

Article

Design and Implementation of a Low-Cost and Low-Power Wireless Sensor Network for Estuary Environmental Monitoring

Tiago Matos ¹, Marcos F. Martins ^{2,3}, João Rocha ^{2,3}, Hugo Dinis ^{2,3}, Marcos Martins ¹, Sérgio F. Lopes ^{2,3}, José A. Afonso ^{2,3,*} and Luís Gonçalves ^{2,3}

¹ INESC TEC, Faculdade de Engenharia da Universidade do Porto, 4200-465 Porto, Portugal; tiago.a.matos@inesctec.pt (T.M.); marcos.martins@inesctec.pt (M.M.)

² Center for Microelectromechanical Systems (CMEMS), University of Minho, Campus of Azurem, 4804-533 Guimarães, Portugal; b14018@cmems.uminho.pt (M.F.M.); id10563@alunos.uminho.pt (J.R.); hdinis@dei.uminho.pt (H.D.); sergio.lopes@dei.uminho.pt (S.F.L.); lgoncalves@dei.uminho.pt (L.G.)

³ LABBELS—Associate Laboratory, University of Minho, Campus of Gualtar, 4710-057 Braga, Portugal

* Correspondence: jose.afonso@dei.uminho.pt

Abstract

Wireless sensor networks have become essential tools for environmental monitoring, enabling distributed data acquisition in remote and dynamic environments. However, challenges related to communication reliability, energy efficiency, synchronization, and real-time data availability remain critical for long-term deployments. This work presents the design, implementation, and validation of a synchronous wireless sensor network tailored for environmental monitoring in the challenging conditions of an estuarine setting. The proposed architecture is based on Digi XBee SX 868 RF modules operating in DigiMesh mode with synchronized cyclic sleep, enabling coordinated measurements and low-power operation. The network comprises monitoring, repeater, and coordinator/gateway nodes, integrated with a web-based platform for real-time data visualization and management. A custom message exchange format was developed to support seamless transmission of monitoring information from sensors to the web server through the wireless mesh infrastructure. Field experiments were conducted to evaluate network coverage, synchronization performance, communication reliability, and energy consumption. The results demonstrated successful multi-hop communication over the estuarine area, stable synchronization among distributed nodes over extended periods, and energy savings through synchronized sleep operation. The developed web platform enabled reliable real-time data access and network management. The proposed system demonstrates the feasibility of deploying scalable, energy-efficient, and synchronized wireless sensor networks for long-term environmental monitoring in estuarine environments.

Academic Editor: Alexandra Bousia

Received: 6 June 2026

Revised: 9 July 2026

Accepted: 13 July 2026

Published: 17 July 2026

Keywords: wireless sensor networks; environmental monitoring; synchronous communications; DigiMesh; web-based platform; internet of things; real-time monitoring

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article

distributed under the terms and

conditions of the [Creative Commons](#)

[Attribution \(CC BY\)](#) license.

1. Introduction

Wireless sensor networks (WSNs), supported by the increasing emergence of the Internet of Things (IoT), have developed rapidly in various sectors and become fundamental across different applications, such as environmental monitoring [1,2], industrial auto-

mation [3–5], precision agriculture [6], health care [7], and smart cities [8,9]. The demand for continuous acquisition of data from multiple types of sensors, particularly in emerging fields with challenging requirements and complexity, is therefore growing rapidly. WSNs rely on multiple distributed nodes to collect and transmit data, often in environments where wired communication is impractical or infeasible. These nodes typically consist of sensors, embedded processors, wireless transceivers, and batteries, allowing autonomous operation and communication without the need for human intervention.

As WSNs become increasingly integrated into IoT infrastructures, they face a range of new challenges, including network interoperability, dynamic scalability, and secure data exchange. Ensuring reliable and real-time communication among heterogeneous devices requires flexible architecture and robust protocol designs. To address these demands, dynamic routing protocols have been developed to reduce control overhead and adapt to changing network conditions, thereby enhancing overall efficiency [10–12]. Clustering-based architectures have shown significant benefits in improving energy efficiency and network longevity [13]. Similarly, advanced spatial node deployment strategies, such as resultant-vector and heap-based algorithms, can optimize the placement of network devices to minimize hardware redundancy while maintaining full sensor coverage [14]. Data aggregation and compression techniques are used to reduce the volume of transmitted data and thereby lower the energy consumed during communication [15,16]. These methods not only help conserve energy but also improve bandwidth utilization [17]. By minimizing redundant transmissions and enabling localized data processing, data aggregation plays a crucial role in supporting the scalability of large-scale WSN deployments.

Ensuring data privacy and preventing unauthorized access has also emerged as a key concern [18,19]. Given the limited computational resources of sensor nodes, security solutions must be both effective and lightweight. End-to-end encryption and authentication protocols are used to protect data confidentiality and verify the identity of communicating devices [20]. To further strengthen security, privacy-preserving data aggregation techniques, including homomorphic encryption, allow encrypted data to be processed without revealing individual values. Additionally, machine learning-enhanced intrusion detection systems have gained attention for their ability to detect and respond to anomalies in real time [21,22].

Since sensor nodes are often deployed in remote or inaccessible locations and rely on battery power, energy optimization is also an important aspect of WSN design. It is essential for maintaining efficient performance while extending network lifespan and ensuring reliable data transmission. Several techniques have been developed to address this challenge. One of the most effective is sleep scheduling, which enables nodes to alternate between active and low-power sleep states, significantly reducing energy consumption during idle periods [23,24]. Protocols such as S-MAC (with a static duty cycle) and T-MAC (with a dynamic duty cycle) have been proposed to efficiently manage these cycles [25]. Other approaches rely on optimizing the hardware end with more energy-efficient RF transceivers [26].

Synchronization plays a pivotal role in coordinating node activity across the network. In many monitoring applications, especially those involving spatiotemporal dynamics, synchronized measurements are essential for achieving temporal accuracy and data coherence [27]. Synchronous WSNs provide several advantages by aligning sleep–wake schedules and data acquisition windows among all nodes. This coordination not only enhances energy efficiency and data consistency but also reduces communication delays and packet collisions, leading to improved network robustness and scalability. In addition to protocol-level optimization, incorporating energy harvesting mechanisms and using low-power hardware components further supports the limited energy budgets of sensor nodes, ultimately extending the operational lifetime of the network [28,29].

The data gathered by WSNs needs to be made easily available to end users, with the IoT paradigm playing a key role in this aspect, allowing sensor data to be stored in a central database and processed according to the application needs. For environmental monitoring applications, making the data available in the form of charts enables end users to make proper analysis and field study research. Off-the-shelf cloud services offer rapid deployment and low initial costs, but introduce vendor dependency, inflexibility to make customizations, and long-term costs as the system scales up. Pre-built data storage and visualization options impose a web application programming interface (API) and rarely match the needs of specific systems, preventing or augmenting the costs to achieve more adequate solutions. Therefore, custom web services and applications are frequently developed to handle specific requirements, like sensor network structure and dynamics, configuration flexibility, hardware failures, and so forth.

While the challenges discussed above are common to most WSN deployments, their practical implementation is inherently application-specific. Each deployment scenario is shaped by its own set of environmental conditions, monitoring objectives, communication constraints, and infrastructure availability. Consequently, the architecture, hardware selection, and communication strategies must be carefully adapted to the unique requirements of the target environment.

Several studies have explored the application of WSNs for environmental and marine monitoring. Xu et al. presented a comprehensive survey of WSN applications in marine environments, highlighting the advantages of wireless monitoring systems for real-time and low-cost oceanographic observations, while also identifying challenges related to energy management, communication reliability, and environmental harshness [30]. Pozzebon et al. developed a Zigbee-based WSN framework for real-time monitoring of sandy beaches and dunes, integrating distributed sensing nodes with a Global System for Mobile Communications (GSM) gateway and cloud infrastructure for remote data visualization [30]. Hidayat et al. proposed a coastal monitoring buoy system using Zigbee communication to collect and transmit environmental data in real time [31]. More recently, Errachdi et al. reviewed the opportunities and challenges associated with IoT and WSN technologies for marine environmental monitoring, emphasizing the importance of communication reliability, energy autonomy, and robust system integration in marine deployments [32].

Different wireless mesh technologies have been explored for WSN implementation, each presenting distinct advantages and limitations. Zigbee remains one of the most widely adopted standards due to its low-power operation and mesh networking capabilities, particularly in short-range monitoring systems. Bluetooth Mesh and Bluetooth Low Energy (BLE) mesh solutions have also gained attention for low-power IoT applications, especially in dense short-range deployments, offering reduced energy consumption and improved interoperability with consumer devices [33]. However, although multi-hop mesh topologies may extend the network coverage compared to single-hop star and peer-to-peer topologies, Zigbee and Bluetooth/BLE are short-range wireless network technologies that are not adequate for long-range ocean environmental monitoring requirements. On the other hand, low-power wide-area network (LPWAN) technologies such as LoRa have demonstrated strong potential for long-range environmental monitoring [34,35]. However, many of these technologies present limitations regarding synchronization support, scalability under sleeping mesh topologies, or communication robustness in dynamic environments.

The main contributions of this paper lie in the problem definition, system design, implementation, and experimental evaluation of a wireless sensor network and respective web server infrastructure, all specifically tailored for environmental monitoring in an estuarine setting. The proposed network enables multiple sensor nodes to operate in syn-

chronization, facilitating coordinated data acquisition and transmission to a central controller, which subsequently forwards the information in real time to a web server for online access. The proposed web server provides a customized storage web service and a web application for sensor network configuration and data visualization, which supports a very high level of variability with fault tolerance. This work details the technical rationale and design choices made throughout the development process, offering practical insights for engineers, researchers, and practitioners into the solutions and challenges associated with deploying WSNs in dynamic and environmentally demanding contexts. A comparison of the main characteristics of the aforementioned marine environment networks and our proposed network is presented in Table 1.

Table 1. Main characteristics of the investigated marine environment networks.

Reference	Application Domain	Communication Technology	Multi-Hop Capability	Synchronization Support	Energy-Saving Strategy	Power Source	Cost Considerations	Custom Hardware	Real-Time Monitoring	Web Visualization
[31] Pozzebon et al. (2018)	Beach and dune	IEEE 802.15.4/ZigBee	Yes	No	Yes (sleep mode)	AA batteries	Yes	Yes	Yes	Yes
[34] Su et al. (2019)	Environmental (unspecific)	LoRa	No (LoRa star topology)	No	No	Electrical grid	No	Yes	Yes	Yes (Microsoft Azure)
[35] Ali & Latiff (2019)	Island	LoRa	No	No	No	AA batteries/electrical grid	No	Yes	Yes	No
This work	Estuarine	DigiMesh (868 MHz)	Yes	Yes	Yes	LiPo batteries	Yes	Yes	Yes	Yes

This paper is organized as follows: Section 2 presents the problem definition and the requirements of the proposed estuarine monitoring system. Section 3 describes the overall system architecture, including the wireless sensor network and its main components. Section 4 details the hardware and firmware implementation, covering the developed node architectures, firmware design, network configuration, and message exchange format. Section 5 presents the web server and application infrastructure responsible for data storage, management, and visualization. Section 6 discusses the results obtained during field deployment, including wireless communication performance, synchronized network operation, energy efficiency, and data visualization capabilities. Finally, Section 7 summarizes the main conclusions and outlines future research directions.

2. Problem Definition

Estuary monitoring is important for understanding and managing some of the most productive and vulnerable ecosystems on Earth. Estuaries act as transition zones between rivers and the sea, supporting high biodiversity, nursery habitats for fish and shellfish, and key ecosystem services such as water filtration, nutrient cycling, and coastal protection. However, they are also highly sensitive to human pressures, including urban development, agricultural runoff, industrial pollution, dredging, and the impacts of climate change, such as sea-level rise and salinity shifts. Continuous monitoring allows for the detection of changes in water quality, sediment dynamics, and ecological health over time, enabling early identification of environmental degradation and supporting evidence-based management decisions. This information is essential for balancing economic use of estuarine areas with conservation goals and for ensuring the long-term resilience of these critical coastal systems.

The successful implementation of a single monitoring station in a previous project [36] led to an increased interest in monitoring the estuary of the Cávado River. Enabling a comprehensive monitoring of the area by adding more monitoring locations intercon-

nected through a WSN holds the potential to enhance the understanding of watershed dynamics. The primary objective of the proposed monitoring system is to acquire and make available the sensor readings of different variables at different locations and depths across an estuary. The estuary area of interest is shown in Figure 1 as a shade of blue. The variables of interest for this application area include, for example, water temperature, depth, sediment concentration, current velocity, salinity, and conductivity. Each monitoring location is represented by a white circle in Figure 1, where there is a node of a WSN.

