Journal of Information Systems and Informatics

**Vol. 5, No. 2, June 2023** e-ISSN: **2656-4882** p-ISSN: **2656-5935**

DOI: 10.51519/journalisi.v5i2.493 Published By DRPM-UBD

# Design of A Security System with Mobile Notifications in Case of Intrusion

**Yanogo Kiswendsida Jean Hermann[1], Gerald Yirga[2], Nabolle Rachid Gaetan[3], Kabore Baowendnere Tanguy[4]**

[1,2,3,4]Institute of Computer Engineering and Telecommunication Polytechnic, Ouagadougou, Burkina Faso
Email: [1]yanogohermann@yahoo.fr, [2]yirgagerald@gmail.com, [3]nabollerg@gmail.com, [4]tanguykabore@yahoo.fr

## Abstract

Physical security is a crucial aspect of cybersecurity, considering the constant risk of unauthorized intrusions and data theft that companies face. While conventional security measures like alarm systems and video surveillance aim to protect physical infrastructure, they often fall short of meeting businesses' actual requirements. It is vital for companies to have immediate awareness of intrusions and the ability to respond promptly. To tackle this challenge, the recommended approach involves implementing an alert security system that offers real-time notifications. This method entails deploying an Internet of Things (IoT) system equipped with sensors throughout the company's premises. These sensors are specifically designed to detect unauthorized entry, triggering instant alerts to the appropriate personnel via a dedicated mobile application. This application enables seamless communication with the IoT system, enabling swift decision-making in response to potential security breaches. In this research, the initial emphasis was on designing the system using Unified Modeling Language (UML) diagrams. Subsequently, the IoT system was implemented, integrating the necessary sensors. Additionally, a user-friendly application was developed to establish a secure connection between the company's system and employees' mobile devices. Through extensive research efforts, an effective and functional system capable of effectively safeguarding businesses has been successfully devised.

**Keywords**: Physical Security, Mobile Notification, Alert Security System, Internet of Things (Iot), Real-Time Notifications

## 1. INTRODUCTION

As urban populations grow and cities become more crowded, the incidence of break-ins and security breaches in companies has become a regular occurrence. Moreover, logistical challenges, such as long distances between workplaces and employees' residences, or nighttime operations relying solely on security guards, contribute to increased absenteeism. Existing security solutions often prove costly and fail to guarantee data protection. In light of these pressing issues, our research aims to explore affordable IT solutions that can bolster business security.

The potential of Internet of Things (IoT) systems, comprising interconnected objects embedded with artificial intelligence, holds great promise. By utilizing electronic components, such as processors and connected sensors, these systems can collect, transmit data, and trigger actions based on changes in state or measurements. IoT technology has the potential to significantly impact various aspects of life and technology, presenting extensive opportunities for technology enhancement and benefiting humankind as a whole [1].

The primary objective of our research is to develop a system that promptly alerts businesses about intrusions, enabling rapid decision-making both within and outside the company. Unlike existing systems, our solution ensures real-time alerts regarding abnormal situations within the company without requiring physical presence in the area. We have based our research on the Unified Modeling Language (UML), a language for specifying, visualizing, and documenting object-oriented systems. Our system leverages the principles of Object-Oriented Analysis/Design (OOA/OOD), Object Modeling Technique (OMT), and Object-Oriented Software Engineering (OOSE) [2]. Additionally, we have developed an application that can be installed on smartphones, facilitating communication with the IoT system. Notably, all user accounts capable of receiving alerts are internal to the company and are not managed by third parties.

Traditionally, objects within an IoT network share their data with a gateway or another device for processing. Increasingly, various industries are embracing IoT technology to enhance performance [3]. The choice of connectivity methods between objects depends on project-specific constraints. When implementing IoT security, the main objectives include preserving privacy, confidentiality, and ensuring the security of users, infrastructures, data, and devices within the IoT ecosystem. Moreover, guaranteeing the availability of services offered by the IoT ecosystem is crucial [4]. Equally important is the absence of vulnerabilities within the application, as vulnerabilities can manifest as weaknesses in internet application construction [5].

This research is essential as it contributes to the improved security of companies by providing instant intrusion alerts directly to mobile phones. Unlike conventional video surveillance systems that require continuous monitoring to detect intrusions, our system automatically sends alerts upon detecting any intrusion, eliminating the need for constant area surveillance. This streamlined approach facilitates efficient time management for personnel responsible for monitoring through cameras. By harnessing the potential of IoT and leveraging the capabilities of innovative system, we are poised to enhance business security and empower companies with real-time awareness of potential threats.

## 2. METHODS

The research methodology employed in this study incorporates several distinct approaches, as illustrated in Figure 1. Figure 1 provides an overview of these approaches, which are further explained below.

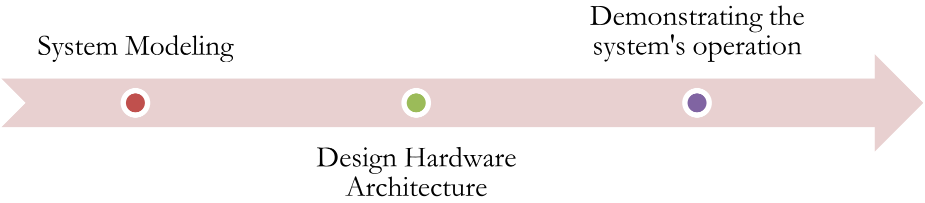

System Modeling
Demonstrating the system's operation
Design Hardware Architecture

**Figure 1.** Research Process

Modeling the system using UML (Unified Modeling Language): UML has emerged as a widely adopted modeling language within the software industry due to its effectiveness. It offers a range of tools designed to simplify the modeling process and enhance communication among developers within a team. While many researchers working on Agent-Based Models (ABMs) acknowledge the benefits of adopting object-oriented (OO) practices for programming their models, UML diagrams remain highly prominent and valuable [6]. One of the key advantages of UML is its programming language independence, providing significant flexibility. In this study, we employ various UML diagrams, including use case diagrams, sequence diagrams, class diagrams, and deployment diagrams, to present our system effectively.

