الجـــمـــهوريـــــة الـجـــــزائريـــــة الـديمقـــراطيـــة الـشعبيـــــة
People’s Democratic Republic of Algeria
وزارة الـتـعـليـــــم الـعـالـي و الـبحـث الـعلـــمـي
Ministery of Higher Education and Scientific Research
جــــامعــــة عبـد الحميـــد بـن باديــــس – مستـغانــــــم
Abdel Hamid Ibn Badis University – Mostaganem
كـليــــــة الـعلــــــوم والـتكنولــوجـــــيـا
Faculty of Sciences and Technology
قـسم الهندسة الكهربائية

Department of Electrical Engineering

ACADEMIC MASTER'S THESIS

Field: Electrical Engineering
Specialty: Automation and Industrial Computing

Thesis Title

Industrial control and monitoring system

Presented By :

1-BENHACHLEF Ikram.

2-BOUAZZA Sirine Minet Allah.

Defended before the jury composed of:

President:	Djelti Benbella	MCB	Université of Mostaganem
Examiner:	Henni Sidahmed	MCA	Université of Mostaganem
Supervisor:	ABDERRAHMANE Abdelkader	MCB	Université of Mostaganem

Academic year 2025 / 2026

الجـــمـــهوريـــــة الـجــــزائريــــــة الـديمقـــراطيـــة الـشعبيــــــة
People's Democratic Republic of Algeria
وزارة الـتـعـليــــــم الـعـالــي و الـبحـث الـعلـــمـي
Ministery of Higher Education and Scientific Research
جـــــامـعـــــة عبـد الحميـــد بـن باديــــس – مستـغانـــــم
Abdel Hamid Ibn Badis University – Mostaganem
كـليــــــة الـعلـــــوم والـتكنولـــوجـــــيـا
Faculty of Sciences and Technology
قـــسم الهندسة الكهربائية

Department of Electrical Engineering

MEMOIRE DE FIN D'ETUDES

DE MASTER ACADEMIQUE

Filière:Génie éléctrique

Spécialité: Automatique et informatique industrielle

Thème

Architecture de Télé-supervision Industrielle Basée sur LoRa P2P et Node-RED.

Présenté par:

1-BENHACHLEF Ikram.

2-BOUAZZA Sirine Minet Allah.

Defended before the jury composed of:

Président:	Djelti Benbella	MCB	Université of Mostaganem
Examinatrice:	Henni Sidahmed	MCA	Université of Mostaganem
Encadrent:	ABDERRAHMANE Abdelkader	MCB	Université of Mostaganem

Année universitaire 2025 / 2026

Acknowledgments

In the name of Allah, the Most Gracious, the Most Merciful. We begin by offering our deepest gratitude to Allah, the Source of all wisdom and light, for illuminating our path, granting us the strength to persevere, and allowing this collective endeavor to bloom.

True success is not merely found in the destination, but in the grace of those who walked beside us. We are profoundly indebted to our supervisor, Abderrahmane Abdelkader, DJELTI BENBELLA and HENNI SIDEHMED whose mentorship has been a steady compass guiding our research. Your insights and patience have been the foundation upon which this work stands, and for that, we remain forever grateful.

We extend our heartfelt appreciation to all the professors who have walked with us throughout our Master's career at University of Mostaganem. You have not only imparted knowledge but have challenged our minds and refined our vision. Your lessons are the seeds from which our academic growth has sprung.

Our sincere thanks go to the staff and colleagues at Research Lab. You provided us with more than just resources and technical support; you fostered a collaborative home where ideas could take root and thrive. Thank you for the environment you created, which allowed this project to flourish.

Finally, to our dear friends and the rest of our circle—thank you for being our sanctuary when the weight of the journey felt heavy. Your belief in us, your laughter, and your steadfast presence provided the warmth we needed to keep going. This achievement is a reflection of the unity and support you have so generously shared with us.

Dedication

We often think that success is measured by money or fame. I believe that success is the glow forming in the eyes of those who believed in me, even when I was feeling low and left behind. Hence, we should be grateful to all those who stood by our side and gave us the strength to continue on our path.

Just like any living creature on this earth, none of us would bloom without God's will; first and foremost, I thank God for brightening my view and guiding my steps.

To my parents, **BOUAZZA Salima and Djilali**: thank you for being my anchors. To my mother, **Salima**, thank you for the endless faith you placed in me and for fighting for me every step of the way. To my father **Djilali**, thank you for your wisdom and for standing by my side with such pride. I only wish to return half of the love and support you have both poured into my life.

To my dearest brother **Mohammed**, my sisters **Ritadj and Nouran**, thank you for being the pillars holding me together when times were tough. Thank you for your unconditional love and care.

To my aunt **Asma**, thank you for your warmth, your guidance, and for always being a source of grace in my life. Your presence has been a steady light throughout this journey.

To my colleague and friend **Ikram**, thank you for your professional collaboration and for the shared burdens that made this path much easier to navigate. Your dedication and support have been invaluable to this work.

Last but never the least, my deepest appreciation to my dear group of friends **Meriem Douae** and all of you who have walked beside me: you are the truest companions. Thank you for the laughter, the patience, and for standing by me through every challenge. This achievement is as much yours as it is mine.

Sirine

Dedication

Success, I believe, is not measured by wealth or fame, but by the glow in the eyes of those who believed in us — even in our lowest moments. To all who stood by my side, this is for you.

First and foremost, all praise to God, for guiding my steps and brightening my path.

To my parents, **Abed** and **Djamila** — thank you for being my anchors, my strength, and my greatest supporters. Your love is the foundation of everything I am.

To my brother **Abderrahmane** and my sister **Ilhem** — thank you for holding me together when times were tough, and for your unconditional love.

To my dear friends **Yamina** and **Amel** — your warmth and grace have been a steady light throughout this journey.

To my colleague and friend **Sirine** — thank you for your collaboration, your dedication, and for making this path so much easier to walk.

And to my wonderful friends **Meriem, Douae,** and each one of you who walked beside me — thank you for the laughter, the patience, and the unwavering support. This achievement is as much yours as it is mine..

Ikram

Résumé :

Face aux exigences croissantes des environnements industriels modernes en matière de surveillance fiable, de longue portée et de faible consommation énergétique, les systèmes de télé-supervision traditionnels montrent leurs limites, notamment dans les sites isolés ou hostiles où le déploiement d'infrastructures réseau classiques reste coûteux et complexe.

Ce projet de fin d'études propose la conception et la réalisation d'une architecture de télé-supervision industrielle basée sur la communication LoRa Point-à-Point (P2P) et la plateforme de traitement de flux Node-RED. Le système développé repose sur des noeuds capteurs intelligents construits autour du microcontrôleur ESP32 couplé à un module LoRa, chargés d'acquérir en temps réel des grandeurs physiques industrielles critiques, puis de les transmettre sans fil sur de longues distances avec une consommation d'énergie minimale.

L'architecture adoptée s'articule autour de deux entités principales : un noeud émetteur assurant l'acquisition et la transmission des données, et un noeud récepteur passerelle chargé de réceptionner, traiter et acheminer ces données vers un tableau de bord de supervision. La logique embarquée est gérée par un firmware développé sous FreeRTOS, modélisé à travers des diagrammes UML (classes, séquences, activités) et une machine d'états rigoureuse pour garantir la fiabilité du cycle acquisition-transmission-réception-visualisation.

Côté supervision, un serveur MQTT est déployé pour assurer le transport léger des données, tandis que Node-RED est utilisé pour construire une interface HMI intuitive accessible depuis smartphone, offrant une visualisation temps réel des paramètres industriels surveillés.

Une validation expérimentale du prototype complet est réalisée afin d'évaluer les performances du système en termes de portée, de latence, de fiabilité de transmission et de réactivité de l'interface de supervision.

Ce travail s'inscrit dans le domaine de l'Industrie 4.0, à l'intersection de l'IIoT, du SCADA et des réseaux LPWAN, et ouvre des perspectives prometteuses vers des architectures de supervision distribuée, évolutives et économiquement accessibles pour les environnements industriels contraignants.

Mots clés : Contrôle industriel – Supervision industrielle – SCADA – Internet des Objets Industriel (IIoT) – ESP32 – LoRa – MQTT – Modbus TCP/IP – Node-RED – Automate Programmable (API/PLC) – Edge Computing – Capteurs industriels – ESP-NOW – IHM (Interface Homme-Machine) – Télésupervision.

Abstract

Facing the growing demands of modern industrial environments for reliable monitoring, long-range communication, and low energy consumption, traditional supervisory systems show their limitations, particularly in isolated or hostile sites where deploying classical network infrastructures remains costly and complex.

This final year project proposes the design and implementation of an industrial tele-supervision architecture based on LoRa Point-to-Point (P2P) communication and the Node-RED data flow platform. The developed system relies on intelligent sensor nodes built around the ESP32 microcontroller coupled with a LoRa module, responsible for acquiring critical industrial physical quantities in real time and transmitting them wirelessly over long distances with minimal energy consumption.

The adopted architecture revolves around two main entities: a transmitter node handling data acquisition and transmission, and a gateway receiver node responsible for receiving, processing, and forwarding data to a supervision dashboard. The embedded logic is managed by firmware developed under FreeRTOS, modeled through UML diagrams (classes, sequences, activities) and a rigorous state machine to ensure the reliability of the acquisition-transmission-reception-visualization cycle.

On the supervision side, an MQTT broker is deployed to ensure lightweight data transport, while Node-RED is used to build an intuitive HMI interface accessible from a smartphone, offering real-time visualization of monitored industrial parameters.

An experimental validation of the complete prototype is carried out to evaluate system performance in terms of range, latency, transmission reliability, and interface responsiveness.

This work falls within the field of Industry 4.0, at the intersection of IIoT, SCADA, and LPWAN networks, and opens promising perspectives toward distributed, scalable, and economically accessible supervision architectures for demanding industrial environments

.**Keywords:** Industrial Control – Industrial Monitoring – SCADA – Industrial IoT (IIoT) – ESP32 – LoRa Communication – MQTT Protocol – Modbus TCP/IP – Node-RED – PLC (Programmable Logic Controller) – Edge Computing – Industrial Sensors – ESP-NOW – HMI (Human-Machine Interface) – Remote Supervision.

ملخص

في مواجهة المتطلبات المتزايدة للبيئات الصناعية الحديثة من حيث المراقبة الموثوقة والمدى البعيد وانخفاض استهلاك الطاقة، تُظهر أنظمة الإشراف عن بُعد التقليدية حدودها، لا سيما في المواقع المعزولة أو القاسية حيث يظل نشر البنى التحتية للشبكات الكلاسيكية مكلفاً ومعقداً.

يقترح مشروع التخرج هذا تصميم وتنفيذ بنية معمارية للإشراف الصناعي عن بُعد، تستند إلى تقنية الاتصال LoRa نقطة إلى نقطة (P2P) ومنصة معالجة تدفق البيانات Node-RED. يعتمد النظام المطوَّر على عقد استشعار ذكية مبنية حول المتحكم الدقيق ESP32 المقترن بوحدة LoRa، تتولى اكتساب القيم الفيزيائية الصناعية الحرجة في الوقت الفعلي ونقلها لاسلكياً عبر مسافات بعيدة بأقل استهلاك ممكن للطاقة.

تتمحور البنية المعتمدة حول كيانين رئيسيين: عقدة مُرسِلة تتولى اكتساب البيانات ونقلها، وعقدة مستقبِلة تعمل كبوابة لاستقبال البيانات ومعالجتها وإرسالها إلى لوحة تحكم للإشراف. تُدار المنطق المدمج ببرنامج ثابت (Firmware) مطوَّر باستخدام FreeRTOS، مُصمَّم وفق مخططات UML (الفصول، التسلسلات، الأنشطة) وآلة حالة صارمة لضمان موثوقية دورة الاكتساب-النقل-الاستقبال-التصور.

على صعيد الإشراف، يُنشأ وسيط MQTT لضمان النقل الخفيف للبيانات، بينما تُستخدم منصة Node-RED لبناء واجهة HMI بديهية يمكن الوصول إليها من الهاتف الذكي، مما يوفر تصوراً في الوقت الفعلي للمعاملات الصناعية المراقَبة.

يُجرى تحقق تجريبي كامل للنموذج الأولي لتقييم أداء النظام من حيث المدى، وزمن الاستجابة، وموثوقية الإرسال، وسرعة تفاعل واجهة الإشراف.

يندرج هذا العمل في إطار الصناعة 4.0، عند تقاطع إنترنت الأشياء الصناعي (IIoT)، وأنظمة SCADA، وشبكات LPWAN، ويفتح آفاقاً واعدة نحو بنيات إشراف موزَّعة وقابلة للتوسع ومتاحة اقتصادياً للبيئات الصناعية المتطلبة.

الكلمات المفتاحية: LoRa P2P، ESP32، FreeRTOS، MQTT، Node-RED، إشراف عن بُعد، IIoT، SCADA، LPWAN، (HMI) الصناعة 4.0.

Table OF contents:

LISTE OF FIGURES:

LIST OF TABLES :

I. GENERAL INTRODUCTION

In today's world, humanity coexists with the developments of the Second Stage of the Industrial Revolution, which reshaped global production methods. From this period onward, agricultural production processes began to be predominantly carried out using machines, which contributed decisively to a massive increase in food production. This highly technified agricultural mechanization fundamentally transformed the historical relationship between humans, land, and production output.

Among the advancements made possible by the widespread use of machinery—especially in agriculture—modern crop irrigation systems stand out. Irrigation is a vital agricultural method that encompasses a precise set of techniques and engineering equipment aimed at supplying water deficiencies to planted, cultivable species. Irrigated agriculture can be defined as an economic activity that explores agricultural, forestry, and ornamental crops and pastures, using advanced irrigation or drainage techniques to secure food supply lines.

It is well known that water is a natural resource of paramount importance across various sectors, and its excessive or unmonitored use severely compromises its long-term availability. Water is a finite, irreplaceable resource. As the absolute foundation of life, civil societies, and global economies, it carries profound ecological and economic value. Considering this vital importance to humanity, its daily use must be made as efficient as possible. In this context, automated water monitoring systems applied directly to agriculture can contribute decisively to a more rational, data-driven approach to consumption.

Among the various strategies for implementing smart irrigation monitoring systems—especially low-cost solutions in the form of functional prototypes—those built using integrated microcontroller boards stand out. Research involving accessible methods, open-source resources, and modern engineering techniques to build and evaluate these prototypes has accelerated rapidly over the past few years. As new approaches are developed, the procedures for constructing and deploying these monitoring systems are refined, expanding our practical engineering knowledge.

However, formal records in technical literature aiming to identify standardized methods, accessible resources, and calibration techniques for building microcontroller prototypes within the Internet of Things (IoT) environment—specifically tailored to open-field irrigation monitoring—remain quite limited. Only a few systematic reviews have actively explored the intersection of low-cost hardware platforms and precise agricultural telemetry.

I.1 Context of SCADA and IoT Systems

I.1.1 SCADA Systems: Definition and Functions

A SCADA system (*Supervisory Control and Data Acquisition*) is an industrial computer system designed for the supervision, monitoring, and control of large-scale industrial processes and infrastructures. It ensures the real-time collection of data from sensors and field instruments, their processing and analysis, as well as automated or manual decision-making based on that information.

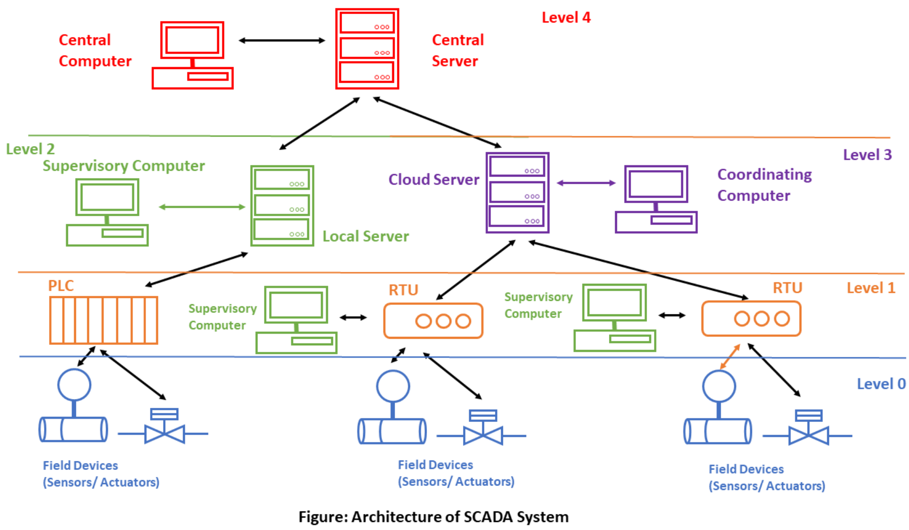


Figure I 1. SCADA System Architecture [1].

SCADA systems play a central role in numerous industrial sectors, including energy production, water management, electrical distribution networks, transportation, and manufacturing. Their functions span several complementary areas. Real-time monitoring constitutes their primary mission: data from sensors and equipment is continuously collected and visualized through graphical interfaces known as HMIs (Human-Machine Interfaces).

These systems also enable remote control of industrial equipment, allowing operators to activate, deactivate, or adjust machine operating parameters from a centralized workstation. Furthermore, the acquisition and archiving of data in dedicated databases support historical analysis and the generation of reports on performance, trends, and incidents. Alarm management ensures the detection of abnormal conditions and the immediate notification of operators, while security protocols protect the system against cyberattacks and unauthorized access.

I.1.2 Fields of Application of SCADA Systems

SCADA systems are deployed worldwide across a wide variety of sectors. In the production, transmission, and distribution of gas and electricity, they monitor current flow, line voltage, and the status of circuit breakers. In the building sector, they manage heating, ventilation, air conditioning, and lighting systems. In manufacturing industries, they contribute to the optimization of process automation and quality control. Finally, in public transportation, they regulate and monitor the electrical power supply for subways, trams, and trolleybuses, ensuring the safety and efficiency of operations.

I.1.3 Main Components of a SCADA System

A SCADA system relies on a set of interdependent hardware and software components. Sensors measure physical quantities such as temperature, pressure, or flow rate and convert them into usable electrical signals, while actuators execute control commands by acting directly on the industrial process. Remote Terminal Units (RTUs) collect sensor data in the field, perform basic control tasks, and transmit information to the central system. Programmable Logic Controllers (PLCs) complement this setup by automating industrial processes; robust and configurable, they find applications in both heavy industry and civil uses such as elevator control or traffic light regulation.

Communications between the various components rely on wired field networks (Ethernet, Modbus) or wireless networks (Wi-Fi, radio), as well as long-distance networks involving cellular networks, satellites, or dedicated telephone lines. On the server side, the architecture includes a database server for historical archiving, an application server for managing interfaces and analytics, and a communication server ensuring exchanges between system components. The Human-Machine Interface (HMI) completes the system by providing the operator with an interactive graphical environment to visualize system states, monitor parameters, receive alerts, and consult data history. Modern HMIs come in several forms, including touchscreens, voice interfaces, gesture-based systems, haptic feedback devices, and three-dimensional visualizations.

I.2 The Industrial Internet of Things (IIoT): A Modern Extension of SCADA

With the rise of digital technologies, the Industrial Internet of Things (IIoT) has established itself as a natural evolution of traditional SCADA systems, bringing enhanced connectivity, greater flexibility, and advanced data exploitation capabilities. Built on the principle that intelligent machines outperform humans in collecting and interpreting real-time data, the IIoT — also referred to as Industry 4.0 — leverages data generated by industrial equipment to improve manufacturing processes and decision-making.

The architecture of an IoT system is organized around three levels: connected objects, which are physical devices equipped with sensors and actuators; IoT gateways, which handle data aggregation and preprocessing; and cloud infrastructure, which enables centralized storage and processing. Among the communication protocols commonly used are Modbus, Wi-Fi, Bluetooth, and ZigBee.

The integration of IoT into modern industrial systems brings several decisive advantages. Wireless connectivity eliminates the constraints associated with costly cabling and promotes flexible deployment. Low-power sensors can operate for several years on batteries or through energy harvesting, considerably reducing maintenance costs. Long-range protocols such as LoRa ensure coverage over several kilometers, well suited to large-scale industrial or agricultural sites. Finally, the combination of edge computing for immediate decisions and cloud computing for long-term analysis makes it possible to transform raw data into actionable insights, particularly for predictive maintenance and resource optimization.

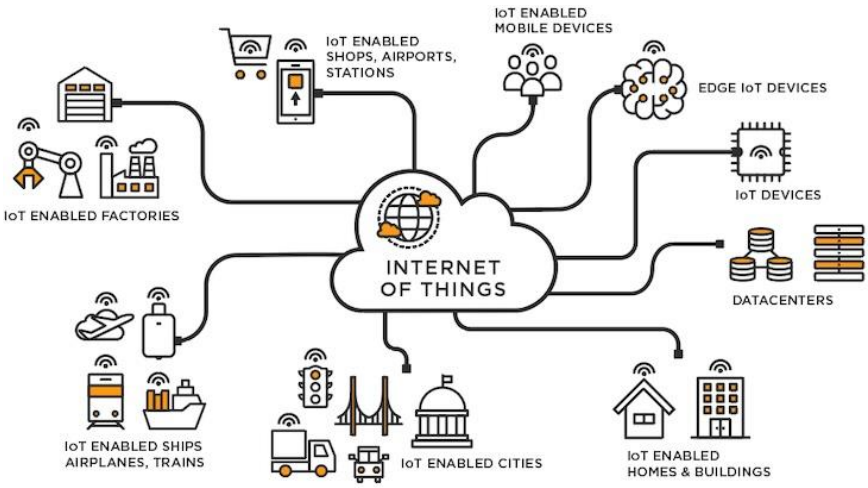


Figure I 2. Internet of Things (IoT) Ecosystem Diagram [2].

Problem Statement

Despite the progress achieved, technological solutions based on Wi-Fi, Bluetooth, or cellular networks (3G/4G) face significant technical constraints. Their high energy consumption makes them unsuitable for isolated battery-powered sensors. Their limited transmission range does not allow coverage of large areas or effective operation in constrained environments. Moreover, their deployment entails high costs and a dependency on heavy network infrastructures.

In light of these limitations, the present study raises the following question:

- **How can a remote supervision architecture be designed to be simultaneously long-range, energy-efficient, and capable of intelligently managing real-time data flows?**

CHAPTER I

STATE OF THE ART

1.1. Industrial Monitoring Technologies: (SCADA, IIoT, and Edge Computing)

Modern industrial environments rely on a combination of complementary technologies to ensure effective supervision, data processing, and decision-making. Three of these technologies — SCADA, IIoT, and Edge Computing — form the backbone of contemporary industrial monitoring architectures, each fulfilling a distinct and essential role.

1.2. SCADA (Supervisory Control and Data Acquisition)

Systems serve as the control center of industrial facilities. Their primary function is the real-time monitoring and control of industrial processes, achieved by gathering data from PLCs and sensors and presenting it through a graphical Human-Machine Interface (HMI) for operators. SCADA systems are recognized for their high reliability, deterministic control capabilities, and localized safety management. However, they have traditionally operated as isolated systems, which limits their ability to scale across multiple geographical sites or integrate with higher-level business intelligence and artificial intelligence platforms.

The Industrial Internet of Things (IIoT) extends industrial data beyond the local plant floor by connecting intelligent devices — such as sensors and instruments — through internet protocols to cloud platforms. This approach enables massive scalability, supports Big Data analytics, and facilitates predictive maintenance by anticipating equipment failures before they occur. Its main limitation lies in its high dependency on internet connectivity, which simultaneously expands the cybersecurity attack surface.

Edge Computing addresses the latency and bandwidth constraints associated with cloud-centric architectures by processing data close to its source, at the edge of the network, rather than transmitting everything to a distant server. This approach enables split-second local decision-making, reduces bandwidth consumption by filtering data before transmission, and enhances security by keeping sensitive information within the local network. Its primary drawback remains its limited storage and processing capacity compared to full cloud environments.

These three technologies do not operate in isolation but rather form an integrated, layered architecture. At the first level, sensors gather raw data directly from machinery. At the second level, edge computing processes this data immediately for high-speed local control. At the third level, SCADA provides supervisors with a comprehensive visual overview of the process. Finally, at the fourth level, IoT and cloud platforms receive critical data for high-level analysis of factory performance over extended periods.

1.3. Industrial Sensors and the Importance of Monitoring Physical Quantities

Sensors constitute a fundamental element of industrial automation, enabling the continuous collection of data on processes, machinery, and the production environment. They convert physical quantities — such as temperature, pressure, flow rate, position, and speed — into electrical signals that can be interpreted and used by automation and control systems.