Formally, a wireless sensor network consists of N monitoring nodes:

$$Network = \{node_1, node_2, \dots, node_N\}. \quad (1)$$

Each monitoring node is georeferenced by its geographical coordinates and can host multiple sensors:

$$node_i = \{lat_i, lon_i, S_i\}, \quad (2)$$

where

$$S_i = \{sensor_1, sensor_2, \dots, sensor_s\} \quad (3)$$

is the set of sensors installed at node i .

One design restriction for this work was to reuse the docking infrastructure from the previous project [36]. That infrastructure consists of a floating platform at the top (water surface) attached to a ballast at the bottom (riverbed/seabed) by a series of pipe and chain sections. Through these pipes and chains runs an electrical cable that enables the connection of underwater sensor devices at different water column depths. This is shown in the zoomed-in view of a node in Figure 1.

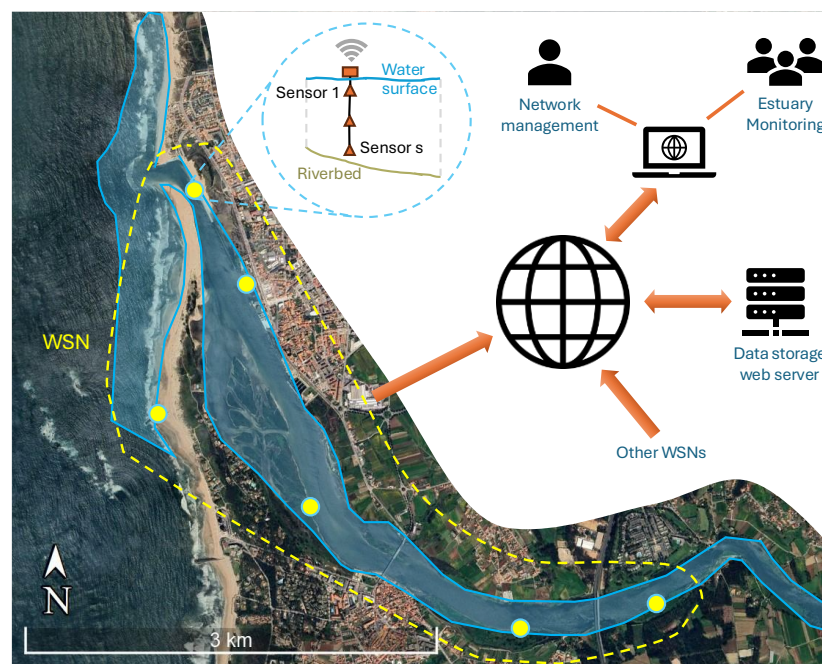


Figure 1. Required estuary monitoring.

Each estuary network sends its data periodically to a central server for storage. Nodes can be added and removed at any time for maintenance, extending the network, or introducing new monitoring locations of interest. Each node can have a different number of sensors, and each sensor can read different variables. While more than a dozen kinds of variables are predefined, the system shall support the addition of new kinds at any time. The system shall also include mobile nodes (with variable geolocation) to be used by un-

manned water surface platforms, such as drifters or buoys, or monitoring campaigns with vessels.

Formally, each sensor periodically acquires one or more environmental variables, producing a measurement record:

$$\text{sensor}_j(k) = \{t_k, v_1, v_2, \dots, v_m\} \quad (4)$$

where t_k is the acquisition timestamp and $v_1, \dots, v_m$ are the measured values of the variables.

The readings from the multiple variables must be synchronized to enable proper analysis of the watershed dynamics. The synchronized operation of the network provides several advantages for environmental monitoring applications. First, it enables temporally aligned measurements among geographically distributed monitoring nodes, which is particularly important for studying dynamic estuarine processes influenced by tides, salinity gradients, and hydrodynamic variability. Second, synchronized communication windows reduce random asynchronous transmissions, minimizing packet collisions and improving network efficiency. Finally, coordinated sleep scheduling significantly decreases idle listening periods, which are typically one of the major contributors to energy consumption in WSNs.

The system shall support the monitoring of numerous marine areas, and the interface shall be a web application providing access to all acquired data in a georeferenced way. Variables' time series shall be presented as graphs and be available for download in a widely used file format for further analysis.

Formally, the data acquired by the server is structured as an historical time-series for each network of E-monitored estuaries:

$$\text{Database} = \{\text{network}_1, \text{network}_2, \dots, \text{network}_E\} \quad (5)$$

Finally, failures are expected to occur at the WSN level, from one sensor to entire nodes, due to the harsh and dynamic environmental conditions inherent to estuarine settings, including continuous exposure to water immersion, high humidity, salinity-induced corrosion, strong tidal currents, wave action, wind stress, sediment abrasion, and pressure variations, all of which can degrade hardware components and compromise long-term sensor reliability and network connectivity. Therefore, the system shall implement fault tolerance features to handle the absence of data at different levels, for example, when there is a sensor hardware fault, and the WSN cannot send its readings to the web server, or a link between two WSN nodes is broken, and data from an entire node or a fraction of the network is absent for some time.

3. System Architecture

The proposed WSN was designed to provide synchronized environmental monitoring across multiple locations of the Cávado estuary, while maintaining low power consumption, wireless communication, and real-time data availability. The system architecture was developed considering the specific environmental and infrastructural conditions of the estuarine region, namely the existence of remote monitoring areas, limited access to communication infrastructure, and the need for synchronized measurements to support environmental and hydrodynamic studies.

Figure 2 presents a high-level overview of the developed system architecture and its main components. The network is composed of three different node typologies: coordinator nodes, monitoring nodes, and repeater nodes. Communication between nodes is established using Digi XBee SX 868 RF modules (Digi International Inc., Hopkins, MN, USA) operating in DigiMesh mode, enabling multi-hop wireless communication and synchronized sleep capabilities across the network.

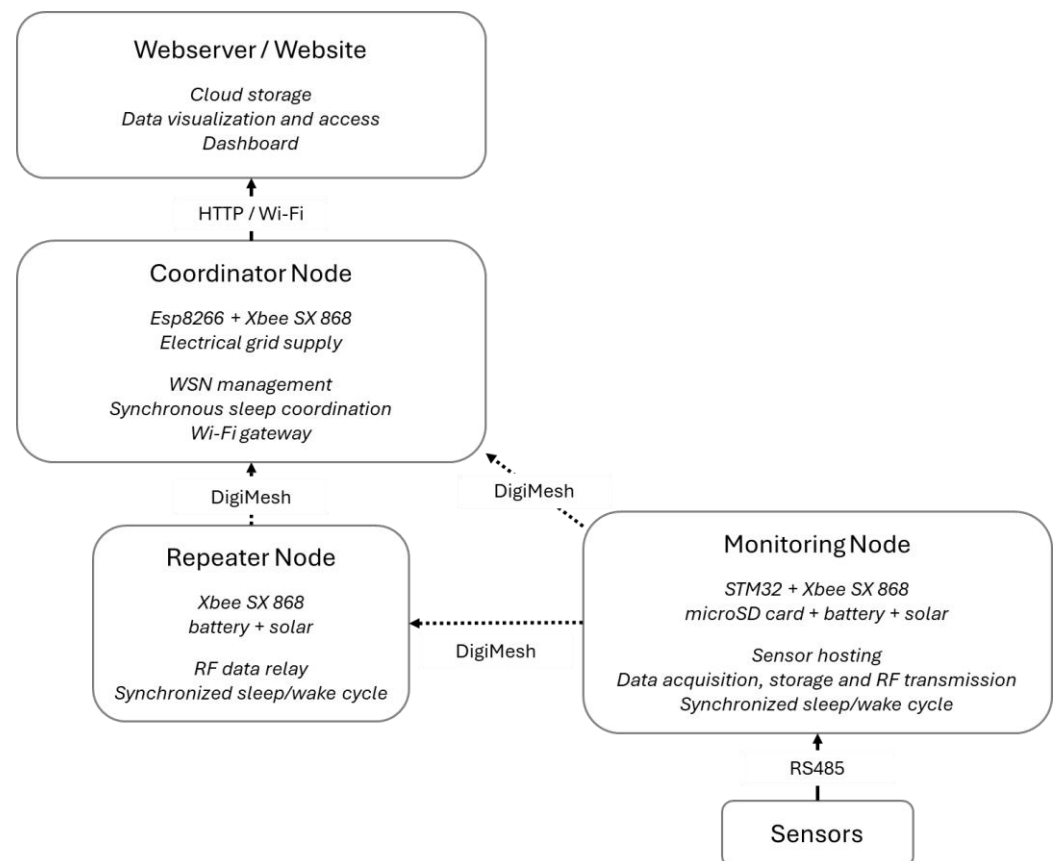


Figure 2. High-level overview of the proposed synchronized wireless sensor network architecture.

The coordinator node acts as the central element of the WSN and is strategically installed where access to electrical power and Wi-Fi connectivity is available. This node is responsible for network management, synchronization of sleep and wake cycles, reception of monitoring data from all network nodes, and transmission of the collected information to a remote web server through an Internet connection. Since energy efficiency is not a critical requirement for the coordinator node, the system is designed to keep the coordinator permanently awake, allowing seamless integration of new nodes and continuous network supervision. The coordinator node also operates as the sleep coordinator of the DigiMesh network, defining and distributing the synchronization parameters used by all network elements.

Monitoring nodes constitute the distributed sensing layer of the system to be deployed at different locations throughout the estuary according to the intended monitoring objectives. Each monitoring node is composed of a low-power data logger, environmental sensors, local data storage, and a Digi Xbee SX 868 RF module for wireless communication. The monitoring nodes periodically wake according to the synchronized network schedule, perform environmental measurements, locally store the collected data, and transmit the information through the wireless mesh network toward the coordinator node. During active periods, monitoring nodes also operate as routing devices, forwarding packets received from neighbouring nodes and contributing to network scalability and robustness.

Repeater nodes were introduced to extend network coverage and improve communication reliability in areas where direct communication between monitoring nodes and the coordinator is not feasible, and monitoring capabilities are not required. These nodes are dedicated routing elements that receive and retransmit wireless packets without performing sensing operations. The use of repeater nodes enables the establishment of additional communication paths and improves network connectivity in complex propagation

environments typical of estuarine and coastal regions. Repeater nodes use the core electronics of monitoring nodes, but without the monitoring hardware, to reduce costs and energy consumption.

The wireless communication infrastructure is based on the DigiMesh protocol operating in the 868 MHz frequency band. This protocol allows peer-to-peer communication among all routing-capable devices while supporting synchronized cyclic sleep mechanisms for low-power operation. Furthermore, the synchronous sleep functionality allows all monitoring and repeater nodes to wake and sleep simultaneously, significantly reducing energy consumption while maintaining coordinated network operation.

Data generated by the environmental sensors is transmitted to the local microcontroller of each monitoring node through a shared RS485 communication bus. The microcontroller processes the received measurements, stores the information in a local microSD card for redundancy, and forwards the monitoring data to the Digi XBee module. The monitoring information is then propagated through the wireless mesh network until it reaches the coordinator node.

At the coordinator node, the received wireless data is transferred to a NodeMCU Lua ESP8266 development board (WEMOS Electronics Co., Ltd., Shenzhen, China) responsible for Internet connectivity and communication with the remote server. The NodeMCU processes the received data packets and performs Hypertext Transfer Protocol (HTTP) requests to send the monitoring information to the web server in real time. The server infrastructure stores the received information and provides data visualization through a dedicated website interface, enabling remote access, visualization, and analysis of environmental data collected throughout the estuary.

The complete system architecture is designed to provide a balance between energy efficiency, communication reliability, synchronization accuracy, and implementation simplicity. The modular structure of the network also allows future scalability, enabling the integration of additional monitoring nodes, sensors, and communication routes without requiring substantial modifications to the existing infrastructure.