Designing the hardware architecture: This involves defining the necessary hardware components, such as the electronic circuitry, the layout of the printed circuit board (PCB), as well as the back-end and front-end of the system application. By carefully considering these aspects, we ensure the system's functionality and performance align with the research objectives.

Demonstrating the system's operation: This phase focuses on presenting a comprehensive understanding of how the system functions. It involves illustrating the step-by-step processes, interactions, and dependencies between different system elements. Through clear and concise explanations, we provide readers with an in-depth understanding of how our system operates.

By employing this multi-faceted methodology, we aim to achieve a comprehensive understanding of the system's design, hardware architecture, and operational mechanisms. This holistic approach enables us to develop an effective and efficient system that addresses the core objectives of our research.

## 2.1. System Modeling

The system modeling consists of four models: the use case diagram, the class diagram, the sequence diagram, and the deployment diagram. Each of these diagrams can be explained as follows.

### 2.1.1 Use Case Diagram

The Figure 2 depicts the use case diagram for the new system, which showcases potential future features. This diagram ensures that the software or system encompasses the required functionalities for different actors and their implications. In the diagram, we identify the following actors and their respective roles:

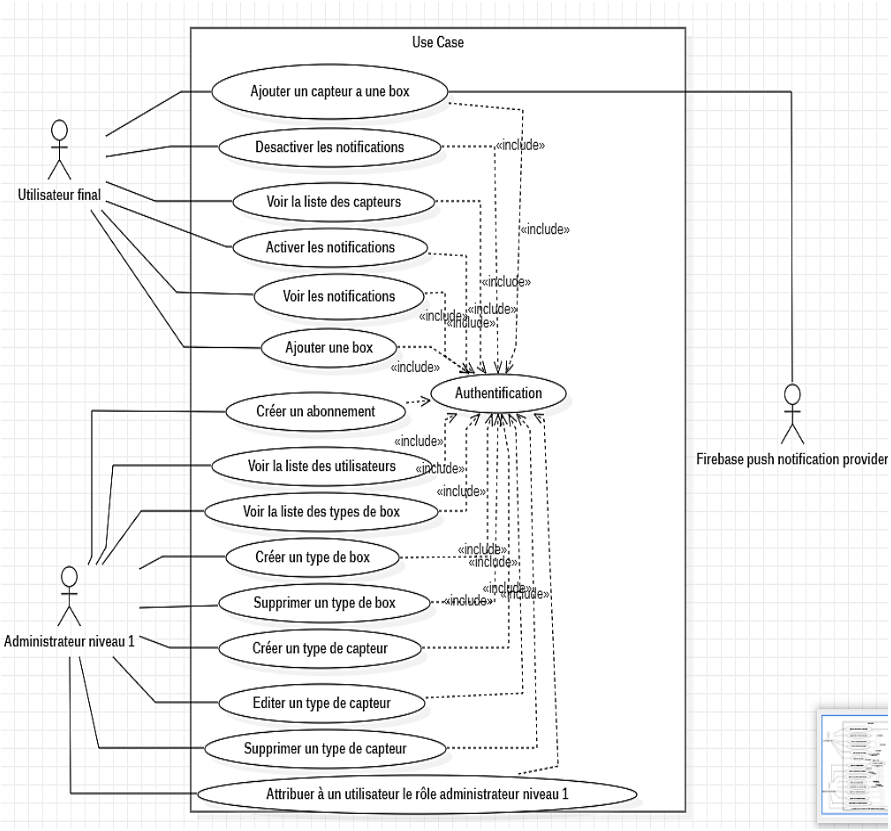

Use Case
Utilisateur final
Administrateur niveau 1
Firebase push notification provider
Ajouter un capteur a une box
Desactiver les notifications
Voir la liste des capteurs
Activer les notifications
Voir les notifications
Ajouter une box
Authentification
Créer un abonnement
Voir la liste des utilisateurs
Voir la liste des types de box
Créer un type de box
Supprimer un type de box
Créer un type de capteur
Editer un type de capteur
Supprimer un type de capteur
Attribuer à un utilisateur le rôle administrateur niveau 1
«include»

**Figure 2.** Use case diagram

1) End User (main actor): This user represents the intended beneficiary of the system. They possess the authority to administer the security system implemented within the company.

2) Administrator: The Administrator is responsible for performing the system's most common operations.
3) Firebase push notification provider (secondary actor): This actor facilitates the creation of notifications on Android mobile devices.

Each actor has their own specific area of operation. It is essential for all main actors to have access to an internet connection in order to utilize the system. Authentication is carried out by entering an email address or phone number, along with a password. The use case diagram serves as a visual representation of the interactions and relationships between actors and the system. It assists in ensuring that the system meets the requirements and expectations of its users. By analyzing the diagram, stakeholders can gain insights into the functionalities available to each actor and how they contribute to the overall system functionality. The diagram helps in identifying potential gaps or areas for improvement, thereby enabling effective system development and user satisfaction.