In industrial processes, sensors provide precise, real-time information regarding equipment status, production levels, product quality, and safety conditions. This data is subsequently processed by control systems to trigger appropriate actions based on predefined operational requirements. The diversity of physical quantities to be monitored has led to the development of a wide range of specialized sensors, each designed to measure a specific variable.

Temperature sensors monitor the thermal state of equipment or the surrounding environment to ensure proper functioning and prevent overheating. Pressure sensors measure fluid pressure in hydraulic or pneumatic systems, enabling regulation within safe operating limits. Flow sensors quantify the movement of liquids or gases through pipelines, which is essential for controlling and optimizing production processes. Position sensors determine the precise location of machine components, allowing accurate control of actuator movements. Proximity sensors detect the presence or absence of nearby objects and are commonly used for positioning and obstacle detection. Level sensors measure the quantity of liquids or raw materials in tanks and vessels, preventing overflow or supply shortages. Finally, vibration sensors detect abnormal oscillations in machinery, serving as early indicators of mechanical wear or impending failure.

All these sensors can be connected to Programmable Logic Controllers (PLCs), distributed control systems, or SCADA supervisory software, thereby forming an integrated monitoring network that automates alerting and regulation tasks and contributes to the efficiency, safety, and reliability of industrial operations.

1.4. Automation via PLCs for Tele-Supervision

1.4.1. Architecture and Operation of the Programmable Logic Controller

In the context of industrial automation, the Programmable Logic Controller (PLC) acts as the central processing unit of the automated system. Its primary role is to bridge the gap between the user program which encodes the control logic and the physical actions carried out on the field by actuators and other operative elements.

1.4.2. Hardware Structure and the Role of the CPU

The PLC is built around a Central Processing Unit (CPU), which interprets input signals and executes the control program to generate the appropriate output commands. The power supply unit converts the main AC voltage into the low DC voltage required by the internal electronics. The processor itself contains an Arithmetic and Logic Unit (ALU) responsible for calculations, alongside a control unit that synchronizes all operations through an internal clock, typically operating between 1 and 5 MHz.

1.4.3. Communication System

The Internal Buses Information flows between the processor, memory, and interfaces through three types of internal buses. The data bus transports actual values, typically encoded in 8 or 16 bits. The address bus locates specific information within the memory space. The control bus carries synchronization signals that coordinate read and write operations across all components.

1.4.4. Memory Management

The PLC relies on three essential types of memory to ensure system reliability. Read-Only Memory (ROM) permanently stores the controller's operating system and remains unaffected by power loss. Random Access Memory (RAM) temporarily holds sensor states, counter values, and the active user program; a backup battery preserves this data in the event of a power failure. Erasable Programmable Read-Only Memory (EPROM) is used for the permanent storage of the finalized control program.

1.4.5. Input/Output Interfaces

The input/output interfaces represent the point of interaction between the PLC and the physical world. Signals are classified into three categories: discrete signals, which represent binary on/off states such as those produced by switches and push-buttons; digital signals, which consist of sequences of binary states; and analog signals, which are continuous variable signals proportional to physical quantities such as temperature or pressure.

1.4.6. Communication and Tele-Supervision of the PLC

In modern automated systems, the PLC is not limited to local process control. It is also capable of communicating with external systems to enable tele-supervision and remote control. Communication interfaces allow the PLC to exchange data with industrial computers, HMI terminals, or other PLCs through industrial networks such as Ethernet, PROFIBUS, Modbus, or MPI. Through these networks, the PLC transmits real-time information including sensor states, actuator statuses, and system alarms, while simultaneously receiving commands from remote supervision stations.

Tele-supervision consists of monitoring and controlling an automated system from a distance using SCADA or HMI software installed on a computer. This software presents process information in graphical form, enabling operators to assess system status at a glance, track parameter evolution, detect faults, and intervene remotely by issuing commands — such as starting or stopping a machine — directly to the PLC.

Information exchange in a tele-supervised system operates bidirectionally. Sensors transmit their measurements to the PLC, which processes them and forwards the relevant data to the supervision system. Conversely, the supervision system sends commands back to the PLC, which applies them to the actuators. This mode of operation improves overall system management, accelerates fault response, and enhances operational safety. Importantly, even when connected to a remote supervision system, the PLC retains full autonomous operation continuing to execute its local program independently in the event of a communication failure.

1.4.7. LPWAN Protocols and IoT Communication Protocols

1.4.7.1 Definition and Characteristics of LPWAN

LPWAN (Low Power Wide Area Networks) are wireless communication protocols specifically designed for IoT applications. They are characterized by four key properties: long-range connectivity reaching up to several kilometers, low power consumption optimized for energy efficiency, low data rates suited to the transmission of small information packets, and extended battery life capable of lasting up to ten years under typical operating conditions. These characteristics make LPWAN protocols particularly well suited for low-traffic IoT sensors and remote monitoring applications in environments where frequent maintenance or battery replacement is impractical

1.4.7.2. Main Characteristics of LPWAN

The following table summarizes the technical parameters, including range, data rates, frequency bands, and typical applications, that define the performance bounds of LPWAN frameworks.

Table 1. 1 Key Operational Characteristics of LPWAN Technologies

Characteristic	Description
Range	2 to 15 km (Urban / Rural)
Data Rate	0.3 kbps to 50 kbps
Power Consumption	Very Low
Frequency	ISM Bands (868 MHz, 915 MHz)
Cost	Low
Topology	Star
Applications	Smart City, Agriculture,

1.4.7.3. Main LPWAN Protocols: LoRa and LoRaWAN

Among the available LPWAN technologies, LoRa and LoRaWAN stand out as particularly well-suited solutions for long-range, low-power IoT deployments. LoRaWAN is a networking specification designed to wirelessly connect battery-operated devices to the internet across regional, national, or global networks. It addresses key IoT requirements including bidirectional communication, end-to-end security, mobility, and localization services.

LoRaWAN operates in unlicensed radio frequency bands and is distinguished by its long-range capabilities, reaching up to 15 kilometers in rural environments, combined with very low power consumption and the ability to connect a large number of devices to a single network. Its architecture is organized across four levels: end devices that collect and transmit data, gateways that relay messages between devices and the network, network servers that manage data routing and protocol handling, and application servers that process the final data for end users. This layered structure enables a scalable and flexible network infrastructure capable of supporting a wide diversity of IoT applications.

1.4.8. Industrial Communication Protocols and Data Transfer Mechanisms

1.4.8.1. Message Queuing Telemetry Transport (MQTT)

MQTT is a lightweight messaging protocol built on the publish/subscribe communication model. It was specifically designed for remote connections where minimal code footprint and limited network bandwidth are critical constraints. Its operation relies on a central broker that manages message exchanges between publishers — typically sensors or field devices — and subscribers, such as monitoring dashboards or applications.

MQTT offers several significant advantages for IoT deployments. Its very low protocol overhead minimizes data consumption, making it highly efficient for resource-constrained environments. It provides three levels of Quality of Service (QoS) to guarantee message delivery according to the reliability requirements of each application. Its event-driven architecture also makes it particularly well suited for real-time notifications and continuous monitoring scenarios. For these reasons, MQTT is considered the protocol of choice for battery-powered IoT devices and systems operating over unstable or low-bandwidth networks.

1.4.8.2. HyperText Transfer Protocol (HTTP)

HTTP is the foundational protocol for data exchange on the World Wide Web. In an IoT context, it operates according to a request/response model based on a client-server architecture. A connected device acting as the client sends a request to a remote server through a REST API, and waits for a response formatted in JSON or XML.

HTTP benefits from broad compatibility with virtually all existing web services and IT infrastructures, making its integration straightforward in most development environments. Security is readily achievable through HTTPS, which encapsulates HTTP communications within TLS/SSL encryption layers. Its familiarity to developers with a background in web technologies further simplifies implementation. However, due to its relatively higher overhead compared to MQTT, HTTP is best suited for devices that are not subject to strict power constraints and for applications involving periodic rather than continuous data uploads.

1.4.8.3. Modbus TCP/IP (Modbus Transmission Control/Internet Protocol)

Modbus TCP/IP is a communication protocol derived from the original Modbus RTU standard and adapted for use over Ethernet networks. It follows a client/server model in which the client, acting as the master, sends requests to a server acting as the slave, which responds with the requested data. Communication is established over TCP port 502, encapsulating standard Modbus frames within IP packets to ensure compatibility with modern network infrastructures.

This protocol is widely adopted in industrial automation and SCADA systems for data exchange between field devices and supervisory platforms. Its foundation on the TCP transport layer guarantees reliable data delivery through built-in error checking and acknowledgment mechanisms. It enables fast and deterministic communication over existing Ethernet infrastructure, and benefits from near-universal support across PLCs, RTUs, sensors, and industrial software platforms. Modbus TCP/IP is therefore particularly well suited for industrial environments requiring real-time monitoring and control of equipment over local networks or the internet.

1.4.9. ESP-NOW Wireless Communication Protocol

Espressif Systems is a leading provider of IoT solutions, dominating the semiconductor industry. In 2023, the company reported nearly 200 million chips sold and has shipped over one billion SoCs since the introduction of its groundbreaking ESP8266 model in 2014.1 These figures not only underline Espressif's substantial influence in the semiconductor market but also affirm its pivotal role in shaping the IoT and smart device sector.

Among other innovations, Espressif has developed ESP-NOW, a proprietary protocol designed for direct device-to-device wireless communication. This protocol, which operates at the data-link layer and relies on Wi-Fi action frames, is provided as an open-source extension module for their ESP-IDF SDK. ESP-NOW is compatible with various Espressif SoCs that feature Wi-Fi connectivity, facilitating seamless and efficient communication between devices. It is specifically engineered to offer quick responses, thereby reducing the delays and packet loss that often occur in congested network environments. This makes ESP-NOW an ideal solution for developers looking to enhance connectivity in their IoT applications.

Nozomi Networks has conducted a comprehensive security analysis and dissection of the ESP-NOW protocol, and this white paper details the findings from our research. We begin by outlining the main features of ESP-NOW and dissecting the frame format utilized in the protocol. Following this introduction, we explain our vulnerability assessment process, which employed automated code analysis tools such as SemGrep and Weggli. These tools helped us identify critical segments of the code that required further investigation. Ultimately, our research uncovered two potential vulnerabilities. The first, an Out-of-Band read access, was determined to be non-exploitable upon deeper analysis. The second, a replay attack vulnerability, proved to be exploitable. We provide a detailed analysis of this vulnerability and include a proof of concept to demonstrate the potential security implications associated with ESP-NOW.

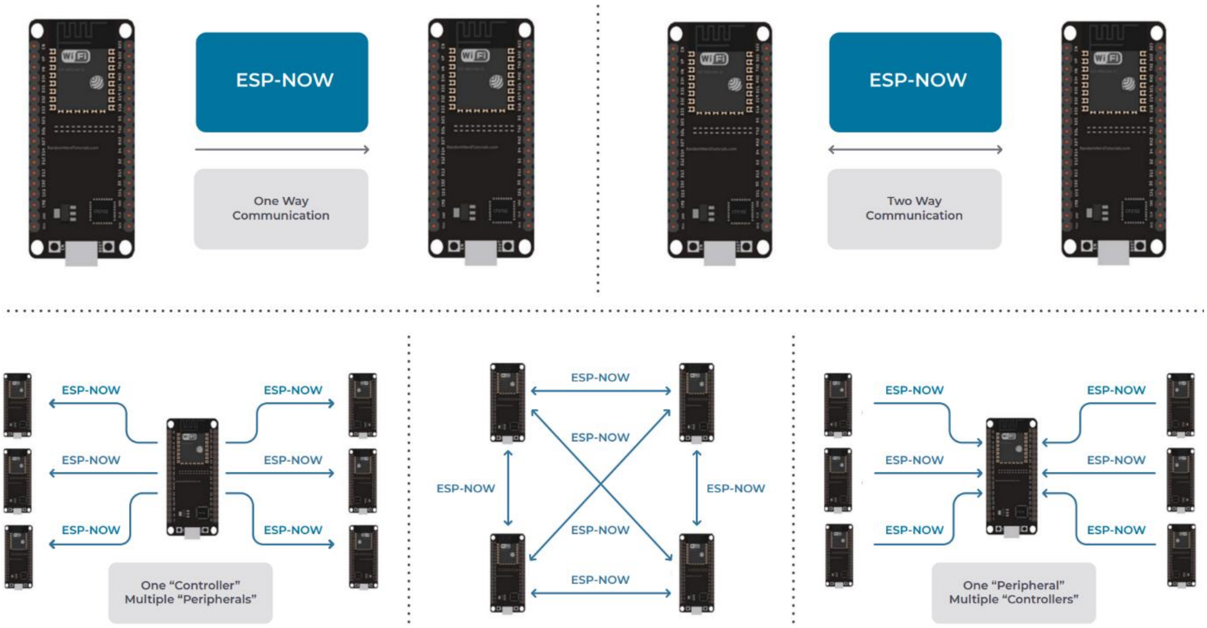


Figure 1. 1. Examples of ESP-NOW One-Hop Topologies[3]

CHAPTER II

HARDWARE AND SOFTWARE DESIGN.

2.1. Introduction

To implement an IoT-based smart agriculture system, we first need to determine the functionalities required for it. In our work, we are considering a system that allows, among other things, data acquisition through digital and analog sensors, access to this data both locally and remotely, in other words, real-time monitoring, and finally, an automatic irrigation system. Based on this, we will present in this chapter the various components and software on which our choice was made.

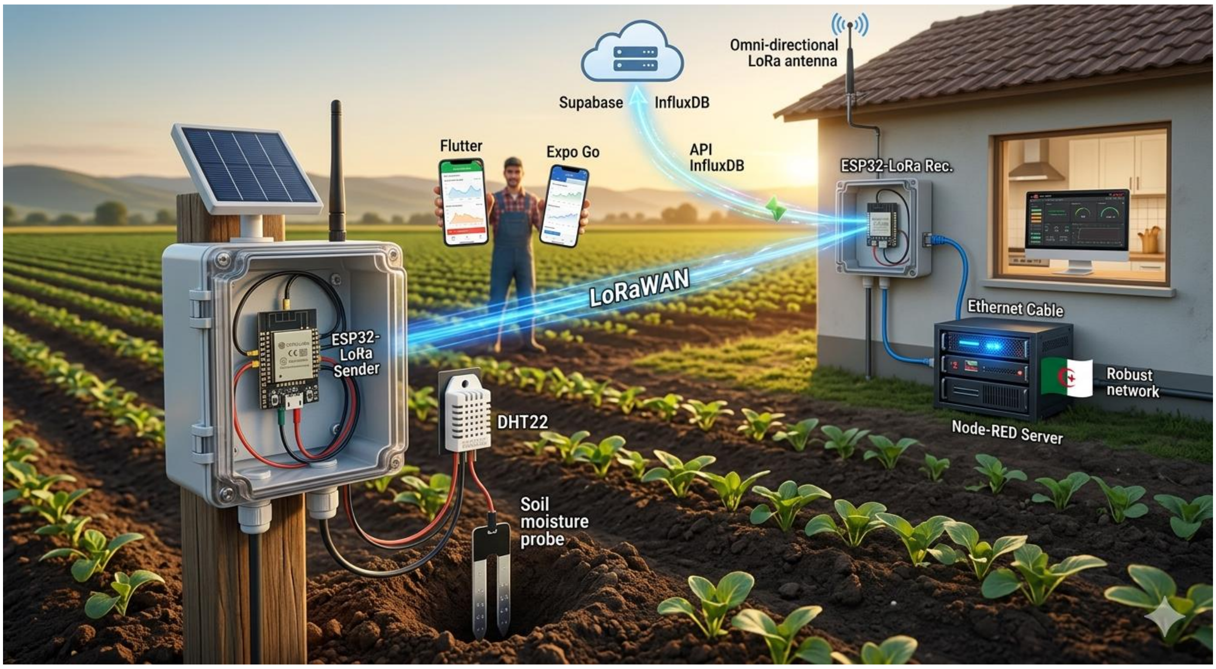


Figure 2. 1. IoT-Based LoRaWAN Architecture for Precision Agriculture Monitoring.

2.2. Hardware

2.2.3. ESP32 Microcontroller

2.2.3.1 Definition

It is a series of microcontrollers of the system-on-a-chip (Soc) type of Expressif Système, based on the Xtensa LX6 architecture of Tensica (en), integrating the management of Wi Fi and Bluetooth (up to LE 5.0 and 5.11) in dual mode, and a DSP, it has met with some success in recent years both for its cost, its capabilities and its integration developed by ExpressifSystème, in the last quarter of 2016 and it is the successor of the ESP8266 microcontroller.

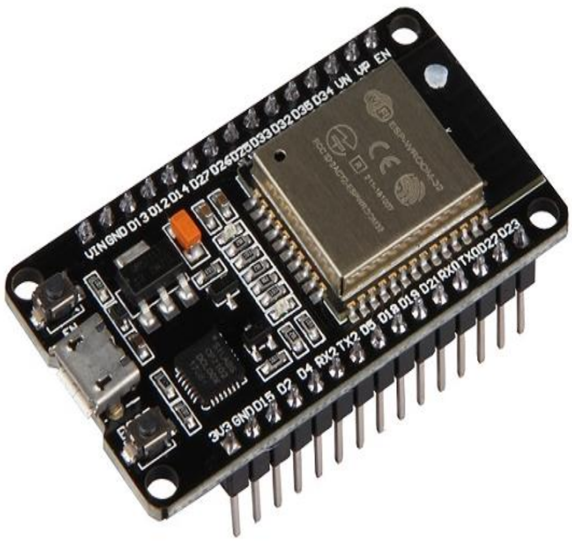


Figure 2. 2. ESP32 Development Board [4].

2.2.3.2. Resources and specifications of ESP32 processors

- CPU (Xtensa dual-core, LX 2-bit microprocessor), ultra-low-power coprocessor (ULO).
- Wifi: 2.4 GHz frequency, transmission speed up to 150Mbits/s, and a standard of 802.11b/g/n.
- Bluetooth: BLE (Bluetooth Low Energy), classic Bluetooth.
- Architecture: 2-bit CPU.
- RAM: 512KB.
- ROM: 448 448KB.
- Flash memory: 4MB.
- Pins: 36Pins.
- Peripherals: Capacitive touch, ADC, DAC, L2C, UART, Can2.0, Spi, L2S, RMII, PWM.
- Dimension: 48×26×11.5mm.

- Temperature: -40°C to 125°C.

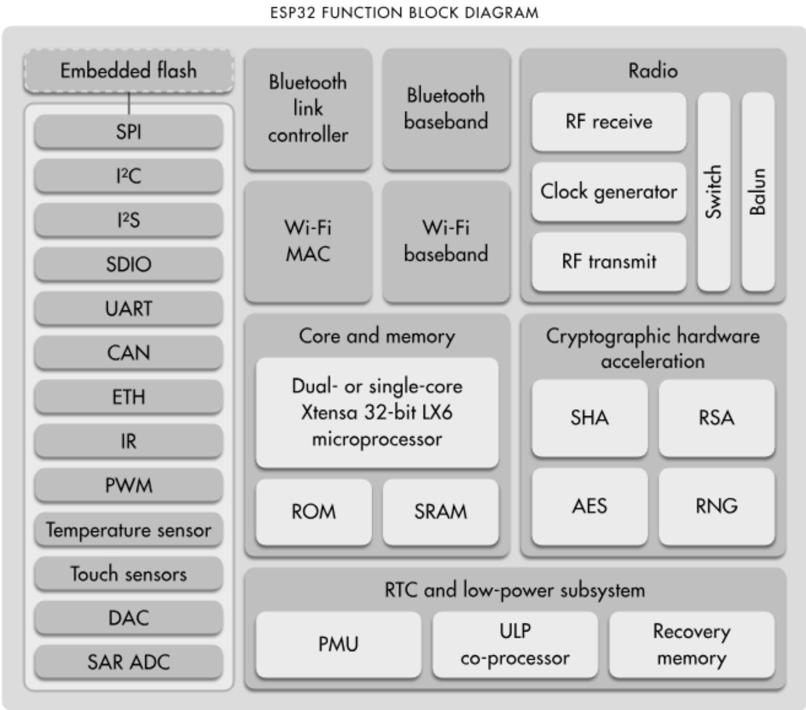


Figure 2. 3 . Functional Block Diagram of ESP32 [5].

2.2.3.3. Components of an ESP32 board

- Wifi antenna: 2.4Ghz antenna.
- Red LED indicates that the card is powered.
- Blue LED connected to pin 2 and it is programmable by the user.
- 3.3v regulator is used to have a voltage of 3.3 volts.
- ENABLE/Reset button to reset the ESP32 board.
- Micro-USB connector that allows you to program and power the ESP 32 board with 5volts.
- The BOOT button is used when downloading the new program.
- CP2102 USB-UART bridge that allows communication between the computer and the ESP32 board.

- Microcontroller (the ESP-WORM-32 chip).

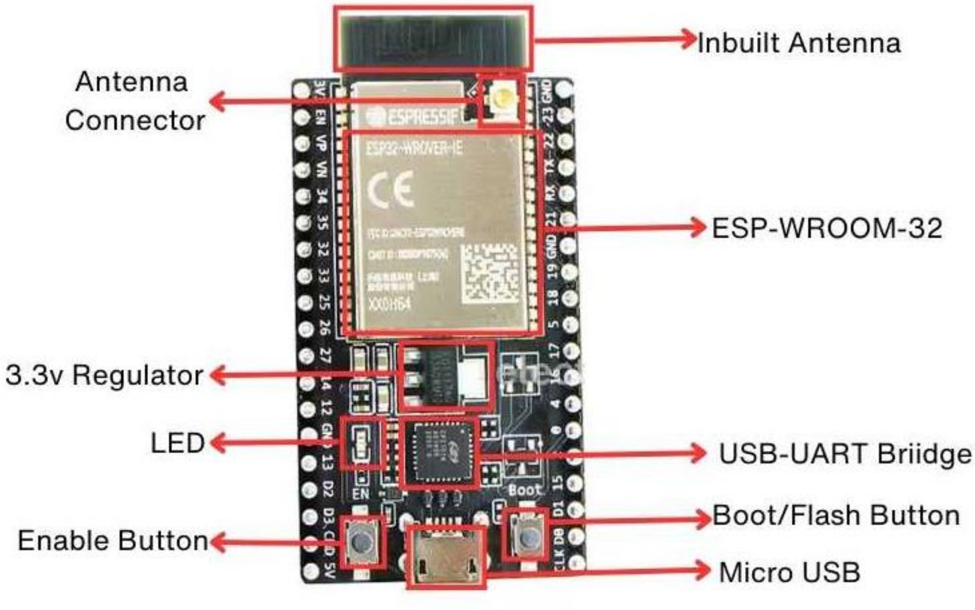


Figure 2. 4. ESP32 Components [6].

2.2.3.4. ESP32 circuit pinout

The ESP32 chip has 38 pins with multiple functions, not all pins are available on all ESP32 development boards, and some pins cannot be used.

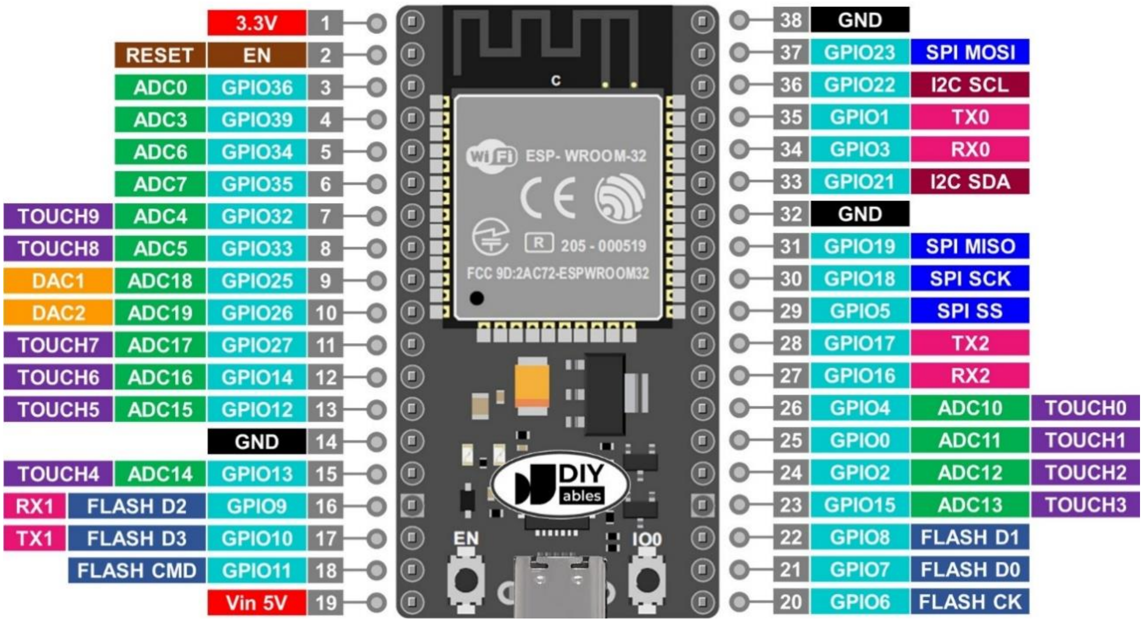


Figure 2. 5. ESP32 Pin Configuration [7].