4. WSN Implementation

4.1. Electronics Design

4.1.1. Monitoring and Repeater Nodes

Custom printed circuit boards (PCBs) were developed for the three node types to accommodate the specific requirements of the wireless sensor network. Each monitoring node is capable of hosting multiple sensors, all of which share a common cable bus for power and RS485 communication. The node uses a low-power STM32 microprocessor (STMicroelectronics, Geneva, Switzerland), which features an integrated real-time clock (RTC) and microSD card, ensuring local timekeeping and redundant physical data storage. The entire system is powered by a 3.7 V Li-Po battery, which can be recharged via a solar panel. Additionally, a 3 V coin cell battery supplies the RTC, preventing timestamp loss when the system is powered down.

A Digi XBee SX 868 RF transmitter enables wireless communication and integration into the sensor network. To complete the RF system, the AEACAQ190012-S868 Abracon antenna (Abracon LLC, Spicewood, TX, USA), recommended by Digi, is used. UART communication was selected as the interface between the RF module and the STM32 microprocessor, allowing for both network configuration using AT commands and RF data transmission and reception using API and Transparent modes.

A general-purpose input/output (GPIO) pin, which signals the sleep status of the Digi XBee SX 868 RF module, is used as an external interrupt to wake up the microprocessor. Additionally, this pin is used to control the power supply of the sensor bus, allowing each

monitoring node to function as a local master that manages power to the data logger and sensors. This approach enhances energy efficiency by ensuring that components are only active during awake periods.

Figure 3 illustrates the schematic of the monitoring node system and includes photographs of the corresponding PCB. The complete electronic schematic is available in Supplementary Material S1, while a 3D visualization of the PCB design can be found in Supplementary Material S2.

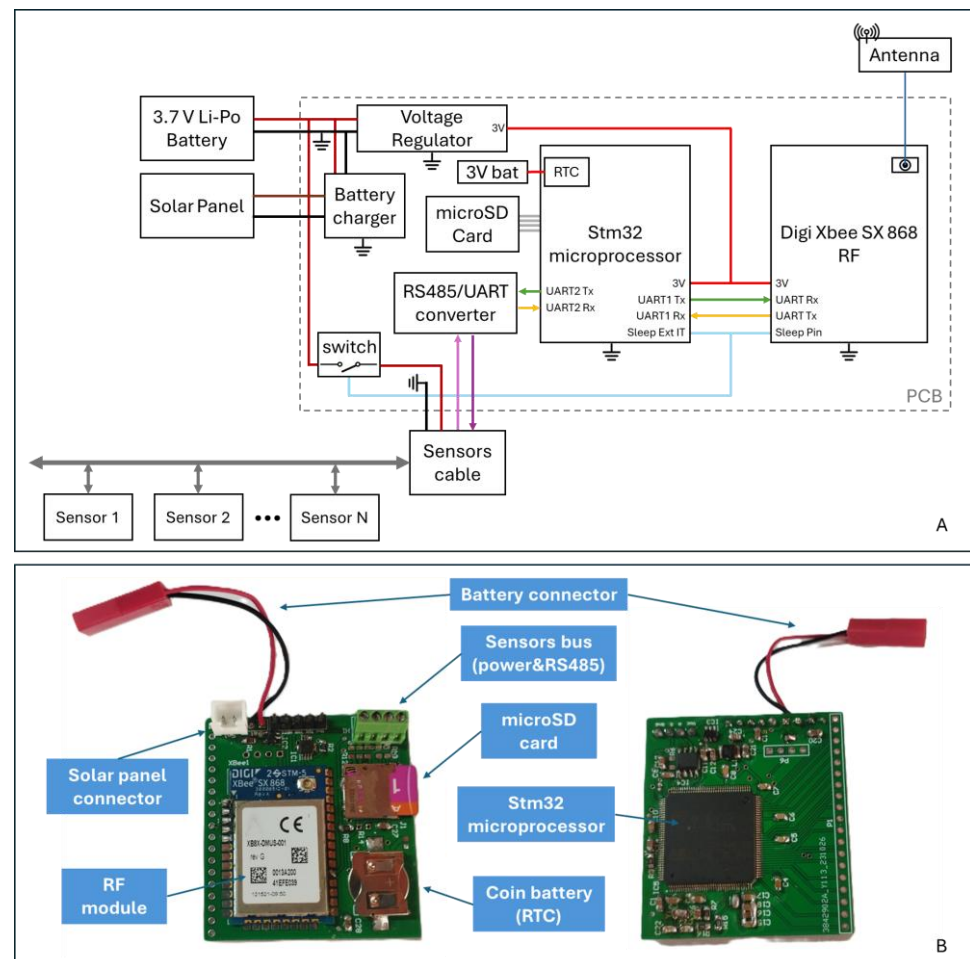


Figure 3. (A) Schematic representation of the monitoring node system. (B) Photographs showing the top and bottom views of the monitoring node PCB.

The hardware used for the repeater nodes follows the same design but includes only the RF hardware and power circuits, featuring a voltage regulator and a solar-powered battery charging system. The data logger hardware (microcontroller, microSD card, and RTC) is not included to reduce costs and energy consumption.

4.1.2. Coordinator Node

Given the availability of electric power for the coordinator node, energy efficiency was not a primary consideration in its hardware development, and the NodeMCU Lua ESP8266 was selected to interface with the Digi XBee SX 868 RF module and enable Wi-Fi connectivity. The NodeMCU Lua ESP8266 is an open-source firmware and development board designed for IoT applications, based on the ESP8266 Wi-Fi module. It features built-in Wi-Fi (802.11 b/g/n) and can operate as both a Station (STA) and an Access Point (AP), supporting TCP/IP stack, MQTT, and HTTP communication. The microcontroller is pow-

ered by a 32-bit Tensilica L106 CPU, running at 80 MHz–160 MHz, with approximately 80 KB of RAM available for user applications and 4 MB of flash memory.

As with the monitoring nodes, UART was chosen as the interface between the microcontroller and the RF module. The entire system is powered via the USB connection of the NodeMCU Lua, which includes an embedded 3 V voltage regulator capable of supporting the entire setup. Additionally, the system is designed to accommodate microSD cards and RTC breakout boards, and it features a sensor bus that shares power and RS485 communication, similar to the one used in the monitoring nodes. While these additional features were not utilized in this network implementation, they provide expanded capabilities for future deployments and allow the coordinator to host sensors and act as a monitoring node itself.

Figure 4 illustrates the schematic diagram of the coordinator node system and includes photographs of the corresponding PCB. The complete electronic schematic is available in Supplementary Material S3, while a 3D visualization of the PCB design can be found in Supplementary Material S4.

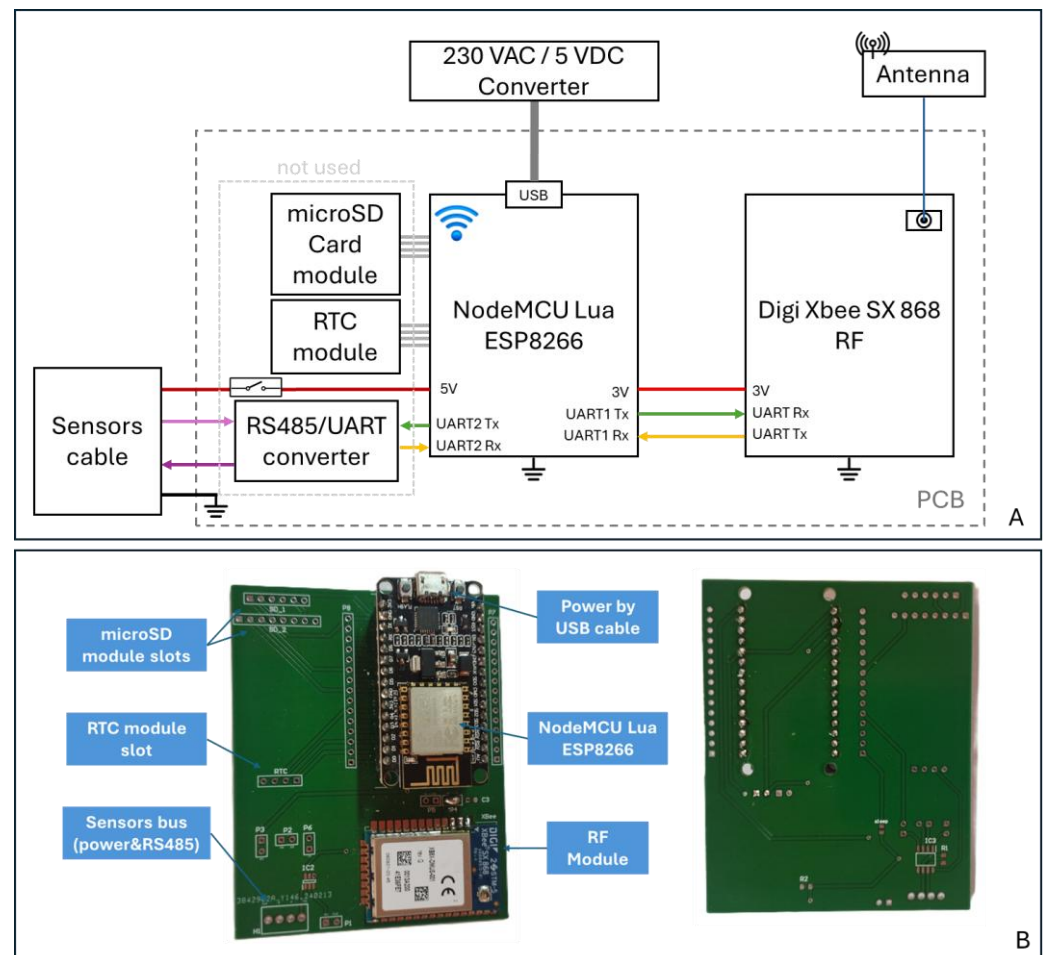


Figure 4. (A) Schematic representation of the coordinator node system. (B) Photographs showing the top and bottom views of the coordinator node PCB.

4.2. Firmware Architecture

4.2.1. Monitoring Node

Figure 5 illustrates the high-level flow diagram of the firmware design for the monitoring nodes. The wake-up periods of these nodes are dictated by the network configuration and are signaled by the Sleep PIN of the Digi XBee SX 868. This GPIO is used to

control the power supply of the sensors within the node, ensuring that they are only powered during awake periods, and to wake up the STM32 microprocessor from sleep mode.

When the system powers up, the sensors perform their designated measurements and transmit the monitoring data via UART using the RS485 bus. When multiple identical sensors (or sensors requiring similar measurement times) are present within a node, a predefined delay is assigned to each sensor before data transmission to prevent collisions. Once data is transmitted, the sensors enter deep sleep mode to conserve energy. During the network's sleep periods, the sensor power supply is completely disabled and is only restored upon network wake-up.

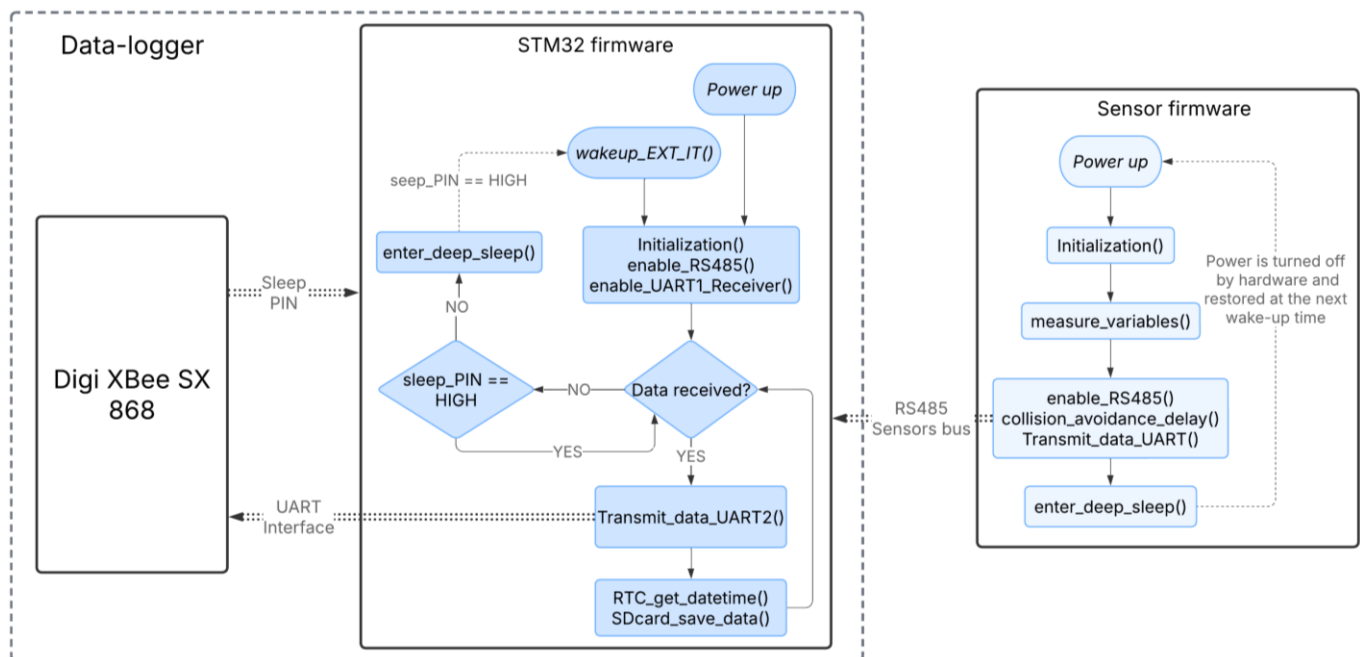


Figure 5. High-level flow diagram of the monitoring node firmware, illustrating the core processes of sensor data acquisition and STM32 microprocessor control.