### 2.1.2 Class diagram

The class diagram is utilized to define the entities present in the system and illustrate their relationships. UML diagrams, such as the class diagram, provide visual models that aid in identifying the requirements and scope of systems and applications [7]. In Figure 3, we can identify the following entities that are essential for the system's functionality:

1) User: This class contains all the user-related information. Its attributes include "firstname," "lastname," "email," "phone," "password," and "accessLevel," which respectively represent the user's first name, last name, email address, phone number, password hash, and access level. It is worth noting that passwords are stored as hashed values, obtained through a hash function. During authentication, provided passwords are hashed and compared to the stored hashed password in the database, enhancing user security. Other attributes in the User class store data relevant to the platform's operation and reference other classes. When a user is created, the "activated" and "deleted" attributes are set to false by default. Only when the user clicks on the activation link sent via email after registration or when the user's account is deleted, these attributes are respectively changed to true. The attributes "securityBoxes," "messages," and "subscriptions" are references to other classes.
2) Message: This class represents the messages received by the user and is identified by its identifiers stored in the "messages" attribute of the User class. It contains information such as the message content, whether it has been read, who sent it, the date of sending, and if it has been deleted.
3) SecurityBox: This class represents the security system gateway. It is linked to the "modelOfBox" class through a composition arrow, indicating that every instance of this class must reference the "modelOfBox" class during

creation. The smart security box operates autonomously without requiring human interaction [8].

4) ModelOfBox: This class serves as a reference and enables the inclusion of different box models within the system. This allows for the continued evolution of modules without the need to restructure the back-end. The modelOfBox class plays a crucial role in designing IoT system communication [9].
5) Sensor: This class represents the sensors used in the security system. Its key attributes are "state" and "notificationAllow," which indicate the opening state and the activation or deactivation of notifications for the represented sensor.
6) ModelOfSensor: Similar to the ModelOfBox class, this class acts as a reference and allows for the inclusion of different sensor models in the system.
7) Event: This class serves as a central element in the system, representing alerts and events triggered in the event of an intrusion within the monitored perimeter.

