmware Design

The firmware of the monitoring node (Fig. **3**) was developed using the Arduino framework, with the backend implemented in C/C++ and the frontend implemented in HTML, CSS, and JavaScript. The firmware architecture of the developed system is illustrated in Fig. **4**. The software architecture integrates WiFi and serial communication, an embedded web server with web-based user interface, EEPROM-based configuration storage, MQTT and HTTP communication, and BSEC sensor algorithms. The firmware supports both Access Point and Station WiFi modes. During initial configuration, the device creates its own wireless hotspot allowing direct user access through a web browser. After configuration, the system reconnects to the selected wireless network and periodically transmits measured data to external IoT platforms.



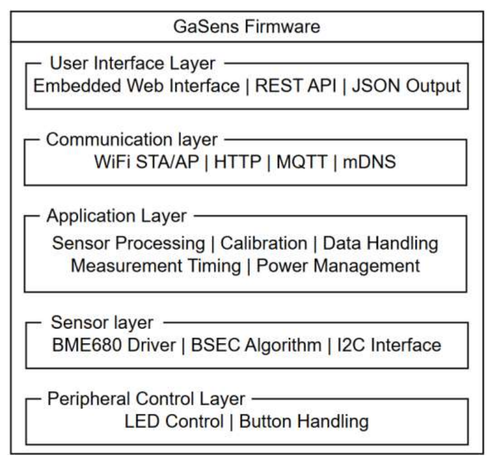


Fig. 3 Web-based user interface embedded in the GaSens device.

Fig. 4 Layered firmware architecture of the GaSens device.

The firmware running on the ESP32 microcontroller is responsible for environmental data acquisition, sensor calibration, communication management, local web-based user interface, and low-power operation. Environmental data including temperature, humidity, pressure, gas resistance, IAQ, CO_2 equivalent concentration, and breath VOC equivalent concentration are dynamically visualized on the integrated web interface using JavaScript-based updates. The embedded web server enables real-time visualization of measured parameters directly in a web browser. The interface is compatible with desktop computers as well as mobile devices. The firmware also provides diagnostic communication through the serial monitor, enabling real-time monitoring of system status, sensor initialization, network connectivity, calibration processes, and measured environmental parameters during operation and debugging. Configuration parameters such as WiFi credentials, MQTT settings, temperature correction, and measurement intervals are stored in EEPROM memory to ensure persistence after restart of the device. An important part of the firmware is the implementation of low-power operation. The ESP32 platform utilizes deep sleep functionality together with timer-based and external interrupt wake-up mechanisms. In low-power mode, the local web interface and WiFi hotspot are disabled while periodic environmental measurements and remote data transmission remain active. This functionality significantly reduces power consumption and enables long-term operation when the device is supplied from battery. The developed firmware also incorporates automatic calibration of the BSEC environmental sensing algorithm [7]. Calibration data are stored in EEPROM memory after achieving stable IAQ accuracy values. During system startup, the stored calibration state is restored, reducing stabilization time and improving long-term measurement reliability. The firmware performs sensor data acquisition, wireless communication, web visualization, HTTP data transmission, and MQTT-based remote telemetry transfer. The firmware additionally implements a REST API for local access to measured parameters in JSON format [3][5].

Communication and Visualization

The environmental monitoring system (Fig. 5) includes a custom cloud visualization platform designed for data storage, processing, and for graphical representation of measured environmental parameters. The platform was developed in the Visual Studio Code framework, with the backend implemented in PHP and the frontend implemented in HTML, CSS, and JavaScript. The visualization interface communicates with the backend through a REST API. The architecture of the developed cloud visualization system is illustrated in Fig. 6. The GaSens device periodically transmits measured data in JSON format using HTTP POST requests to dedicated REST API endpoints. The backend application stores received data into SQL database tables corresponding to individual sensing nodes. The visualization platform supports simultaneous operation of multiple GaSens devices. Separate REST API routes and database tables are assigned to individual sensor nodes, enabling simultaneous deployment in laboratory environments. The server-side application processes incoming environmental data, validates JSON structures, stores measured values into SQL databases, and provides historical data retrieval services for the frontend dashboard. REST API endpoints are implemented separately for individual sensing nodes, enabling independent management of multiple environmental monitoring devices. Interactive graphical visualization of measured environmental parameters was implemented using the Chart.js library [8].

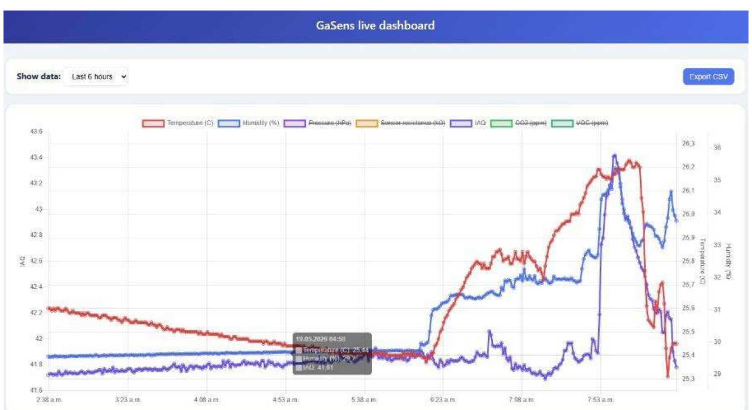


Fig. 5 Cloud visualization interface developed for the GaSens devices.