Upon power-up, the STM32 microprocessor activates the RS485 and UART interfaces to receive data from the sensors. Whenever data is received, it is saved on the microSD card with a local timestamp, and another UART bus is used to transmit the sensor data to the Digi XBee SX 868 for RF transmission. The sensor data are received through UART interrupts rather than by polling, preventing the monitoring node from becoming blocked while waiting for a sensor response. This approach makes the monitoring node independent of the number of connected sensors, improving its plug-and-play capability and allowing sensors to be added or removed at any time without requiring firmware modifications or reflashing. This feature is also fundamental for fault-tolerance, by isolating the monitoring node from the sensor's hardware or software malfunction.

Throughout wake-up periods, the Sleep PIN state is monitored via polling. When the GPIO enters a LOW state, indicating that the Digi XBee and the network have entered sleep mode, the microprocessor transitions to deep sleep mode. It remains in this state until a LOW to HIGH transition on the Sleep PIN triggers an external interrupt, waking the system. During awake periods, the Digi XBee SX 868 broadcasts all received RF data, acting as a router for the network, while also transmitting monitoring data from its node when received via UART.

The Digi XBee SX 868 RF modules support both UART and SPI interfaces for integration with external microcontrollers and embedded systems. In the developed architecture, UART communication was selected due to its implementation simplicity and compatibil-

ity with the Transparent operating mode. In this mode, all serial data received through the UART is automatically encapsulated and transmitted through the RF network without requiring frame construction at the microcontroller level. This approach simplifies firmware development and reduces processing overhead in embedded systems.

4.2.2. Coordinator Node

Figure 6 illustrates the high-level flow diagram of the firmware design for the coordinator node, which consists of the RF module and Wi-Fi module. Upon power-up, the NodeMCU Lua enables the UART port and connects to the available Wi-Fi network. Multiple Wi-Fi connections can be predefined in the firmware, allowing the module to select the strongest available network based on SSID and password authentication. After initialization, the NodeMCU Lua listens for UART transmissions from the Digi XBee module. When data is received, it sends an HTTP request to transmit the monitoring data to the web server. The Wi-Fi connection status is continuously monitored to detect and prevent disconnections.

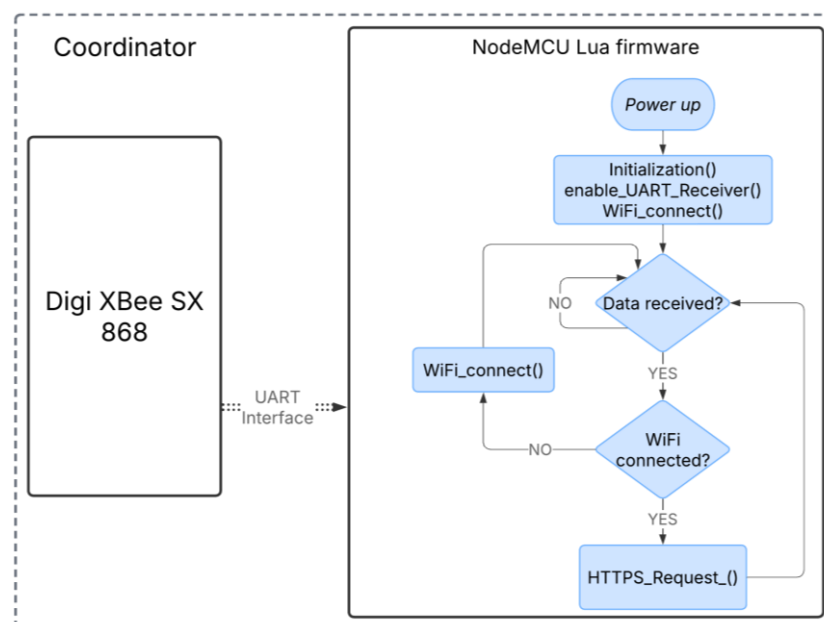


Figure 6. High-level flow diagram of the coordinator node firmware, detailing the NodeMCU Lua communication processes for Wi-Fi connectivity and data transmission.

The Digi XBee SX 868 in the coordinator node remains awake at all times to manage the network. All RF transmissions are addressed to the coordinator, which then transfers the received data to its serial transmit UART buffer, allowing the NodeMCU Lua to retrieve and process the data.

4.2.3. Digi XBee Transceivers

Access to the application, network, and medium access control (MAC) layers by the Digi XBee SX 868 is restricted and not supported by users. Instead, the module's operations are controlled through configurable parameters, which can be modified via the UART interface using AT commands or through the dedicated XCTU software. For this WSN implementation, all Digi XBee modules were pre-configured using XCTU before deployment. AT commands were not used during operation to maintain a fixed network configuration.

The network was configured to operate in Single Frequency Mode at 869.85 MHz, with the Channel Mask (CM) parameter ensuring that all nodes share the same frequency.

Additionally, all nodes in the network were assigned to the same Network ID (ID) and Preamble ID (HD) to ensure proper communication within the network. The RF baud rate (BR) was set to 80 kbps to enable DigiMesh networking, including its synchronous operation and sleep modes.

The monitoring and repeater RF transceivers were set as standard routers (CE = 0) to allow packet forwarding, with DH (Destination High) and DL (Destination Low) configured for broadcast, ensuring that all received RF data is forwarded within the network. Since all monitoring data is ultimately unicasted to the coordinator, the coordinator transceiver was set as a non-routing coordinator (CE = 3), meaning it does not participate in routing operations.

For Transmit Options (TO), monitoring nodes were configured to use DigiMesh unicast transmission with acknowledgement and retries for reliable data delivery to the coordinator. Repeater nodes, however, could either follow the same configuration or be set to point-to-point/multipoint mode with unacknowledged broadcast to reduce transmission latency. Since repeaters and the coordinator do not send monitoring data, unicast transmission is not required for these nodes. The coordinator transmits network management data (such as synchronization control signals) as broadcasts to all nodes, rather than unicasting to specific devices. The maximum number of hops in the network was set to 7 (the default value) but can be increased, if necessary, as the network expands. The TX power was configured to its maximum value (13 dBm) to maximize communication range. Conversely, a lower TX power would reduce the range significantly while yielding only a minimal reduction in power consumption.

Regarding the serial interface, the UART baud rate (BD) was set to 9600 bps, matching the communication settings of both the STM32 microprocessor and the NodeMCU Lua to ensure seamless data exchange. The transparent mode (AP = 0) was chosen for data transmission. While API mode (AP = 1 or 2) offers a more structured, compact, and efficient data exchange with improved reliability, transparent mode was preferred for its simplicity, reduced processing overload, and greater flexibility in data handling.

All transceivers were set to operate in synchronized cyclic sleep mode (SM = 8) to ensure energy efficiency, with the coordinator node assigned as the sleep coordinator (SO = 1), and monitoring and repeater nodes set as non-sleep coordinators (SO = 2). The sleep time (SP) and wake time (ST) values can be adjusted based on the specific application requirements. Initially, monitoring and repeater nodes operate using their own SP and ST values. However, once they successfully join the network, the coordinator's SP and ST settings override initial values, ensuring network-wide synchronization.

4.3. Message Exchange Format

A message exchange format was designed to seamlessly transfer monitoring information across all network layers, from the data collection points (sensors) to the final data handler (web server). Figure 7 illustrates the flow of the protocol through the various network components.

The monitoring data generated by each sensor deployed in the network is structured into a text string consisting of the sensor ID (1 byte), the monitored variables along with their respective IDs and measurements (string format), and a terminator character (1 byte). Each element in the sensor packet is separated by a designated delimiter, except for the final element before the terminator character. Considering a sensor with sensor ID 'A', that measures water temperature at 15.5 °C (temperature variable ID: 't') and absolute pressure at 2100 mbar (pressure variable ID: 'p'), and using '\$' as the terminator character, the formatted data payload from the sensor would be "A&t=15.5&p=2100\$". This string is transmitted via UART to the STM32 microcontroller of the respective monitoring node.

The STM32 of the respective monitoring node collects all received bytes until it detects the terminator character. Since the received packet is in string format, it is directly saved to the microSD card along with a timestamp (date and hour) without requiring additional processing. The only modification applied is the replacement of the terminator character with a newline character ('\n'). The adopted format enables users to open the saved data as a comma-separated values (CSV) file, where the '&' character serves as the delimiter, allowing for easy offline analysis and processing.

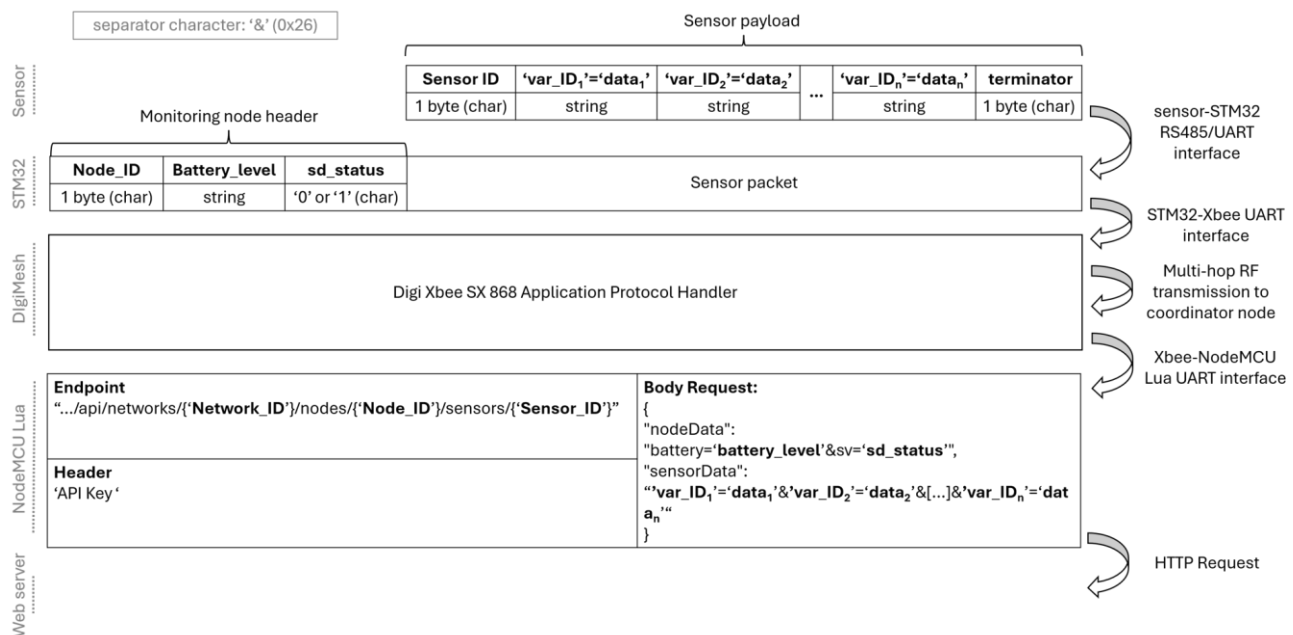


Figure 7. Flow of the message exchange format across all network elements, from sensor data collection to final transmission to the web server.