2.2.3.5. Selection Criteria for the ESP32 in Low-Power Telemetry Systems

As part of the design of the industrial tele-supervision system based on LoRa P2P, the choice of the microcontroller was the ESP32 after a comparative analysis with the STM32 and the Arduino Uno. The ESP32 features a 32-bit dual-core architecture clocked up to 240 MHz, offering superior performance and better multitasking thanks to the native integration of FreeRTOS, which is essential for simultaneously performing sensor acquisition, LoRa transmission and WiFi communication tasks. Unlike the Arduino Uno, limited by its 8-bit architecture and low memory, and the STM32 which requires the addition of an external WiFi module, the ESP32 directly integrates WiFi connectivity as well as many communication interfaces (SPI, I2C, UART, ADC), thus reducing the hardware complexity, the overall cost and the energy consumption of the system. This strong integration, combined with sufficient memory capacity and computing power suitable for industrial embedded applications, fully justifies its choice as the central unit of the proposed system.

2.2.4. LoRa Communication Module

LoRa technology refers to the proprietary spread spectrum modulation technology based on the Chirp Spread Spectrum (CSS) technology and is specifically used for the Low power wide area network (LPWAN) communications. LoRa enables long-range wireless transmission of data (up to over 10 miles in rural areas) at relatively low bit rates (typically up to 50 kbit/s) while minimizing power consumption.

- LoRa technology operates in unlicensed ISM bands like 868 MHz in Europe, 915 MHz in North America and 433 MHz in Asia.It was initially developed by the company Semtech.
- LoRaWAN, managed by LoRa Alliance, is a radio access technology that provides bi-directional, secure communication from end device to cloud through one of the RF links. It includes features such as Adaptive Data Rate (ADR), MAC layer encryption and device operation by classes (Class A/B/C) , it is based on a star network topology where end nodes transmit data to multiple gateways, which forward data to the network server.

Due to long range and low power consumption characteristics, LoRa (Long Range) technology is being widely used in various applications of Internet of Things (IoT) projects.

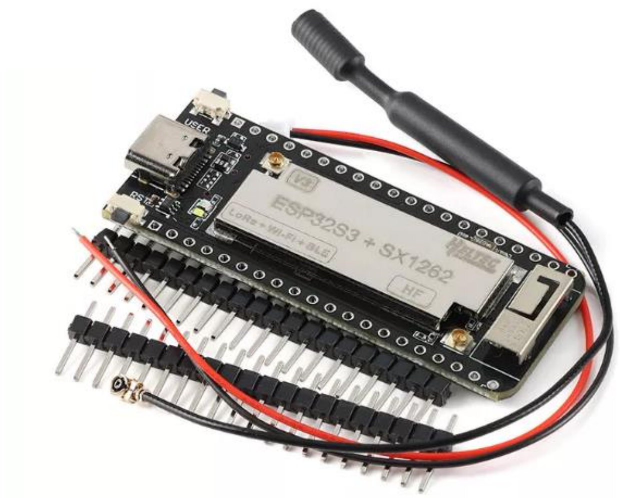


Figure 2. 6. ESP32-LoRa Development Board [8].

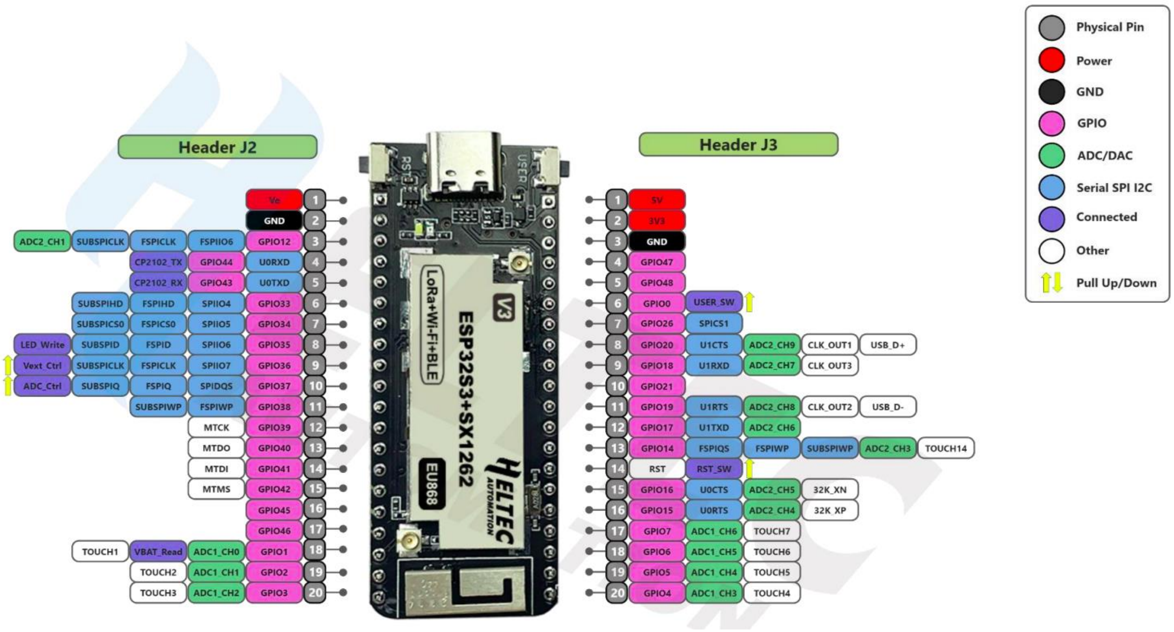


Figure 2. 7. Heltec ESP32-S3 LoRa V3 Pinout Map.

2.2.4.1. Principles and Technical Characteristics

LoRa technology uses a modulation technique called Chirp Spread Spectrum (CSS) which grants the signal robustness against noise and interference during transmission.

It is important to ensure robustness of communication in IoT applications, especially for the more challenging environments. Parameters that can be adjusted for the LoRa layer are carrier frequency, channel bandwidth, spreading factor (SF) and code rate.

The spreading factor, which can be thought of as affecting the data range and rate, must be selected according to the requirements of the system. In general, a larger spreading factor will increase the range of the system but decrease the rate.

Understanding the LoRa PHY and its Parameters for Better Transmission Range, SNR and Packet reception rate. Understanding the LoRa PHY protocol and the parameters that make up the LoRa PHY transmission is key to optimizing the performance of LoRa. Here are some important parameters and how they affect LoRa’s performance.

Although LoRa is robust against channel noise, it can still suffer from non-received packets in LoRaWAN networks due to packet collisions that may occur during concurrent transmissions, a phenomenon that is common in dense network scenarios.

To mitigate this problem, robust MAC protocols, such as CH-MAC, have been proposed to improve reliability and reduce latency.

2.2.4.2. Communication LoRa en mode Peer-to-Peer (P2P)

LoRa P2P, or Peer-to-Peer, is a term referring to direct communication between two LoRa devices outside the LoRaWAN standardized protocol that defines the LoRa network architecture. As we all know LoRaWAN uses a 'star-of-stars' topology, where gateways sit in between end-devices and the network server, also known as the gateway bridge. LoRa P2P mode removes the need for any infrastructure as it enables communication between only two nodes.

The choice of this communication technology was driven by a desire to keep system design simple and costs low. Setting up a full LoRaWAN network – with gateways, network server and application server – seems overkill for a local remote monitoring system for which the range will be limited.

2.2.4.3. LoRa Physical Layer and P2P Communication Parameters

LoRa P2P communication is based on the LoRa physical layer, implemented via the SX1278 transceiver. This technology uses Chirp Spread Spectrum (CSS) modulation, recognized for its large range (several kilometers in free field), its low energy consumption, and its robustness against noise and interference. LoRa operates in unlicensed ISM bands, notably 433 MHz, 868 MHz, and 915 MHz. In this project, the 433 MHz band was chosen for its regional compatibility and good penetration in semi-industrial environment. To optimize this communication for industrial use, in a P2P environment, the radio setting is entirely local and needs to be optimized manually. Essential parameters include Spreading Factor (SF), Bandwidth, Coding Rate, and Transmission power (TxPower). A high Spreading Factor increases the range but reduces the transmission rate. Therefore, there is a fundamental trade-off between Scope, Debit, Occupation time of the canal, and Energy consumption. In the context of industrial remote supervision, a compromise-oriented robustness and reliability is generally favored.

2.2.4.4. Différences entre LoRa P2P et LoRaWAN

The LoRaWAN architecture adopts a centralized topology where:

- Nodes→ Gateway→Network→ server→ Application

On the other hand, the LoRa P2P mode works according to direct communication:

- Sending node→Receiving node

This difference has several implications:

- Lack of centralized network management.
- No Adaptive Data Rate (ADR).
- Manual parameterization of physical parameters (SF, bandwidth, power).

The P2P mode offers:

- Simplicity of deployment.
- Cost reduction.
- Independence from a network infrastructure.
- Increased local resilience.

2.2.5. ESP32-LoRa Development Board

The combination of ESP32 and LoRa technology is an effective model to provide for the development of Internet of Things (IoT) solutions that need to operate using low power and have the ability to communicate over long range, while providing the functionality of a processor. The ESP32 is a series of low cost, low power system-on-a-chip (SoC) microcontrollers with built-in WiFi and dual mode Bluetooth. The use of ESP32 provides for the capability to perform both computing and connectivity functions, thereby enabling the ESP32 as an acceptable platform for a wide variety of IoT applications. Additionally, when the ESP32 is used with LoRa (Long Range), a technology that enables long range, low power and low bit rate wireless communications by utilizing CSS (Chirp Spread Spectrum) modulation, the resulting ESP32/LoRa combination will enable users to develop robust and efficient IoT devices that are capable of communicating from one location to another across large distances in varying environmental conditions.

2.2.5.1. Technical Synergies and Advantages

The integration of ESP32 and LoRa leverages several key advantages. Processing and Connectivity: The ESP32 provides significant processing power and memory for embedded applications, alongside built-in Wi-Fi and Bluetooth. This allows for local data processing, sensor interfacing, and flexible connectivity options. For instance, an ESP32-LoRa node could collect data from sensors, process it locally, transmit summary data via LoRa over long distances, and also offer local configuration or data access via Wi-Fi or Bluetooth. Long-Range Communication: LoRa's primary advantage is its ability to transmit data over several kilometers, even up to 15-30 miles in rural areas, while consuming minimal power. This makes it suitable for applications in remote locations where cellular or Wi-Fi infrastructure is unavailable or impractical. Low Power Consumption: Both ESP32 (when optimized) and LoRa are designed for low-power operation, enabling battery-powered devices to function for extended periods, which is crucial for many IoT deployments. LoRa achieves low power consumption through its modulation scheme and by supporting various communication modes (e.g., Class A, B, C in LoRaWAN), where devices can spend most of their time in a deep sleep state. Robustness and Interference Resistance: LoRa's CSS modulation provides excellent resistance to noise and interference, making it reliable in challenging RF environments. The adaptability of physical layer parameters like spreading factor (SF), channel bandwidth, and code rate allows for optimization based on environmental conditions and application requirements, balancing range and data rate. Flexibility in Network Topologies: An ESP32-LoRa combination can be used in various network configurations including LoRaWAN End-Device where the ESP32 handles sensor data acquisition and processing, then uses the LoRa module to communicate with LoRaWAN gateways, connecting to a larger LoRaWAN infrastructure; LoRa Peer-to-Peer (P2P) for localized or direct device-to-device communication, where the ESP32 can manage P2P links between LoRa modules, eliminating the need for gateways or network servers, particularly useful for ad-hoc networks or situations where centralized infrastructure is not feasible or desired; and LoRa Mesh Networks where ESP32-LoRa nodes can form mesh networks building on P2P, with devices relaying messages for each other, extending coverage and enhancing network resilience beyond the range of a single transmission.

2.2.5.2. Communication Protocol

Table 2. 1. SPI and Interrupt Signaling Mechanisms for ESP32-LoRa Communication

Protocol	Description
SPI	High-speed serial communication between ESP32 and SX1278
Interrupt-driven	DIO0 pin notifies ESP32 when packet is sent or received

2.2.5.3. ESP32 – LoRa Hardware Interface:

Table 2. 2. Hardware Pin Mapping and Signal Definitions Between ESP32 and SX1278

ESP32 Pin	SX1278 Pin	Signal Type	Function Description
GPIO18	SCK	SPI Clock	Serial clock generated by ESP32
GPIO23	MOSI	SPI Data	Data sent from ESP32 to SX1278
GPIO19	MISO	SPI Data	Data sent from SX1278 to ESP32
GPIO5	NSS (CS)	Chip Select	Enables communication with LoRa module
GPIO14	DIO0	Interrupt	Signals packet transmission/reception complete
GPIO26	RESET	Digital Output	Resets the SX1278 module
3.3V	VCC	Power	3.3V regulatedsupply
GND	GND	Ground	Common groundreference

2.2.5.4. Power Supply and Battery Charging Module

The reliability and life expectancy of electronic circuits depend heavily on a stable power supply. For this remote monitoring node, the power supply must consistently deliver a regulated 3.3 VDC to both the ESP32 and SX1278 modules, preventing voltage drops during high-load events.

While the typical combined current draw of these components hovers between 120 mA and 150 mA, the system can experience transient peaks of up to 240 mA when the ESP32 transmits coded modulated packets via the LoRa transceiver. Because the field node is deployed in an isolated location without access to the electrical grid, an off-grid solar energy harvesting system is utilized. This setup integrates a photovoltaic panel and a rechargeable battery to sustain continuous, uninterrupted operation through nighttime and overcast conditions, requiring a voltage regulator capable of suppressing fluctuations during maximum transmission current spikes.

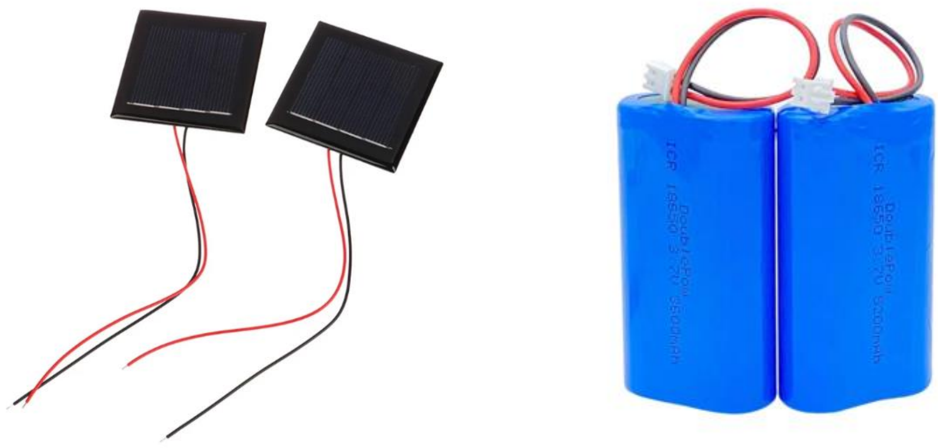


Figure 2. 8. Solar Panel and Lithium Battery [9].

2.2.5.5. Antenna

The selection of the antenna plays a vital part in the seamless interaction between the ESP32 and LoRa. We can say that the antenna is a critical aspect of reliability, transmission range, and energy efficiency. In addition, talking about LoRa, a low-power wide-area network (LPWAN) specification which typically operates in unlicensed band. It incorporates frequency bands of 433 MHz, 868 MHz and 915 MHz for ISM. Apart from that, the antenna in a LoRa system requires careful consideration as far as operating frequency. In addition, if the impedance is not tuned for the designated frequency range, it will affect the communication capabilities and the signal attenuates. This also leads to energy and power loss. To develop an efficient IoT system, it is essential to pay careful attention to the antenna selection.

The antenna used in conjunction with the LoRa transceiver requires careful impedance matching at 50 ohms to maximize power transfer efficiency. In general, the impedance of the LoRa transceiver and the antenna is designed to be 50 Ω. In addition, any changes in the impedance may reflect power to the LoRa transceiver as well. Moreover, the reflected power will result in a decrease in the effective radiated output power, which is again very important in long-range communication.

Table 2. 3. Antenna parameters.

Parameter	Description	Impact on System
Operating Frequency	433 / 868 / 915 MHz (ISM band)	Determines antenna tuning and efficiency
Impedance	50 Ω standard RF	Ensures maximum power transfer
Gain	0–5 dBitypical	Affects communication range
Radiation Pattern	Omnidirectional or Directional	Definescoverage area
Antenna Type	Whip, Helical, PCB, Ceramic, Dipole	Influences size and performance
Installation Position	Vertical recommended	Improves signal propagation
EnvironmentResistance	Industrial-grade recommended	Ensuresdurability and reliability
RecommendedChoice (P2P)	External 433 MHz omnidirectionalantenna	Stable long-range communication

2.2.5.6. Drivers

The software drivers that manage the communication protocol between the microcontroller (MCU) and the LoRa transceiver (SX1278) play a key role in the ESP32 + LoRa system. The SPI driver of the ESP32 controls the MCU communication with the LoRa module through the MOSI, MISO, SCK and NSS pins. The LoRa driver is implemented via a specific library and is responsible for forming packets, changing the modulating parameters, adjusting the frequency, and handling interrupts via the DIO0 pin. Each of the implemented driver functions is crucial for precise timing of the send and receive packet transmission, controlling of the number of retries, and adjusting the physical layer parameters such as spreading factor, bandwidth and transmission power. By integrating the drivers with FreeRTOS (or other real-time operating systems), the tasks can operate in non-blocking mode and thus the CPU is utilized in the most efficient way possible. The drivers in the LoRa P2P communication between nodes in industrial IoT applications play a crucial role.

Table 2. 4. Software drivers used in the system.

Driver	Function	ESP32 Role	Impact on System
SPI Driver	Low-level communication via SPI	Data transfer to/from SX1278	Reliable hardware connection established
LoRa Driver / Library	Packet handling, modulation, frequency tuning	Transmitter setup including output power	Settings for ensuring successful bi-directional radio link (Link Check)
Interrupt Handling	Provides DIO0 interrupt for TX/RX completion	Supports event-driven applications	Minimizes CPU blocking and improves timing
FreeRTOSIntegration (optional)	Task scheduling, non-blocking operation	LoRa tasks can be managed together with other tasks	- Better real time response - More efficient code - Simplified development flow
Parameter Configuration	SF, BW, TxPower, Coding Rate	Physical layer parameters configuration	Optimize communication range, data transmission speed and save energy

2.2.6. Power Supply and Battery Charging Module

A charge module is a circuit that safely manages battery charging by regulating voltage and current, typically following a CC/CV (constant current then constant voltage) process. It includes protections against overcharge, short circuits, and overheating. Common examples include the TP4056 for small Li-ion cells and BMS modules for larger battery packs, found in everything from DIY electronics to electric vehicles.

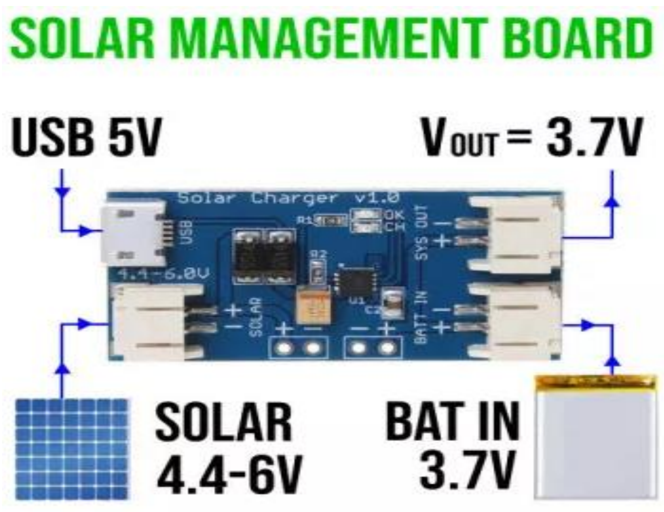


Figure 2. 9. Charger module [10].

2.2.7. Sensors Used in the Monitoring System

2.2.7.1. Definition

A sensor is a device that converts a physical phenomenon (such as temperature, pressure, or humidity) into information that can be used by an electronic system. It does not directly measure this quantity, but detects a change in the environment and transforms it into a signal (often electrical). This signal is then processed by electronic devices, such as a microcontroller, to perform measurements or make decisions in a process.

2.2.7.2. DHT22 humidity and temperature sensor:

The DHT22 emits a calibrated digital signal. It uses a proprietary digital signal collection technique and humidity detection technology, which ensures its reliability and stability. Its detection elements are connected to an 8-bit single-chip computer, which guarantees its reliability and stability. Each sensor of this model is temperature compensated and calibrated in a precise calibration chamber. The calibration coefficient is saved as a program in the OTP memory, when the sensor is detected, it quotes the coefficient of the memory. Memory coefficient.

The small size, low power consumption and long transmission distance (20m) allow the DHT22 to be adapted to all kinds of difficult applications.

Single row packaging with four pins, making the connection very convenient

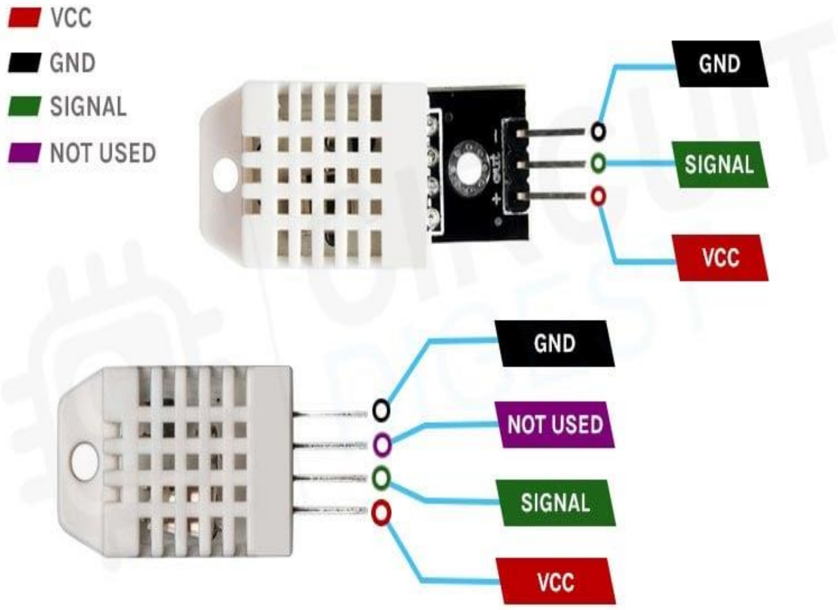


Figure 2. 10. DHT22 sensor [11].

2.2.7.3. Technical Specifications of DHT22

Table 2. 5. Technical specifications of the Dht22 sensor

Model	DHT22
Power Supply	3.3–6V DC
Output Signal	Digital signal via single-wire bus
Sensing Element	Polymer capacitor
Operating Range	Humidity: 0–100% RH; Temperature: -40 to 80°C
Accuracy	Humidity: ±2% RH (Max ±5% RH) ; Temperature: ±0.5°C
Resolution	Humidity: 0.1% RH; Temperature: 0.1°C
Repeatability	Humidity: ±1% RH; Temperature: ±0.2°C
Humidity Hysteresis	±0.3% RH
Long-Term Stability	±0.5% RH/year
Sampling Period	Average: 2 seconds

2.2.7.4. Soil Moisture Sensor

This soil moisture sensor module is used to detect soil moisture. It measures the volumetric content of water in the soil and gives us the moisture level as an output. The module has digital and analog outputs and a potentiometer to adjust the threshold level.

This humidity detection module is composed of a humidity sensor, resistors, a capacitor, a potentiometer, an LM393 IC comparator, a power and status LED in an integrated circuit.

Figure 2. 11. Soil moisture sensor [12].