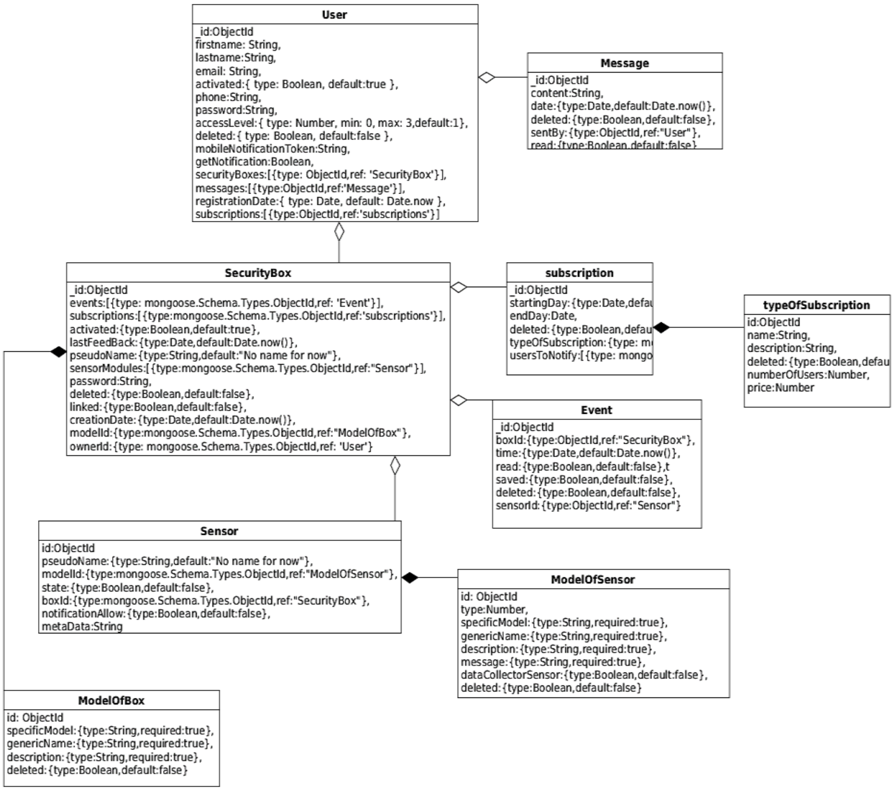

User
_id:ObjectId
firstname: String,
lastname:String,
email: String,
activated:{ type: Boolean, default:true },
phone:String,
password:String,
accessLevel:{ type: Number, min: 0, max: 3,default:1},
deleted:{ type: Boolean, default:false },
mobileNotificationToken:String,
getNotification:Boolean,
securityBoxes:[{type: ObjectId,ref: 'SecurityBox'}],
messages:[{type:ObjectId,ref:'Message'}],
registrationDate:{ type: Date, default: Date.now },
subscriptions:[{type:ObjectId,ref:'subscriptions'}]
Message
_id:ObjectId
content:String,
date:{type:Date,default:Date.now()},
deleted:{type:Boolean,default:false},
sentBy:{type:ObjectId,ref:"User"},
read:{type:Boolean,default:false}
SecurityBox
_id:ObjectId
events:[{type: mongoose.Schema.Types.ObjectId,ref: 'Event'}],
subscriptions:[{type:mongoose.Schema.Types.ObjectId,ref:'subscriptions'}],
activated:{type:Boolean,default:true},
lastFeedBack:{type:Date,default:Date.now()},
pseudoName:{type:String,default:"No name for now"},
sensorModules:[{type:mongoose.Schema.Types.ObjectId,ref:"Sensor"}],
password:String,
deleted:{type:Boolean,default:false},
linked:{type:Boolean,default:false},
creationDate:{type:Date,default:Date.now()},
modelId:{type:mongoose.Schema.Types.ObjectId,ref:"ModelOfBox"},
ownerId:{type: mongoose.Schema.Types.ObjectId,ref: 'User'}
subscription
_id:ObjectId
startingDay:{type:Date,defau
endDay:Date,
deleted:{type:Boolean,defau
typeOfSubscription:{type: m
usersToNotify:[{type: mongo
typeOfSubscription
id:ObjectId
name:String,
description:String,
deleted:{type:Boolean,defau
numberOfUsers:Number,
price:Number
Event
_id:ObjectId
boxId:{type:ObjectId,ref:"SecurityBox"},
time:{type:Date,default:Date.now()},
read:{type:Boolean,default:false},t
saved:{type:Boolean,default:false},
deleted:{type:Boolean,default:false},
sensorId:{type:ObjectId,ref:"Sensor"}
Sensor
id:ObjectId
pseudoName:{type:String,default:"No name for now"},
modelId:{type:mongoose.Schema.Types.ObjectId,ref:"ModelOfSensor"},
state:{type:Boolean,default:false},
boxId:{type:mongoose.Schema.Types.ObjectId,ref:"SecurityBox"},
notificationAllow:{type:Boolean,default:false},
metaData:String
ModelOfSensor
id: ObjectId
type:Number,
specificModel:{type:String,required:true},
genericName:{type:String,required:true},
description:{type:String,required:true},
message:{type:String,required:true},
dataCollectorSensor:{type:Boolean,default:false},
deleted:{type:Boolean,default:false}
ModelOfBox
id: ObjectId
specificModel:{type:String,required:true},
genericName:{type:String,required:true},
description:{type:String,required:true},
deleted:{type:Boolean,default:false}

**Figure 3.** System class diagram

The class diagram provides a comprehensive overview of the system's structure, entities, and their relationships. It facilitates better understanding of the system's architecture and aids in designing and implementing the various components. By analyzing the class diagram, developers can identify the necessary classes, attributes, and associations, enabling efficient system development and maintenance. The diagram also supports scalability and modularity by allowing the addition of new models and sensors without significant backend modifications. Furthermore, the use of hashed passwords enhances the security of user data, safeguarding their information even in the event of a database breach. The autonomous nature of the security box streamlines its operation and reduces the need for constant human intervention.

### 2.1.3 Intrusion alert sequence diagram

To provide a step-by-step description of a use case and the entities involved in its execution, sequence diagrams are commonly used. These diagrams resemble scenarios that unfold during the execution of a use case. In the case of detecting an intrusion through a door opening sensor or a motion sensor, the sequence of events involves the following entities: the sensor, the box, the weaner, and the designated individuals for receiving notifications. When an intrusion is detected, an esp-Now protocol packet is transmitted from the sensor to the box. Depending on the current configuration of the box, it will then either send a signal or refrain from doing so to the weaner. Subsequently, the weaner sends a notification to the mobile device of the individuals previously designated to receive notifications from the specific device. This process ensures that the appropriate individuals are promptly informed about the detected intrusion.

Intrusion detection systems must possess the ability to differentiate between normal and abnormal activities in order to detect malicious attempts in a timely manner [10]. Figure 4 presents a detailed sequence diagram illustrating the various steps involved in the event of an intrusion, starting from the sensor sending the signal to the box, then to the server, and finally leading to the notification being sent to the user's mobile device.

By utilizing sequence diagrams, system designers and developers can visualize the flow of events and interactions between entities involved in a specific use case. These diagrams aid in understanding the order of actions, the exchange of information, and the roles of different components within the system. Analyzing the sequence diagram allows for the identification of potential bottlenecks or areas for improvement, ensuring the efficient execution of the use case and enhancing the overall functionality of the system.

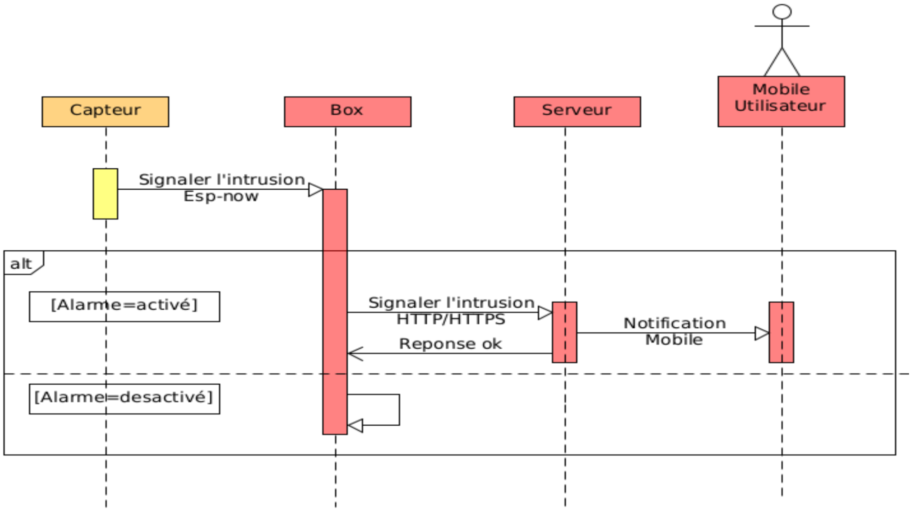


**Figure 4.** Sequence diagram in case of intrusion detection

### 2.1.4 System Deployment Diagram

The deployment diagram provides an overview of how a system is deployed, showcasing the various machines and entities involved in its operation. Figure 5 precisely illustrates the deployment strategy employed to make our system operational and efficient. Within this diagram, several key components can be identified:

1) Sensors: These devices play a crucial role in collecting data and reporting intrusions. They utilize the proprietary communication protocol, esp-now, to transmit information to the designated box.
2) Boxes: Serving as gateways, these local controllers receive esp-now requests from the sensors and translate them into HTTP/HTTPS requests to interact with the back-end servers. The boxes act as intermediaries, enabling seamless communication between the sensors and the system's logic.
3) Back-end servers: Functioning as the central points of the system, these servers contain the core logic and functionality. All other components of the system communicate with the back-end servers, primarily via the internet. They handle processing requests, executing business logic, and delivering responses to connected devices.
4) Data server and replicas: These servers are responsible for data persistence and availability. They ensure the reliability of the system by duplicating data in replicas whenever data is written or deleted in the primary data server. In the event of a failure of the primary data server, one of the replicas can assume the role of the primary server, ensuring uninterrupted access to the data.

5) Telephones and computers: These devices actively participate in the system by sending requests to the back-end servers to retrieve the required data. Mobile and web applications running on these devices interact with the main back-end servers to obtain the necessary information for their proper functioning.

The deployment diagram provides a comprehensive understanding of how different components of the system are distributed across machines and entities. It aids in visualizing the network of interconnected devices, servers, and communication protocols involved in delivering a functional system. Analyzing the deployment diagram assists in identifying potential bottlenecks, optimizing resource allocation, and ensuring the seamless operation of the system across various deployment environments.

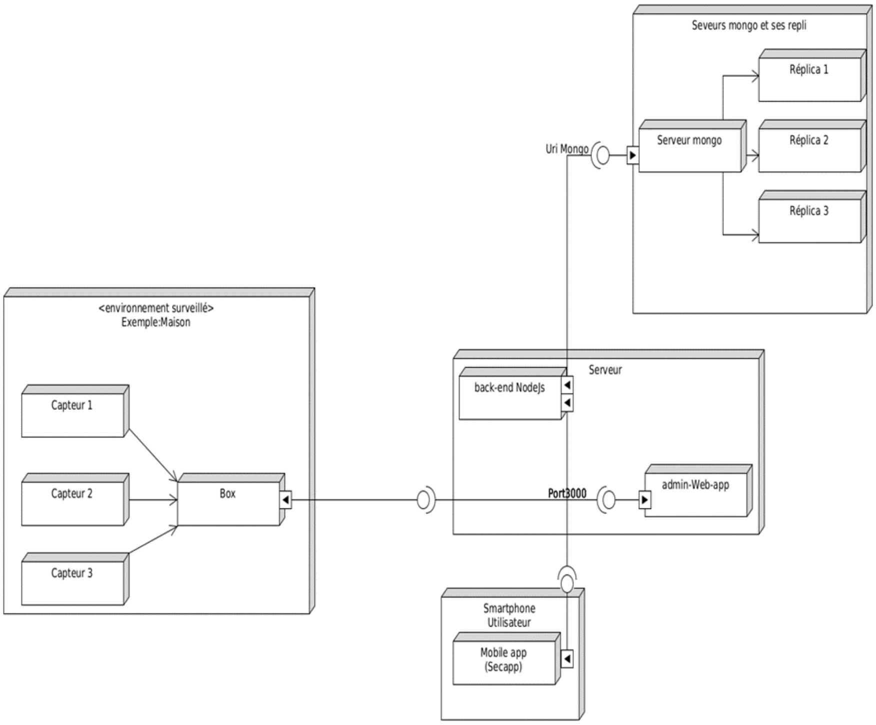


**Figure 5.** Deployment diagram.

### 2.2. Applications Used To Interact With The System

The implementation of a comprehensive security system necessitates the utilization of numerous software and hardware technologies. The backbone of the system resides within the back-end, where the system's logic is housed. Both the mobile application and the web application, which constitute the front-end, rely on the back-end to function effectively. The technologies employed for this crucial part of the system include:

1) Node.js: Node.js is a JavaScript/TypeScript back-end framework that enables the development of back-end servers using the JavaScript programming language. Previously, JavaScript was primarily used for front-end development until the introduction of Node.js. Its dynamic typing makes it an ideal framework for rapid development, forming the foundation for our back-end development.
2) MongoDB: MongoDB is a NoSQL database management system that offers an alternative to traditional SQL databases. NoSQL databases, including MongoDB, provide efficient and intuitive solutions for storing large-volume data such as web service data, marketing data, IoT device logs, and security logs [11]. MongoDB also supports the creation of replicas, allowing for the creation of database copies for enhanced data availability.
3) Visual Studio Code: Visual Studio Code is a free text editor developed by Microsoft, offering extensive functionality for a wide range of programming languages through the use of extensions. It provides a versatile and user-friendly environment for developers.
4) Android Studio: Android Studio is the official integrated development environment (IDE) for creating applications on the Android platform. Built on the IntelliJ platform, it allows developers to simulate various devices on their computer, facilitating application testing and deployment on Android devices.
5) Postman: Postman is an indispensable tool for API development. It enables developers to test APIs by sending requests of different types, including JSON, XML, GraphQL, and more. Its intuitive interface allows for the inclusion of attributes of various types within requests, streamlining the testing process.
6) Mongo-gui: Mongo-gui is a database visualization tool that eliminates the need for command-line inputs when interacting with databases. It provides a graphical interface for selecting and manipulating databases and collections, enhancing efficiency and ease of use. It is comparable to phpMyAdmin, commonly used for managing MySQL databases.
7) Arduino IDE: Arduino IDE is open-source software that offers a simplified API for embedded systems. Initially developed for Arduino prototyping boards (AVR), it can now be utilized for other boards and microcontrollers. Arduino microcontrollers provide a user-friendly tool for developing small projects involving sensors, as they are easy to learn and program. Arduino IDE serves as a programming platform for Arduino microcontrollers [12].
8) Ubuntu Server: Ubuntu Server is the chosen operating system for our Virtual Private Server (VPS). Ubuntu is one of the most widely used Linux distributions, benefiting from a large community that can provide support and solutions through forums. Ubuntu Server is a robust operating system that performs excellently in such environments [13].

The front-end of the software encompasses the user-visible components such as pages, buttons, and text entries. In our system, the front-end consists of two components: the mobile application and the web application. Both are developed using the Ionic framework on Angular. Ionic is a versatile framework that can be integrated with other frameworks such as React, Vue, and Flutter, allowing for efficient cross-platform development. By leveraging these software and hardware technologies, our security system achieves a robust and effective implementation, providing users with reliable and user-friendly functionality.

### 2.3. Hardware Architecture

The hardware architecture for the ToI project had to meet several specific constraints, including:

1) Low power consumption: Due to the autonomous nature and size of the sensors, it was crucial to minimize power consumption. This requirement ensures extended battery life and operational efficiency.
2) Fast data transmission: The system needed to swiftly deliver data to the relevant individuals to ensure timely response and action.
3) Compact size: The hardware components had to be small in size to enable discreet installation in various locations.

Considering these constraints, we selected the esp-01 as our preferred component, as it fulfills all the aforementioned conditions. The esp-01 offers the following advantages:

1) Low power consumption: It incorporates a deep-sleep mode that significantly reduces power usage to approximately 10µA during idle periods. The module awakens only when necessary, allowing for prolonged battery operation.
2) Support for esp-now protocol: The esp-01 enables the utilization of the esp-now protocol, which facilitates the transmission of small packets at high speeds. Under certain conditions, this protocol can achieve a range of up to 1000m. Compared to HTTP protocols, esp-now is approximately 20 times faster on average. This enhanced speed ensures swift data transfer, minimizing delays and facilitating real-time communication.
3) Compact size: The esp-01 is remarkably small, with dimensions comparable to a 100 CFA franc coin in terms of both thickness and size. Unlike most development modules, the esp-01 does not incorporate its programming microcontroller, contributing to its compact design. These features make it an ideal component for discreetly installing the device on various openings and corners within businesses.

By selecting the esp-01 for hardware architecture, we have successfully addressed the requirements for low power consumption, fast data transmission, and compact size. This choice ensures an efficient and inconspicuous implementation of the security system in diverse business environments. The esp-01 for hardware architecture as shown in Figure 6 and 7.

**Figure 6.** Esp-01 seen from above

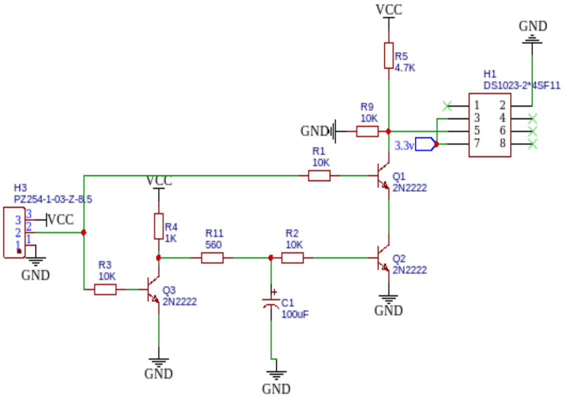


**Figure 7.** Electronic interface circuit between the sensors and the esp-01

After successfully creating a prototype on a breadboard, the next phase involved the production of printed circuit boards (PCBs). This process entails several essential steps. Firstly, we needed a circuit diagram, which we had already developed. With the circuit diagram in hand, we proceeded to design the PCB layout.

To accomplish this, we utilized EasyEDA, an online platform specifically designed for designing and visualizing PCBs based on electrical diagrams. EasyEDA provided us with the necessary tools to transform our circuit diagram into a detailed PCB layout. Once the design was complete, we were able to preview the final PCB before manufacturing. This preview allowed us to examine the layout and ensure its accuracy and functionality. Figures 8, 9, and 10 showcase the preview of the PCB, providing a visual representation of the finished product.