Before RF transmission, the STM32 appends additional metadata to the sensor packet, including the monitoring node ID, the battery level, and the microSD card status (indicating whether data storage is functioning correctly). Assuming the monitoring node ID 'A', battery level 3750 mV, and SD card status '1' (indicating successful data storage), the STM32 payload would be "A&3750&1&A&t=15.5&p=2100\$". This formatted data packet is transmitted to the Digi XBee module of the monitoring node, which subsequently broadcasts the packet over the DigiMesh network until it reaches the coordinator node.

The NodeMCU Lua receives monitoring data from the coordinator XBee module via the UART interface. Upon receiving a data packet, the NodeMCU parses and processes the information to generate an HTTP request for transmission to the web server. The Lua firmware is preconfigured with the network ID, API key, endpoint URL, and necessary HTTP headers and body formatting to ensure seamless communication. Data parsing is performed using the string format, with '&' as the delimiter to separate packet elements. The structure of the received data is as follows: the first element corresponds to the Node ID (e.g., 'A'), the second element to the battery level (e.g., '3750'), the third element to the SD card status (e.g., '1', indicating success), the fourth element to the Sensor ID (e.g., 'A'), and the fifth element onward (until the terminator character) contains the monitoring data, including variable IDs and values (e.g., "t=15.5&p=2100"). Each extracted element is then formatted accordingly and concatenated into the body and endpoint of the HTTP request for proper data submission.

Although string format was selected for data exchange, binary formatting was also considered as an alternative. A binary format would reduce the payload size, thereby

minimizing RF transmission time and improving energy efficiency. However, it would also increase processing requirements for both the STM32 and NodeMCU Lua, as additional computation would be necessary to encode and decode data. For monitoring nodes with a single sensor, a binary format would be the most efficient choice due to its compact size. However, as the number of hosted sensors increases, the network's limited awake time imposes constraints on packet processing. In this scenario, string formatting ensures a seamless data transfer without requiring additional processing overhead.

Furthermore, string formatting offers greater flexibility, allowing users to define nodes, sensors, and variables dynamically without strict data structure constraints (except for the reserved separator and terminator characters). Additionally, it supports various numerical formats, including integers, decimals, and negative values, without the need to predefine byte sizes and data types, as required in binary formatting. Considering the need for scalability, flexibility, and ease of data handling, string formatting was determined to be the optimal choice for this wireless sensor network implementation.

Lastly, concerning fault-tolerance, if one sensor fails, the STM32 does not receive its data, and consequently, the respective message is not sent to NodeMCU and then to the web server. By sending a separate message for each sensor data, the protocol naturally handles sensor hardware or software faults.

5. Web Server and Application

5.1. Database

Most of the developed systems' data consists of time series, besides the network structure and a few configuration and status variables. Conceptually, the relationships between entities of the database are: a network contains one or more nodes, each node contains one or more sensors, and each sensor contains one or more variables. The kinds of variables for each sensor are different, and new kinds of variables may be added at any time.

To store the data, the chosen option was a NoSQL database, namely MongoDB, whose physical model is presented in Figure 8. This model defines 3 collections—network, node and sensor—providing a structure for these entities. When data from one sensor is necessary, what makes more sense is to consider all of its variables in the analysis; therefore, it was decided to be the finest grain of data to be obtained from the database. To deal with the flexibility of sensors' variables and that they are supposed to be obtained altogether, the option was to store sensors' data as a JSON array (datalog) with each object in the same format as the (Request body) of Figure 7. With this kind of storage, the absence of data from a given sensor is handled naturally and does not require any special coding or structure. This is essential for the fault tolerance capabilities of the system.

Nodes have timestamps for the battery level and location, with all three fields being saved as an array. Node timestamps could be used for sensors, but each sensor may fail, resulting in the lack of data for a given period. To relate variables' data to the proper timestamp without a 1:1 correspondence between arrays, it would be necessary to introduce logic in the system at different levels. Instead, we opted for simplicity (and robustness), keeping the timestamps separate on each sensor and with a 1:1 correspondence with the datalog. Timestamps are fundamental to handling the absence of data. Since the system operates on a fixed monitoring period, when one or more timestamps are missing, it means there is a fault with the respective node or sensor.

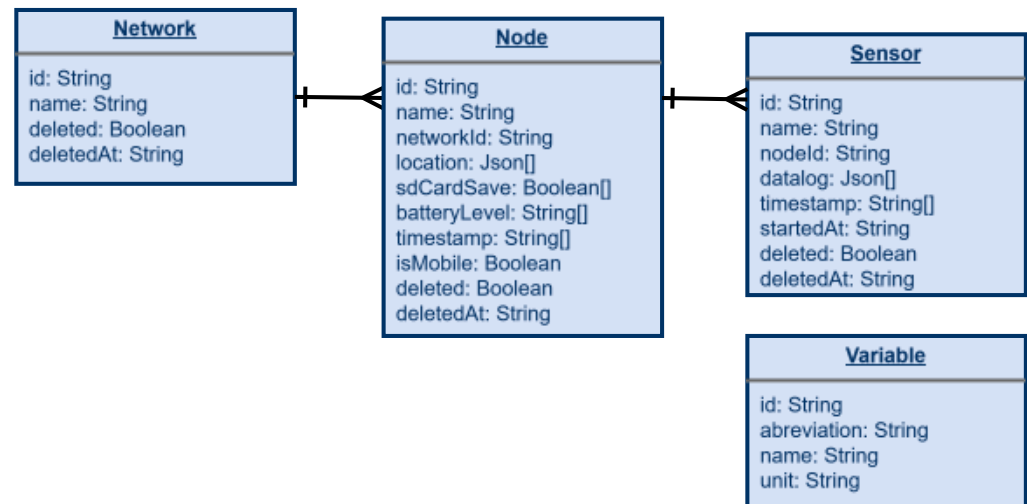


Figure 8. Physical model of the database.

A fourth collection contains the definitions of all the kinds of variables in the system. The definitions include the units for the variables, allowing fewer bytes in the value string sent by the WSN messages and simplifying message processing. To help with the definition of new networks, the database is populated from scratch with a list of predefined kinds of variables, for example, conductivity in S/m, turbidity in NTU (Nephelometric Turbidity Units), water velocity in m/s, pressure in mbar, temperature in °C, and salinity in PSU (Practical Salinity Units). These kinds include not only the variables that are currently monitored in the Cávado estuary, but also others that are usual in such a type of monitoring.

A delete flag is included in the network, node and sensor collections, enabling a safer management interface by supporting logical data removal. A command line script is provided to physically delete data in the database, which also allows listing networks and putting back on data marked as deleted.

5.2. Backend API

The web server provides an API for the management and monitoring of all created networks. The web API is RESTful with HTTPS requests protected by a secret key, and it provides the endpoints listed in Table 2. Networks, nodes and sensors are identified by IDs, and the body of both requests and responses is in JSON format.

Table 2. Endpoints and HTTP methods supported by the webserver backend API.

Endpoints	HTTP Methods
.../api/networks	GET ¹ , POST
.../api/networks/{netid}	GET ¹ , DELETE
.../api/networks/{netid}/nodes	GET ¹ , POST, DELETE
.../api/networks/{netid}/nodes/{nid}	GET ¹ , DELETE
.../api/networks/{netid}/nodes/{nid}/sensors	GET ¹ , POST, DELETE
.../api/networks/{netid}/nodes/{nid}/sensors/{sid}	GET ¹ , PUT ² , DELETE
.../api/variables	GET ¹ , POST
.../api/variables/{vid}	GET ¹ , DELETE

¹ Requests that serve both monitoring and management purposes. ² Request specifically provided for WSN data upload.

All endpoints have a GET request. For monitoring purposes, the GET requests form a chain that provides access to all the available data. The requests work in the following

sequence: network IDs are obtained, then information about each one (see Network collection in Figure 8) can be obtained, and so forth for nodes and then sensors.

Data from WSNs is uploaded to the server using the PUT request of the “.../api/networks/{netid}/nodes/{nid}/sensors/{sid}” endpoint, which means it is issued once per sensor on each network monitoring period. The variables’ values are sent in the request body, in the same format as the (Request body) of Figure 7, to be stored as an object in the datalog array of Figure 8. The isMobile field of the node (in Figure 8) is used to handle the location received in PUT requests: if the node is fixed, the single value in the array is updated, otherwise the value is added to the array.

The WSN network keeps data acquisition synchronous, as explained in Section 3. PUT requests are received per sensor (for each network), arriving at slightly different times for all the possibly dozens of them. The WSN is active for 10 s and all requests shall arrive within an equal or smaller time window, which is several orders of magnitude smaller than the acquisition period (currently 10 min). To simplify data analysis, it was decided that the server would synchronize timestamps. For that operation, a time window mechanism was implemented that assigns the same timestamp to all requests received during a configurable interval (currently 15 s). This approach is feasible because the variance is small compared to the sampling time. The timestamp value is that of the first request that is received after the previous time window has elapsed.

The GET requests also serve the purpose of network management to assist managers in their tasks. All other requests in Table 2, namely POST and DELETE, are exclusively dedicated to management. These API requests allow the creation of a new network that has been deployed, (re)configure an existing network, and delete it when it is removed from the field. Network (re)configuration involves adding and removing individual nodes, and it is possible to delete all nodes in a network. For each node, the API allows the deletion of all sensors, adding a new sensor and deleting an existing one. It is not possible to reconfigure one variable of a sensor: when changes are necessary to the set of variables of a sensor, it should be deleted and a new one added. The list of all existing variables defined in the system can be obtained, and new variables can be defined.

Due to configuration errors (node, sensor or variable that was defined but does not exist) and operation failures at any level (hardware, software or communication), the web server must support the absence of data at different levels. Thus, several robustness measures have been implemented, for example:

- If one variable that has been received before is missing, a gap is introduced in the JSON datalog; this is a redundancy measure, since each missing value could be inferred from the absence in the JSON object corresponding to the respective timestamp.
- If a variable is received that is not in the collection of defined variables, the response includes a status “Updated partially” with “unknown variables” description, and the variable reading is discarded;
- If the same variables appear more than once in a request, the response includes a status “Updated partially” with “duplicated variables” description, with the identification of which one(s);
- If during the synchronization time window more than one request is received for the same sensor (e.g., due to auto reset), the data updates the same datalog array element.

5.3. Web Application

Network management is performed by researchers with in-depth knowledge of the system and only from time to time. Therefore, no web application was built for it, and tools such as Postman can be used to easily carry out network configuration.

The web application is dedicated to network monitoring, and all data is freely accessible at <https://nextsea.dei.uminho.pt> (accessed on 12 July 2026). The server backend and web application architecture are represented in Figure 9. Both were built using Next.js (version 14.2.4), a full-stack web framework, thus using a single language (JavaScript) for all web development.

On the server side, access to the MongoDB database was implemented using Prisma (version 5.15.0), an object-relational mapper that automates connection and queries to the database. The Prisma client and MongoDB database definitions were both generated from the schema file with the data model definition. The backend API uses the Prisma client to interface with the database. The frontend API provides static web page content (i.e., server-side rendering). The Nginx server (version 1.24.0) is used as a reverse proxy, making it easier to handle HTTPS, providing file (images, scripts, and CSS) caching and better security.

On the client side, several libraries were used to implement the user interface. The Axios HTTP client (version 1.7.2) is used to more efficiently make requests to the frontend web API. Web page hooks were developed by wrapping Axios fetches using Tanstack Query (version 5.50.1) to asynchronously manage global state and local cache. TailwindUI (version 3.4.4) and MaterialUI (version 5.16.4) were used for graphical style and elements, such as buttons and dropdown boxes. React ECharts (version 5.5.1) was used to implement the time series charts, making it easier to provide interactive and dynamic charts. Interactive maps showing the estuary area and the location of monitoring nodes as markers were implemented using the React Google Maps library (version 2.19.3). The code was divided into small components used in different pages to improve the efficiency of Next.js content downloading from the server. Page navigation was implemented using the proper mechanism (useRouter) to enable Single Page Application behavior, avoiding full reloads.

