



Research article

Secure smart home architecture for ambient-assisted living using a multimedia Internet of Things based system in smart cities

Ridha Ouni¹ and Kashif Saleem^{2,*}

¹ Department of Computer Engineering, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

² Center of Excellence in Information Assurance (CoEIA), King Saud University, Riyadh 11653, Saudi Arabia

* **Correspondence:** Email: ksaleem@ksu.edu.sa.

Abstract: Recent advances in smartphones and remote monitoring based on the Internet of Things (IoT) have enabled improved multidimensional intelligent services. The advent of IoT-based wearable and multimedia sensors has prevented millions of mishaps through seamless and systematic monitoring. An IoT-based monitoring system is composed of several sensor devices to measure vital signs, fall detection, energy consumption, and visual recognition. As the data collected by the sensors are transmitted to cloud storage through the Internet, data security is a major concern when transmitting data from remote locations. To improve data security and prediction accuracy, in this study, we proposed a smart and secure multimedia IoT monitoring system for smart homes backed up by smart grid supervisory control and data acquisition (SCADA). The proposed system employs state-of-the-art IoT microcontrollers and hardware devices and integrates them in a manner that significantly affects the accuracy and speed of the entire system. Furthermore, the information gathered from IoT is securely transferred through private channels and stored on the cloud, which can be accessed authentically and reliably using an information system built into an IoT application. The output was extensively compared in terms of power consumption and delivery ratio, which were based on the values collected with sequence numbers. The comparative analysis demonstrated that the proposed approach provides increased prediction accuracy and better security. Hence, the proposed power-efficient prototype model monitors the entire smart home environment in real time and serves as an early warning system for critical situations.

Keywords: Internet of Things; remote sensing system; data security; smart homes

1. Introduction

The Internet of Things (IoT) has enormous potential in a wide range of fields, including smart grids, earth observation, environmental monitoring, agriculture, resource management, public health, public security, transport, and the military. However, over the past few years, remote sensing has emerged as an attractive new source of data for environmental applications. These applications have acquired considerable momentum in terms of spatial, radiometric, and spectral resolution. Remote sensing and monitoring systems are responsible for acquiring, storing, and using collected data to make decisions and analyze complex problems. Recent studies have evaluated the collateral operation of IoT and remote sensing systems and how IoT constitutes an integral part of remote sensing [1,2]. In particular, the studies considered the technological relationship between IoT and remote sensing edge components, that is, the global positioning system (GPS) and geographical information system (GIS).

Recently, smart homes have garnered considerable interest [3] because they help enable smart cities and complete the smart grid, as shown in Figure 1(a); smart farming and e-healthcare, as shown in Figure 1(b); water quality; and other smart systems [4–8]. These advancements ensure an efficient lifestyle and safety for people through 24/7 monitoring. Most smart homes are equipped with passive infrared sensors to detect motion and report binary data for visualization. The IoT provides a wide range of personalized treatment possibilities when utilized in conjunction with traditional healthcare. Furthermore, multimedia IoT (MIoT) monitoring system for smart homes have been developed for several possible applications and their integration has been suggested with the existing smart house testbeds for further research [9]. MIoT provides enhanced device, system, and service connectivity in the context of smart homes beyond machine-to-machine communication. The purpose of creating a smart home ontology, as explained in previous studies [4,7], is to provide semantics for the data shared by the systems and devices and establish semantic interoperability in smart homes. From a technological perspective, cloud computing and IoT/MIoT are combined to provide pervasive sensing services and robust processing of the collected data that go beyond what individual “things” can do, leading to advancements in both domains. Another approach employs several machine learning techniques as well as data management and analysis approaches. However, the major challenges in differentiating cases located in the same space include allowing the interoperability of different technologies and guaranteeing data security and privacy [4,10,11].

Corporal sensors face the same challenges as general wireless sensor networks (WSNs). Indeed, the major concern is how to extend the lifetime of wireless sensor nodes and provide security with an appropriate Quality of Service (QoS) for each type of event. Energy efficiency is another principal criterion [12] that should be considered in the design of remote monitoring systems [13]. Many of these criteria have been addressed by various WSN Media access control (MAC) protocols that focus only on energy conservation without a major concentration on urgent data transmissions, remote monitoring, security, and privacy [14].

Current monitoring systems suffer from several problems such as alarm settings, convoluted wires, complex artifacts, excessive information, adaptability of many hardware devices made by different manufacturers, in-person monitoring, and challenges resulting from human performance. To overcome these problems, we aimed to establish a smart and secure home based on IoT for real-time remote

monitoring of smart home architectures. The initial research has been reported in our previous study [15]. Our primary objectives of the present study were as follows:

- To develop an intelligent end-to-end secure IoT-based health monitoring system for smart homes. This system uses current monitoring mechanisms and employs the most recent ambient sensing devices and wireless technologies in addition to cloud services. It also includes security and privacy mechanisms that allow reliable end-to-end transfer of information through secure channels.
- To define the entire architecture and set up suitable processes and interconnectivity to help automate the complete operation of the system (monitor and secure) with a preventive solution for critical situations. This contribution includes both software development and communication establishment in addition to a heterogeneous hardware environment.
- To develop techniques for enhancing the performance of remote monitoring systems. This contribution involves alleviating the complexity of the mechanisms to make them adequate for hardware platform resources. Therefore, the process involves enhancing communication processes, cryptographic algorithms, data collection, aggregation, analysis, and visualization.
- To implement a real testbed to perform the experiments and generate results for conducting comparisons in terms of power consumption. This phase allows for building and running an entire heterogeneous platform, including several ambient sensors, communication protocols, hypertext transfer protocol secure (HTTPS), and API Key mechanisms for security, analysis, and remote visualization.

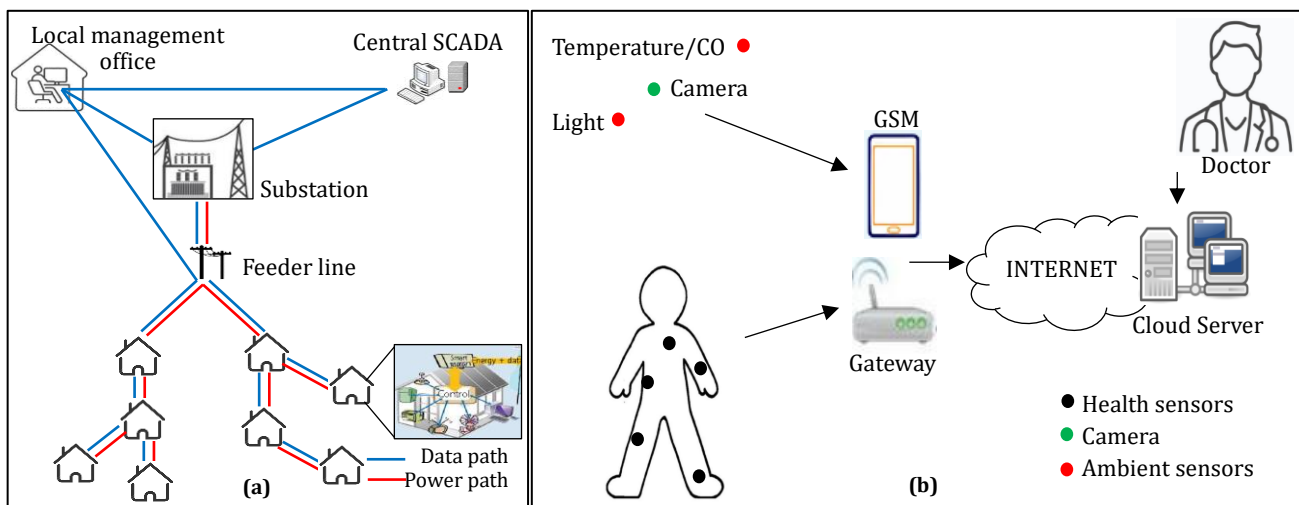


Figure 1. (a) Smart Grid communication network architecture, and (b) an ehealth care monitoring system.

Based on these contributions, the research conducted in this study can be employed in various monitoring applications in smart cities, which include event sensing, data communication, information security, analysis, and visualization. The remainder of this paper is organized as follows: In Section 2, we introduce the background, including the major directions of this study in terms of IoT-based security, confidentiality, and remote monitoring approaches. In Section 3, we performed a literature review and conducted a comparative study of related works. In Section 4, we describe the methodology of the proposed method, and, in Section 5, we outline the experimental results. Finally, in Section 6, we

present our conclusions and discuss the future scope of this research.

2. Background

Swift progress in electronics and wireless communication technologies has allowed the evolution of IoT, which provides practical solutions for several applications such as smart grids, home security, military, target tracking, and pollution level monitoring [4]. In particular, researchers in the field of computing, networking, and software are working together to implement this promising technology, which provides high-quality services to people even in their homes, primarily for older individuals, children, and those with persistent maladies. WSNs are located on the human body or implemented on the premises to form clusters. The WSN allows for the collection of sensed data, aggregating them in a coordinator and sending them to a central database to be used by the administration [7]. At the IoT level, three challenges must be considered: sensor limitations, network performance and security, and confidentiality support.

With rapid advancements in information and communication technology, the advent of multimedia and innovation in systems using IoT and WSNs has become increasingly significant. In a previous study [9], the researchers presented a comprehensive review of distinct frameworks, real-time applications, data compression, sensory data, challenges, limitations, and open research issues related to the MIIoT in industrial production systems. The integration of multimedia and IoT leads to exciting possibilities across various domains, including healthcare, smart cities, and precision agriculture. Khan et al. [9] explored different frameworks that combine multimedia and IoT. These frameworks enable seamless communication, data exchange, and intelligent decision-making. Real-time multimedia applications benefit from IoT systems, and examples of this include video streaming, serious games, rehabilitative exercises, and e-healthcare. The integration of multimedia and cloud services has enhanced these applications. Efficient data compression techniques are crucial for MIIoT systems; however, balancing data quality and resource constraints is challenging. IoT sensors capture sensory data from the environment, and integrating these data with multimedia content allows for context-aware applications.