2.2.7.5. Technical Specifications of Soil moisture sensor

Table 2. 6. Technical specifications of the soil moisture

Specification	Description
Operating Voltage	3.3V to 5V DC
Operating Current	15 mA
Digital Output	0V to 5V, adjustable trigger level from a preset threshold
Analog Output	0V to 5V based on infrared radiation from flame detected by the sensor
IndicatorLEDs	LEDs indicating power and output status
PCB Size	3.2 cm × 1.4 cm
Design Based On	LM393 comparator

Main Features	Easy to use with microcontrollers or standard digital/analog ICs; small size, low cost, and widely available

2.3. Software

2.3.1. Arduino IDE

The embedded layer of the industrial monitoring and control system utilizes Arduino as the development framework for programming the ESP32 microcontroller, specifically the Heltec Wireless Stick Lite V3, an integrated ESP32-LoRa development board designed for IoT and LPWAN applications. Arduino is an open-source electronics platform based on easy-to-use hardware and software, widely used in IoT and prototyping due to its simplicity, extensive library support, and active community. The Heltec Wireless Stick Lite V3 integrates the ESP32-S3 microcontroller with SX1262 LoRa transceiver on a single compact board, featuring a dual-core 32-bit processor (up to 240 MHz, 512 KB SRAM), built-in Wi-Fi and Bluetooth 5, LoRa Sub-GHz frequency support (433 MHz, 868 MHz, 915 MHz), multiple I/O pins (GPIO, SPI, I2C, UART, ADC, DAC), 3.3V logic operation, -40°C to +85°C temperature range, and an integrated OLED display for debugging. It was chosen for its integrated ESP32-LoRa architecture (eliminating external modules), low power consumption, compact form factor, and high processing capability.

The Arduino-based ESP32 firmware performs core functions: sensor data acquisition (reading temperature, humidity, pressure, vibration, voltage, or current every 5–10 seconds using the 12-bit ADC), data processing (packaging readings into JSON format with timestamps and device IDs), wireless transmission (sending data via SX1262 LoRa transceiver over 2–15 km in low-power mode), and power management (deep sleep mode between readings for battery-powered deployment). Development uses Arduino IDE or PlatformIO with Heltec ESP32 Board Manager, Arduino Framework, LoRa Library (Heltec_ESP32.h/SX1262.h), and sensor libraries. Code follows the standard setup() and loop() pattern: initialization in setup() (GPIO, sensors, LoRa, OLED), then main loop() (read sensors, package JSON, transmit via LoRa, sleep, repeat).

The Heltec Wireless Stick Lite V3 offers advantages: integrated ESP32-LoRa design (reducing failure points), simplicity, extensive library support, rapid prototyping with onboard OLED debugging, cost-effectiveness, community support, cross-platform compatibility, and scalability. This solution is suitable for industrial monitoring due to its reliability, real-time operation, and low power consumption enabling battery/solar power operation.

2.3.2. Node-RED Platform

The middleware layer of the industrial monitoring and control system utilizes **Node-RED** as the IoT orchestration platform and dashboard framework, an open-source, graphical programming environment developed by IBM and hosted by the OpenJS Foundation, widely used in IoT, Industrial IoT (IIoT), and industrial automation due to its visual programming approach, rapid prototyping, extensive node ecosystem, and seamless integration with databases and protocols. Node-RED is built on **Node.js** (JavaScript runtime), requiring Node.js LTS v14.x or higher to be installed before Node-RED via npm install -g node-red.

Node-RED serves as the **gateway/orchestration layer** between ESP32-LoRa edge devices and the database layer, performing core functions: **data reception** (receiving sensor data from Heltec Wireless Stick Lite V3 via HTTP POST or MQTT from the LoRa gateway), **data processing** (parsing JSON payloads, validating integrity, applying filtering/transformation, enriching with metadata), **data routing** (directing time-series data to InfluxDB and relational data to Supabase), **dashboard visualization** (real-time web-based monitoring via charts, gauges, tables showing live readings and trends), **alert management** (threshold-based monitoring triggering alerts when values exceed limits), and **database integration** (InfluxDB output nodes for sensor measurements, Supabase/HTTP nodes for device registry and user management).

Development uses Node-RED on a server or single-board computer (Raspberry Pi, industrial PC) with Node.js, including Node-RED Dashboard package, InfluxDB nodes, Supabase/HTTP request nodes, HTTP input nodes, MQTT nodes, function nodes (JavaScript processing), and debug nodes. The flow architecture follows: input nodes (receive via HTTP/MQTT) → function nodes (parse JSON, transform) → switch/rule nodes (filter) → database output nodes (InfluxDB/Supabase) → dashboard nodes (visualize).

Node-RED advantages: visual programming (rapid development), open-source/cost-free (no licensing fees vs. SCADA), extensive node ecosystem (thousands of nodes via npm), easy integration (InfluxDB, Supabase, MQTT, HTTP, Modbus), real-time processing (10–100 ms latency), automatic reconnection (3–20 s recovery), cross-platform (Windows, macOS, Linux, Raspberry Pi), scalability (multiple devices), modular architecture, and web-based dashboard (remote access). It is suitable for industrial monitoring due to real-time operation, reliability (automatic reconnection), low cost, flexibility, ease of deployment, community support, and Industry 4.0 alignment.

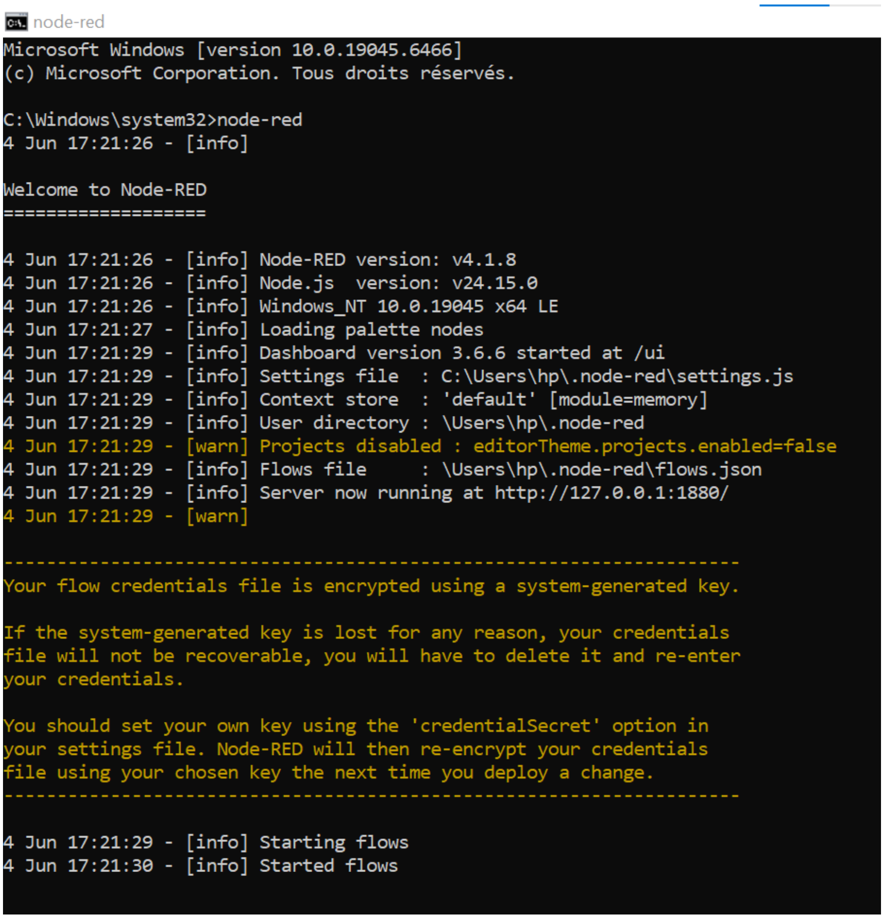


Figure 2. 12. Starting node-red.

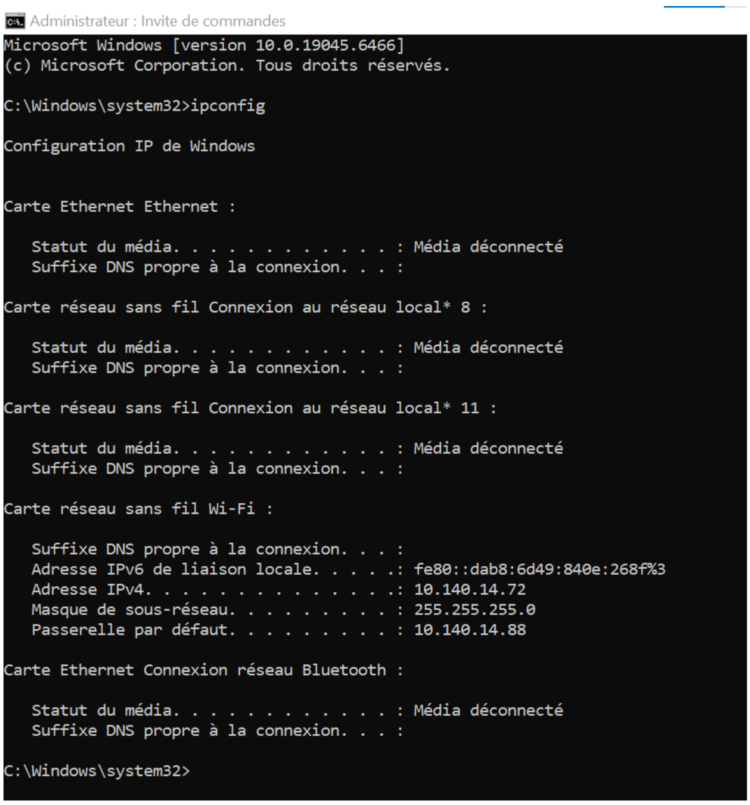


Figure 2. 13. Copying the ip address.

2.3.3. Database Management System

2.3.3.1. Supabase

Supabase is a platform that offers a collection of tools for building applications. It provides backend services, such as authentication, database, file storage, auto-generated APIs and edge functions. The Supabase client library is supported in JavaScript, Python, and Dart. The platform was founded in 2020 and is marketed as an open-source alternative to Firebase.

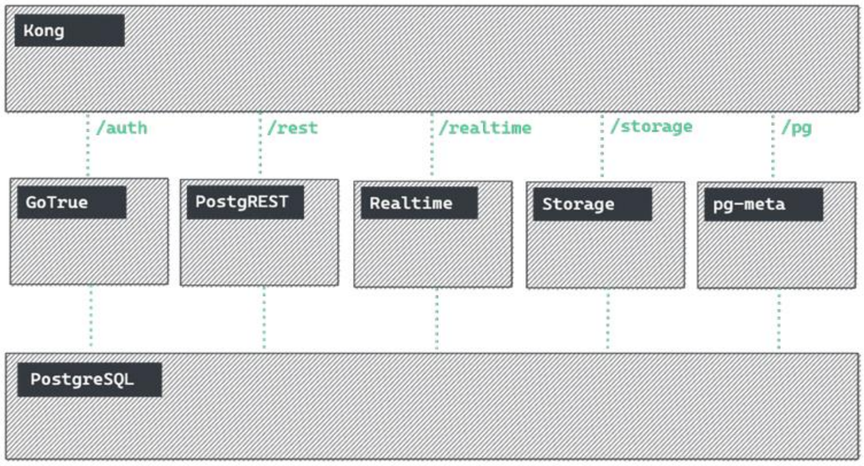


Figure 2. 14. Supabase architecture

As illustrated in figure 2.14, Supabase is created using a combination of different open-source tools that are scalable. If the features already provided by Supabase are inadequate, supabase offers the option of self-hosting.

Every project created in Supabase includes a full Postgres database. Since the database is built on Postgres, it is relational. This means that data can be organized into tables that are linked based on defined relationships. The items inside the tables can be organized as rows and columns. Supabase offers a few ways of creating tables. The easiest way to create a table is directly from the dashboard using the intuitive table view provided by Supabase but it is also possible to do so by running SQL queries. When creating a table each column must have a predefined data type. Supabase offers default types but it is also possible to create custom types if the ones already provided are insufficient. Database tables are secured using Row Level Security (RLS) policies. Since data can be accessed directly from the client using the anon key provided by Supabase, it is important to restrict access to private tables using RLS policies.

This can be achieved by running code inside the SQL editor or via the Supabase dashboard. The policies can be as complex as required. Supabase also supports SQL functions that exist inside the database. A database function can be created using SQL or from the dashboard. After a function is created it can be executed inside the database or directly from the client library. A database function is best suited for handling data intensive operations. The platform also includes other additional features, such as a realtime engine that is extended by postgres for listening to database changes. It also offers extensions that can be enabled to further expand the functionality of the database. Moreover, Supabase self manages data backups, and allows data imports from comma-separated values (CSV) or excel spreadsheet.it also auto-generates self-documenting, secure, fast, and scalable APIs directly from the database schema. [35] The three APIs generated by Supabase include RESTful, Realtime, and GraphQL (Graph Query Language). The RESTful API handles CRUD operations and can be useful for fetching data from multiple tables by using deeply nested joins. The Realtime API listens to database changes and the GraphQL API functions as a GraphQL endpoint which is currently in beta. Supabase also has a system for authorization which offers several ways of authenticating users.

For instance, users can be authorized with email and password, phone number, magic links (one-click sign-in), or third-party social providers. For the third-party authentication, Supabase offers a great deal of providers to choose from. Third party authentication can be enabled from the settings tab of the authentication page by providing a client ID and a secret for the selected provider. When a user signs up, Supabase automatically creates a user's table with unique ids which makes user management, and authorization easier. For authorization, RLS and Policies can be used to set complex SQL rules.

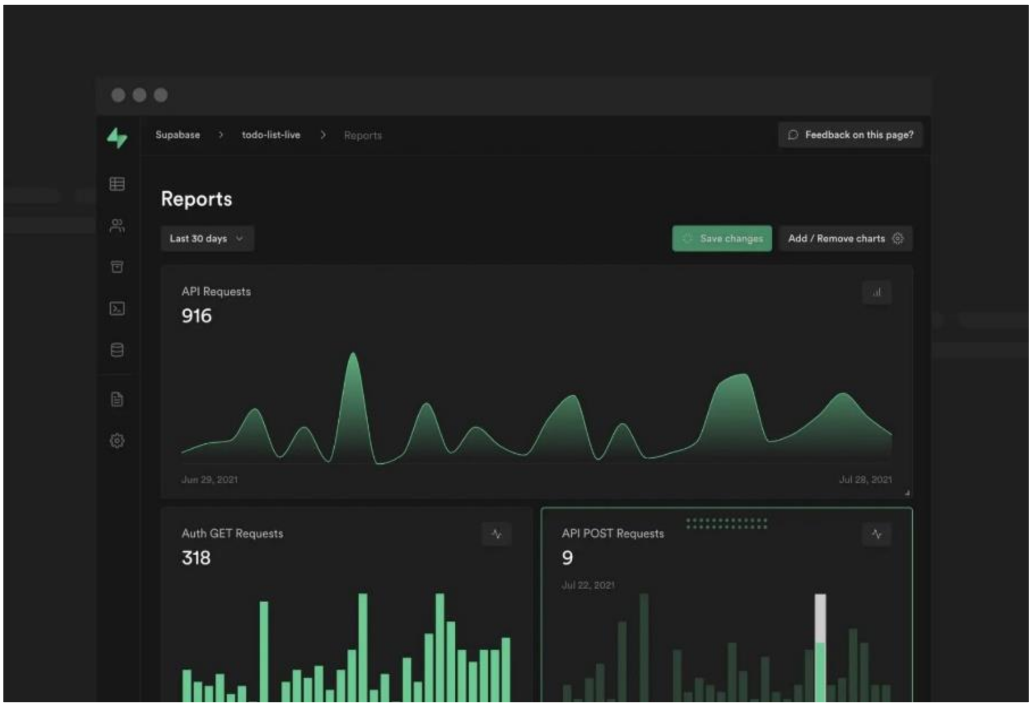


Figure 2. 15. Example of a Supabase Database

2.3.3.2. influxDB

The data layer of the industrial monitoring and control system utilizes InfluxDB as the time-series database, an open-source database written in Go and Rust, purpose-built for handling high-write workloads of timestamped sensor data and metrics, designed specifically for IoT and Industrial IoT (IIoT) monitoring; InfluxDB serves as the primary storage for time-series sensor measurements performing core functions including high-performance data ingestion (accepting thousands of writes per second from ESP32-LoRa sensor nodes via HTTP API, MQTT, or Modbus TCP/RTU with minimal latency) time-stamped data storage (nanosecond-resolution timestamps), data organization (measurements, tags for metadata like device_id and location, fields for sensor values like temperature and humidity), efficient compression (reducing storage by up to 90%), retention policies (automatic data expiration), Modbus integration (collecting industrial sensor data through Modbus TCP/IP or Modbus RTU protocols via Telegraf or Node-RED Modbus nodes for reading registers from industrial devices), and real-time querying (sub-second dashboard response times); the InfluxDB data model consists of measurements (like tables), tags (indexed metadata), fields (sensor values), and timestamps (nanosecond-precision); InfluxDB offers advantages including optimized for time-series data, high write throughput, built-in time functions, efficient compression, retention policies, real-time querying, industrial protocol support including Modbus, MQTT, and HTTP, open-source and cost-effective, and scalability; InfluxDB is particularly suitable for industrial monitoring due to its time-optimized architecture, high performance, reliability for 24/7 data ingestion, low storage overhead, easy integration with Node-RED through InfluxDB output nodes and Modbus nodes, and complementary to Supabase where InfluxDB handles time-series sensor measurements from Modbus-enabled industrial sensors while Supabase manages device registry and user authentication.

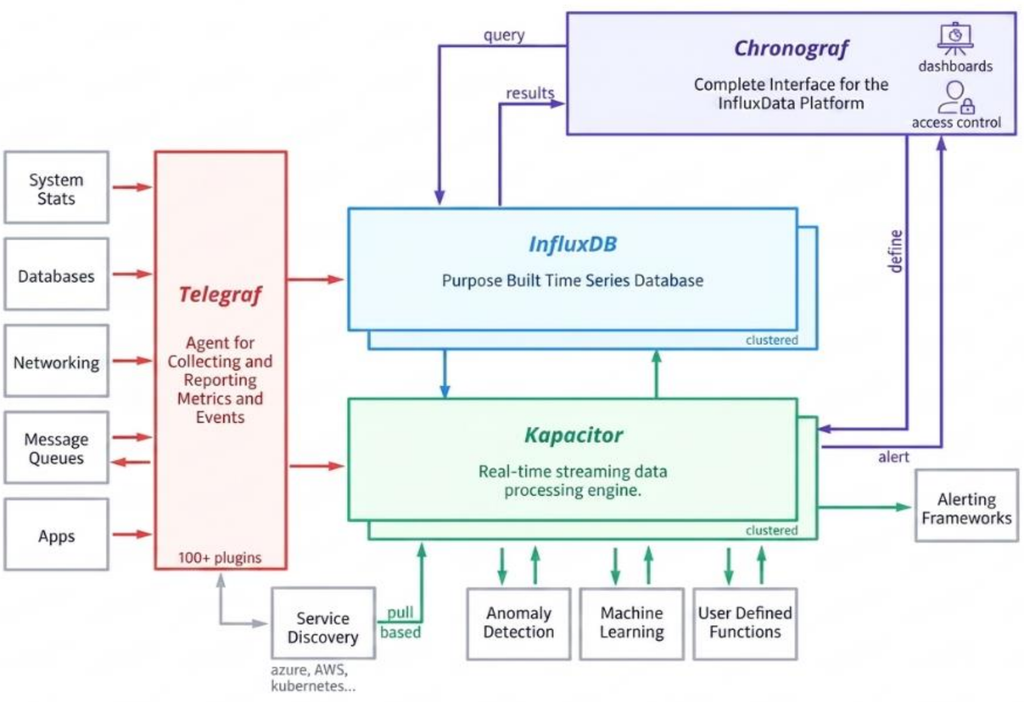


Figure 2. 16. InfluxDB Platform Architecture [17].

2.4. Firmware Architecture

The firmware is developed in the Arduino IDE using the following libraries: Heltec_ESP32.h (for Heltec Wireless Stick Lite V3 board support and OLED display), LoRa.h or SX1262.h (for LoRa communication with SX1262 transceiver at 433 MHz), DHT sensor library (for DHT22 temperature and humidity sensor reading), and ArduinoJson (for packaging sensor data into JSON format). The middleware layer utilizes Node-RED installed on a server with Node.js (LTS v14.x or higher), installed via npm install -g node-red, employing the following nodes and packages: node-red-dashboard (for web-based UI components: charts, gauges, tables showing real-time sensor readings from DHT22), node-red-contrib-influxdb (for InfluxDB time-series database operations - writing sensor temperature/humidity data with timestamps), node-red-contrib-supabase or HTTP request nodes (for Supabase PostgreSQL database interactions - device registry, user authentication, alert configurations), HTTP input nodes (receiving sensor data from Heltec Wireless Stick Lite V3 sender via LoRa gateway P2P communication), Function nodes (custom JavaScript for parsing JSON payloads, data transformation, filtering), Switch nodes (filtering data based on threshold conditions for alert management), and Debug nodes (testing and troubleshooting flows). InfluxDB stores time-series sensor measurements (temperature, humidity) with tags (device_id, location) and fields (temperature value, humidity value), while Supabase stores relational data (devices table with ESP32 sender/receiver info, users table for authentication, alerts table for threshold configurations). The flow architecture follows: **HTTP input node (receive LoRa gateway data) → Function node (parse JSON from ESP32) → Switch node (check thresholds) → InfluxDB output node (store sensor data) + Supabase node (update device status) → Dashboard nodes (visualize real-time temperature/humidity readings)**.

2.4.1. Role of the Firmware

In our project, the firmware ensures Data Acquisition by reading soil moisture, temperature, and ambient humidity from sensors; Local Processing by comparing readings with thresholds, making decisions to activate the pump, and managing errors; LoRa Communication by encapsulating data, transmitting it to the LoRa receiver, and receiving commands (e.g., manual ON/OFF); and Energy Management by optimizing consumption for battery/solar-powered operation in the isolated sender location.

2.4.2. Relationship between Firmware and Node-RED

In our architecture, the data flow follows this sequence:

Sensors (Modbus) → ESP32 (Firmware) → LoRa → Receiver → Node-RED → Dashboard

The firmware developed on the ESP32 (Heltec Wireless Stick Lite V3) reads data from Modbus industrial sensors (such as DHT22 for temperature and humidity) via Modbus protocol and sends it via LoRa P2P communication at 433 MHz to the receiver node. The Node-RED middleware receives the data from the LoRa gateway via HTTP, processes it, displays real-time sensor readings on the dashboard (charts, gauges, tables), and enables remote control capabilities for the industrial monitoring system. The system also integrates InfluxDB for storing time-series sensor measurements and Supabase for managing device metadata, user authentication, and alert configurations.

2.5. Interface utilisateur (HMI du recepteur en utilisant Node-RED)

2.5.1. Development of the HMI Using Node-RED

Installation and configuration of Node-RED To create a dashboard in Node-RED, first install the node-red-dashboard package via the Node-RED palette manager (click the palette icon, search for "node-red-dashboard", and click install). Once installed, drag dashboard nodes from the sidebar onto your flow canvas, including ui_gauge nodes to display real-time sensor values (temperature, humidity, soil moisture) with customizable scales and colors, ui_chart nodes to show historical data trends over time, ui_text nodes to display current readings or system status, ui_switch or ui_button nodes for remote control (e.g., pump ON/OFF), and ui_slider nodes for setting threshold values. Double-click each node to configure its properties: assign a name, select or create a tab (to organize dashboard groups), set the group within the tab, configure the topic to match your incoming data format, and customize display options (gauge type, chart history duration, text format). Connect these dashboard nodes to your data processing flow by wiring them after the function nodes that parse the ESP32 sensor data, ensuring the dashboard receives the correct JSON payload with sensor values. Finally, deploy the flow and access the dashboard by clicking the dashboard icon in the top-right corner of Node-RED or navigating to *http://your-server-ip:1880/ui* in a web browser, where you can monitor real-time sensor readings, view historical trends, and control the industrial monitoring system remotely from any device with a browser.

CHAPTER III

PRACTICAL IMPLEMENTATION AND RESULTS.

3.1. Introduction

This chapter outlines the practical implementation of our system, covering the hardware and software architecture, the integration of sensors (DHT22, soil moisture) and communication modules (LoRa), as well as validation tests on transmission reliability, measurement accuracy, and the performance of the InfluxDB and Supabase databases.