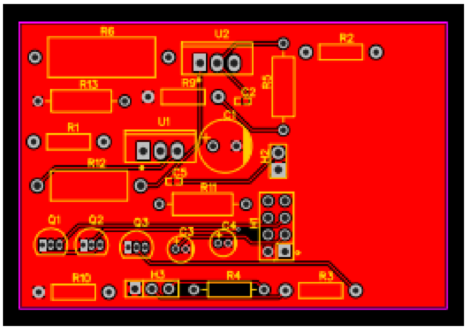


**Figure 8.** PCB top layer

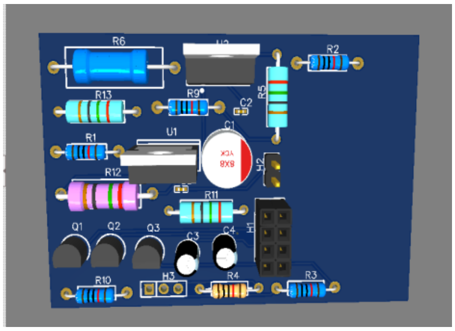


**Figure 9.** 3D view of the PCB

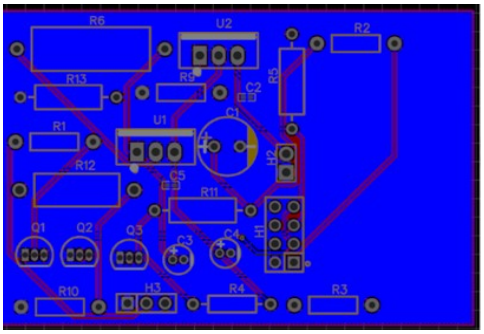


**Figure 10.** Bottom layer of pcb

## 3. RESULTS AND DISCUSSION

### 3.1 System Function Test

Figure 11 illustrates the fully functional device that should be deployed within the monitored environment. This device consists of a minimum of two modules, operating in a master and slave configuration. The purpose of this configuration is to minimize the number of requests sent to the main server (the back-end). Specifically, the slave module is responsible for transmitting messages to the server, based on the predefined configuration.

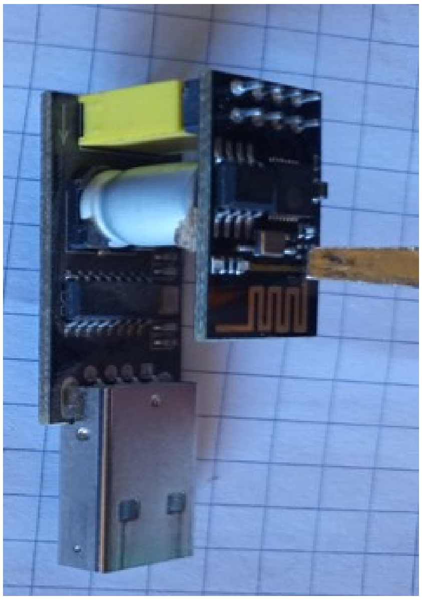


**Figure 11.** Esp-01 with its programmer seen from above

In our case, the masters refer to state change sensors, namely motion sensors and opening/closing sensors for doors or windows. Motion sensors utilize PIR (Passive Infrared) technology to detect movement by sensing the heat emitted by the human body. These sensors are commonly found in outdoor solar lamps. They can be configured to generate a brief, low-level signal. As a cost-effective alternative to occupancy sensors that rely on light or ultrasound, motion sensors are extensively examined [14].

Opening and closing sensors utilize Hall sensors, which detect changes in a magnetic field. These sensors are combined with magnets installed on openings such as doors. When the door is closed, the magnetic field of the magnets runs parallel to the Hall sensor but not aligned with it. However, when the door is opened, the magnetic field of the magnets momentarily aligns with the Hall sensor, generating a low-level signal during this exposure period. Figure 12 showcases a simulation test of the interface circuit between the ESP module and the sensor.