Sensor limitations: Corporal sensors share the same challenges as general WSNs. Energy efficiency is the principal requirement for designing low-level capacity node protocols [12]. Indeed, the challenge lies in extending the lifetime of the body sensor nodes. Sensor capabilities in terms of processing rate and buffer size also constitute an additional major sensor limitation.

Network performance: QoS metrics, such as the coverage rate, reliability, end-to-end delay, throughput, and error rates for delivery or sensing, are network performance measures that should satisfy an acceptable level [13]. A large proportion of this problem has been addressed using various WSN protocols. Most studies have focused only on energy conservation, and only a few studies have concentrated on urgent data transmission [14]. The timely and reliable delivery of life-crucial medical data must be assured; that is, a patient who requires the most instantaneous service should be given major priority, and the patient's vital signs should be transferred with higher reliability and shorter delay. Furthermore, IoT-based e-health systems assist diverse services with various characteristics, such as different quantities of energy, different-sized data, and various transmission rates. Therefore, it has become important to arrange a service-differentiated scheme to allow multilevel QoS support, as most approaches process all packets in an identical manner [4].

Challenges in security and confidence: Owing to the multiplicity of devices and the dynamic

nature of networks, healthcare IoT systems face many security and privacy challenges because they are frequently vulnerable to threats and attacks that can compromise the system. To address these challenges, security and privacy issues must first be identified. Second, a physically unclonable functional authentication approach is needed, considering both sensor limitations and network capacity. Prominent companies are incorporating artificial intelligence into printed circuit board (PCB) designs [16], producing embedded programs that are tailored to the specifications of the circuit. In electronics, PCBs have revolutionized the creation of circuits. Although many designs exist worldwide, PCB designs are becoming increasingly valued owing to their automation. Several companies have concentrated on PCB designs that improve the user experience by integrating artificial intelligence into the system.

3. Related works

Technological development in the areas of telecommunications and electronics has led to rapid advances in IoT, which is a promising technology for several applications such as healthcare. Because wireless body area networks (WBANs) are used in healthcare to monitor the vital signs of patients, any delayed or missing data can threaten the patient's life. Urgent data are immediately transmitted to supply a QoS guarantee to the WBAN. Recent advances in smartphones and remote healthcare monitoring based on IoT provide multidimensional and intelligent services. With continuous monitoring and improved interventions, the development of IoT-based wearable sensor devices has prevented millions of deaths worldwide. The SIM7600E GSM and GNSS HAT Module enable on-demand positioning of the patient as well as remote monitoring of body temperature (DS18B20), oxygen saturation (SPO₂; MAX30100), and heart rate [17]. Information from sensors is sent across the network and stored in the cloud. The proposed method in the present study uses the most recent IoT microcontroller and device versions, which have a substantial impact on the overall accuracy and speed of the system.

Information security encounters serious challenges while sending data. In addition, predicting diseases using gathered data is a complex task. To address these problems, Alandjani [18] suggested a new deep learning method called convolutional neural network-based disease prediction. The Rives-Shamir-Adleman cryptographic technique can be used to ensure the security of data stored in the cloud. We compared the performance of the method proposed in the present with that of methods proposed in previous studies. A comparative analysis demonstrated that the proposed approach provides improved prediction accuracy and better security.

QoS is a key factor of the WBAN. Assigning various priority levels for sensed data allows traffic differentiation and provides a suitable QoS level based on event, anomaly, and emergency cases. QoS support considers both the IoT device capability and network response/performance. The major metrics used to evaluate the QoS support are latency, throughput, link quality, and loss rate.

Alharbi et al. [19] introduced a novel approach called label augmentation. It assigns a joint label to each generated image by considering both the label and data augmentation. Furthermore, they addressed the challenge posed by augmented samples, which increases the intraclass diversity in the training set. To enhance the classification accuracy, the method employs the Kullback–Leibler divergence to constrain the output distribution of two training samples with the same scene category. Extensive experiments on widely used datasets (UCM, AID and NWPU) demonstrated that the method proposed in the present study outperforms other state-of-the-art approaches in terms of classification accuracy.

The same idea was applied by Puustjärvi and Puustjärvi [7], wherein the goal of the scheme was

to classify and prioritize sensed data to improve real-time delivery using differentiated services, traffic scheduling, and multipath transmission. Sensor nodes sense the data and begin an anomaly detection process based on a predefined threshold. The coordinator node has a criticality table used to classify patients based on data from the source nodes. The sensed data are sent to the packet classification mechanism through the path selector using the path state information. The received signal strength indicator between the nodes, residual energy, and path delay is used to estimate the path states by indicating link quality information.

Other protocols for real-time applications exploit the decoding and forwarding mechanisms for cooperation [20]. Thus, decoding each packet at the relay nodes causes these networks to experience security and privacy attacks. Almost all the current network coding (NC)-based studies on WBSNs use linear combinations or the Exclusively-OR (XoR) process for coding. Almost all current NC-based error recovery methods are not QoS-aware for medical applications. However, NC-based error recovery techniques have a substantial capacity to be appropriate for network channel conditions. Based on this, Razzaque et al. [20] employed Marinkovic and Popovici's XoR coding-based method for WBSNs. In this study, we enhanced the NC-based approach to make it suitable for network channel conditions and user QoS requirements. Two QoS mechanisms are involved in this approach: adaptive service differentiation and adaptive error recovery.

Suganthi et al. [21] employed a mobile healthcare technology to enhance the quality of patient monitoring. This objective was achieved by employing sensor nodes affixed to the patient's body to monitor physiological data and vital signs. To protect patient privacy and security while balancing security and performance, they proposed an end-to-end mutual authentication system. The proposed authentication method also uses a smartphone or personal digital assistant (PDA) as a gateway node, allowing for ongoing patient monitoring even outside the clinical setting. The approach also includes emergency protocol measures necessary to maintain the standard of patient care in dire circumstances.

Saleem et al. [15] addressed the need for secure monitoring in smart environments. By leveraging the cellular IoT technology, the proposed system ensures robust communication and data exchange and provides features that include real-time monitoring, security, and scalability. The system integrates various sensors (such as temperature, humidity, and motion sensors) to monitor the environment. Cellular connectivity allows seamless communication even in remote or challenging locations. This system can be applied to diverse scenarios, including home automation, industrial settings, and agriculture. Security measures include the transmission and authentication of encrypted data. The researchers provided a comprehensive overview of the network's architecture, implementation, and performance evaluation.

Manikandan et al. [22] introduced a novel system designed to promptly detect fires while pinpointing the affected area. Raspberry Pi 3 was employed as a control; this compact yet powerful device integrates several sensors and cameras. A validation mechanism was employed to prevent false alerts, and the system validated the fire suspicions. Upon detecting a fire, the system not only captures an image of the afflicted area but also sends an automated message. If the administration confirms a fire, an alert is promptly raised and a message is sent to the local fire department. By leveraging IoT and Raspberry Pi, this approach enhances safety, minimizes false alarms, and facilitates rapid responses in critical situations.

Meddeb et al. [23] introduced a Raspberry Pi 3 Model B-based robot that can be seamlessly integrated into any household. Furthermore, equipped with a webcam, the robot monitors the area and sends notifications upon detecting trespassing or obstruction. The camera also features a face recognition

algorithm, enabling it to identify the person responsible for triggering the motion. If an individual is authorized, an onboard voice assistant engages in conversation. Notifications are dispatched only to authorized personnel, accompanied by pictures of the trespasser and live streaming of the webcam feed. Pi's live streaming capability allows for the remote analysis of the camera feed via the Internet. Such a system provides users with enhanced security and peace of mind when they are away from home or when they leave vulnerable family members alone. In the case of authorization, the system does not notify the owner, which is a vulnerability that may cause harmful situations.

Upadrista et al. [10] addressed healthcare data security and privacy challenges, where health information is critical and requires fail-safe methods against unauthorized access, leakage, and manipulation. They proposed a remote health monitoring (RHM) solution using blockchain technology that allows immutability, decentralization, and transparency to address data security and privacy challenges. In summary, they provided a comprehensive review of related works on blockchain for RHM, focusing on privacy, security, reliability, and latency.

Shreya et al. [24] made use of the Internet of Medical Things and cloud computing to replace the old medical system, making healthcare easier, customized, and more efficient. They employed medical sensors to capture the health information transmitted to a remote server via a cloud-based solution for processing or diagnostic purposes. They outlined different types of threats that affect the integrity, confidentiality, and privacy of health data. Consequently, the data are shared between stakeholders in an encrypted format, resulting in a reduction in the consumed energy (79.19%), communication costs (15.62%), and execution time (80.03%) compared with the existing methods.

Sangeethalakshmi et al. [25] proposed an IoT-based real-time health monitoring system to overcome the limitations of traditional medical treatments. A mobile application and GSM were combined in the proposed system to provide hospitalized or at-home patients with a dependable patient management system. They used sensors to regularly collect vital parameters (temperature, heartbeat rate, echocardiogram [ECG] readings, blood pressure, and SPO₂), which are sent to the cloud via a WiFi module. The system consists of sensors, a data acquisition unit, a microcontroller (ESP32), and software and provides real-time online health information monitoring. They assigned a threshold to each parameter once a message was sent to the doctor.

Kekre and Gawre [5] provided a solution for monitoring and maintaining solar photovoltaics. As the cost of renewable energy equipment has decreased, the number of solar photovoltaic installations has increased significantly. Many of these installations serve as auxiliary power sources and are in diverse and, at times, inaccessible areas ranging from rooftops to remote desert locations. To address the need for efficient monitoring, they proposed an inexpensive IoT-based embedded solar photovoltaic monitoring system. The system utilizes a general packet radio service (GPRS) module and an affordable microcontroller to collect data from the production end and transmit them over the Internet. This real-time information can be accessed globally, aiding maintenance, fault detection, and data recording at regular intervals.