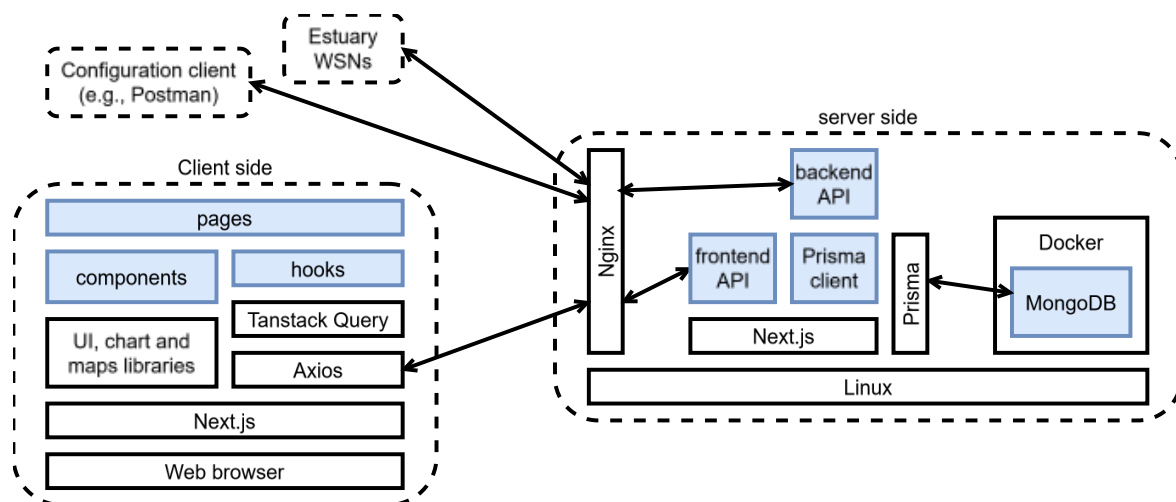


Figure 9. Web application and server architecture.

The data is made accessible by navigating from a list of networks to the node of interest and displaying the respective variables from all sensors (this is detailed in the following Section 6.4). Regarding fault tolerance, the web application draws time-series charts with blank spaces when the variables' data is missing. This way, the user interface handles gracefully when faults happen at either the sensor, node, or communication level.

6. Results and Discussion

6.1. Wireless Communication and Network Coverage

An initial field validation campaign was conducted to evaluate the communication range, multi-hop connectivity, and overall reliability of the proposed WSN architecture under real deployment conditions in the Cávado estuary. The test consisted of a temporary deployment of six monitoring nodes distributed along the estuarine margins and configured to periodically transmit dummy monitoring data through the DigiMesh network. The nodes were positioned at representative locations covering different communication scenarios, including line-of-sight propagation, partially obstructed paths, and extended communication distances along the river margins (Figure 10).



Figure 10. Temporary installation of the WSN in the Cávado estuary.

Monitoring node 1 was the only node located within the direct communication range of the coordinator node, maintaining a direct wireless connection (enhanced by the presence of the repeater node). Monitoring nodes 2 to 6 relied on multi-hop communication, forwarding data through intermediate nodes along the route 6–5–4–3–2–1–(R)–C until reaching the coordinator.

During the experiment, all nodes were configured to transmit packets at one-minute intervals over a continuous period of approximately two hours. The coordinator node remained permanently active, ready to receive the incoming packets. The transmitted information consisted of predefined dummy data packets of 10 bytes emulating environmental measurements and node diagnostic information. The wireless transmission was set to 80 kbps, and the nodes were configured to 3 unicast retries. A repeater node was installed to strengthen the link between monitoring node 1 and the coordinator, given that the coordinator was located inside a building with thick stone walls. The purpose of the test was not to validate sensing capabilities but rather to assess the practical coverage of the implemented mesh network. Table 3 presents the expected and received packages, as well as the packet delivery ratio computing of the experiment.

Table 3. Packet delivery ratio (PDR) during the experiment.

Node	Expected	Received	Lost	Corrupted	PDR	Hop Count
1	120	120	0	0	100.0%	1
2	120	119	1	0	99.2%	2
3	120	120	0	0	100.0%	3
4	120	120	0	0	100.0%	4
5	120	118	2	0	98.3%	5
6	120	120	0	0	100.0%	6

During the two-hour experiment, the coordinator received 120 out of 120 packets from each node, except for nodes 2 and 5, from which one and two packets were lost, respectively. The obtained results demonstrate reliable network operation throughout the test duration, with successful packet delivery from all deployed nodes. Nodes located outside the direct communication range of the coordinator successfully transmitted data through intermediate routing nodes, confirming the effectiveness of the DigiMesh multi-hop communication capabilities. Despite the longest communication path requiring multiple forwarding stages, packet delivery remained above 98%, demonstrating the feasibility of the selected network architecture and communication parameters.

The results further highlight the suitability of the 868 MHz frequency band for environmental monitoring applications in estuarine environments, where vegetation, terrain irregularities, high humidity, and changing tidal conditions may affect radio propagation. Reliable communication was maintained over distances of up to 1.2 km between neighboring nodes, including partially obstructed links. The successful operation of the mesh routing mechanisms between node 1, repeater, and coordinator also demonstrates the scalability of the proposed architecture, allowing additional monitoring and repeater nodes to be incorporated without requiring direct connectivity to the coordinator. This capability is particularly advantageous for environmental monitoring applications, where sensor deployments often need to be progressively expanded to cover larger areas or regions with difficult accessibility.

6.2. Synchronized Network Operation

Following the initial validation, a permanent deployment consisting of one coordinator, one repeater node, and two monitoring nodes (Node 1 and Node 2) was implemented to evaluate synchronized operation under real-world monitoring conditions. The deployment aimed to assess synchronized sleep behavior, data transmission reliability, and clock synchronization stability. The monitoring nodes were configured to wake every 10 min for a 10 s active period, during which they acquired pressure (depth) and water temperature measurements using an MS5837-30BA sensor and transmitted the collected data through the DigiMesh network to the coordinator node. The 10 s interval allows sufficient time for sensor stabilization and data collection. After each acquisition and transmission cycle, the nodes returned to sleep mode for a 10 min interval. The system was deployed from 30 March to 7 April 2026.

Figure 11 presents the monitoring data collected during the eight-day experimental deployment. The received monitoring data demonstrates a temporal alignment between the pressure measurements acquired by monitoring nodes 1 and 2. The observed 12 h sinusoidal pattern corresponds to the tidal cycle in the estuary, with transitions between high and low tide occurring every 6 h, highlighting the importance of synchronized data acquisition in this type of environmental monitoring application. A similar trend can be observed in the water temperature measurements, which are influenced by the temperature differences between ocean water (approximately 10–13 °C) and river water (approximately 14–16 °C) at this time of the season. Consequently, the estuary experiences cooling

during high tides, when ocean water intrudes inland, and warming during low tides, when riverine influence becomes dominant. Furthermore, monitoring node 1 exhibits greater temperature variability due to its proximity to the estuary inlet. As the distance from the inlet increases, the influence of ocean water decreases, and temperature fluctuations become less pronounced. These results highlight the importance of monitoring node placement for accurately capturing the spatial variability of estuarine processes.

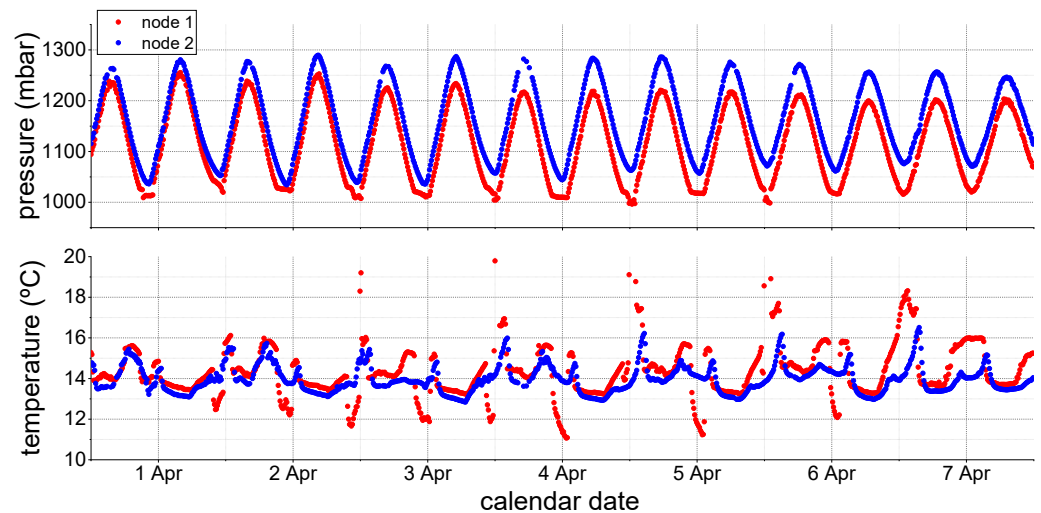


Figure 11. Results of the monitoring experiment with two synchronized nodes collecting measurements of pressure and water temperature.

The received data confirmed the successful transmission and reception of measurements from both monitoring nodes, demonstrating stable end-to-end communication across the synchronized mesh network. The implemented message exchange format enabled the coordinator node to correctly parse and process the transmitted information before forwarding it to the web server. During the deployment period, the coordinator successfully received 100% of the expected data packets from monitoring node 1 and 96.45% from monitoring node 2, with 3.25% corresponding to corrupted packets and 1.20% not received during the scheduled wake-up periods. The slightly lower performance observed for monitoring node 2 can be attributed to its dependence on a multi-hop communication path involving an intermediate repeater node, which increases the probability of transmission failures and packet collisions. Analysis of the local data storage revealed that all missing measurements were successfully recorded on the microSD card, indicating that data acquisition and processing at the monitoring node were not affected. This result highlights the importance of incorporating redundant local storage mechanisms in wireless sensor networks, ensuring data preservation even when temporary communication failures occur. The availability of physical data storage, therefore, enhances the reliability of the monitoring system and provides an effective safeguard against packet losses associated with wireless communications. Nevertheless, the obtained packet delivery rates demonstrate the reliability of the proposed architecture for long-term environmental monitoring applications, particularly considering the low-power operation and synchronized sleep behavior of the network.

One of the primary objectives of the proposed architecture was the implementation of synchronized sleep and wake cycles across the network. In this monitoring scenario, pressure (water depth) and water temperature measurements enabled the observation of tidal cycles associated with estuarine hydrodynamics. The synchronization mechanism ensured temporally aligned measurements among geographically distributed monitoring nodes, allowing direct comparison of environmental conditions at different locations

within the estuary. The observed alignment of tidal signatures in both pressure and temperature records confirms the effectiveness of the synchronization strategy and demonstrates its suitability for studying spatial and temporal variations in estuarine processes. The results further indicate that all monitoring and repeater nodes operated according to the synchronization schedule distributed by the coordinator node acting as the sleep coordinator. This mechanism automatically compensated for local clock drift while maintaining accurate timestamps for data storage. Throughout the deployment, the coordinator continuously redistributed synchronization information, enabling the automatic correction of accumulated timing deviations without requiring manual recalibration or user intervention. This capability represents a significant advantage for long-term environmental monitoring applications, where maintenance operations should be minimized, and reliable temporal synchronization is essential for data interpretation.

6.3. Energy Efficiency Assessment

Energy consumption measurements were conducted to evaluate the energy efficiency of the developed hardware. The monitoring nodes exhibit power consumptions of 135 mW and 225 mW during idle and RF transmission modes, respectively, which occur during active periods, and 100 μ W during sleep mode. The repeater nodes show similar power consumption during active operation and 6 μ W while in sleep mode. The coordinator node presents an average power consumption of 265 mW; however, this value is not critical since the coordinator was designed to be powered directly from the electrical grid. The sensor used during the field experiment consumes 210 μ W during active operation and its power supply is disconnected during sleep periods.

Using the measured values, the average power consumption (P_{avg}) of the monitoring node during the field experiment was estimated according to

$$P_{avg} = P_{active} * D_{cycle} + P_{sleep} * (1 - D_{cycle}), \quad (6)$$

where P_{active} is the power consumption during active operation calculated using the transmission power (P_{Tx}) of 225 mW, the transmission time duration (T_{tx}), and 135 mW power consumption in idle mode (P_{idle}):

$$P_{active} = P_{Tx} * \frac{T_{tx}}{T_{active}} + P_{idle} * (1 - \frac{T_{tx}}{T_{active}}) \quad (7)$$

P_{sleep} is the power consumption during sleep mode and D_{cycle} is the duty cycle, which is given by

$$D_{cycle} = \frac{T_{active}}{T_{active} + T_{sleep}} \quad (8)$$

where T_{active} corresponds to the active period and T_{sleep} to the sleep period.