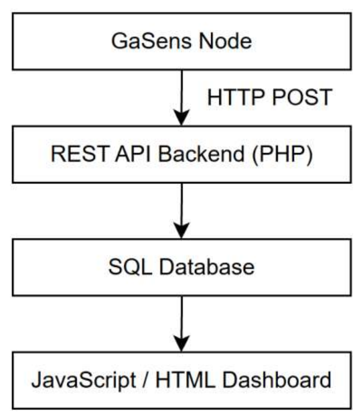


Fig. 6 Block diagram of the cloud visualization platform.

The dashboard allows interactive visualization of temperature, humidity, pressure, gas resistance, IAQ, CO_2 equivalent concentration, and VOC concentration. Users can dynamically select different time intervals including six-hour, daily, and weekly views. The platform also supports CSV export functionality for further offline processing and statistical analysis. Measured environmental parameters are visualized using interactive charts with multiple independent vertical axes. Individual measured environmental parameters can be selectively enabled or disabled depending on user requirements. The responsive frontend design allows access from desktop computers, tablets, and smartphones connected to the Internet. The monitoring system additionally supports MQTT-based communication for integration with external IoT visualization platforms and industrial monitoring infrastructures [1][5]. After MQTT configuration through the embedded web user interface, measured environmental data are simultaneously transmitted using both HTTP REST communication and MQTT messaging protocols. GaSens node also supports device control and configuration via MQTT commands. The hybrid communication architecture integrating dedicated and external cloud platform is illustrated in Fig. 7 and Fig. 8. The external visualization platform enables real-time monitoring, historical data analysis, remote laboratory supervision, and long-term environmental evaluation. Experimental testing of the developed monitoring system was performed over several months under different environmental conditions and operational configurations. Stable sensor operation together with reliable HTTP and MQTT communication was confirmed during long-term operation.

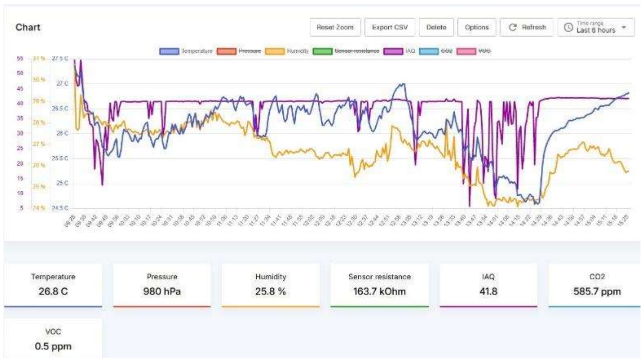


Fig. 7 MQTT based data transmission to the IoT dashboard.

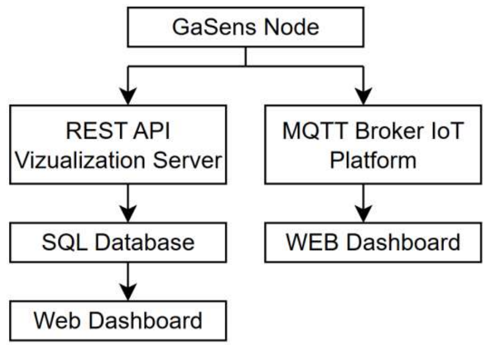


Fig. 8 Hybrid communication architecture integrating dedicated and external cloud platform.

Conclusion

The presented GaSens system represents a compact and scalable solution for monitoring of environmental conditions in laboratories. The developed platform integrates an ESP32 microcontroller with the BME680 environmental sensor and combines local visualization, wireless communication, cloud data transfer, and remote visualization into a unified monitoring system. The proposed architecture enables simultaneous operation in local Wi-Fi mode and infrastructure network mode while supporting both HTTP and MQTT communication. The firmware architecture ensures autonomous sensor calibration, periodic environmental measurements, EEPROM-based configuration and calibration storage, and low-power operation using deep sleep functionality. Experimental operation confirmed stable acquisition of temperature, humidity, pressure, gas resistance, IAQ, CO_2 equivalent, and VOC equivalent parameters. The web-based visualization platform provides real-time graphical visualization, historical data access, and CSV export functionality, which simplifies long-term laboratory monitoring and analysis. The proposed system is suitable for laboratories, indoor air-quality monitoring, IoT research applications and for educational purposes. After minor modifications in the cloud software and after medium modifications in the firmware and casing the monitoring platform is also possible to use with another sensors.

Acknowledgement

This work was funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the projects No. 09I03–03-V04–00154 and No. 09I04-03-V02-00033.

References

[1] Boyes H., Hallaq B., Cunningham J., Watson T.: Comput. Ind. 101, 1 (2018).
[2] Gubbi J., Buyya R., Marusic S., Palaniswami M.: Future Gener. Comput. Syst. 29, 1645 (2013).
[3] Espressif Systems: ESP32 Series Datasheet (2024).
[4] Bosch Sensortec: BME680 Environmental Sensor – Datasheet (2024).
[5] Minoli D., Sohraby K., Occhiogrosso B.: IEEE Internet Things J. 4, 269 (2017).
[6] MQTT.org: MQTT Version 3.1.1 Protocol Specification. OASIS Standard (2014).
[7] Bosch Sensortec: BSEC Software Library Integration Guide. Bosch Sensortec GmbH (2024).
[8] Chart.js: Simple yet flexible JavaScript charting library for the web. Available: https://www.chartjs.org (2026).