Xie et al. [26] discussed the challenges related to scalability, interoperability, and privacy. They also highlighted the potential benefits of integrating edge computing into systems. Consequently, they introduced a conceptual architecture that combined IoT principles with smart grid functionalities. This framework aims to predict and prevent potential threats to the grid. The system employs an electronic fence mechanism to detect unauthorized access or abnormal behavior within the grid. Real-time video monitoring enhances situational awareness and helps identify security breaches. Ultrawideband technology contributes to precise location tracking and rapid response during

emergencies. By leveraging IoT, the system anticipates and mitigates malicious activities and safeguards critical infrastructure.

Albataineh et al. [27] proposed a hybrid solution that leveraged both cloud and edge computing for data processing. Edge computing processes smart grid information at the edge of the IoT network, where microgrids are located. The authors introduced a machine learning engine that employs decision trees to facilitate communication among the edge layer, failure mechanisms, and cloud layer. By combining edge and cloud computing, this novel smart home control system aims to enhance efficiency, reliability, and overall performance. The authors addressed the challenges of existing power grids facing outages, unpredictable power disturbances, inflexible energy rates, and customer fraud. These issues have contributed to the rising fossil fuel demand and service costs. This approach minimizes latency and storage compared with centralizing all processing in the cloud.

Sreenivasu et al. [8] proposed an innovative cloud-based electric vehicle (EV) temperature monitoring system by leveraging the power of artificial intelligence (AI) and IoT. The sensors were integrated into the system interfaces by embedding them in the EV battery. These sensors continuously collect data on critical parameters, such as voltage, current, and temperature. The collected data are transmitted to the cloud via the IoT network. This real-time flow of information enables remote monitoring and analysis. Users can access the performance data of the battery through a mobile application connected to the cloud. This interface provides insights into the state of charge and overall health of the battery. The cloud-based system employs AI algorithms to analyze historical data, predict potential issues, and recommend preventive measures. This proactive approach enhances battery management and reduces the risk of unexpected failures.

Nasraoui et al. [28] investigated authenticated key agreements as a crucial part of providing end-to-end security solutions for WSNs using LoRa connectivity. As the applications of WSNs and IoT continue to expand, ensuring security has become paramount. However, owing to the limited resources of the connected nodes, new challenges arise. The researchers provided a primer that covers the general security aspects of WSNs and IoT at different levels. This foundational understanding sets the stage for the proposed lightweight key exchange method. We focused on implementing lightweight key agreement methods based on a standardized Internet Key Exchange protocol. These methods aim to establish secure communication channels while minimizing the computational overhead. The method was experimentally evaluated, and communication costs in terms of power consumption and memory usage were considered in the assessment. Realistic scenarios were used to demonstrate the practical implications of these lightweight approaches.

Sangeethalakshmi et al. [29] investigated the security aspects of WSNs in the context of environmental monitoring systems. They emphasized that any security mechanism within a WSN becomes ineffective if an attacker gains access to the firmware of the sensor device. This study revealed that the bootstrap loader (BSL) password for the MSP430 microcontroller unit (MCU) is insecure. The attacker successfully obtains WSN cryptographic keys by reversing the firmware. In a few days, the BSL password can be cracked. The researchers introduced a two-factor authentication using the capabilities of the MSP430 MCU by adding an additional layer of security. A solution was proposed to enhance the security of the firmware of MSP430 processors. The Secure-BSL application generates random BSL passwords to ensure robustness against brute-force attacks.

Nguyen-Tan et al. [30] proposed an autonomous irrigation solution that combines historical soil moisture data stored in a database with real-time weather forecasts. Two deep-learning models based on the transformer architecture were employed to predict weather conditions and soil moisture levels.

Surprisingly, despite having fewer training variables than the long short-term memory model, the transformer model achieved comparable accuracy (91.41% for weather forecasts and 82.06% for soil moisture forecasts). However, being an IoT system, security concerns in smart agricultural solutions must be addressed. To safeguard against eavesdropping, control command spoofing, and poisoned machine learning models, the researchers have proposed end-to-end encryption and authentication approaches. This solution employs AES 256-bit encryption, HMAC, and the CRYSTALS-Kyber key exchange technique, even in the quantum age. The evaluation results demonstrated that the proposed security measures can be effectively deployed in IoT devices such as Arduino, STM32, and Raspberry Pi 4. We contribute to sustainable and efficient agricultural practices by combining intelligent irrigation and robust security.

4. Materials and methods

An end device is a wireless device responsible for detecting events in its environment and transmitting them to the sink node. The end device receives the data from the sensors, processes it, and sends it to the sink node when there is a danger. To perform these tasks efficiently, the end device should be equipped with a microcontroller with specific features [31]. In this context, a low-power and fast microcontroller is required to execute complex computational instructions without ignoring the power constraints, as shown in Figure 2.

Many types of microcontrollers with different features and capabilities have been proposed. In the proposed platform, we used ATmega328P, which was integrated into the Arduino Uno board. For the data communication protocol, the ZigBee XBee module was chosen, which includes the XBee-S2C gateway, as shown in Figure 2.

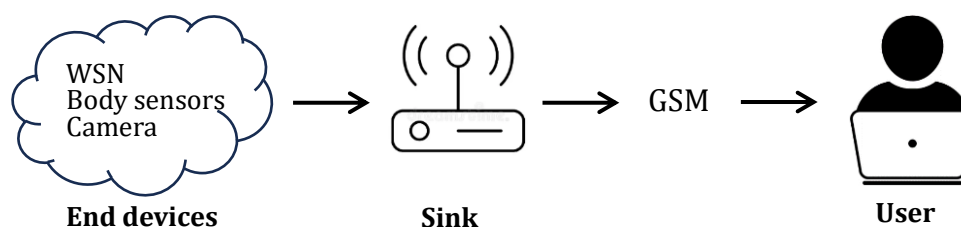


Figure 2. Schematic of the platform's processes, including event sensing, data communication, and remote monitoring.

4.1. Setting the XBee with Arduino

The XBee USB Adapter Board is sold as a partially assembled kit and provides a cost-effective solution to interfacing a PC or microcontroller to any XBee or XBee Pro module. A PC connection can be used to configure the XBee module using Digi's X-CTU software. This module works with XBee Series 1, 2, and Pro modules.

Using this adapter board, an easy interface can be provided to the XBee or XBee Pro modules. The adapter board also provides a means of connecting pluggable wires, solder connections, and mounting holes, as shown in Figure 3.

- Advantages:
 - Provides easy pluggable wire or solder connections
 - Provides an easy interface for configuring XBee modules using Digi's X-CTU software.
 - No shield is required (decreases the cost).
- Disadvantages:
 - To set up GPIOs, a serial software library must be downloaded.
 - Requires coding expertise.

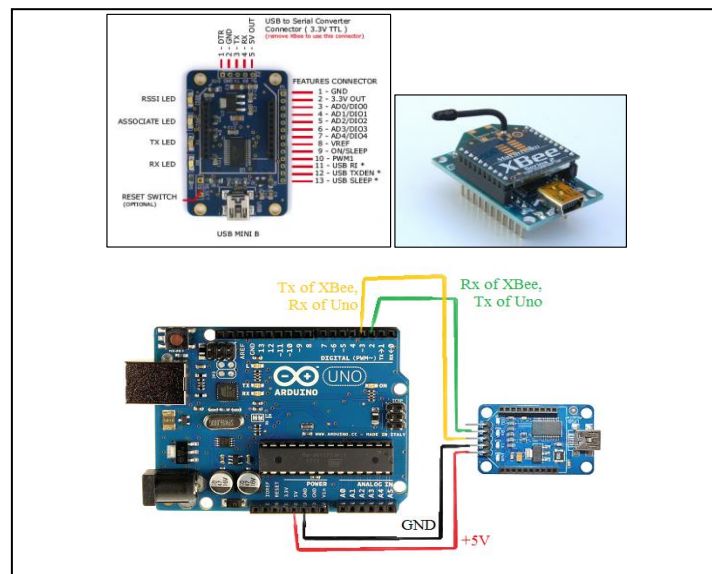


Figure 3. Interfaces and interconnectivity of Arduino Uno and the XBee module using the XBee adapter.

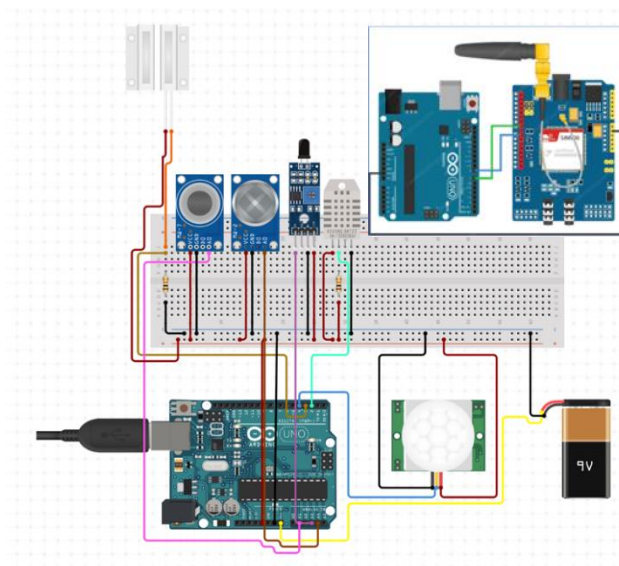


Figure 4. Schematic of the experimental testbed developed for event sensing and communication.

From the two previously suggested approaches (XBee shield and XBee adapter), we used the

XBee adapter to configure (XBee s2c) and establish communication between the two XBees. The XBee shield used to configure the (XBee s2c) on top of the Arduino is shown in Figure 4, and the real experimental testbed is shown in Figure 5.