The battery lifetime of the monitoring node (T) in the absence of energy harvesting was estimated using a battery capacity (C) of 4.07 Wh, corresponding to a 3.7 V, 1100 mAh Li-Po battery, according to

$$T = \frac{C}{P_{avg}} \quad (9)$$

Table 4 summarizes the calculated average power consumption and the estimated node autonomy. The calculated average power consumption of 2.35 mW highlights the impact of duty-cycled operation, reducing the effective power demand by nearly two orders of magnitude compared with the active operating state. The active power consumption value assumes a fixed RF transmission duration of 10 ms per acquisition cycle, with the remaining active time spent in idle operation and no packet retransmissions. Consequently, the presented values represent a theoretical best-case lifetime estimate and do

not account for additional energy consumption associated with retransmissions, route discovery procedures, or the forwarding of packets originating from other monitoring nodes.

Table 4. Estimated average power consumption and battery lifetime of the monitoring node.

P_{active} (mW)	P_{sleep} (mW)	P_{avg} (mW)	T (days)
135.1	0.1	2.35	72.2

The obtained results indicate an estimated lifetime of approximately 72 days without energy harvesting with the configured 10 min sampling interval, demonstrating the effectiveness of the proposed low-power architecture and synchronized sleep strategy. Although the active power consumption reaches 135 mW during measurement and communication periods, the very low sleep consumption of approximately 100 μ W significantly reduces the average power demand due to the low duty cycle of the network.

Figure 12 presents node battery lifetime across various sampling times, assuming a fixed 10 s active time and a battery capacity of 4.07 Wh.

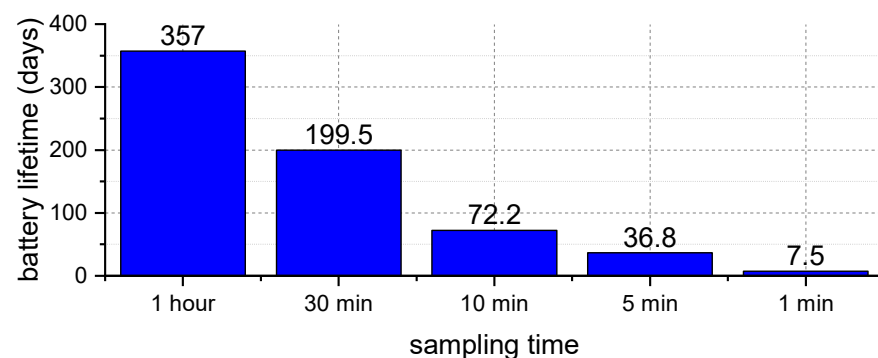


Figure 12. Estimated battery lifetime for different sampling times.

The energy and autonomy assessment results confirm that synchronized operation is not only beneficial for temporally aligned measurements but also plays a fundamental role in extending node lifetime by maximizing the duration spent in low-power states. As referred before, synchronized active-sleep communication cycles among the monitoring nodes decrease idle listening periods significantly, which are typically one of the major contributors to energy consumption in WSNs, while preventing loss of data due to transmission of packets to sleeping nodes. Furthermore, while the current hardware utilizes a compact 1100 mAh LiPo battery, it can be easily replaced with a higher-capacity alternative to extend autonomy if required.

Yet, during the field deployment, the monitoring nodes were equipped with solar panels for battery recharging, effectively extending the lifetime of the nodes indefinitely. Figure 13 presents the battery voltage of monitoring node 1 throughout the deployment period.

The battery voltage monitoring reveals regular charging cycles during daylight periods and a stable battery voltage over the entire experiment, demonstrating that the harvested solar energy was sufficient to compensate for the energy consumed by the node. No progressive battery discharge trend was observed, indicating that the energy harvesting system could sustain continuous operation under the environmental conditions encountered during the deployment. These findings demonstrate the feasibility of long-term autonomous monitoring without battery replacement or external intervention, which is a key requirement for large-scale environmental monitoring networks deployed in remote or difficult-to-access locations.

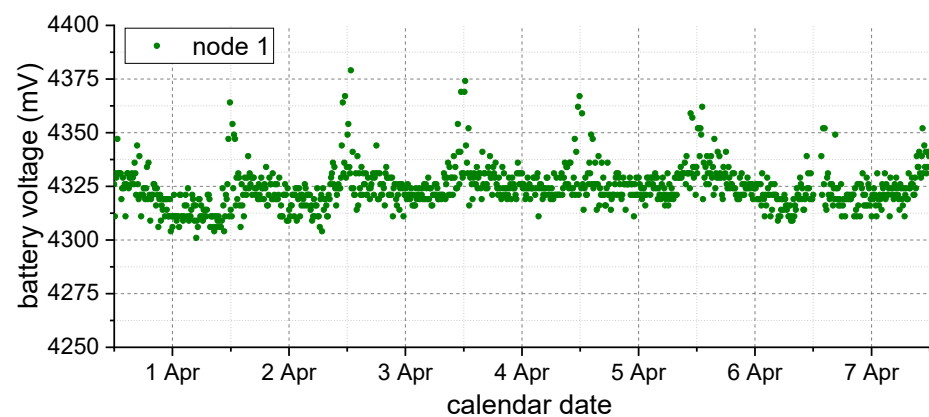


Figure 13. Battery level of monitoring node 1 during the field experiment.

6.4. Website and Data Visualization

The web-based platform was developed to provide real-time access to the data collected by the wireless sensor network and to facilitate network management and data interpretation. Acting as the final layer of the monitoring infrastructure, the platform presents all stored data in an intuitive graphical interface accessible from any device with internet connectivity.

Figure 14 illustrates the main pages of the developed platform. The initial page allows users to select the monitoring network to be visualized. This approach enables the monitoring of multiple independent sensor networks within the same platform, providing scalability for future deployments and monitoring applications.

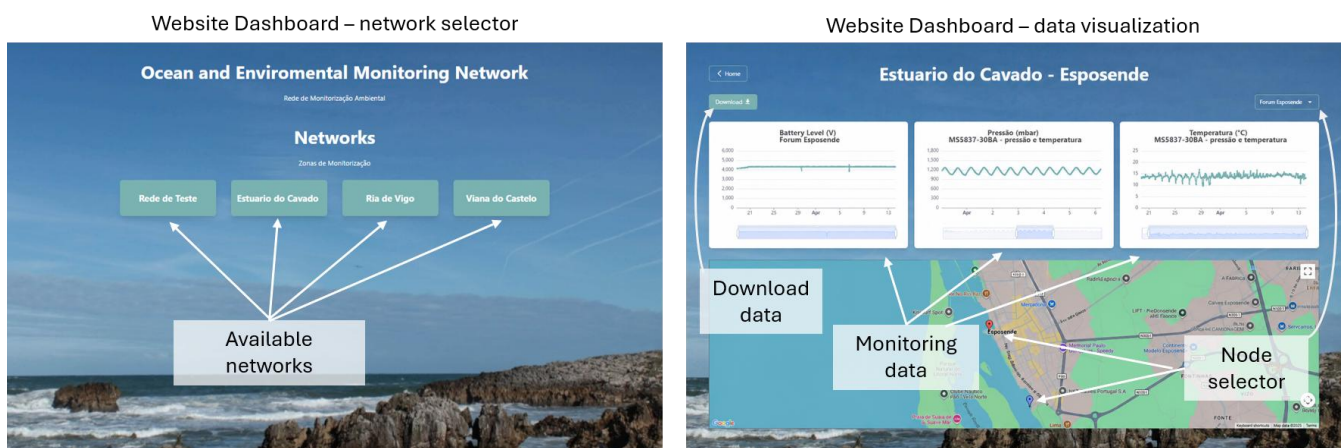


Figure 14. Web-based monitoring platform showcasing the network selection page and the dashboard of the Cávado Estuary deployment.

Once a network is selected, the user is redirected to the corresponding monitoring dashboard. The dashboard provides real-time visualization of the collected environmental and system variables, which for the Cávado estuary deployment include the collected water pressure, water temperature, and battery voltage. The measurements are presented through interactive time-series graphs that allow users to explore historical and recent data. A sliding time window enables dynamic adjustment of the displayed time interval, facilitating both short-term analysis of recent measurements and long-term assessment of environmental trends. The monitoring data can be downloaded into a CSV file for further offline processing.

In addition to graphical data visualization, the dashboard incorporates a georeferenced map displaying the location of all monitoring nodes belonging to the selected network. Each node is represented by an individual marker, allowing users to quickly identify the spatial distribution of the deployment. By selecting a marker on the map, the user can switch between monitoring nodes, automatically updating the displayed graphs to present the measurements corresponding to the selected location. This functionality provides a seamless integration between spatial and temporal data visualization, allowing users to analyze environmental variability across different monitoring sites within the estuary.

The platform proved effective throughout the deployment period, enabling continuous remote access to both environmental measurements and network status indicators. The combination of real-time data visualization, historical records, and geospatial information facilitated the identification of tidal patterns, temperature variations, and node operating conditions without requiring direct access to the monitoring sites. Furthermore, the ability to monitor battery voltage remotely provides valuable support for predictive maintenance and long-term network management. These capabilities demonstrate the importance of integrating data visualization tools, transforming raw sensor measurements into actionable information for environmental monitoring and decision-making.

7. Conclusions

This work presents the complete design, implementation, and field validation of a synchronized wireless sensor network for environmental monitoring in the Cávado estuary (Esposende, Portugal). The proposed system comprises a multi-layer architecture including custom-designed monitoring, repeater, and coordinator nodes, dedicated firmware for network operation and synchronization, message exchange format, and a web-based platform for real-time data visualization and management. The architecture was developed to address the requirements of estuarine monitoring applications, namely synchronized measurements, low-power operation, reliable wireless communication, and remote access to environmental data.

The implementation was based on Digi XBee SX 868 RF modules operating in DigiMesh mode with synchronized cyclic sleep support. The developed hardware and firmware enabled the seamless integration of environmental sensors, local data storage, wireless communications, and energy harvesting capabilities. The message exchange format was designed to ensure efficient transfer of monitoring information from the sensors to the web server while maintaining simplicity, flexibility, and scalability. The web platform complemented the monitoring infrastructure by providing real-time access to environmental measurements, network status information, historical records, and georeferenced visualization of monitoring nodes.

The developed network architecture was successfully validated under real deployment conditions. Initial field experiments demonstrated reliable wireless communication over extended distances using multi-hop routing, confirming the suitability of the 868 MHz frequency band and DigiMesh networking for estuarine environments. The synchronized monitoring deployment further demonstrated the ability of the network to maintain temporally aligned measurements across geographically distributed nodes while automatically compensating for clock drift through the synchronization mechanisms provided by the network coordinator. The collected pressure and water temperature measurements captured the expected tidal dynamics of the estuary and highlighted the benefits of synchronized data acquisition for environmental studies involving spatially distributed processes.

Energy efficiency assessments confirmed the effectiveness of the proposed low-power architecture. The combination of synchronized sleep operation, low-power hard-

ware design, and solar energy harvesting enables long-term autonomous operation with minimal maintenance requirements. The results obtained throughout the deployment demonstrate that the proposed system constitutes a reliable, scalable, and energy-efficient solution for environmental monitoring applications in dynamic and remote environments.

Although the presented results demonstrate the feasibility and effectiveness of the proposed monitoring system, the current validation campaign has some limitations. The long-term deployment was conducted over an eight-day period using two monitoring nodes and one repeater in a single estuarine environment. While these experiments were sufficient to validate the proposed architecture, synchronized operation, and low-power design under real operating conditions, they do not fully characterize the network performance under different seasonal conditions, extreme weather events, or larger-scale deployments. Moreover, the experimental evaluation focused primarily on monitoring data delivery and system functionality rather than on a detailed characterization of the wireless communication link, such as received signal strength indication (RSSI), latency, retransmission statistics, or link-quality indicators. Future work will therefore include longer deployment campaigns involving a larger number of monitoring nodes, comprehensive communication performance analyses, and validation under a broader range of environmental conditions to further assess the scalability and robustness of the proposed architecture.

Supplementary Materials: The supporting information can be downloaded at <https://www.mdpi.com/article/10.3390/electronics15143155/s1>.

Author Contributions: Conceptualization, T.M., S.F.L. and L.G.; methodology, T.M., M.F.M., J.R., H.D., M.M., S.F.L. and L.G.; software, T.M., M.F.M., J.R. and S.F.L.; validation, T.M., S.F.L., M.M., L.G. and J.A.A.; investigation, T.M., M.F.M., J.R., M.M., S.F.L. and L.G.; data curation, T.M.; writing—original draft preparation, T.M. and S.F.L.; writing—review and editing, T.M., S.F.L. and J.A.A.; supervision, M.M., S.F.L. and L.G.; project administration, L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the European Union, through CCDRN, project Atlântida II, NORTE2030-FEDER-0179920.