3.2. Operational Flow: Sensing, Transmission & Control

The activity diagram illustrates the operational logic of the SmartFarm system, structured around three core components. The sensor node (ESP32) periodically awakens to acquire measurements from the DHT22 and soil moisture sensors, autonomously controlling irrigation based on predefined humidity thresholds prior to transmitting data via LoRa and entering deep sleep mode. The gateway (ESP32 Receiver) captures incoming LoRa packets, parses the contained data, and relays them to Node-RED, while also handling the forwarding of user-generated commands. The Node-RED interface and Dashboard provide real-time data visualization, ensure persistent storage in InfluxDB and Supabase databases, and enable remote command issuance through the Flutter application, thereby establishing a comprehensive control loop and ensuring effective system supervision. See Figure 3.1:

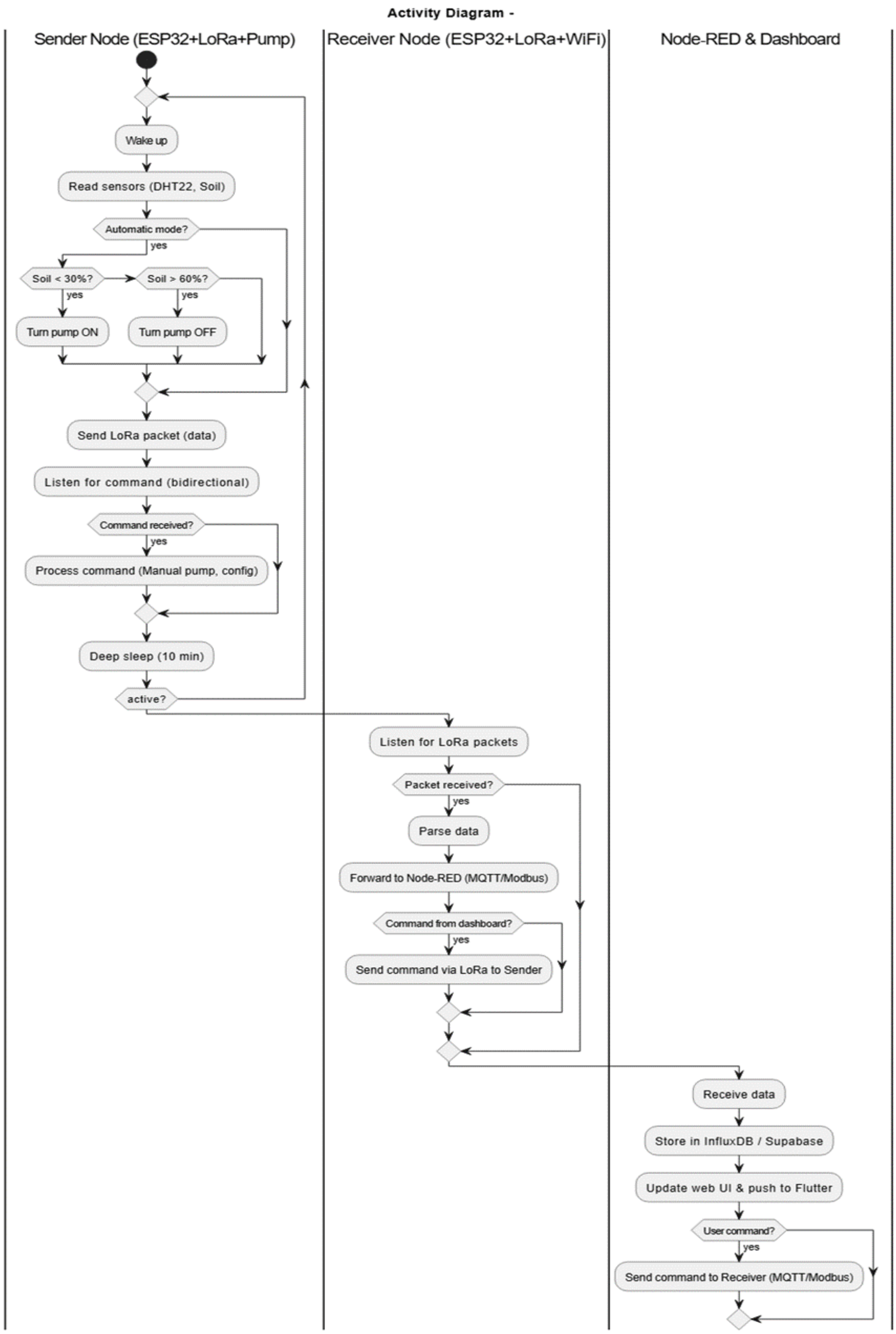


Figure 3.1. Activity diagram.

3.3. ESP32 Sensor Node Software Architecture

The class diagram depicts the software architecture of the ESP32 sensor node, organized around the core ESP32Sender class. This central component orchestrates LoRa communication, sensor data acquisition from the DHT22 and SoilSensor modules, photovoltaic power management through the PowerManager class, and irrigation pump actuation via the Pump class. Such a modular design promotes a clear separation of concerns and enhances overall system maintainability. See figure 3.2:

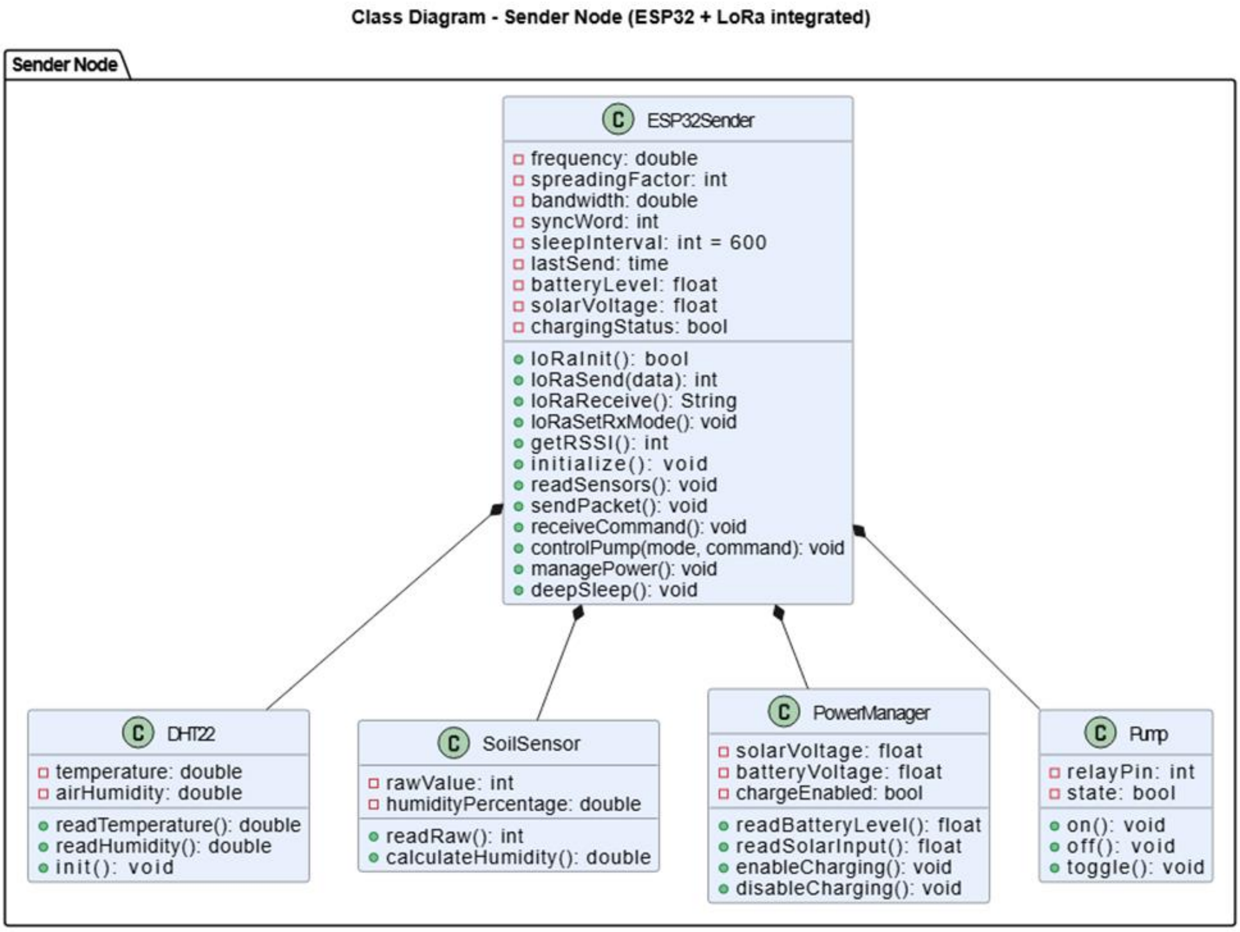


Figure 3. 2. class diagram -sender

3.3.1. ESP32 Sensor Node Software Architecture

The ESP32 receiver state diagram illustrates the different operational phases of the gateway: hardware initialization (INIT), WiFi connection, MQTT connection, permanent listening for LoRa data with forwarding to Node-RED, and processing of downstream commands for transmission to the sensor node via LoRa, thus ensuring robust bidirectional communication.

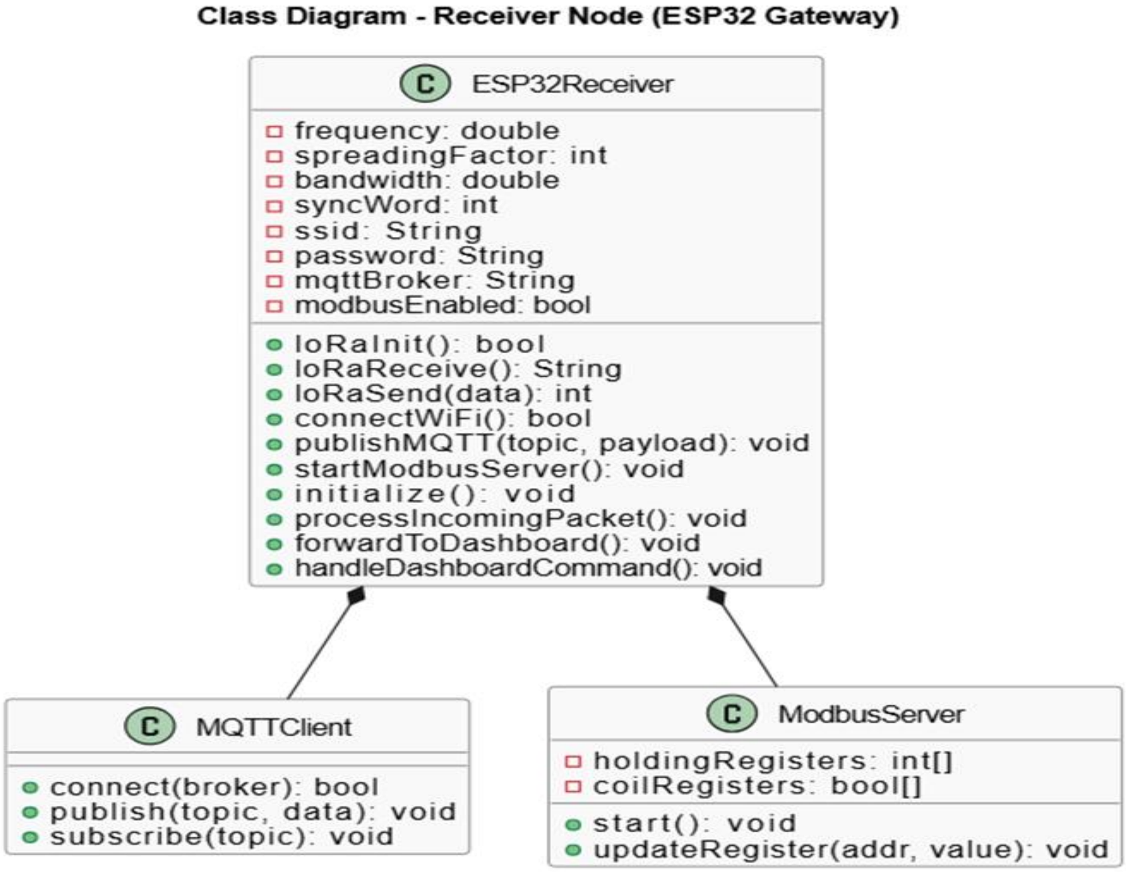


Figure 3. 3. Class diagram-receiver.

3.4. ESP32 Sensor Node Cyclic Behavior & Energy Management

The Sender node state diagram illustrates the cyclic operational behavior of the ESP32 sensor node. Following initialization (INIT), the system performs sensor data acquisition (READING) and executes automatic pump control logic (LOCAL_CONTROL) prior to data transmission (SENDING). It subsequently listens for incoming gateway commands (LISTEN_FOR_CMD) during a 500 ms timeout window, processes any received commands (PROCESS_CMD), and finally enters a 10-minute deep sleep mode (SLEEP) to optimize energy consumption. An error recovery mechanism (ERROR) with a maximum of five retry attempts ensures system resilience and reliability. See figure 3.4

Figure 3. 4. State machine-sender.

3.4.1. ESP32 Gateway Sequential Communication & Command Processing

The Receiver node state diagram illustrates the sequential operation of the ESP32 gateway. After initialization (INIT), the system establishes WiFi connection, then ModBus connection. It then enters permanent listening for LoRa packets (LISTEN LORA) to forward data to Node-REDThe Receiver node state diagram illustrates the sequential operation of the ESP32 gateway. After initialization (INIT), the system establishes WiFi connection, then MQTT connection. It then enters permanent listening for LoRa packets (LISTEN LORA) to forward data to Node-RED. Simultaneously, it processes downstream commands from the dashboard through the HANDLED and SEND CMD states, ensuring bidirectional communication between the gateway and the sensor node.See figure 3.5:

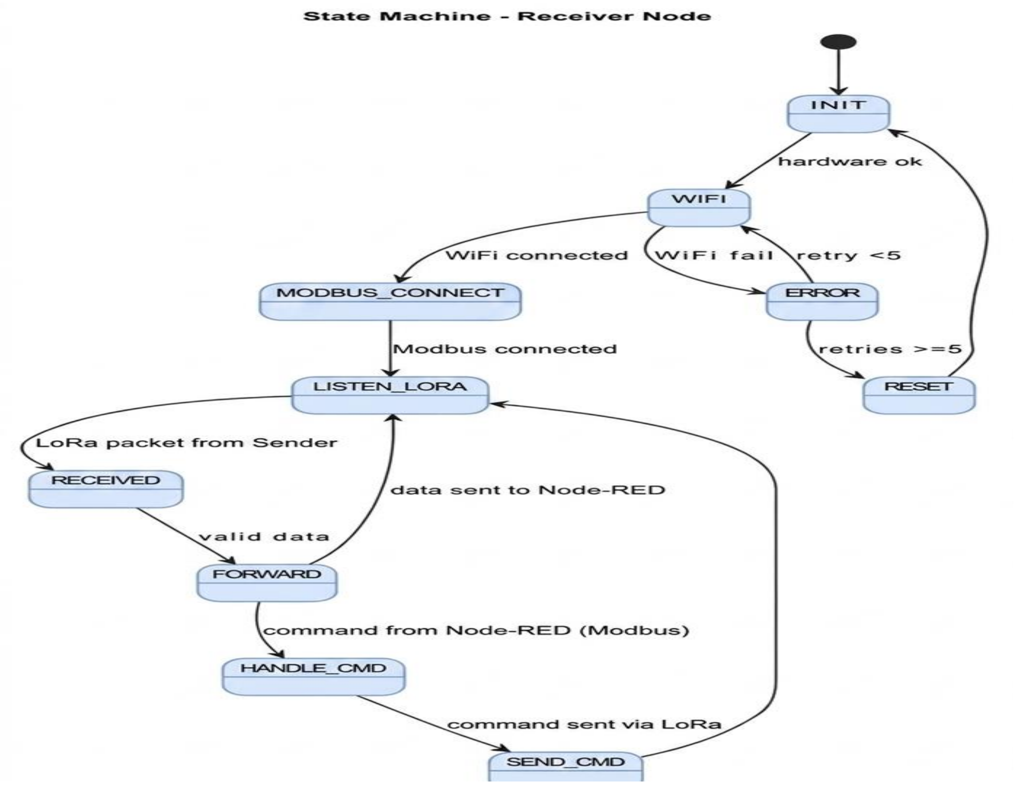


Figure 3. 5. State machine receiver

3.5. SmartFarm Layered Software Architecture & Data Flow

The component diagram illustrates the software architecture of the SmartFarm system, organized into four main layers. The sensor node (ESP32) acquires data and communicates bidirectionally via LoRa with the receiver node. The receiver node forwards data to Node-RED via MQTT or Modbus TCP. The Cloud layer centralizes storage in InfluxDB and Supabase, while the Flutter application provides remote monitoring and control capabilities. See figure 3.6:

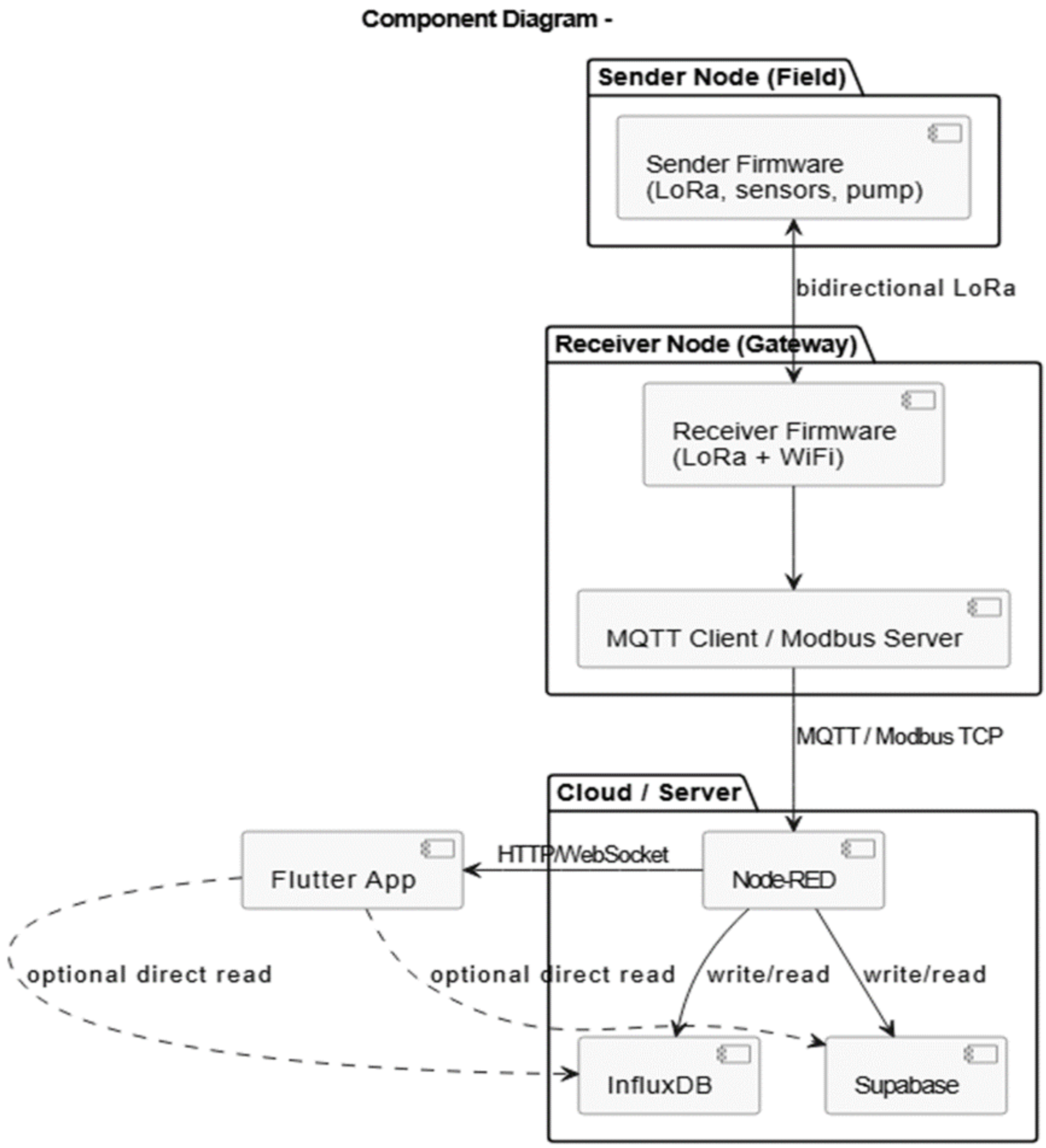


Figure 3. 5. Components diagram.

3.6. SmartFarm Physical Infrastructure & Network Topology

The deployment diagram illustrates the physical architecture of the SmartFarm system across three levels. The sensor node (ESP32 + LoRa), powered by solar panel and battery, acquires sensor data and controls the pump. The gateway (ESP32 + LoRa + WiFi) relays data to the network via MQTT/Modbus. The user interacts through the Flutter application connected via WiFi. The Windows server centralizes Node-RED, InfluxDB and Supabase services for data processing, storage and supervision.See figure 3.7:

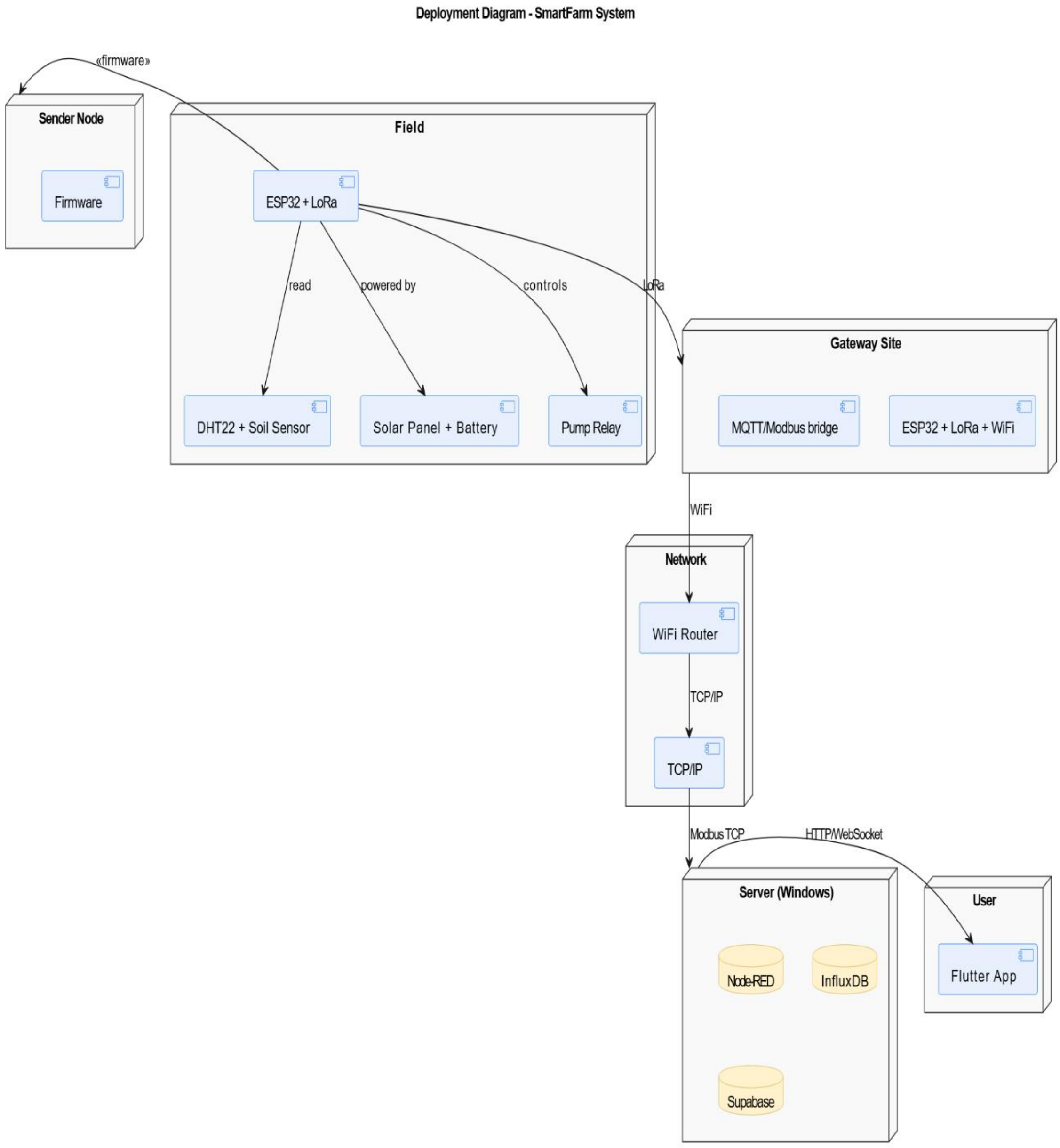


Figure 3. 6. Deployment diagram.

3.7. End-to-End Monitoring & Automated Irrigation Cycle

The sequence diagram illustrates the operational cycle of the SmartFarm system: the ESP32 sensor node wakes up every 10 minutes to read sensors (DHT22, soil moisture) and transmit data via LoRa to the gateway. The gateway exposes the data via Modbus TCP for visualization on the Flutter application and storage in InfluxDB and Supabase. . The user can manually control the pump from the Flutter interface. This cycle ensures continuous monitoring and optimized irrigation.