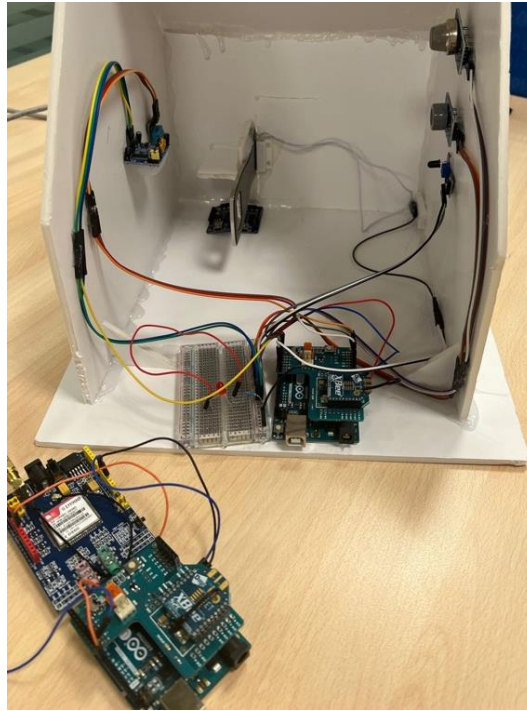


Figure 5. Real testbed implementation for practical experiments.

4.2. Setting the Arduino IDE

Arduino is an open-source electronic platform based on easy-to-use hardware and software. Arduino boards can read inputs—light on a sensor and a finger on a button—and convert them into an output. To program it, the Arduino integrated development environment (IDE) or Arduino software must be used, which contains a text editor for writing a code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload and communicate with programs.

Initially, we visited (<https://www.arduino.cc/en/software>) and downloaded the most recent version of the Arduino IDE. Subsequently, the driver was downloaded. However, if the driver is not downloaded for some reason, it must be manually downloaded to post the code onto the platform.

Libraries for XBee, Software Serial, DHT, and MQ2 sensors were downloaded. The IDE's Sketch > Include Library > Manage Libraries menu will allow the individuals to download the software.

4.3. Setting the XBee software

The XBee module was configured using the XCTU software. It was developed by DIGI and is available at (<https://www.digi.com/products/embedded-systems/digi-xbee/digi-xbee-tools/xctu>). The first step consisted of starting the XCTU software and connecting the XBee module using a USB adapter, as shown in Figure 6. Once XBee is connected to the PC, the command (Ctrl + Shift + D)

allows the module to be located at the XCTU graphical interface, as shown in Figure 7. Once detected, the XBee module is set up to enable effective communication. The XBee module is configured at least once. The IDs of the source and destination should be reversed. As discussed earlier, the XBee module was disconnected from the USB adapter and connected to the XBee shield on top of the Arduino Uno. Next, the two XBee modules were configured, as shown in Figure 8.

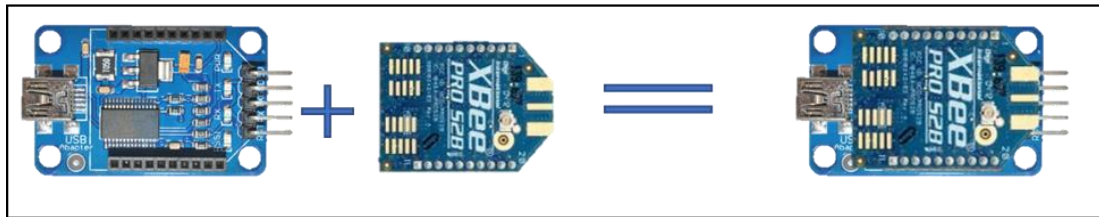


Figure 6. XBee configuration using XCTU and attached with the USB adapter.

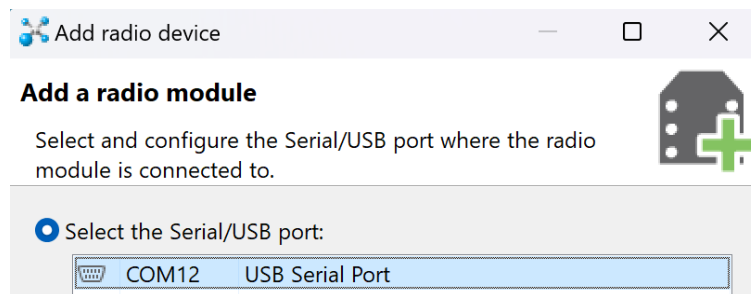


Figure 7. Discovering the XBee module using XCTU software through the USB adapter.

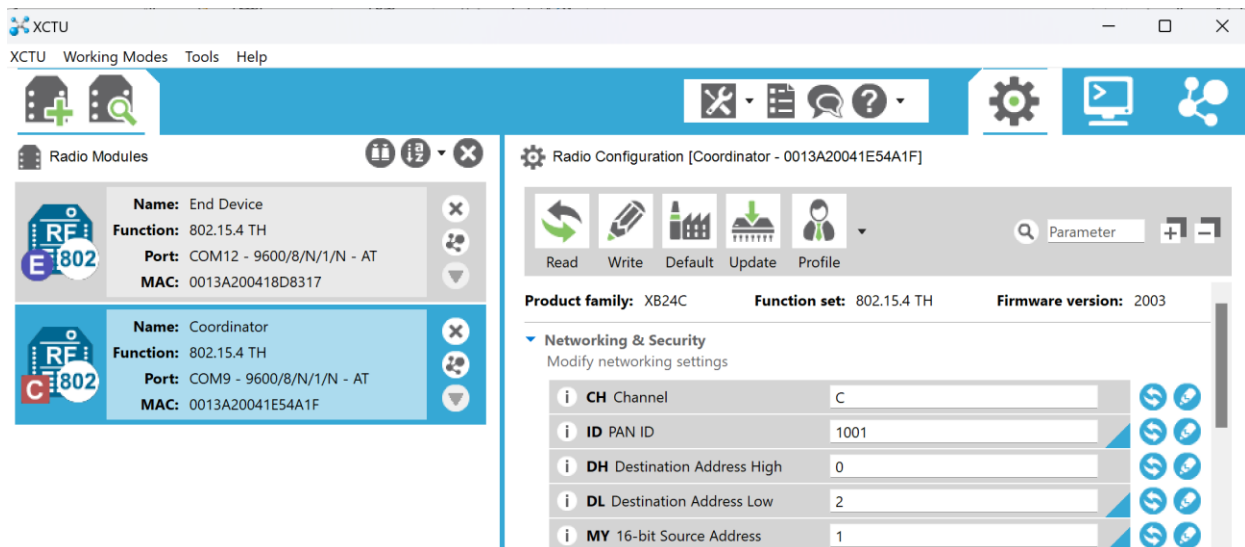


Figure 8. XCTU graphical interface for XBee connectivity and configuration.

5. Results and performance evaluation

Our primary objective of this study was to create a sensor-based monitoring system for smart homes. We constructed a WSN using the ZigBee protocol. Additionally, we used the Global System for Mobile Communication (GSM) to enable Arduino to send an SMS message to the phone. The user should be able to receive specific data from the WSN through the GSM when it detects an incident. The sensors controlled by the Arduino first read the data. The processed data that detect an event send a message through the XBee module via the ZigBee protocol. The sink node then receives the data and automatically sends them to the user through the GSM.

5.1. Energy consumption

Power consumption is a crucial consideration when working with IoT-based hardware platforms, which include various types of sensing devices. Various board components, including processors, sensors, and communication modules, consume power. Identifying the factors that affect the power consumption of IoT devices, as well as the Arduino Uno platform, can help make adequate decisions regarding hardware and software configurations. By utilizing low-power sensors and efficient coding practices, energy usage can be minimized, potentially extending the battery life. To investigate strategies for reducing power consumption and demonstrate how to measure the power usage of Arduino Uno, it is necessary to explore various components of Arduino Uno and the sensors that consume power.

The test results of power usage were measured using two variables: 1) variation in the sensing duty cycle (i.e., the time interval between two consecutive sensors) and 2) variation in the number of operating sensors. Arduino Uno was used to collect information regarding the amount of electric energy consumed according to the configuration of the hardware platform for several experiments. Specifically, the ACS612 sensor was powered by a 9-V battery as a unique source of energy. Using N sensors, first, the power usage P_N was measured based on the current and voltage levels using Eq (1). Notably, the voltage was considered constant during the experiment. The energy dissipated by these sensors considers the sensing duty cycle and is calculated using Eq (2).

$$P = V \times I \quad (1)$$

$$E = P \times t \quad (2)$$

5.1.1. Experiment 1 (one sensor; sensing time cycle: 2 s)

In this experiment, only one sensor was involved in event sensing, with a sensing time cycle of 2 s. Figure 9 shows a small reduction in the remaining energy in the battery because only one sensing device was in operation. The red curve represents the average variation in the remaining energy. Table 1 lists certain consecutive values of power, voltage, and consumed energy collected from the hardware platform within the experimental period. In this experiment, measurements were performed every 2 s.

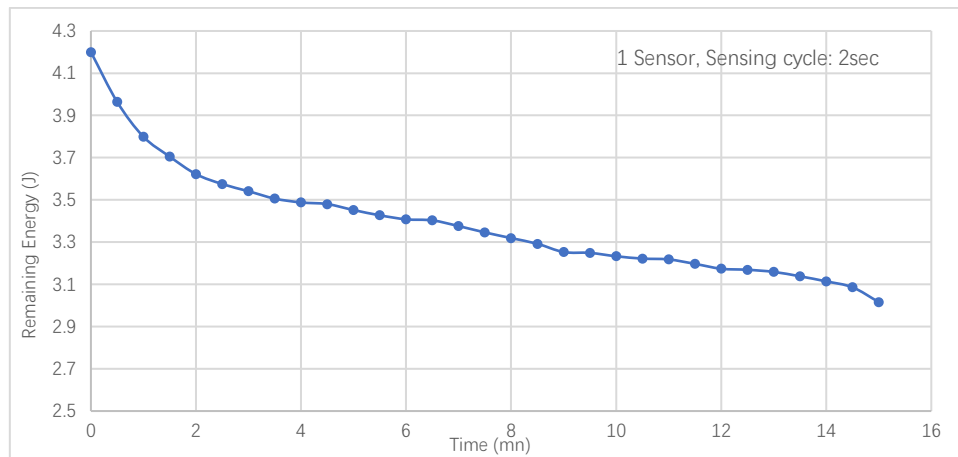


Figure 9. Battery's remaining energy for one operating sensor with a sensing cycle of 2 s.