Data Availability Statement: Data are contained within the article and Supplementary Materials. The data collected by the sensors is freely available at <https://nextsea.dei.uminho.pt>.

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

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
BLE	Bluetooth Low Energy
CPU	Central Processing Unit
CSV	Comma-Separated Values
PDR	Packet Delivery Ratio
GPIO	General-Purpose Input/Output
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
JSON	JavaScript Object Notation
LPWAN	Low-Power Wide-Area Network
MAC	Medium Access Control
MQTT	Message Queuing Telemetry Transport

PCB	Printed Circuit Board
REST	Representational State Transfer
RF	Radio Frequency
RSSI	Received Signal Strength Indication
RTC	Real-time Clock
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
TCP/IP	Transmission Control Protocol/Internet Protocol
UART	Universal Asynchronous Receiver–Transmitter
URL	Uniform Resource Locator
USB	Universal Serial Bus
WSN	Wireless Sensor Network

References

1. Fascista, A. Toward Integrated Large-Scale Environmental Monitoring Using WSN/UAV/Crowdsensing: A Review of Applications, Signal Processing, and Future Perspectives. *Sensors* **2022**, *22*, 1824. <https://doi.org/10.3390/S22051824>.
2. Adu-Manu, K.S.; Abdulai, J.D.; Engmann, F.; Akazue, M.; Appati, J.K.; Baiden, G.E.; Sarfo-Kantanka, G. WSN Architectures for Environmental Monitoring Applications. *J. Sensors* **2022**, *2022*, 7823481. <https://doi.org/10.1155/2022/7823481>.
3. Tabella, G.; Ciuonzo, D.; Paltrinieri, N.; Rossi, P.S. Bayesian Fault Detection and Localization Through Wireless Sensor Networks in Industrial Plants. *IEEE Internet Things J.* **2024**, *11*, 13231–13246. <https://doi.org/10.1109/JIOT.2024.3359646>.
4. Chen, W. Intelligent manufacturing production line data monitoring system for industrial internet of things. *Comput. Commun.* **2020**, *151*, 31–41. <https://doi.org/10.1016/J.COMCOM.2019.12.035>.
5. Majid, M.; Habib, S.; Javed, A.R.; Rizwan, M.; Srivastava, G.; Gadekallu, T.R.; Lin, J.C.W. Applications of Wireless Sensor Networks and Internet of Things Frameworks in the Industry Revolution 4.0: A Systematic Literature Review. *Sensors* **2022**, *22*, 2087. <https://doi.org/10.3390/S22062087>.
6. Huerta, M.K.; Garcia-Cedeno, A.; Guillermo, J.C.; Clotet, R. Wireless Sensor Networks Applied to Precision Agriculture: A worldwide literature review with emphasis on Latin America. *IEEE Geosci. Remote Sens. Mag.* **2021**, *9*, 209–222. <https://doi.org/10.1109/MGRS.2020.3044235>.
7. Ahmed, H.M.; Rashid, A.N. Wireless Sensor Network Technology and Adoption in Healthcare: A Review. *AIP Conf. Proc.* **2022**, *2400*, 20021. <https://doi.org/10.1063/5.0112148>.
8. Hosseinzadeh, M.; Hemmati, A.; Rahmani, A.M. Clustering for smart cities in the internet of things: A review. *Cluster Comput.* **2022**, *25*, 4097–4127. <https://doi.org/10.1007/S10586-022-03646-8>.
9. Sharma, H.; Haque, A.; Blaabjerg, F.; Du, Q.; Sookhak, M.; Shu, L.; Sarkar, N.I.; Abawajy, J.H.; Falcone, F. Machine Learning in Wireless Sensor Networks for Smart Cities: A Survey. *Electronics* **2021**, *10*, 1012. <https://doi.org/10.3390/ELECTRONICS10091012>.
10. Naveen, G.; Prathap, P.M.J. Network energy optimization and intelligent routing in WSN applicable for IoT using self-adaptive coyote optimization algorithm. *Int. J. Commun. Syst.* **2023**, *36*, e5464. <https://doi.org/10.1002/dac.5464>.
11. Santhosh, G.; Prasad, K.V. Energy optimization routing for hierarchical cluster based WSN using artificial bee colony. *Meas. Sensors* **2023**, *29*, 100848. <https://doi.org/10.1016/J.MEASEN.2023.100848>.
12. Manikandan, A.; Venkataramanan, C.; Dhanapal, R. A score based link delay aware routing protocol to improve energy optimization in wireless sensor network. *J. Eng. Res.* **2023**, *11*, 404–413. <https://doi.org/10.1016/J.JER.2023.100115>.
13. Jubair, A.M.; Hassan, R.; Aman, A.H.M.; Sallehudin, H.; Al-mekhlafi, Z.G.; Mohammed, B.A.; Alsaffar, M.S. Optimization of Clustering in Wireless Sensor Networks: Techniques and Protocols. *Appl. Sci.* **2021**, *11*, 11448. <https://doi.org/10.3390/APP112311448>.
14. Qi, Z.; Gu, H.; Chao, Z.; Dong, Z.; Wang, H.; Gershome, A.G. RVH-Cover: A Resultant-Vector and Heap-Based WSN Coverage Optimization Algorithm. *IEEE Internet Things J.* **2026**, *13*, 20811–20821. <https://doi.org/10.1109/JIOT.2026.3664115>.
15. Badiger, V.S.; Ganashree, T.S. Data aggregation scheme for IOT based wireless sensor network through optimal clustering method. *Meas. Sens.* **2022**, *24*, 100538. <https://doi.org/10.1016/J.MEASEN.2022.100538>.
16. Ketshabetswe, K.L.; Zungeru, A.M.; Mtengi, B.; Lebekwe, C.K.; Prabakaran, S.R.S. Data Compression Algorithms for Wireless Sensor Networks: A Review and Comparison. *IEEE Access* **2021**, *9*, 136872–136891. <https://doi.org/10.1109/ACCESS.2021.3116311>.
17. Mishra, M.; Sen Gupta, G.; Gui, X. Investigation of Energy Cost of Data Compression Algorithms in WSN for IoT Applications. *Sensors* **2022**, *22*, 7685. <https://doi.org/10.3390/S22197685>.

18. Sagar, B.B.V.; Munjul, M. Security issues in wireless sensor network—A survey. *J. Discret. Math. Sci. Cryptogr.* **2021**, *24*, 1415–1427. <https://doi.org/10.1080/09720529.2021.1932937>.
19. Faris, M.; Mahmud, M.N.; Salleh, M.F.M.; Alnoor, A. Wireless sensor network security: A recent review based on state-of-the-art works. *Int. J. Eng. Bus. Manag.* **2023**, *15*, 18479790231157220. <https://doi.org/10.1177/18479790231157220>.
20. Adu-Manu, K.S.; Engmann, F.; Sarfo-Kantanka, G.; Baiden, G.E.; Dulemordzi, B.A. WSN Protocols and Security Challenges for Environmental Monitoring Applications: A Survey. *J. Sensors* **2022**, *2022*, 1628537. <https://doi.org/10.1155/2022/1628537>.
21. Bhushan, B. Intrusion Detection System (IDS) for Security Enhancement in Wireless Sensing Applications. *Lect. Notes Netw. Syst.* **2022**, *355*, 39–49. https://doi.org/10.1007/978-981-16-8512-5_5.
22. Sadia, H.; Farhan, S.; Haq, Y.U.; Sana, R.; Mahmood, T.; Bahaj, S.A.O.; Khan, A.R. Intrusion Detection System for Wireless Sensor Networks: A Machine Learning Based Approach. *IEEE Access* **2024**, *12*, 52565–52582. <https://doi.org/10.1109/ACCESS.2024.3380014>.
23. Beltramelli, L.; Mahmood, A.; Osterberg, P.; Gidlund, M.; Ferrari, P.; Sisinni, E. Energy Efficiency of Slotted LoRaWAN Communication with Out-of-Band Synchronization. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 5501211. <https://doi.org/10.1109/TIM.2021.3051238>.
24. Kanoun, O.; Bradai, S.; Khriji, S.; Bouattour, G.; El Houssaini, D.; Ben Ammar, M.; Naifar, S.; Bouhamed, A.; Derbel, F.; Viehweger, C. Energy-Aware System Design for Autonomous Wireless Sensor Nodes: A Comprehensive Review. *Sensors* **2021**, *21*, 548. <https://doi.org/10.3390/S21020548>.
25. Bhatia, S.; Bhadauria, R.V.S.; Mehrotra, K.; Malik, J.; Goel, A.K.; Naib, B.B. A Comparative Analysis and Design Criteria of MAC, SMAC, and TMAC Protocols in Wireless Sensor Network. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*; IEEE: Piscataway, NJ, USA, 2023. <https://doi.org/10.1109/ICCCNT56998.2023.10306758>.
26. Li, Z.; Jin, J.; Zhang, D.; Li, Z. Design and Simulation of Integrated Radio Frequency Transmitter for Wireless Sensor Network Applications. *Rom. J. Inf. Sci. Technol.* **2025**, *28*, 341–352. <https://doi.org/10.59277/ROMJIST.2025.4.03>.
27. Chaturvedi, P.; Daniel, A.K. A Comprehensive Review on Scheduling Based Approaches for Target Coverage in WSN. *Wirel. Pers. Commun.* **2022**, *123*, 3147–3199. <https://doi.org/10.1007/s11277-021-09281-7>.
28. Liu, L.; Guo, X.; Liu, W.; Lee, C. Recent Progress in the Energy Harvesting Technology—From Self-Powered Sensors to Self-Sustained IoT, and New Applications. *Nanomaterials* **2021**, *11*, 2975. <https://doi.org/10.3390/NANO11112975>.
29. Williams, A.J.; Torquato, M.F.; Cameron, I.M.; Fahmy, A.A.; Sienz, J. Survey of Energy Harvesting Technologies for Wireless Sensor Networks. *IEEE Access* **2021**, *9*, 77493–77510. <https://doi.org/10.1109/ACCESS.2021.3083697>.
30. Xu, G.; Shen, W.; Wang, X. Applications of Wireless Sensor Networks in Marine Environment Monitoring: A Survey. *Sensors* **2014**, *14*, 16932–16954. <https://doi.org/10.3390/S140916932>.
31. Pozzebon, A.; Andreadis, A.; Bertoni, D.; Bove, C. A Wireless Sensor Network Framework for Real-Time Monitoring of Height and Volume Variations on Sandy Beaches and Dunes. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 141. <https://doi.org/10.3390/ijgi7040141>.
32. Errachdi, H.; Felis, I.; Madrid, E.; Martínez, R. Bridging the Gap: Challenges and Opportunities of IoT and Wireless Sensor Networks in Marine Environmental Monitoring. *Eng. Proc.* **2023**, *58*, 102. <https://doi.org/10.3390/ECSA-10-16158>.
33. Collotta, M.; Pau, G.; Talty, T.; Tonguz, O.K. Bluetooth 5: A concrete step forward towards the IoT. *IEEE Commun. Mag.* **2017**, *56*, 125–131. <https://doi.org/10.1109/MCOM.2018.1700053>.
34. Su, Z.Y.; Lin, Y.N.; Shen, V.R.L. Intelligent Environmental Monitoring System based on LoRa Long Range Technology. In *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*; IEEE: Piscataway, NJ, USA, 2019; pp. 354–357. <https://doi.org/10.1109/ECICE47484.2019.8942778>.
35. Azmi Ali, N.A.; Abdul Latiff, N.A. Environmental Monitoring System Based on LoRa Technology in Island. In *2019 IEEE International Conference on Signals and Systems (ICSigSys)*; IEEE: Piscataway, NJ, USA, 2019; pp. 160–166. <https://doi.org/10.1109/ICSIGSYS.2019.8811066>.
36. Matos, T.; Rocha, J.L.; Dinis, H.; Faria, C.L.; Martins, M.S.; Henriques, R.; Goncalves, L.M. A low-cost, low-power and low-size multi-parameter station for real-time and online monitoring of the coastal area. In *OCEANS 2022, Hampton Roads*; IEEE: Piscataway, NJ, USA, 2022. <https://doi.org/10.1109/OCEANS47191.2022.9977347>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.