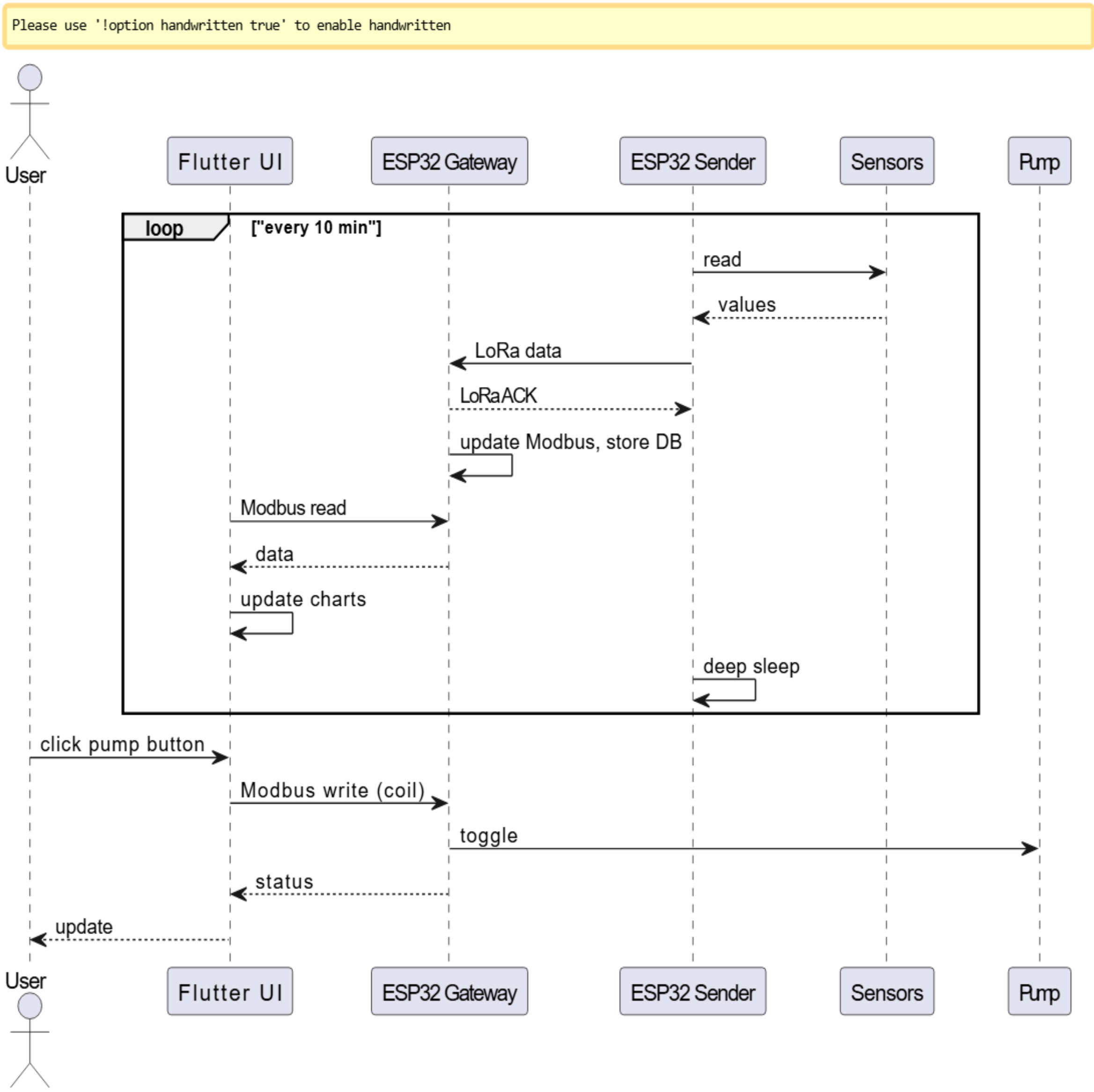


Figure 3. 7. Sequence diagram.

3.8. Esp32-now test

Esp32-now circuit

Figure 3. 8. Esp32-now circuit

sender

```
void connectMQTT() {
    //    Connect ESP32 to the Wi-Fi network
    // Establish MQTT connection with the broker
    // Reconnect automatically if the connection is lost
}

void sendSensorData() {
    // Read temperature and humidity from the DHT22 sensor
    // Read soil moisture from the analog sensor
    // Format the sensor values
    // Publish the data to the MQTT topics
}
```

3.9. MQTT End-to-End Test Framework

MQTT Integration Flow

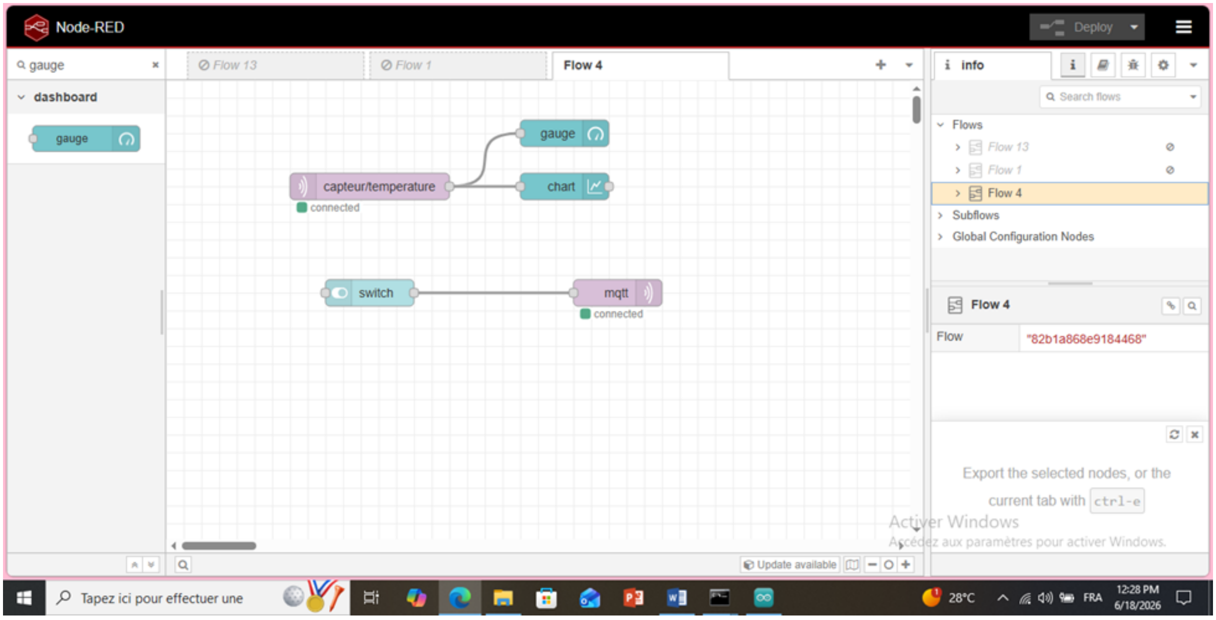


Figure 3. 9. Mqtt flow.

MQTT Monitor Dashboard

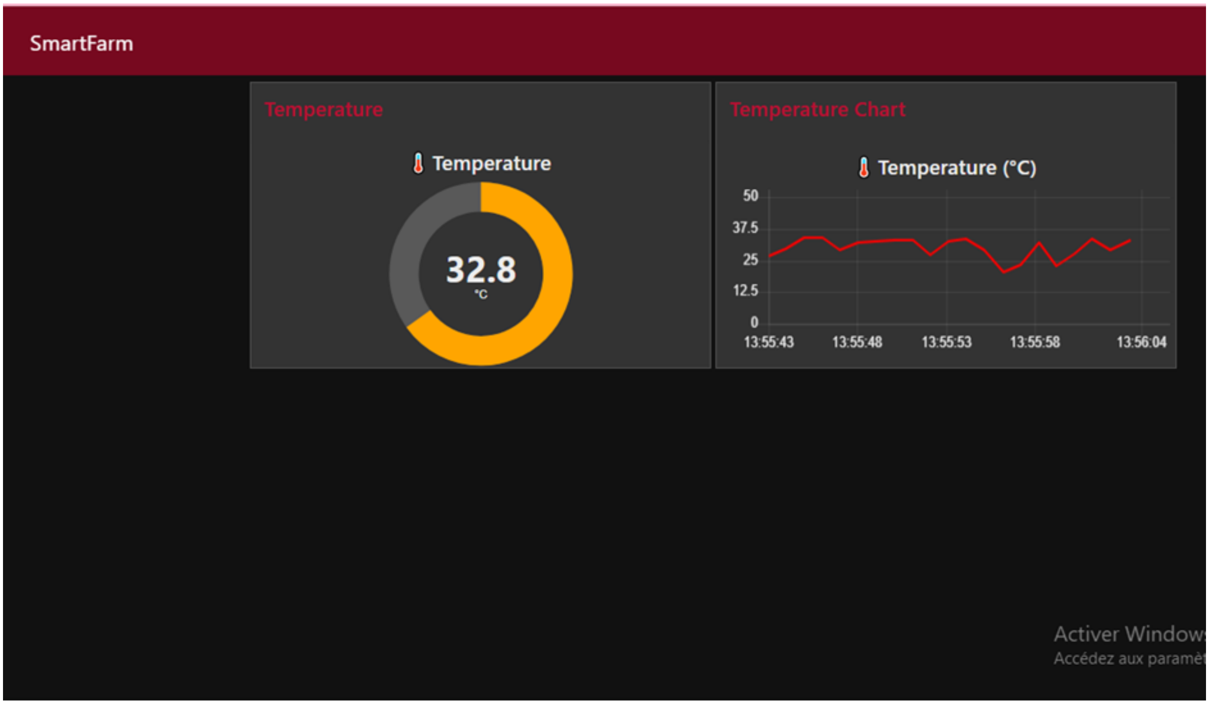


Figure 3. 10. Mqtt dashboard.

3.10 Modbus Register Read/Write Tester

Modbus Integration Flow

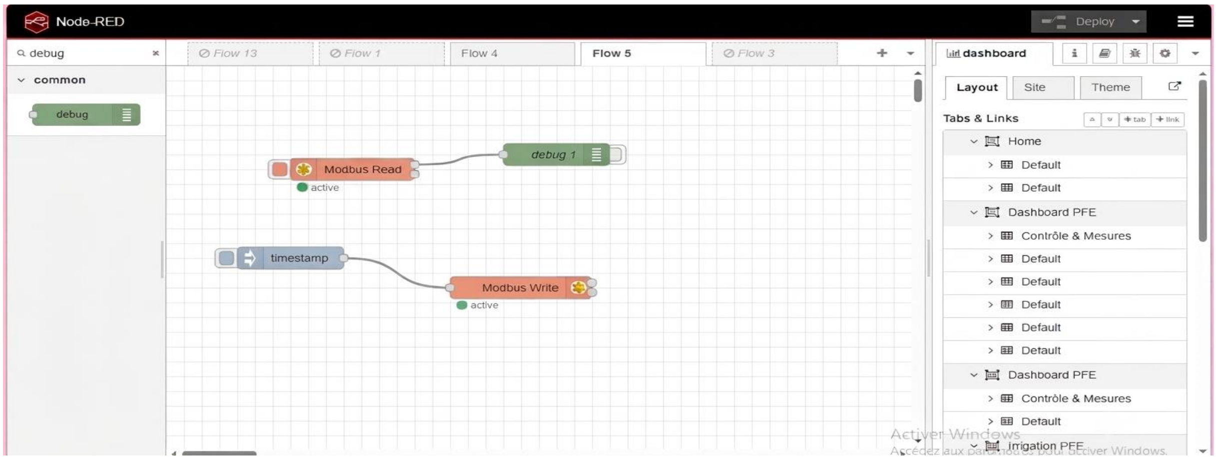


Figure 3. 11. Modbus flow.

Modbus dashboard

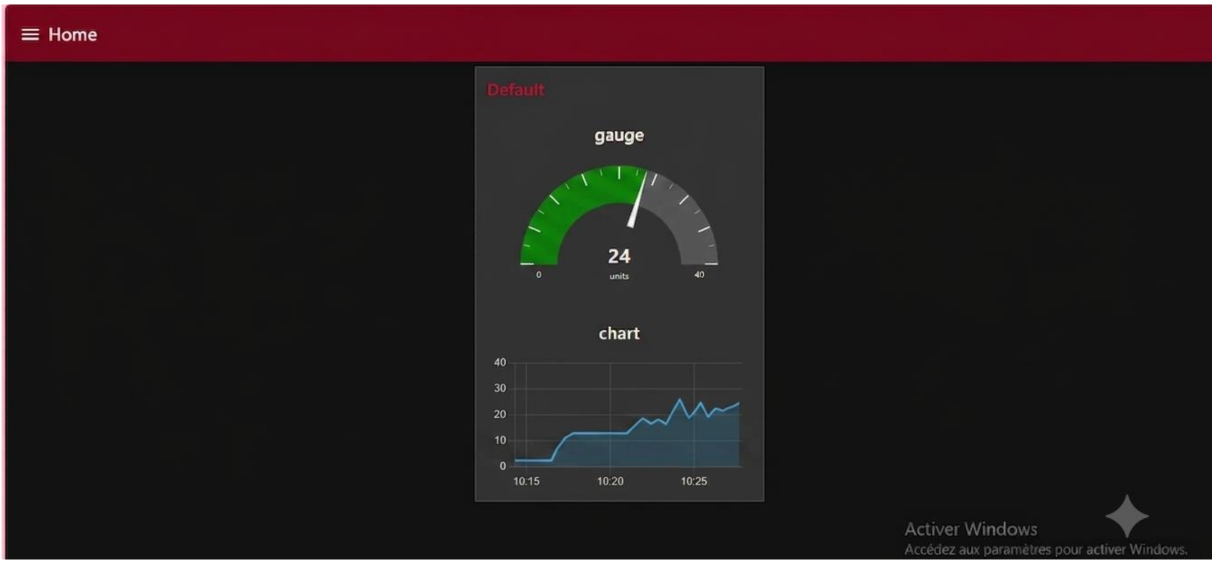


Figure 3. 12. Modbus dashboard

Modbus code

```
void initializeModbusServer() {
    // Connect ESP32S3 to the Wi-Fi network
    // Start the Modbus TCP server on port 502
    // Create holding registers for sensor data
    // Create coils for remote actuator control
}

void updateModbusRegisters() {
    // Read sensor values or generate simulation data
    // Store temperature, humidity, and soil moisture values in
registers
    // Monitor the LED control coil state
    // Update the LED status and make data available to Modbus
clients
}
```

3.11. Bidirectional code

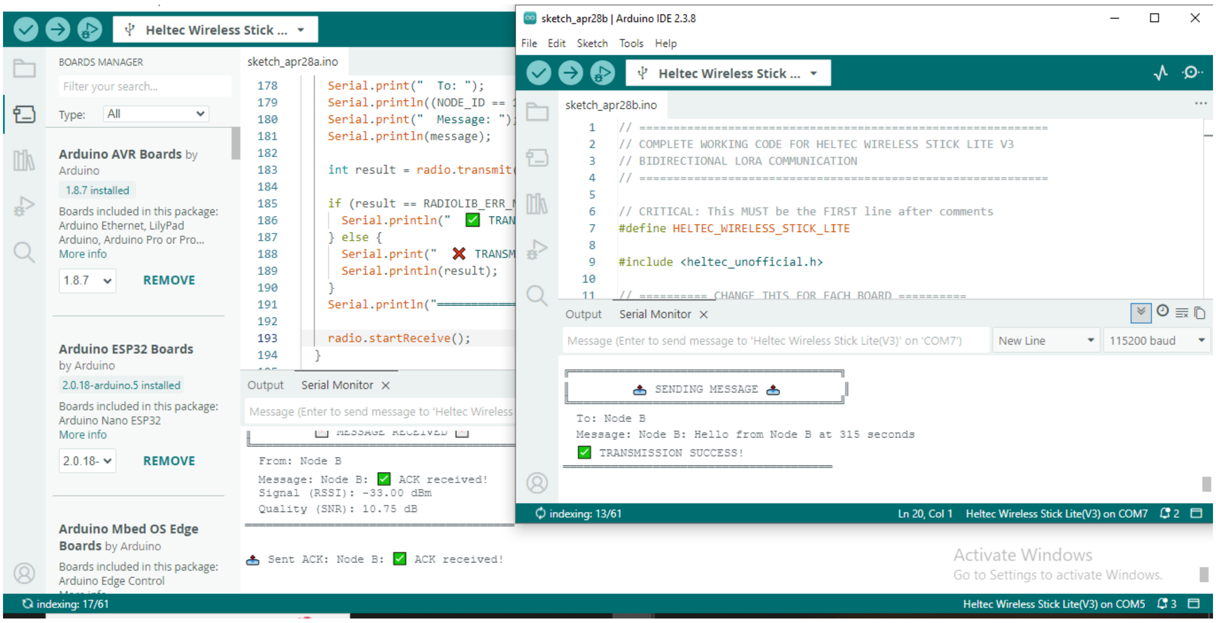


Figure 3. 13. Bidirectional code.

Sender

```
void initializeModbusServer() {
    // Connect ESP32S3 to the Wi-Fi network
    // Start the Modbus TCP server on port 502
    // Create holding registers for sensor data
    // Create coils for remote actuator control
}

void updateModbusRegisters() {
    // Read sensor values or generate simulation data
    // Store temperature, humidity, and soil moisture values in register
    // Monitor the LED control coil state
    // Update the LED status and make data available to Modbus clients
}
```

Receiver

```
void sendData() {
    // Read sensor values from the connected sensors
    // Format the collected measurements
    // Transmit data to the receiver through the communication protocol
```

```
    // Repeat the transmission periodically
}

void receiveData() {
    // Listen for incoming messages from the sender
    // Decode the received data and commands
    // Update registers or actuator states accordingly
    // Send acknowledgments or control commands back if necessary
}
```

3.12. Experimental Results

Lora circuit final

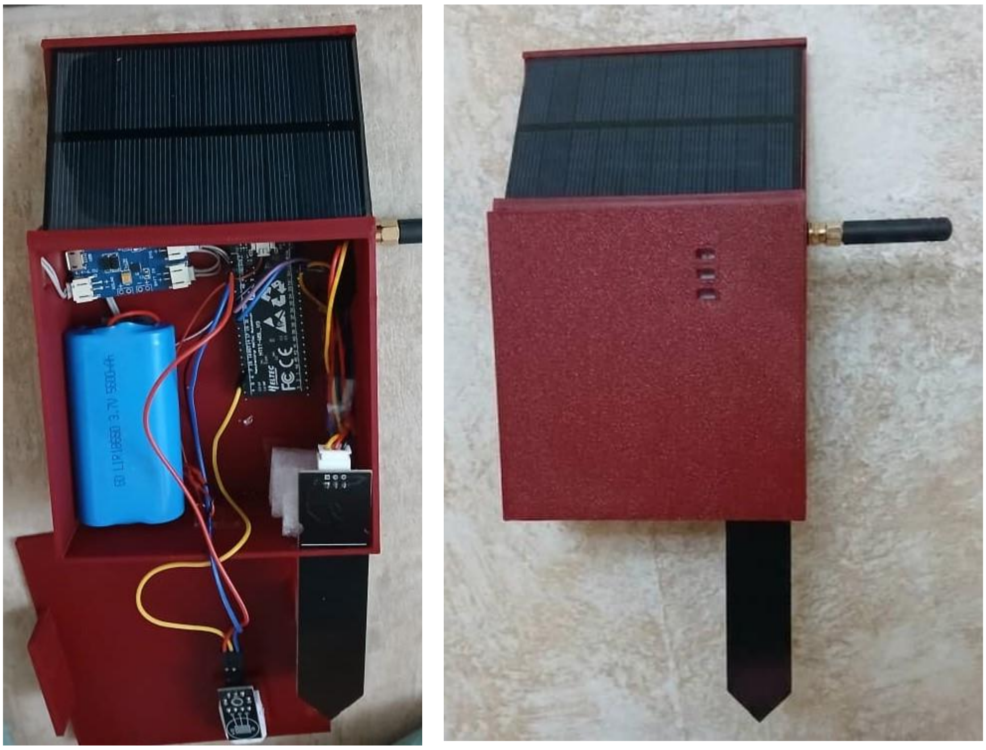


Figure 3. 14. Lora circuit final

3.13. Result shown by the sender and receiver serial monitor

```
T: 26.4°C | H: 47.2% | Soil: 100% (raw:2251) | Sent OK ✓
T: 26.4°C | H: 47.0% | Soil: 100% (raw:2253) | Sent OK ✓
T: 26.4°C | H: 47.4% | Soil: 100% (raw:2252) | Sent OK ✓
Reply received: PUMP:ON
>>> PUMP TURNED ON ✓
T: 26.4°C | H: 47.6% | Soil: 100% (raw:2250) | Sent OK ✓
T: 26.4°C | H: 47.7% | Soil: 100% (raw:2252) | Sent OK ✓
T: 26.4°C | H: 47.7% | Soil: 100% (raw:2251) | Sent OK ✓
Reply received: PUMP:OFF
>>> PUMP TURNED OFF ✓
T: 26.4°C | H: 47.5% | Soil: 100% (raw:2251) | Sent OK ✓
T: 26.4°C | H: 47.3% | Soil: 100% (raw:2251) | Sent OK ✓
T: 26.4°C | H: 47.0% | Soil: 100% (raw:2251) | Sent OK ✓
T: 26.4°C | H: 46.7% | Soil: 100% (raw:2249) | Sent OK ✓
T: 26.4°C | H: 46.5% | Soil: 100% (raw:2251) | Sent OK ✓
```

```
Received: 26.80,48.20,100
Received: 26.80,48.20,100
Received: 26.80,48.30,100
Received: 26.80,48.40,100
Received: 26.80,48.40,100
Received: 26.80,48.50,100
Outside Temp: 28.10 | Outside Humidity: 48.00
Received: 26.80,48.40,100
========================
PUMP COMMAND: OFF
========================
Received: 26.80,48.40,100
LoRa reply sent: PUMP:OFF ✓
Received: PUMP:OFF.40,100
Received: 26.80,48.40,100
========================
PUMP COMMAND: ON
========================
Received: 26.80,48.40,100
LoRa reply sent: PUMP:ON ✓
Received: PUMP:ON8.40,100
Received: 26.80,48.50,100
```

Node-red flow

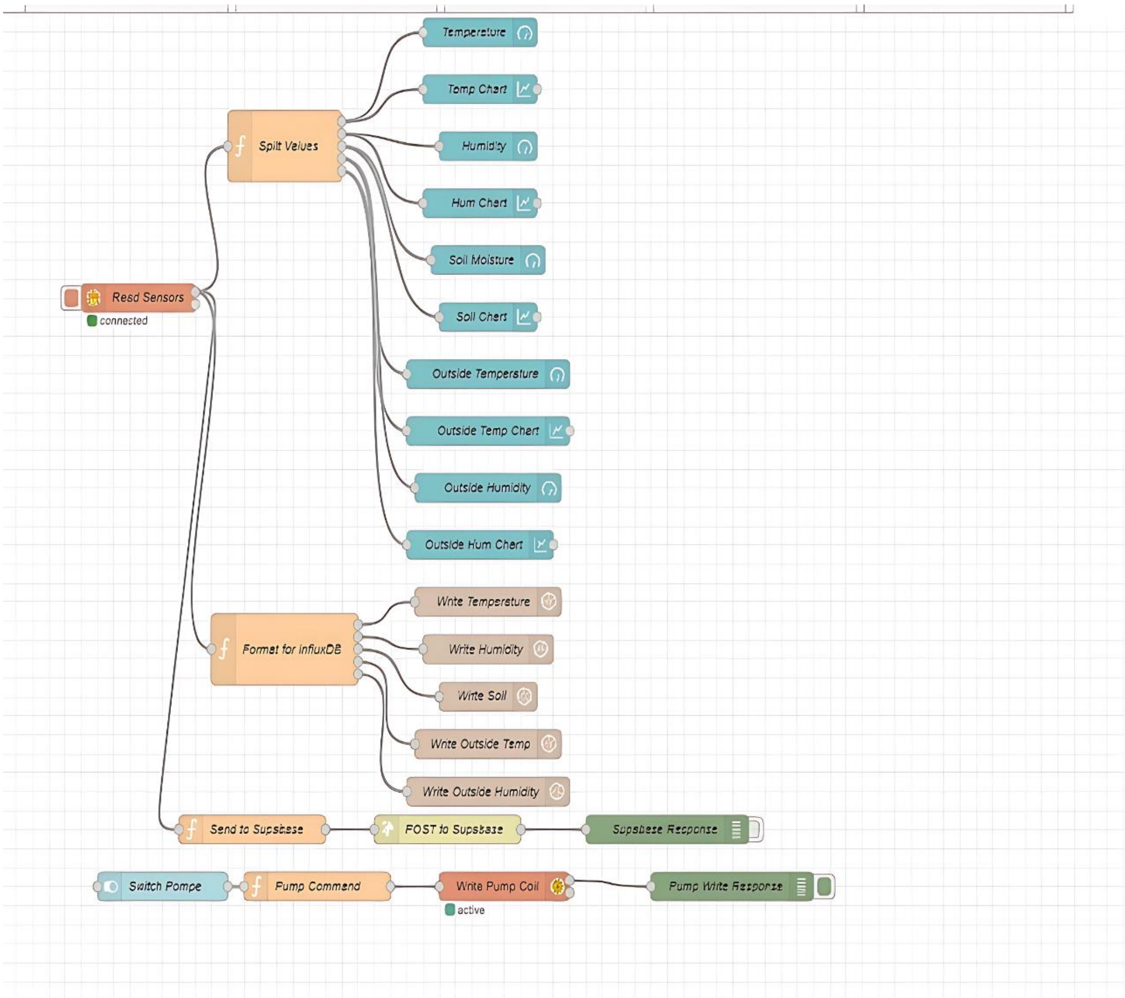


Figure 3. 15. Node-red flow.