Table 1. Collected values of power, voltage level, and consumed energy in experiment 1.

Sec	Power	V	CH1	Energy	Time
2	1.44	9	0.16	2.88	22:00:12.16
2	1,44	9	0.16	2.88	22:00:09.69
2	1.53	9	0.17	3.06	22:00:07.22
2	1.53	9	0.17	3.06	22:00:04.75
2	1.53	9	0.17	3.06	22:00:02.28
2	1.53	9	0.17	3.06	21:59:59.81
2	1.53	9	0.17	3.06	21:59:57.33
2	1.53	9	0.17	3.06	21:59:54.86
2	1.44	9	0.16	2.88	21:59:52.40
2	1.53	9	0.17	3.06	21:59:49.92
2	1.53	9	0.17	3.06	21:59:47.45
2	1.53	9	0.17	3.06	21:59:44.99
2	1.44	9	0.16	2.88	21:59:42.52

5.1.2. Experiment 2 (three sensors; sensing time cycle: 2 s)

In this experiment, the number of sensors was increased to three, and the sensing time cycle was maintained at 2 s. The range of energy dissipated during this experiment was estimated to be twice that in experiment 1. Figure 10 shows the energy variations in experiment 2 and are presented in Table 2. For experiment 2, the same process as described in experiment 1 was followed.

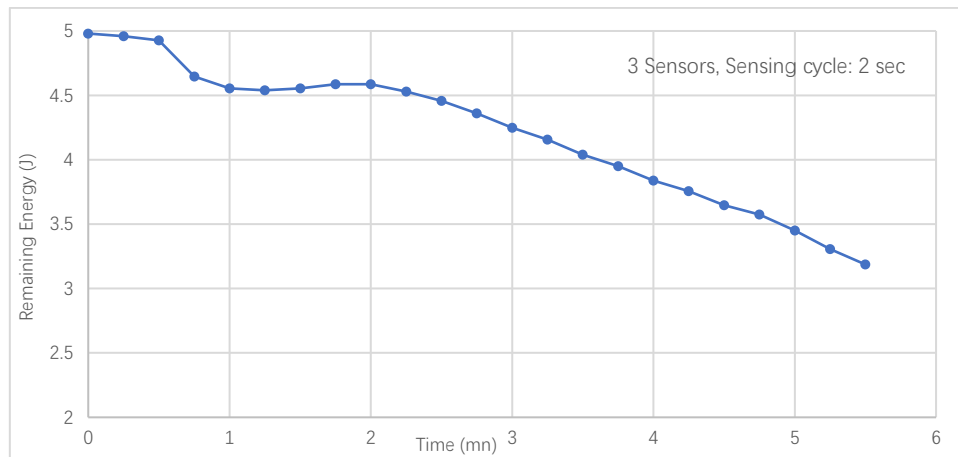


Figure 10. Battery's remaining energy for three operating sensors with a sensing cycle of 2 s.

Table 2. Collected values of power, voltage level, and consumed energy in experiment 2.

Sec	V	CHI	Power	Energy	Time
2	9	0.17	1.53	3.06	19:05:33.62
2	9	0.17	1.53	3.06	19:05:31.16
2	9	0.17	1.53	3.06	19:05:28.70
2	9	0.18	1.62	3.24	19:05:26.23
2	9	0.18	1.62	3.24	19:05:23.76
2	9	0.17	1.53	3.06	19:05:21.28
2	9	0.18	1.62	3.24	19:05:18.81
2	9	0.18	1.62	3.24	19:05:16.33
2	9	0.18	1.62	3.24	19:05:13.86
2	9	0.18	1.62	3.24	19:05:11.39
2	9	0.18	1.62	3.24	19:05:08.93
2	9	0.18	1.62	3.24	19:05:06.46
2	9	0.18	1.62	3.24	19:05:03.99

5.1.3. Experiments 3, 4, and 5 (six sensors; variable sensing time cycle)

Three other experiments were performed to obtain more descriptive results regarding energy consumption. In experiment 3, six sensors were operated with a sensing time cycle of 2 s, as shown in Figure 11; the exact values are listed in Table 3. Experiment 4 included six sensors configured for sensing every 30 s, as shown in Figure 12; the exact values are listed in Table 4. Finally, in experiment 5, six sensors were operated with a sensing time cycle of 60 s, as shown in Figure 13; the exact values are listed in Table 5. In these experiments, the sensing time cycle was increased to optimize energy consumption without ignoring the events occurring in the smart home. At this level, a major compromise was observed between energy optimization and event detection. In other words, it is useful to increase the time cycle without exceeding a certain threshold to avoid the non-detection of events or delayed detection.

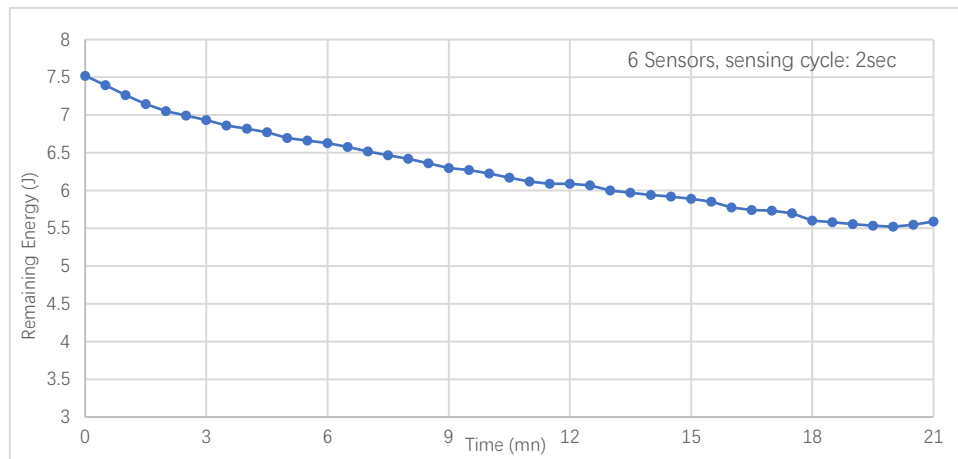


Figure 11. Battery's remaining energy for six operating sensors with a sensing cycle of 2 s.

Table 3. Collected values of power, voltage level, and consumed energy in experiment 3.

Sec	Power	V	CH1	Energy	Time
2	1.44	9	0.16	2.88	22:00:12.16
2	1.44	9	0.16	2.88	22:00:09.69
2	1.53	9	0.17	3.06	22:00:07.22
2	1.53	9	0.17	3.06	22:00:04.75
2	1.53	9	0.17	3.06	22:00:02.28
2	1.53	9	0.17	3.06	21:59:59.81
2	1.53	9	0.17	3.06	21:59:57.33
2	1.53	9	0.17	3.06	21:59:54.86
2	1.44	9	0.16	2.88	21:59:52.40
2	1.53	9	0.17	3.06	21:59:49.92
2	1.53	9	0.17	3.06	21:59:47.45
2	1.53	9	0.17	3.06	21:59:44.99
2	1.44	9	0.16	2.88	21:59:42.52

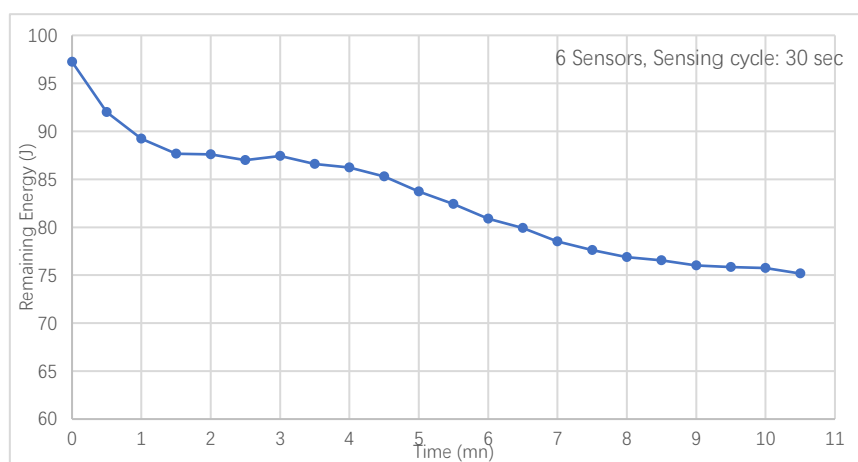
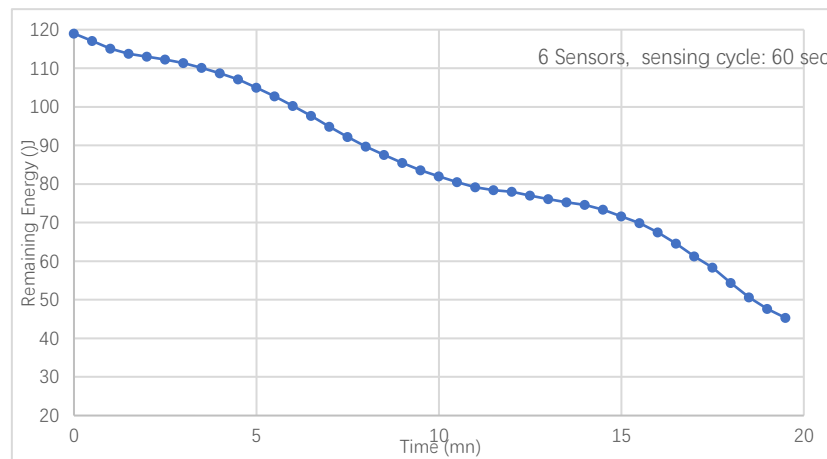


Figure 12. Battery's remaining energy for six operating sensors with a sensing cycle of 30 s.

Table 4. Collected values of power, voltage level, and consumed energy in experiment 4.

Sec	Power	V	CH1	Energy	Time
30	2.52	9	0.28	75.6	20:09:13.73
30	2.52	9	0.28	75.6	20:08:43.25
30	2.52	9	0.28	75.6	20:08:12.78
30	2.52	9	0.28	75.6	20:07:42.31
30	2.61	9	0.29	78.3	20:07:11.83
30	2.52	9	0.28	75.6	20:06:41.35
30	2.61	9	0.29	78.3	20:06:10.86
30	2.61	9	0.29	78.3	20:05:40.40
30	2.61	9	0.29	78.3	20:05:09.91
30	2.7	9	0.3	81	20:04:39.45
30	2.61	9	0.29	78.3	20:04:08.97
30	2.79	9	0.31	83.7	20:03:38.50
30	2.88	9	0.32	86.4	20:03:08.02

**Figure 13.** Battery's remaining energy for six operating sensors with a sensing cycle of 60 s.**Table 5.** Collected values of power, voltage level, and consumed energy in experiment 5.

Sec	Power	V	CH1	Energy	Time
60	0.72	9	0.08	43.2	22:30:36.30
60	0.81	9	0.09	48.6	22:29:35.83
60	0.99	9	0.11	59.4	22:28:35.36
60	0.99	9	0.11	59.4	22:27:34.88
60	1.08	9	0.12	64.8	22:26:34.41
60	1.26	9	0.14	75.6	22:24:33.47
60	1.26	9	0.14	75.6	22:23:32.98
60	1.35	9	0.15	81	22:22:32.52
60	1.35	9	0.15	81	22:21:32.05
60	1.35	9	0.15	81	22:20:31.57
60	1.44	9	0.16	86.4	22:19:31.10

5.1.4. Power consumption

An average power consumption chart was used to visualize the amount of power consumed by the device over a specific period. By plotting the power consumption data in a chart, it becomes easier to identify patterns and trends in energy usage that can influence the design and optimization of devices. This can be particularly useful when working with battery-powered devices, where power consumption is a critical factor. Understanding the power consumption of a device over time also helps identify components that use more power than necessary, potentially leading to more efficient and sustainable designs.

Furthermore, measuring and monitoring power consumption can provide valuable insights into the performance of the device and help make informed decisions regarding future design choices. With these considerations, Figure 14 shows that energy consumption is inversely proportional to the sensing time cycle. Because the time cycle is long, the energy consumed is reduced. The reduction in energy did not obey a linear function; experiment 5 involved considerable optimization than experiment 4, with experiment 3 as a reference. Therefore, to develop energy-efficient and sustainable projects, the sensing time cycle can be increased to an optimal point where the consumed energy is minimal and almost all the events are detected.

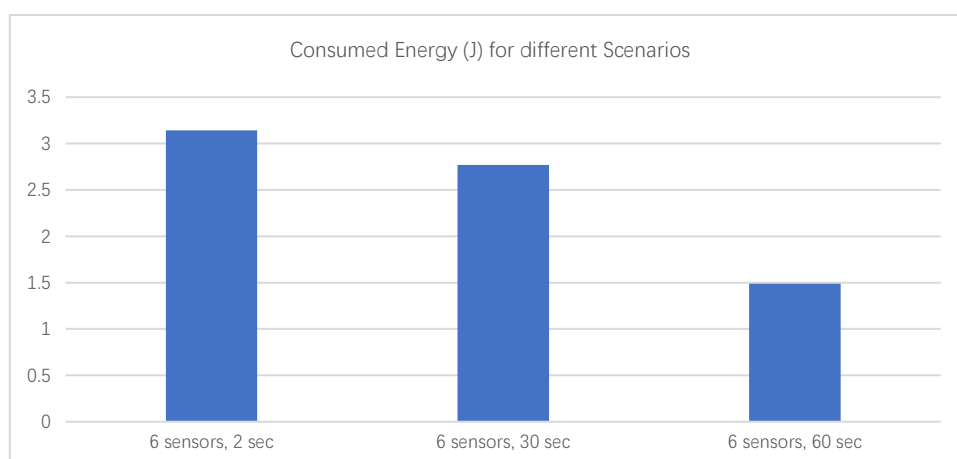
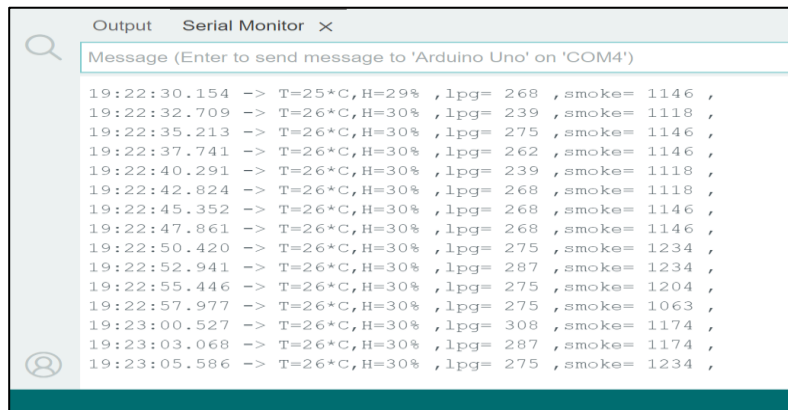


Figure 14. Power consumption in the sensing platform in different scenarios (Nb sensors; sensing cycle).

5.2. Channel setup and data collection

The sensor readings were obtained and whether the code had been successfully uploaded to the platform was determined. In this simple demonstration, we employed only DHT and MQ2 sensors. Although the MQ2 sensor was responsible for determining the liquefied petroleum gas (LPG), smoke, and methane gas, the DHT sensor was used to determine the temperature and humidity. The humidity unit was expressed as a percentage, with 100% denoting the highest possible humidity. The temperature was displayed in Celsius, as shown in Figure 15. The DHT and MQ2 readings were viewed on a serial monitor in Arduino IDE; these readings include the room temperature (T), humidity (H), LPG, and smoke. The sensor readings were sent to the sink node using the XBee module.



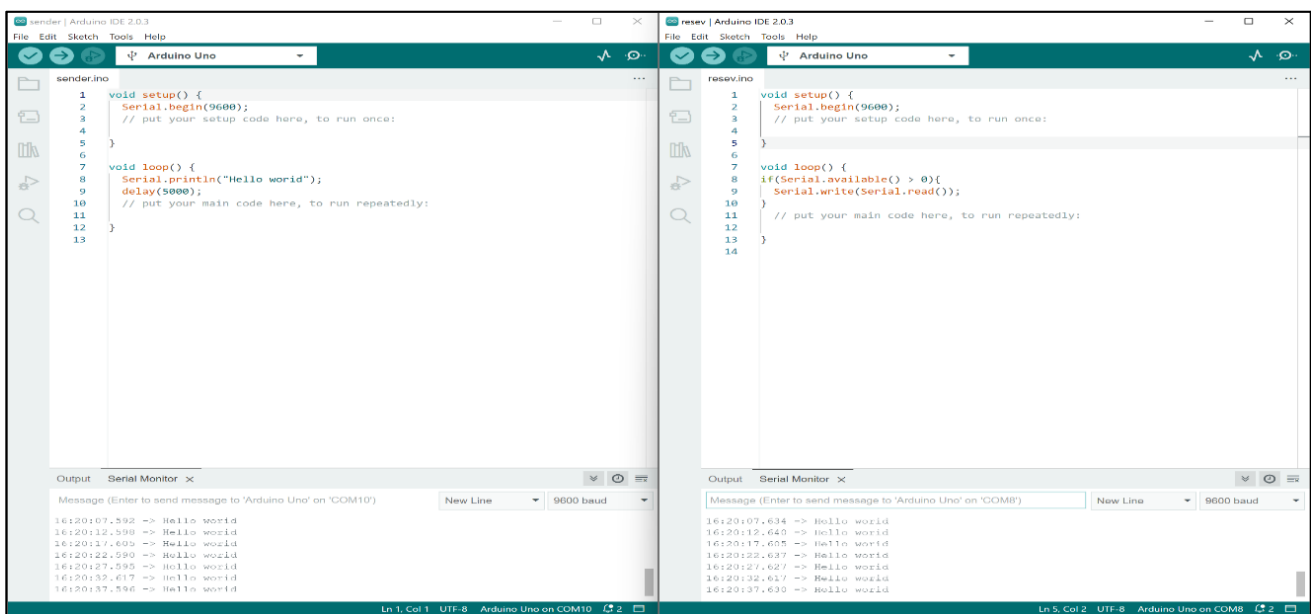
```

Output Serial Monitor X
Message (Enter to send message to 'Arduino Uno' on 'COM4')
19:22:30.154 -> T=25°C,H=29% ,lpg= 268 ,smoke= 1146 ,
19:22:32.709 -> T=26°C,H=30% ,lpg= 239 ,smoke= 1118 ,
19:22:35.213 -> T=26°C,H=30% ,lpg= 275 ,smoke= 1146 ,
19:22:37.741 -> T=26°C,H=30% ,lpg= 262 ,smoke= 1146 ,
19:22:40.291 -> T=26°C,H=30% ,lpg= 239 ,smoke= 1118 ,
19:22:42.824 -> T=26°C,H=30% ,lpg= 268 ,smoke= 1118 ,
19:22:45.352 -> T=26°C,H=30% ,lpg= 268 ,smoke= 1146 ,
19:22:47.861 -> T=26°C,H=30% ,lpg= 268 ,smoke= 1146 ,
19:22:50.420 -> T=26°C,H=30% ,lpg= 275 ,smoke= 1234 ,
19:22:52.941 -> T=26°C,H=30% ,lpg= 287 ,smoke= 1234 ,
19:22:55.446 -> T=26°C,H=30% ,lpg= 275 ,smoke= 1204 ,
19:22:57.977 -> T=26°C,H=30% ,lpg= 275 ,smoke= 1063 ,
19:23:00.527 -> T=26°C,H=30% ,lpg= 308 ,smoke= 1174 ,
19:23:03.068 -> T=26°C,H=30% ,lpg= 287 ,smoke= 1174 ,
19:23:05.586 -> T=26°C,H=30% ,lpg= 275 ,smoke= 1234 ,

```

Figure 15. Humidity, temperature, and light intensity readings displayed on the Arduino IDE serial monitor.