Dashboard

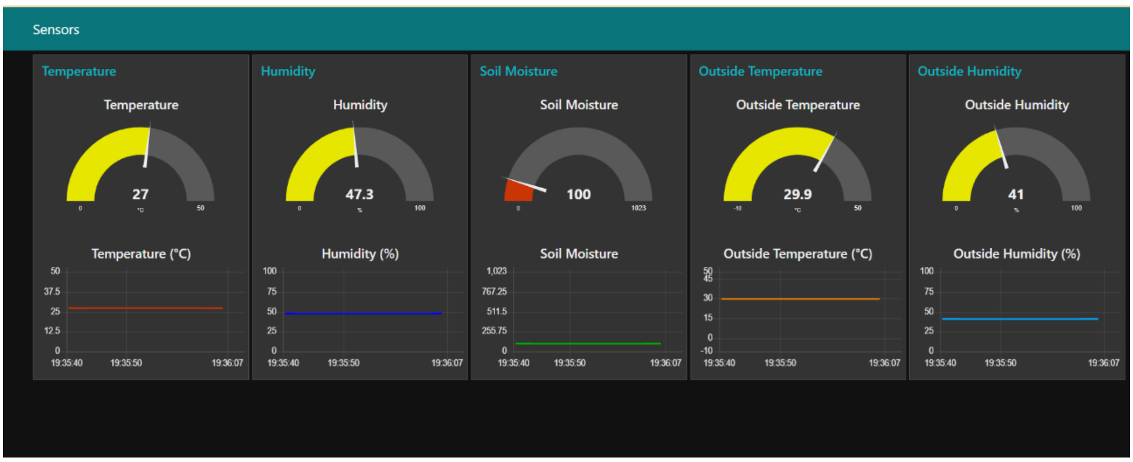


Figure 3. 16. Dashboard.

3.14. Database Performance Assessment

Experimental Results

•Supabase

id int8	created_at timestamptz	command text	acknowledged b..
47	2026-06-20 16:35:59.907623+C	off	FALSE
48	2026-06-20 17:08:08.228571+0	on	FALSE
49	2026-06-20 17:08:13.200749+0	off	FALSE
50	2026-06-20 17:08:18.372883+0	on	FALSE
51	2026-06-20 17:21:47.536217+01	off	FALSE
52	2026-06-20 17:22:36.771846+01	on	FALSE
53	2026-06-23 11:30:32.636468+0	on	FALSE
54	2026-06-23 11:30:34.4052+01	on	FALSE
55	2026-06-23 11:30:36.812053+01	off	FALSE
56	2026-06-23 13:24:05.525711+01	on	FALSE
57	2026-06-23 13:24:44.102012+01	off	FALSE
58	2026-06-23 13:29:44.004282+C	off	FALSE
59	2026-06-23 13:29:50.139394+0	on	FALSE
60	2026-06-23 19:25:20.846675+0	off	FALSE

Figure 3. 17. Pump table.

id int8	created_at timestamptz	temperature float8	humidity flo...	soil_moisture int4	outside_temp float8
4070	2026-06-17 22:44:16.518+01	28.6	72.6	93	27.1
4071	2026-06-17 22:49:24.509+01	28.6	70.6	89	27.1
4072	2026-06-17 22:49:30.53+01	28.6	70.7	92	27.1
4073	2026-06-17 22:53:28.399+01	28.6	70.6	87	27.1
4074	2026-06-17 22:57:40.316+01	28.4	68.9	86	27.1
4075	2026-06-17 22:57:58.378+01	28.4	68.8	86	27.1
4076	2026-06-17 22:58:13.402+01	28.4	69	86	27.1
4077	2026-06-17 23:00:25.036+01	28.3	67.7	94	27.1
4078	2026-06-17 23:00:49.116+01	28.3	67.5	87	27.1
4079	2026-06-17 23:00:52.106+01	28.3	67.5	86	27.1
4080	2026-06-17 23:01:33.354+01	28.3	67.1	87	27.1
4081	2026-06-17 23:02:53.028+01	28.2	66.6	89	27.1
4082	2026-06-17 23:02:56.04+01	28.2	66.6	89	27.1

Figure 3. 18. Sensors table.

- **InfluxDB**

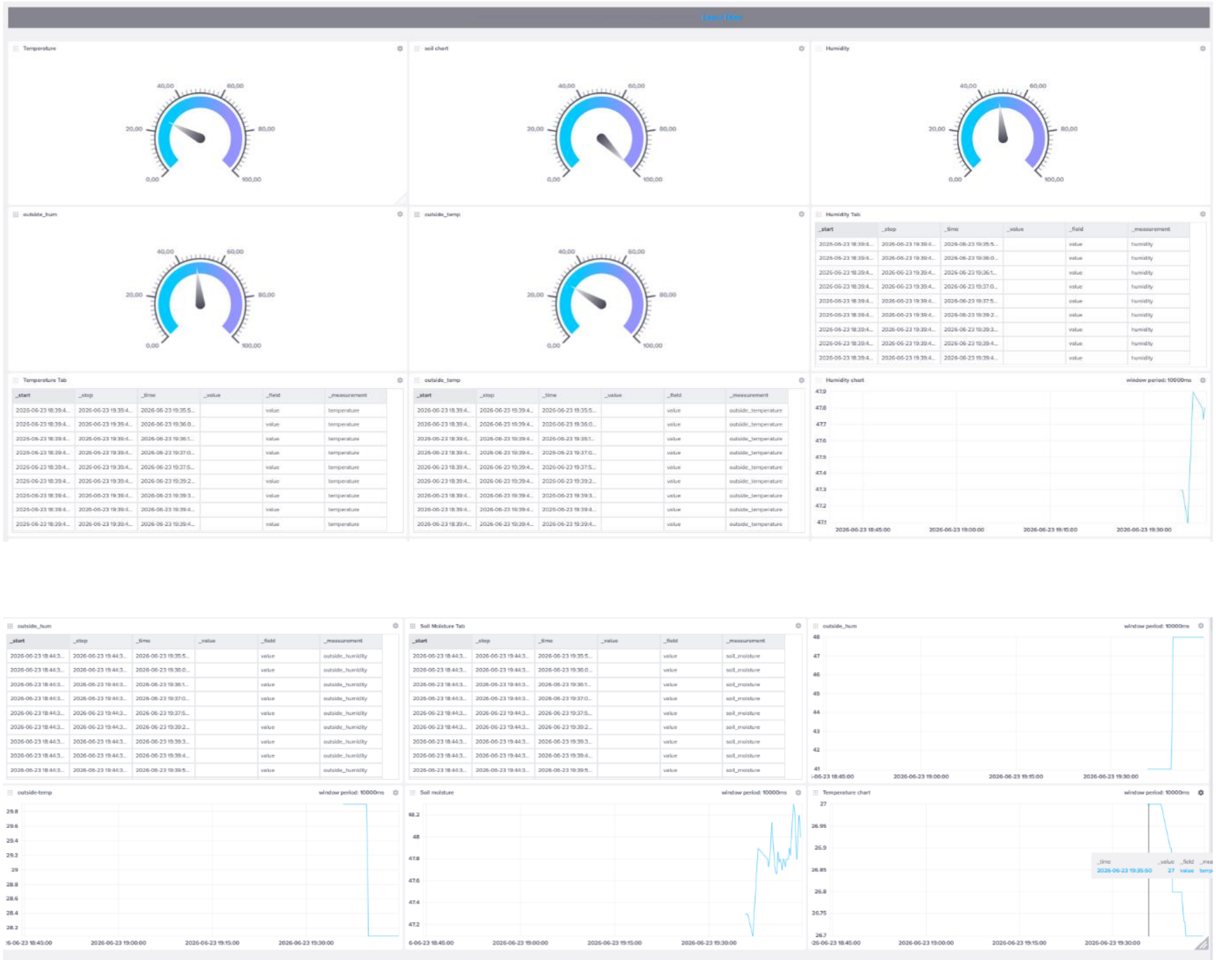


Figure 3. 19. InfluxDB.

- **Mobile application**

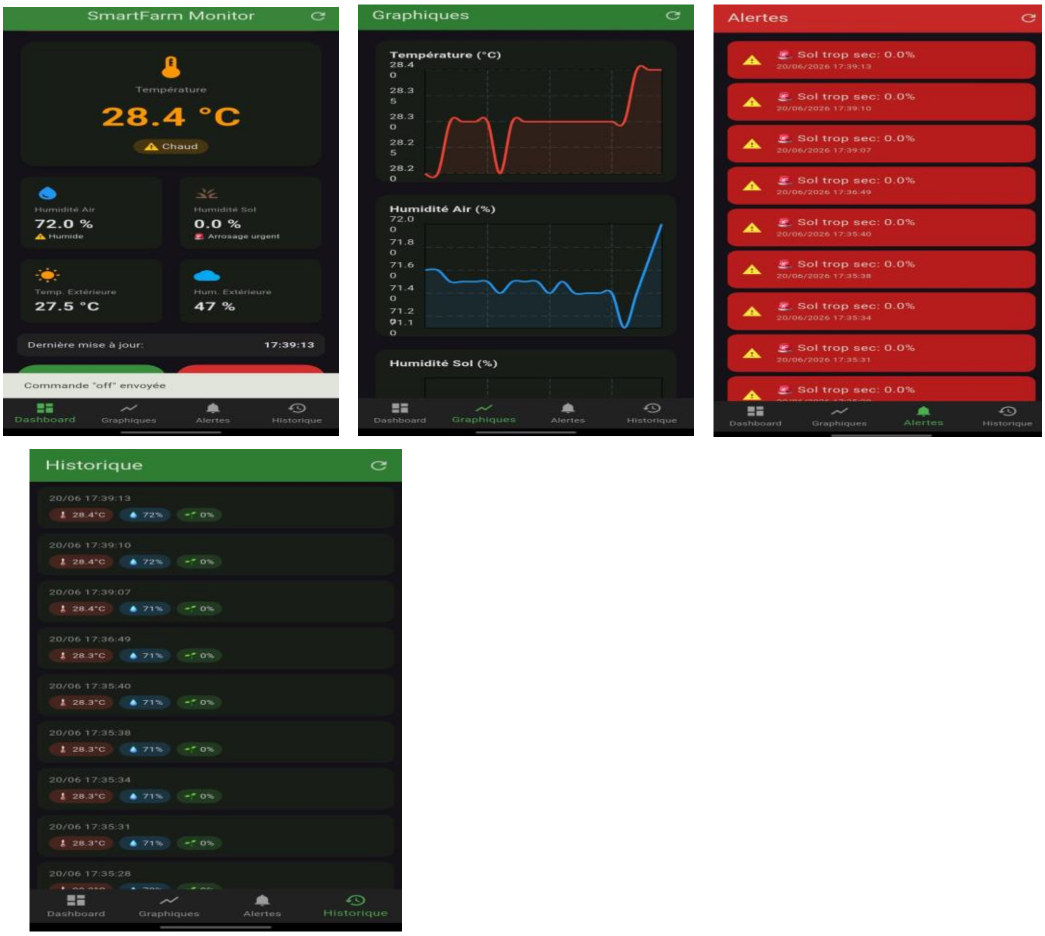


Figure 3. 20. Flutter app.

Appendix

Appendix A- SOLUTIONS SCADA IoT EXISTANTES

A.1. Introduction

The field of industrial supervision is undergoing a profound transformation driven by the emergence of Internet of Things (IoT) technologies. Historically, SCADA (Supervisory Control and Data Acquisition) systems relied on proprietary architectures, closed protocols and expensive hardware infrastructures, limiting their accessibility and scalability. Today, these constraints are easing with the advent of more open, flexible and interconnected solutions, leveraging lightweight communication protocols (MQTT, LoRa, Modbus TCP), open-source platforms (Node-RED, InfluxDB) and low-cost connected devices. This section provides an overview of existing SCADA IoT solutions, highlighting their contributions and limitations, before comparing them with the architecture proposed for our SmartFarm project..

A.2. SCADA IoT solutions open-source

The emergence of open-source platforms has transformed the industrial supervision landscape by democratizing access to control and data acquisition systems. These solutions now offer viable, cost-effective and technically performant alternatives to traditional proprietary offerings. Among these, Node-RED, developed by IBM, stands out as a visual programming tool that orchestrates data flows between services and protocols through an intuitive web interface. JSON-SCADA provides a portable and scalable SCADA/IIoT platform centered on MongoDB, deployable via Docker and Kubernetes. Eclipse Hono offers an industrial-grade MQTT communication solution for large-scale deployments, while ThingsBoard delivers a turnkey visualization platform with customizable dashboards and integrated device management..

A.3. SCADA IoT solutions sectorielles (Agricole)

The agricultural sector has seen the emergence of specific SCADA IoT solutions to address the challenges of precision agriculture. CENTAR-IA (Intelligent Telecontrol Centre for Olive and Almond Trees) in Spain integrates a SCADA system (WinCC OA) with innovative SEPHTA sensors for measuring tree water potential, combined with artificial intelligence algorithms for irrigation optimization. The PMKSY projects in India illustrate the large-scale integration of SCADA-IoT solutions for the modernization of irrigation networks, with canal monitoring and water accounting systems. Additionally, crop monitoring solutions using thermal cameras (MLX90620) and environmental sensors have been developed, communicating via Modbus TCP to remotely accessible cloud interfaces.

A.4. Comparison with conventional industrial architectures

Traditional SCADA systems (Siemens WinCC, Schneider Citect, AVEVA Wonderware) rely on a hierarchical layered architecture (CIM pyramid) where each level performs a specific function: sensors/actuators at the field level, PLCs for control, SCADA for visualization, and ERP for enterprise management. While robust, these solutions have major limitations: high cost (licenses, hardware, maintenance), closed architecture (proprietary protocols, limited interoperability), deployment complexity (specialized skills required), and wired communication that is difficult to deploy at scale. Our SmartFarm project offers an agile and cost-effective alternative using standard components (ESP32), open protocols (LoRa, MQTT) and open-source supervision (Node-RED). The system integrates automated control based on configurable irrigation thresholds (trigger at <30% soil moisture, stop at >60%) and remote control via the Flutter application.

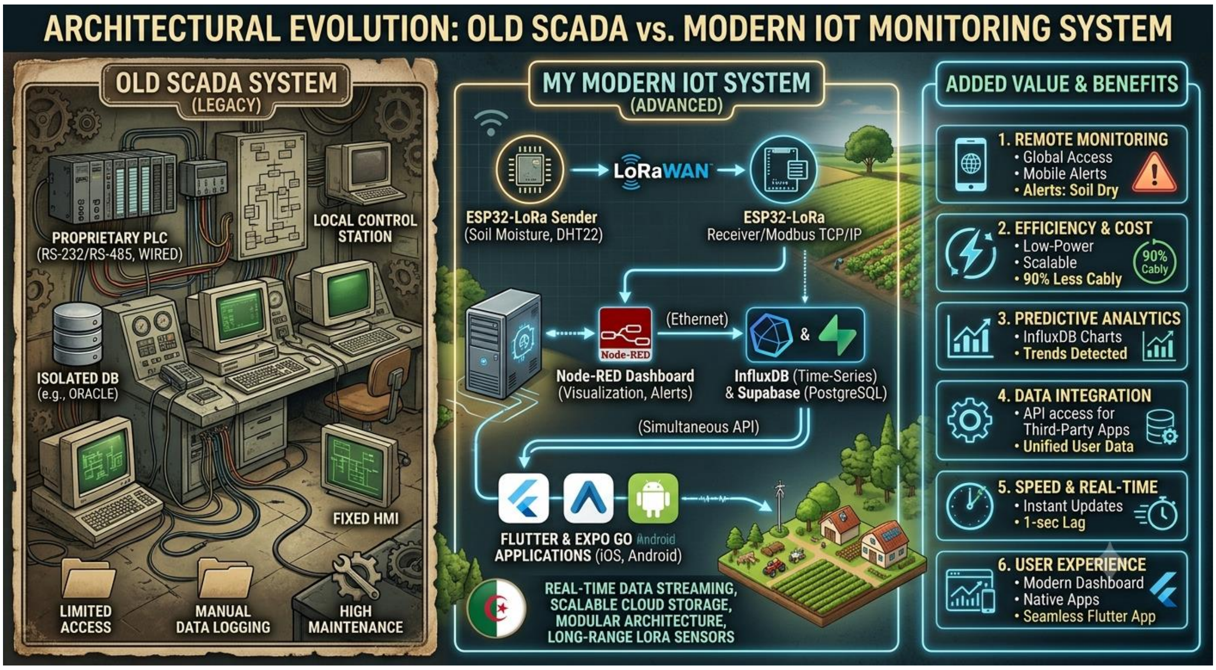


Figure A. 1. Architectural Evolution of SCADA Systems.

Appendix B- ABRÉVIATIONS TECHNIQUES

Table B. 1

IoT	Internet of Things
SCADA	Supervisory Control and Data Acquisition
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
TCP/IP	Transmission Control Protocol / Internet Protocol
MQTT	Message Queuing Telemetry Transport
LORA	Long Range
LORAWAN	Long Range Wide Area Network
Esp-now	Espressif Now
Wi-fi	Wireless Fidelity
BLE	Bluetooth Low Energy
GPIO	General Purpose Input/Output
API	Application Programming Interface
IDE	Integrated Development Environment
DB	database

JSON	JavaScript Object Notation
SQL	Structured Query Language
DHT22	Digital Humidity and Temperature sensor 22
ESP32	Espressif Systems 32-bit microcontroller
P2P	Peer-to-Peer
PLC	Programmable Logic Controller
RTU	Remote Terminal Unit
CPU	Unité Centrale de Traitement

Appendix C- Business Model Canvas

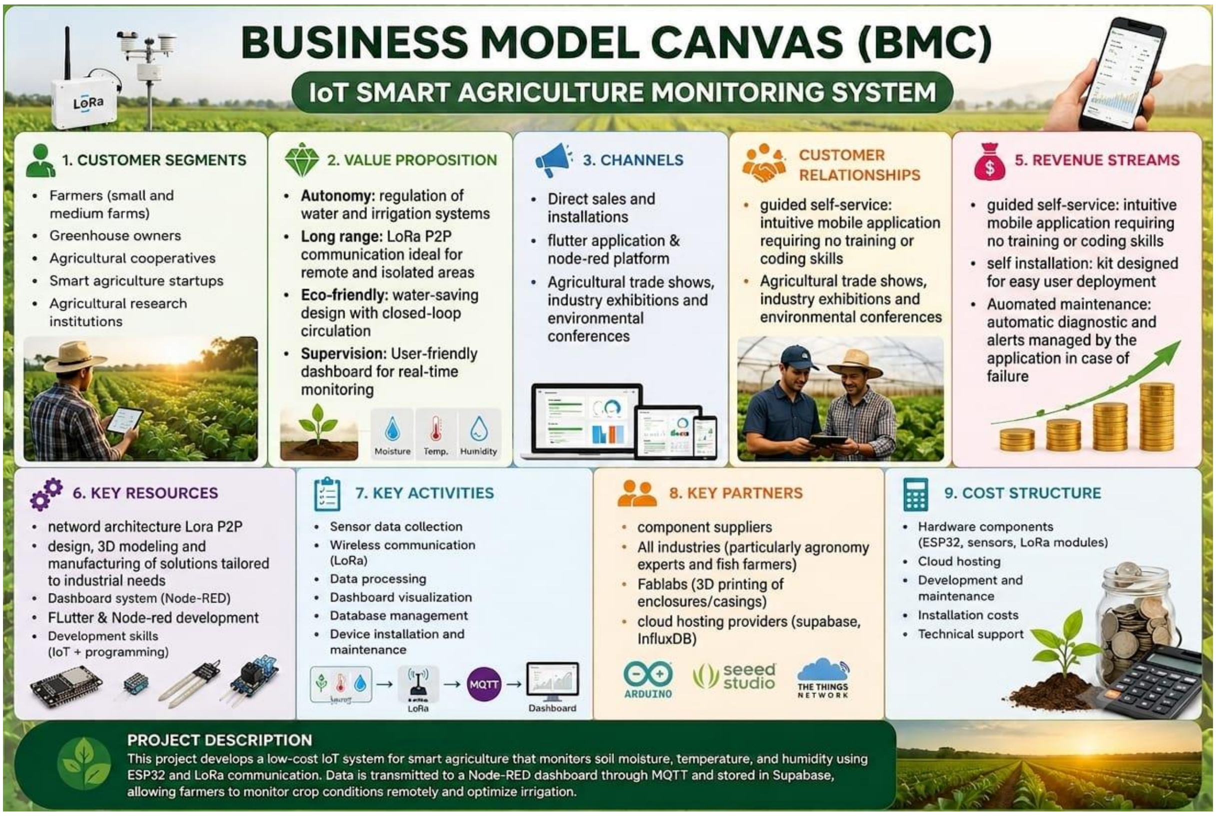
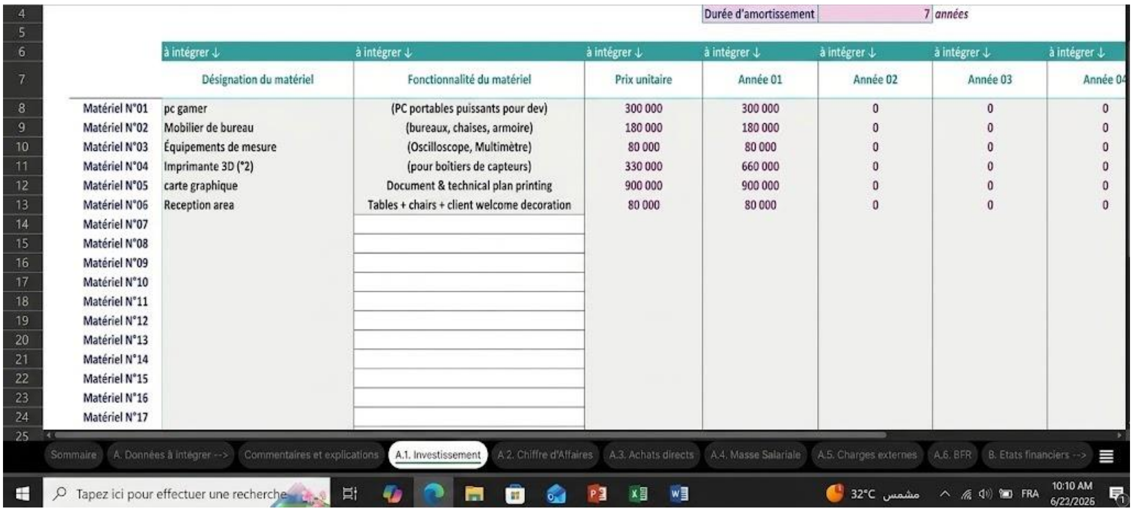


Figure C. 1. Business Model Canvas.