The Zigbee protocol is a wireless networking protocol widely used in IoT because of its low power consumption, robustness, and simplicity. With ZigBee, devices can communicate wirelessly, thereby providing a flexible and scalable solution for various IoT applications. In this section, we tested the wireless communication link between two Arduinos using the Zigbee protocol. Specifically, we used the Xbee module to enable wireless connectivity between the devices, as shown in Figure 16.



```

sender.ino
1 void setup() {
2   Serial.begin(9600);
3   // put your setup code here, to run once:
4
5 }
6
7 void loop() {
8   Serial.println("Hello world");
9   delay(5000);
10  // put your main code here, to run repeatedly:
11
12 }
13

Output Serial Monitor X
Message (Enter to send message to 'Arduino Uno' on 'COM10')
16:20:07.392 => Hello world
16:20:12.099 => Hello world
16:20:17.605 => Hello world
16:20:22.090 => Hello world
16:20:27.596 => Hello world
16:20:32.617 => Hello world
16:20:37.596 => Hello world

receiver.ino
1 void setup() {
2   Serial.begin(9600);
3   // put your setup code here, to run once:
4
5 }
6
7 void loop() {
8   if(Serial.available() > 0){
9     Serial.write(Serial.read());
10
11   // put your main code here, to run repeatedly:
12 }
13
14

Output Serial Monitor X
Message (Enter to send message to 'Arduino Uno' on 'COM8')
16:20:07.634 => Hello world
16:20:12.640 => Hello world
16:20:17.605 => Hello world
16:20:22.637 => Hello world
16:20:27.627 => Hello world
16:20:32.617 => Hello world
16:20:37.630 => Hello world

```

Figure 16. Arduino IDE interface for network establishment.

The sensor readings were sent to the IP gateway using the XBee module over a secure channel. Communication with the gateway can be established manually or automatically using a secure HTTP protocol. The manual configuration uses MAC addresses for IP communication because the XBee module does not integrate the IP layer. Communication considers the creation of a frame using XCTU software. To achieve this, the XBee module must be connected to the IP gateway, which is considered as the coordinator in the “network working mode.” Switching to the console mode was then achieved

using (Ctrl + C) in XCTU (Figure 17). Next, a connection was opened and a frame (frame_0) was created. The three major attributes of the manual configuration are: Frame type, destination, and content. “Transmit Request” represents the type of the frame. The IP-gateway MAC address was provided through a 64-bit destination identifier. The message content to be transmitted was indicated by the “RF data” using ASCII or HEX formats. Because the data were received at base 64, a decoding step was required to make the received message readable by humans. The manual configuration was used only for debugging purposes.

5.3. Event visualization

In this study, we proposed a method to provide remote monitoring services using an Internet server that allows the supervision of smart home events from any location in the world and at any time. This solution results in continuously showing sensor readings using an application-based end system. This result allows establishing a comparison with the input readings and a system efficiency analysis. Any sensing metric provides the user with three levels: green as normal with no reaction, yellow as the average level requiring a follow-up, and red as the highest level requiring a serious action.

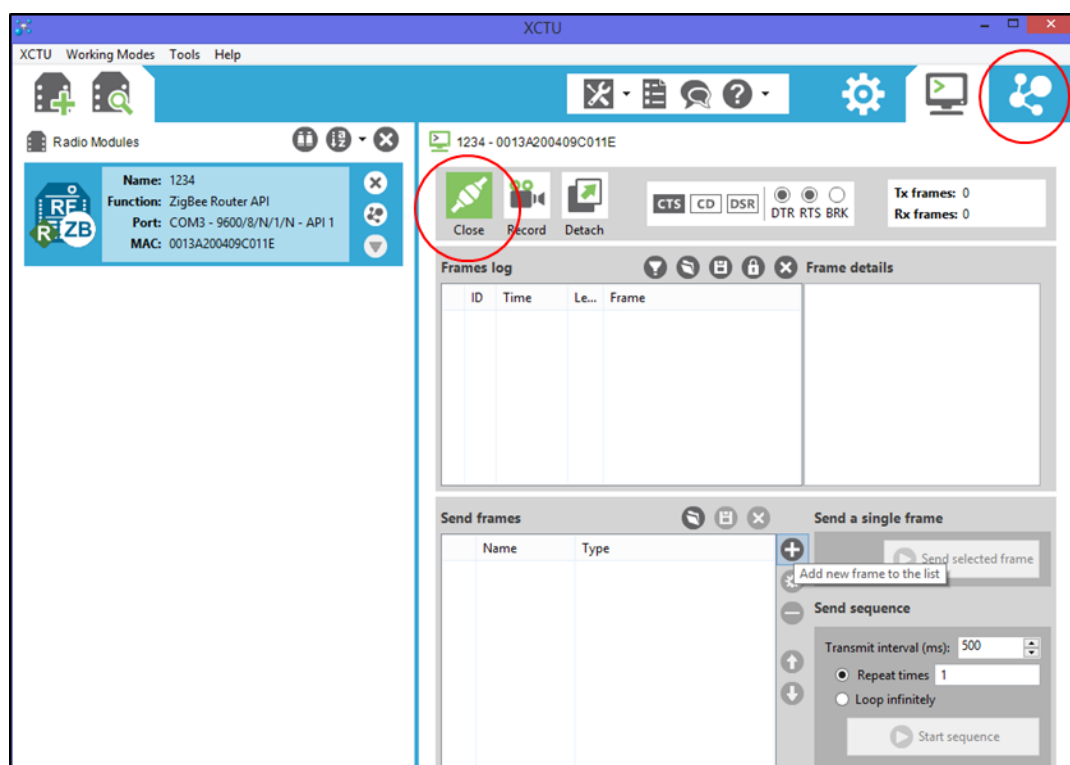


Figure 17. Creating a frame in XCTU.

Servers constitute an important component in accessing and managing IoT events in remote locations. The server-finding process should consider certain characteristics that are suitable for wireless sensor networks. To begin, we surveyed the features of many types of servers such as Amazon Web Services, Google Cloud, and Microsoft Azure. A comparative study of these services, as well as their capabilities, was conducted. Furthermore, a comparison with the local cloud servers of Saudi

Arabia, such as STC Cloud and Sahara Net, was also conducted. Unfortunately, cloud services are unavailable to individuals in Saudi Arabia. Therefore, ThingSpeak was selected for use in the comparative study for several reasons: ThingSpeak is a free cloud platform used to monitor and visualize live data provided in both values and graphs, and it supports the MATLAB programming language, which simplifies graph analysis. In addition to ThingSpeak, the Blynk application was used as a server in the GSM device. Blynk supports both iOS and Android devices.

After signing up with ThingSpeak, a certain number of channels must be created according to the list of metrics collected from the IoT sensors. In its free version, ThingSpeak offers up to eight fields for monitoring sensing events from the smart home. Therefore, the most significant environmental and other vital sign metrics were considered to create the channels (Figure 18).

Blynk allows readings collected from various home sensors to be monitored. It supports GSM, Wi-Fi, USB, and several other connection methods. In addition, Blynk operates on many microcontrollers and microcomputer platforms, such as Arduino, Raspberry Pi, and NodeMCU. In this study, Arduino constituted the first connectivity method, whereas GSM constituted the second way of connectivity. To communicate the readings to the server, the “API Key” and “Channel ID” represent the main parameters that successfully set up the secure connection [28,30]. The Arduino Uno was then configured with these parameters.

Blynk supports three types of pins: Analog, digital, and virtual. In other words, the first type allows direct connection with the analog pin of the platform, whereas the second type enables a similar feature from the digital pin of the platform; however, the virtual type is needed for any variable metric in the code. As a result, the work requires the use of virtual pins, as Blynk provides more than 1000 virtual pins with no limitation on the GPIO pins of the platform.

New Channel

Name SH-RHM

Description Smart-home for remote health monitoring

Field	Metric	Selected	Category
Field 1	Temperature	<input checked="" type="checkbox"/>	Environmental And Multimedia metrics
Field 2	Humidity	<input checked="" type="checkbox"/>	
Field 3	Motion detection	<input checked="" type="checkbox"/>	
Field 4	Blood pressure	<input checked="" type="checkbox"/>	Vital Sign
Field 5	Heart rate	<input checked="" type="checkbox"/>	
Field 6	Blood sugar	<input checked="" type="checkbox"/>	
Field 7	SpO2	<input checked="" type="checkbox"/>	
Field 8		<input type="checkbox"/>	Other metrics

Figure 18. ThingSpeak channel fields displaying the sensor data.

6. Conclusions

In this study, several concepts related to the IoT, remote monitoring, and their applications were studied, and the most recent approaches for remote monitoring were reviewed. We proposed a flexible architecture that includes real electronic components used for event detection, data communication, and information visualization and analysis. The major steps for building an IoT-based monitoring system (i.e., configuring the XBee module, IP gateway, GSM transmitter, and Arduino platform, including various sensors) were outlined. In addition, remote monitoring was incorporated through the inclusion of a server and its application to the entire architecture. Finally, we described the establishment of heterogeneous hardware and software connections to achieve the main objectives of remote monitoring. The 2-level secure channel was enabled based on the HTTPS and built-in API Key features of the Cloud. In conclusion, the obtained results justify the usefulness of the proposed method for various monitoring applications in smart cities, which include event sensing, secure data communication, information security, analysis, and visualization. Based on the collected values, the results demonstrate the efficiency of the proposed system in terms of power consumption and delivery ratio.

In the future, the system is expected to be enhanced with an AI algorithm that detects sensed data anomalies at an advanced stage and manages them automatically in critical situations. In other words, the system will sense danger, apply AI to determine its type, and communicate with the competent government agencies to address the situation. In this manner, the confidentiality of the sensed information is ensured, and security risks can be addressed using sophisticated end-to-end security and privacy mechanisms supported during communication between individual partitions of the system.

Use of AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors extend their appreciation to the Deputyship for Research & Innovation, “Ministry of Education” in Saudi Arabia for funding this research work through the project number (IFKSUDR_D104).

Conflict of interest

The authors declare that there are no conflicts of interest.

References

1. P. Muruganantham, S. Wibowo, S. Grandhi, N. H. Samrat, N. Islam, A systematic literature review on crop yield prediction with deep learning and remote sensing, *Remote Sens.*, **14** (2022), 1990. <https://doi.org/10.3390/rs14091990>

2. N. Casagli, E. Intrieri, V. Tofani, G. Gigli, F. Raspini, Landslide detection, monitoring and prediction with remote-sensing techniques, *Nat. Rev. Earth Environ.*, **4** (2023), 51–64. <https://doi.org/10.1038/s43017-022-00373-x>
3. A. Chakraborty, M. Islam, F. Shahriyar, S. Islam, H. U. Zaman, M. Hasan, Smart home system: a comprehensive review, *J. Electr. Comput. Eng.*, **2023** (2023), 7616683. <https://doi.org/10.1155/2023/7616683>
4. B. Hammi, S. Zeadally, R. Khatoun, J. Nebhen, Survey on smart homes: vulnerabilities, risks, and countermeasures, *Comput. Secur.*, **117**, (2022), 102677. <https://doi.org/10.1016/j.cose.2022.102677>
5. A. Kekre, S. K. Gawre, Solar photovoltaic remote monitoring system using IOT, in *2017 International Conference on Recent Innovations in Signal processing and Embedded Systems (RISE)*, Bhopal, India, (2017), 619–623. <https://doi.org/10.1109/RISE.2017.8378227>
6. M. Manoj, V. D. Kumar, M. Arif, E. R. Bulai, P. Bulai, O. Geman, State of the Art techniques for water quality monitoring systems for fish ponds using IoT and underwater sensors: a review, *Sensors*, **22** (2022), 2088. <https://doi.org/10.3390/s22062088>
7. J. Puustjärvi, L. Puustjärvi, The role of smart data in smart home: health monitoring case, *Procedia Comput. Sci.*, **69** (2015), 143–151. <https://doi.org/10.1016/j.procs.2015.10.015>
8. S. V. N. Sreenivasu, T. S. Kumar, O. B. Hussain, A. R. Yeruva, S. R. Kabat, A. Chaturvedi, Cloud based electric vehicle's temperature monitoring system using IOT, *Cybern. Syst.*, (2023) 1–16. <https://doi.org/10.1080/01969722.2023.2176649>
9. A. A. Khan, A. A. Laghari, A. A. Shaikh, Z. A. Shaikh, A. K. Jumani, Innovation in multimedia using IoT systems, in *Multimedia Computing Systems and Virtual Reality*, Taylor & Francis, (2022), 171–187. <http://dx.doi.org/10.1201/9781003196686-8>
10. V. Upadrista, S. Nazir, H. Tianfield, Secure data sharing with blockchain for remote health monitoring applications: a review, *J. Reliab. Intell. Environ.*, **9** (2023), 349–368. <https://doi.org/10.1007/s40860-023-00204-w>
11. K. Saleem, H. Abbas, J. Al-Muhtadi, M. A. Orgun, R. Shankaran, G. Zhang, Empirical studies of ECG multiple fiducial-points based binary sequence generation (MFBSG) algorithm in E-health sensor platform, in *2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)*, Dubai, United Arab Emirates, (2016), 236–240. <https://doi.org/10.1109/LCN.2016.053>
12. P. Morsali, S. Dey, A. Mallik, A. Akturk, Switching modulation optimization for efficiency maximization in a single-stage series resonant DAB-based DC-AC converter, *IEEE J. Emerging Sel. Top. Power Electron.*, **11** (2023), 5454–5469. <https://doi.org/10.1109/JESTPE.2023.3302839>
13. M. S. Akbar, Z. Hussain, M. Sheng, R. Shankaran, Wireless body area sensor networks: survey of MAC and routing protocols for patient monitoring under IEEE 802.15.4 and IEEE 802.15.6, *Sensors*, **22** (2022), 8279. <https://doi.org/10.3390/s22218279>
14. D. D. Olatinwo, A. M. Abu-Mahfouz, G. P. Hancke, H. C. Myburgh, Energy efficient priority-based hybrid MAC protocol for IoT-enabled WBAN systems, *IEEE Sens. J.*, **23** (2023), 13524–13538. <https://doi.org/10.1109/JSEN.2023.3273427>
15. K. Saleem, F. Y. Alfariheedi, R. Ouni, J. Al-Muhtadi, Cellular IoT based secure monitoring system for smart environments, in *2022 IEEE International Conference on E-health Networking, Application & Services (HealthCom)*, Genoa, Italy, (2022), 1–5. <https://doi.org/10.1109/HealthCom54947.2022.9982776>

16. A. H. Montazeri, S. K. Emami, M. R. Zaghiyan, S. Eslamian, Stochastic learning algorithms, in *Handbook of Hydroinformatic*, Elsevier, (2023), 385–410. <http://dx.doi.org/10.1016/B978-0-12-821285-1.00016-6>
17. B. G. Mohammed, D. S. Hasan, Smart healthcare monitoring system using IoT, *Int. J. Interact. Mob. Technol.*, **17** (2023), 141–152. <https://doi.org/10.3991/ijim.v17i01.34675>
18. G. Alandjani, IoT enabled healthcare monitoring system using convolutional neural network, *ARPN J. Eng. Appl. Sci.*, **18** (2023), 245–250. <http://dx.doi.org/10.59018/022343>
19. R. Alharbi, H. Alhichri, R. Ouni, Y. Bazi, M. Alsabaan, Improving remote sensing scene classification using quality-based data augmentation, *Int. J. Remote Sens.*, **44** (2023), 1749–1765. <https://doi.org/10.1080/01431161.2023.2184213>
20. M. A. Razzaque, S. S. Javadi, Y. Coulibaly, M. T. Hira, QoS-aware error recovery in wireless body sensor networks using adaptive network coding, *Sensors*, **15** (2015), 440–464. <https://doi.org/10.3390/s150100440>
21. S. D. Suganthi, R. Anitha, V. Sureshkumar, S. Harish, S. Agalya, End to end light weight mutual authentication scheme in IoT-based healthcare environment, *J. Reliab. Intell. Environ.*, **6** (2020), 3–13. <https://doi.org/10.1007/s40860-019-00079-w>
22. G. Manikandan, D. Karunkuzhali, D. Geetha, V. Kavitha, Design of an IoT approach for security surveillance system for industrial process monitoring using Raspberry-Pi, in *AIP Conference Proceedings*, **2519** (2022), 030024. <https://doi.org/10.1063/5.0109769>
23. H. Meddeb, Z. Abdellaoui, F. Houaidi, Development of surveillance robot based on face recognition using Raspberry-PI and IOT, *Microprocess. Microsyst.*, **96** (2023), 104728. <https://doi.org/10.1016/j.micpro.2022.104728>
24. S. Shreya, K. Chatterjee, A. Singh, A smart secure healthcare monitoring system with Internet of Medical Things, *Comput. Electr. Eng.*, **101** (2022), 107969. <https://doi.org/10.1016/j.compeleceng.2022.107969>
25. K. Sangeethalakshmi, S. P. Angel, U. Preethi, S. Pavithra, V. S. Priya, Patient health monitoring system using IoT, *Mater. Today: Proc.*, **80** (2023), 2228–2231. <https://doi.org/10.1016/j.matpr.2021.06.188>
26. J. Xie, X. Xiao, Y. Xu, B. Jin, An IoT assisted early warning system for smart grid, *J. Phys.: Conf. Ser.*, **2218** (2022), 012027. <https://doi.org/10.1088/1742-6596/2218/1/012027>
27. H. Albataineh, M. Nijim, S. Ballampalli, The design of a novel smart home control system using a smart grid based on edge and cloud computing, *Int. J. Smart Grid Clean Energy*, **11** (2022), 57–71. <https://doi.org/10.1109/SEGE49949.2020.9181961>
28. L. Nasraoui A. Cabani, H. Trimech, Implementing lightweight key exchange solutions for WSN with LoRa connectivity, *Int. J. Sens. Netw.*, **39** (2022), 192–204. <https://dx.doi.org/10.1504/IJSNET.2022.124569>
29. K. Sangeethalakshmi, S. J. Ganesh, K. Dhivya, V. Kannagi, M. Rajkumar, Internet of Things assisted wireless environment monitoring system using smart sensors supportivity, in *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, Tuticorin, India, (2022), 628–632. <https://doi.org/10.1109/ICEARS53579.2022.9751994>
30. T. Nguyen-Tan, C. Dang-Ngoc, Q. Le-Trung. A smart agriculture solution includes intelligent irrigation and security, in *Industrial Networks and Intelligent Systems, INISCOM 2023*, Springer Nature, (2023), 3–18. https://doi.org/10.1007/978-3-031-47359-3_1

-
31. D. Hercog, T. Lerher, M. Truntič, O. Težak, Design and implementation of ESP32-based IoT devices, *Sensors*, **23** (2023), 6739. <https://doi.org/10.3390/s23156739>



AIMS Press

©2024 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0>)