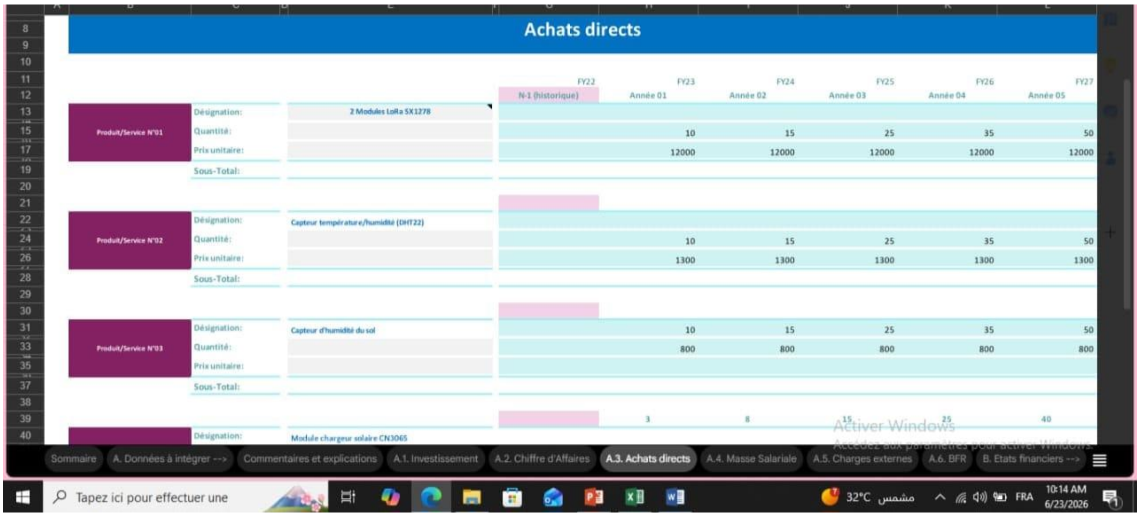
Appendix D- Business Plan

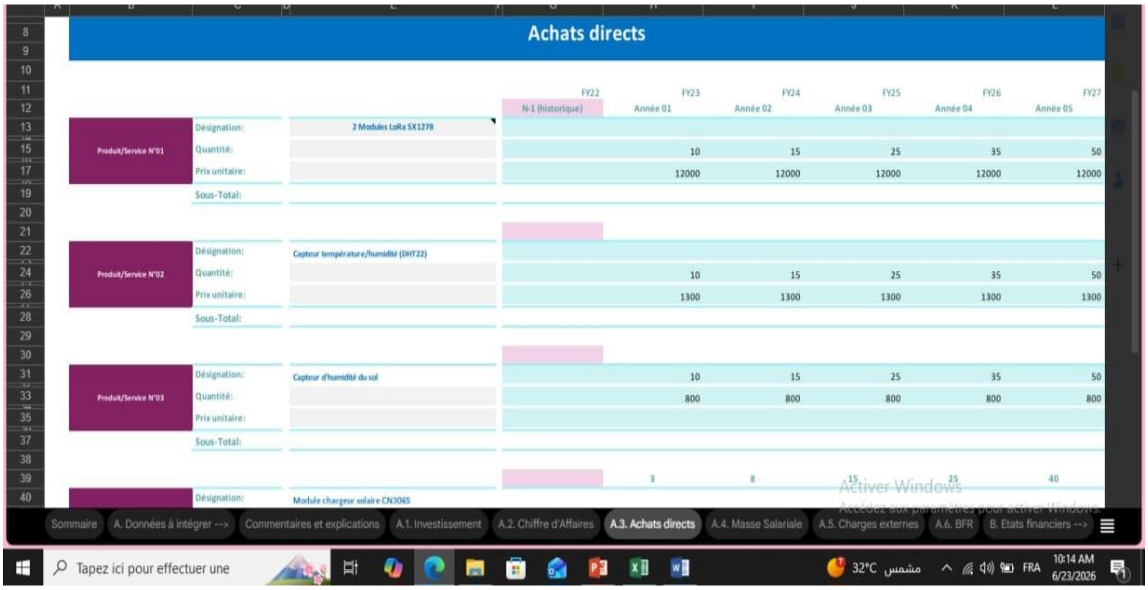


Durée d'amortissement: 7 années

	Désignation du matériel	Fonctionnalité du matériel	Prix unitaire	Année 01	Année 02	Année 03	Année 04
Matériel N°01	pc gamer	(PC portables puissants pour dev)	300 000	300 000	0	0	0
Matériel N°02	Mobilier de bureau	(bureaux, chaises, armoire)	180 000	180 000	0	0	0
Matériel N°03	Équipements de mesure	(Oscilloscope, Multimètre)	80 000	80 000	0	0	0
Matériel N°04	Imprimante 3D (*2)	(pour boîtiers de capteurs)	330 000	660 000	0	0	0
Matériel N°05	carte graphique	Document & technical plan printing	900 000	900 000	0	0	0
Matériel N°06	Reception area	Tables + chairs + client welcome decoration	80 000	80 000	0	0	0
Matériel N°07							
Matériel N°08							
Matériel N°09							
Matériel N°10							
Matériel N°11							
Matériel N°12							
Matériel N°13							
Matériel N°14							
Matériel N°15							
Matériel N°16							
Matériel N°17							

Figure D. 1. Investissement.





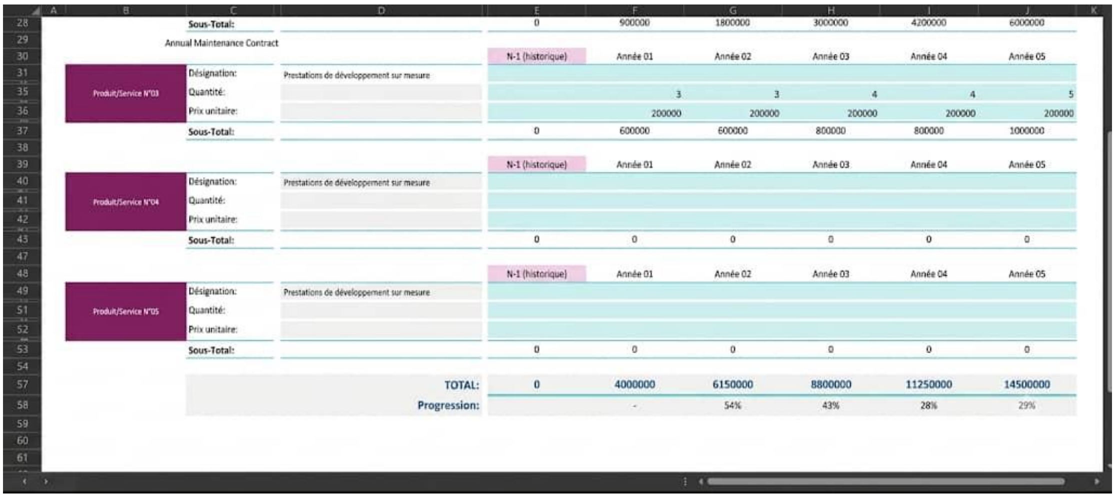


Figure D. 2. Achats.

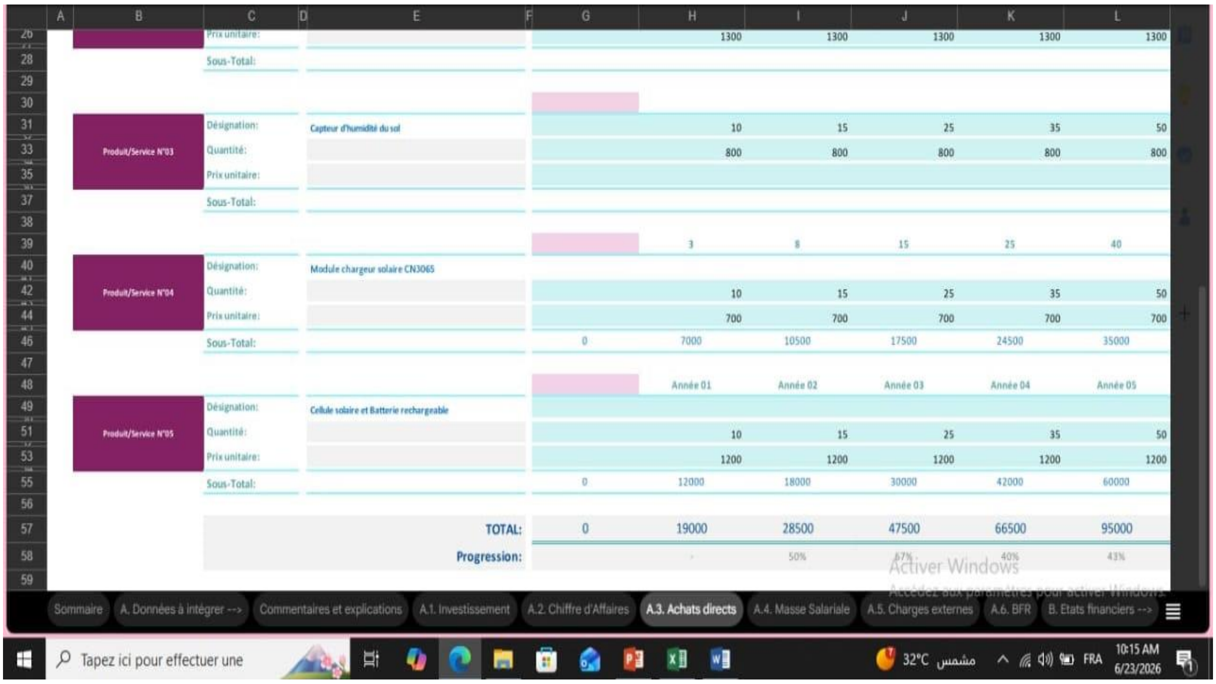


Figure D. 4 Achats-Suite.

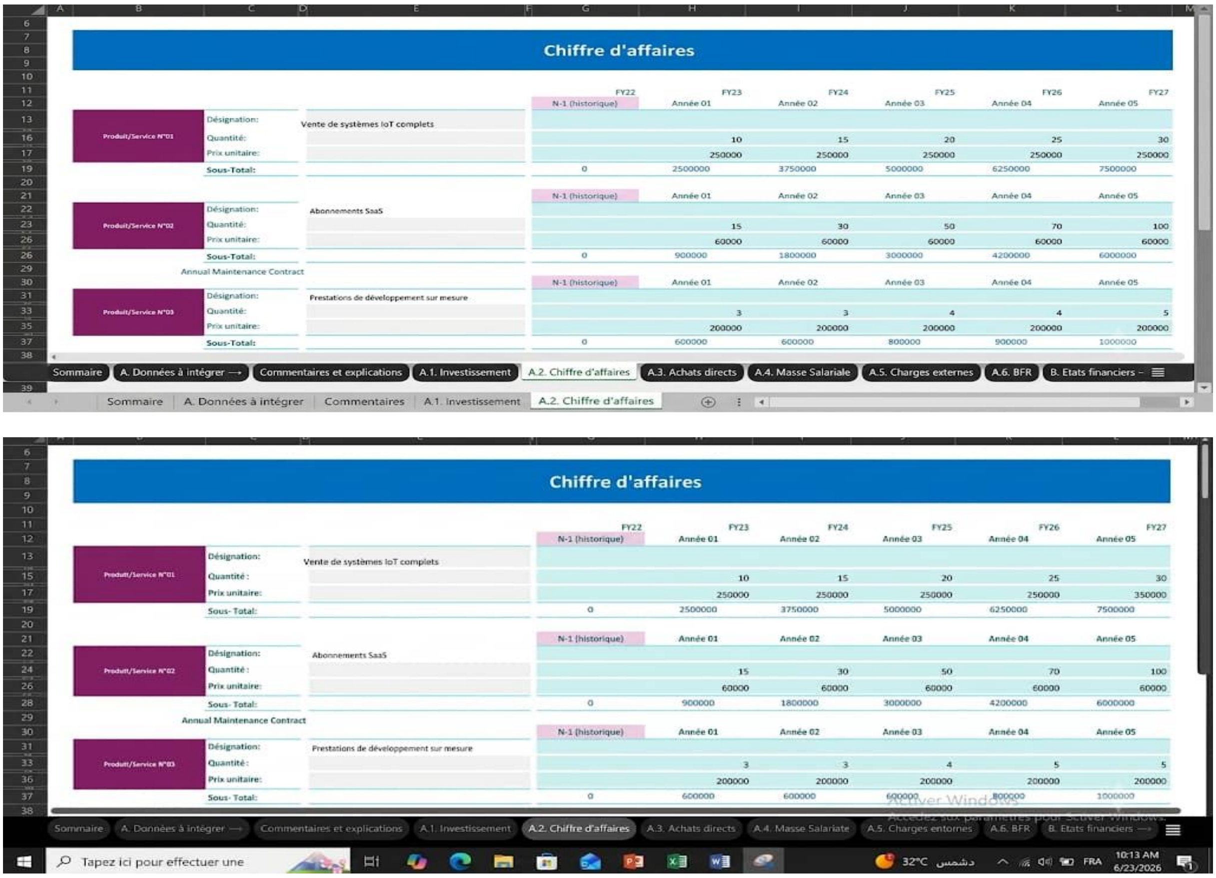


Figure D. 3. Chiffre d’affaires.

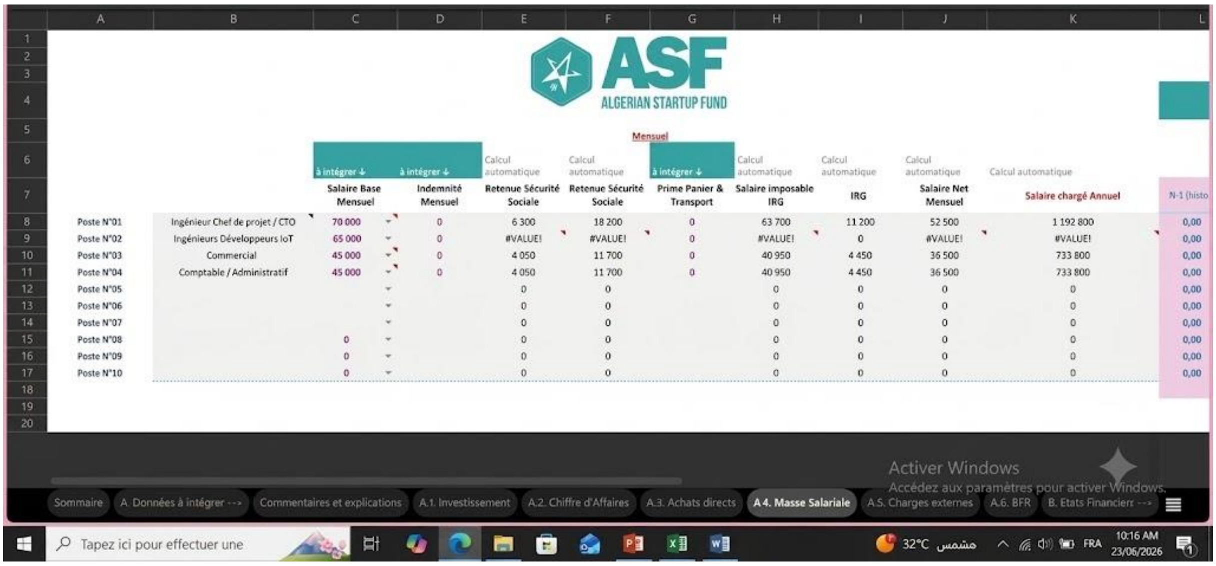


Figure D. 5. Masse salariale.

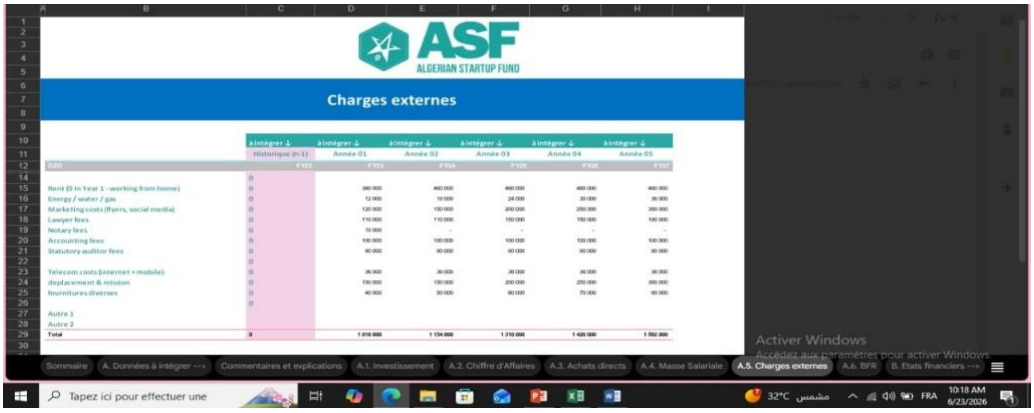


Figure D. 6. Charges externes.

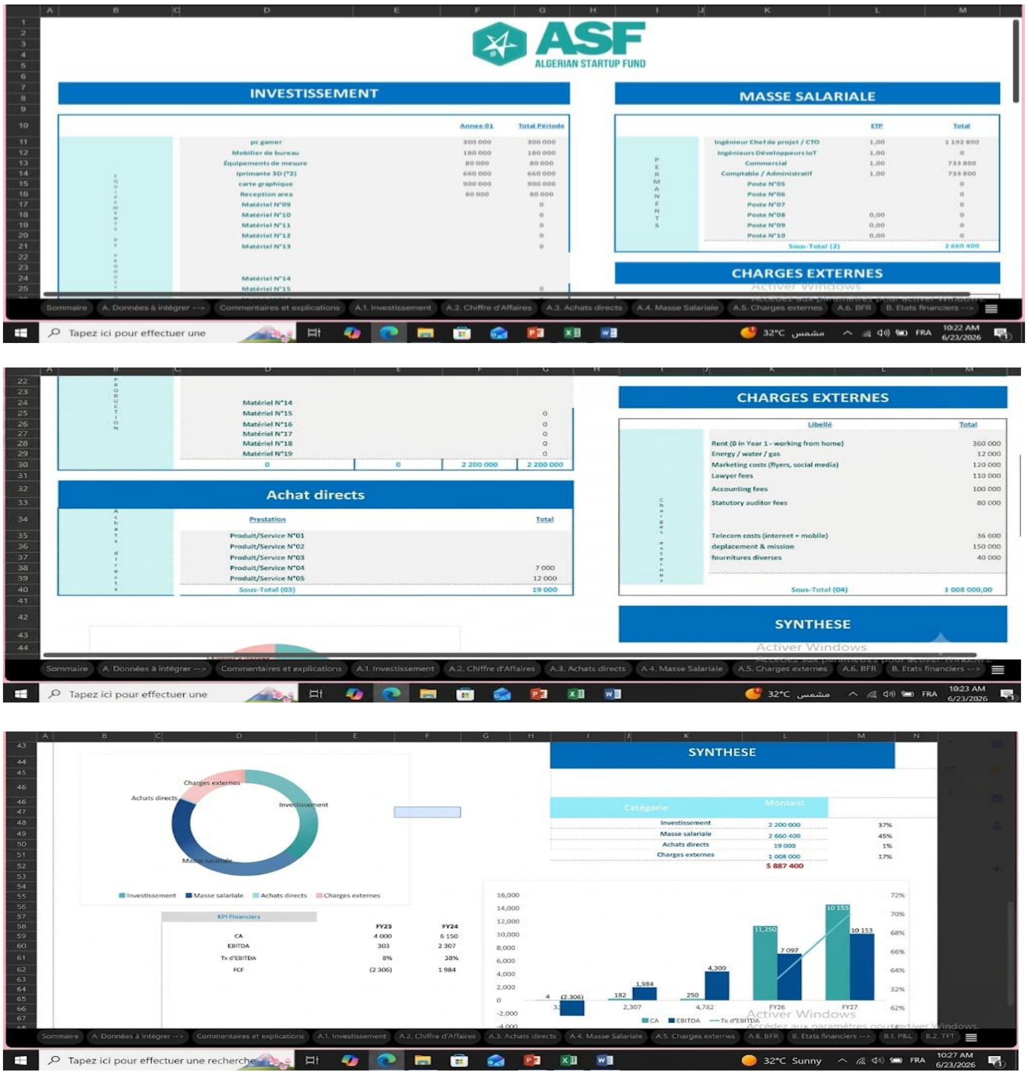


Figure D. 7. Synthese.

REFERENCES

References:

[1] Jankov, S., Novaković, B., Marković, M., Šarenac, U., Landup, D., Premčevski, V., & Đorđević, L. (2026). Optimization of Oil Production Using Sucker Rod Pumps via Predictive Elimination of Paraffin Issues. *Applied Sciences*. https://doi.org/10.3390/app16031590

[2] Cavazza, F. (2026, June 16). Internet des objets. FredCavazza.net. https://fredcavazza.net/internet-des-objets/

[3] Nozomi Networks. (2026, June 16). ESP-NOW protocol analysis and vulnerability research. https://www.nozominetworks.com/resources/esp-now-protocol-analysis-and-vulnerability-research

[4] Sciences Physiques et Numérique. (2026, June 16). ESP32. https://physique.david-therincourt.fr/microcontroleurs/esp32/

[5] espblockly.com. (s.d.). Introduction | ESP32 tutorial. https://espblockly.com/intro.html

[6] Scribd. (s.d.). Présentation de la carte ESP32. https://www.espressif.com/

[7] uPesy. (s.d.). ESP32 pinout: How to use GPIO pins? https://www.upesy.com/blogs/tutorials/esp32-pinout-reference-gpio-pins-ultimate-guide

[8] ElectroPi. (s.d.). ESP32 LoRa SX1278 0.96 inch blue OLED display BT WiFi module for Arduino (Type C connector). https://heltec.org/project/wireless-stick-lite-v2/

[9] Made-in-China.com. (s.d.). Fabricants de panneaux solaires stratifiés en résine époxy. https://fr.made-in-china.com/manufacturers/epoxy-resin-laminated-solar-panels.html

[10] HiLetgo. (s.d.). HiLetgo - Electronic Components & Modules. http://www.hiletgo.com/

[11] Seeed Studio. (2025, December 29). Grove - Temperature & Humidity Sensor Pro (DHT22). *Seeed Studio Wiki*. https://wiki.seeedstudio.com/Grove-Temperature_and_Humidity_Sensor_Pro/

[12] DFRobot. (s.d.). DFRobot - Open-Source Hardware Electronics and Kits. https://www.dfrobot.com/

[13] Ayezabu, A. (s.d.). Supabase vs Firebase: Evaluation of performance and development of Progressive Web Apps. Theseus. https://www.theseus.fi/bitstream/handle/10024/771009/Ayezabu_Amanuel.pdf?sequence=2&isAllowed=y

[14] InfluxData. (s.d.). Time series platform. https://www.influxdata.com/time-series-platform/

[15] DB-Engines. (s.d.). DB-Engines ranking. https://db-engines.com/en/ranking

[18] IIOT Products. (2022, 27 juillet). Implement and deploy Node-RED program — MC-EX-FQ4 fast transistor output expansion module Node-RED startup guide. https://iiotproducts.com/blog/docs/mc-ex-fq4-fast-transistor-output-expansion-module/expansion-module-node-red-startup-guide/implement-and-deploy-node-red-progra

[16] Aghiles, R. (2023). Conception et réalisation d'un système réseau de serres à base des cartes ESP-32 (Doctoral dissertation, FGEI. UMMTO).

https://dspace.ummto.dz/handle/ummto/26882

[17] Soumia, Z. I. D. O. U. R., & Ibtissem, K. E. B. A. I. L. I. (2022). Etude et réalisation d'un système de télésurveillance avec détection de présence, Basé sur une carte ESP32-CAM (Doctoral dissertation, faculté des sciences et de la technologie univ bba).http://10.10.1.6:4000/handle/123456789/2513

[18] López, J. J. (s.d.). Rapid IoT prototyping: A visual programming tool and hardware solutions for LoRa-based devices. ResearchGate. https://www.researchgate.net/publication/373534295_Rapid_IoT_Prototyping_A_Visual_Programming_Tool_and_Hardware_Solutions_for_LoRa-Based_Devices

[19] Bonilla, V., Campoverde, B., & Yoo, S. G. (2023). A systematic literature review of LoRaWAN: Sensors and applications. *Sensors, 23*(20), 8440. https://www.mdpi.com/1424-8220/23/20/8440

[20] Silva, E. F., Figueiredo, L. M., de Oliveira, L. A., Chaves, L. J., Rosário, D., & Cerqueira, E. (2023). Adaptive parameters for LoRa-based networks physical-layer. *Sensors, 23*(10), 4597. https://www.mdpi.com/1424-8220/23/10/4597

[22] Berto, R., Napoletano, P., & Savi, M. (2021). A LoRa-based mesh network for peer-to-peer long-range communication. *Sensors, 21*(13), 4314. https://www.mdpi.com/1424-8220/21/13/4314

[23] Magzym, Y., Eduard, A., Urazayev, D., Fafoutis, X., & Zorbas, D. (2023, July). Synchronized ESP-NOW for improved energy efficiency. In 2023 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 57-62). IEEE..

https://www.researchgate.net/publication/370952583_Synchronized_ESP-NOW_for_Improved_Energy_Efficiency