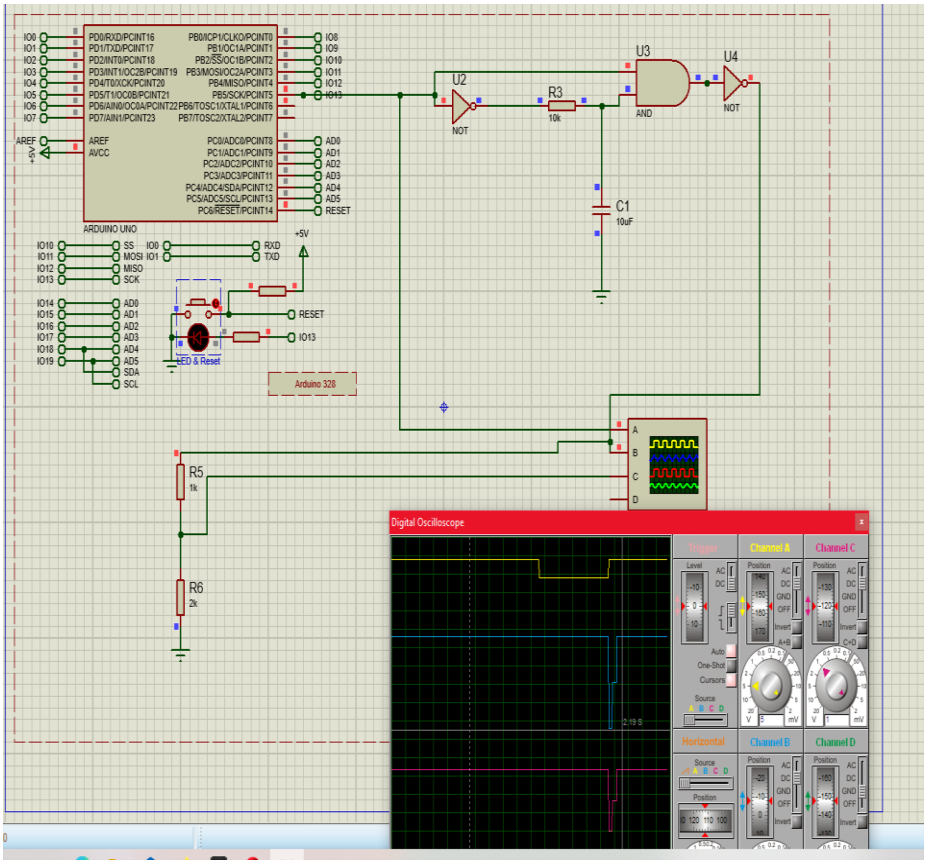


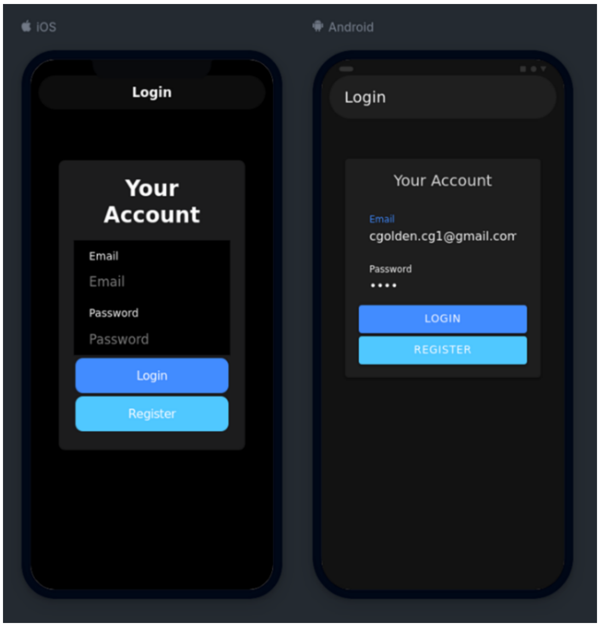
**Figure 12.** simulation test of the interface circuit between esp and sensor

The circuit is designed to produce a pull-down effect when any of the sensor types are triggered. The resistor and capacitor components enable independent adjustment of the desired pull-down time, separate from the pull-down time generated by the connected sensor. After being triggered, the module connected to the sensor sends a packet via the esp-Now protocol to the slave module.

The slave module acts as a gateway, facilitating communication between esp-Now requests and the HTTP protocol if the information needs to be relayed to the server. Since the slave module needs to remain powered continuously to receive requests from the masters, it requires a constant power source. Existing approaches nowadays rely on electronic devices that depend on sensors with onboard power sources and continuous power supply for sensing and data logging [15].

### 3.2 Functional Test of The Mobile Application

ToI's mobile application is primarily tailored for users who need to stay informed about their company's security. It provides notifications and allows users to view the status of various installations within their company, including the current state of different openings. Informational mobile applications play a crucial role in today's world, as they offer convenience, interactivity, and compatibility for individuals [16]. The accompanying images, displayed in Figure 13, offer an overview of the mobile application.



cgolden.cg1@gmail.com
••••

Figure 13. Mobile app login on Iphone and Android

In Figure 13, we can observe that the application, which handles notifications from the system, is fully functional. The login process is based on the user's email and password, as specified in the use case diagram. By logging into their account, which is configured on the server, the system user can easily access the application and monitor the relevant information. This monitoring capability provides a simple and convenient solution, allowing users to check on their company's security from anywhere and at any time. Mobile applications have become an integral part of daily life, dominating individuals' digital habits due to advancements in mobile technologies, high-speed internet access on mobile devices, and the interactive nature of mobile interfaces [17]. The market for mobile apps is continuously expanding and has become a significant source of revenue generation. Mobile apps facilitate the timely dissemination of essential information in a user-friendly format [18].

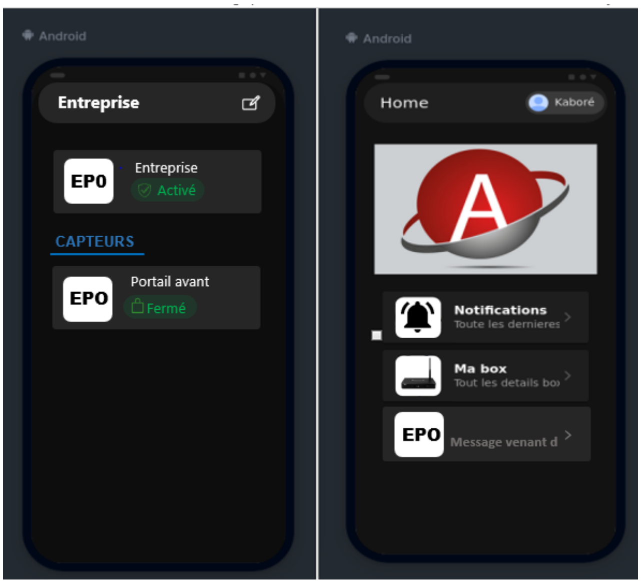


Figure 14. Mobile app screenshot: a) Left: Installation details b) Right: Mobile app home page

In Figure 14, it is evident that the application, after the user logs in, provides detailed information about an installation. This includes the status of sensors, the front gate, and the available notifications. This research highlights the value of mobile app design activities and the utilization of web-based visual programming tools [19]. Our solution sets itself apart from existing alternatives by offering precise and instantaneous alerts directly to employees' phones. Other solutions often rely on video surveillance cameras, requiring continuous monitoring of the designated area. However, our system's alerts do not necessitate constant monitoring. The objectives of this research were to establish a security system capable of ensuring safety and delivering instant notifications. We can conclude that these objectives have been successfully achieved, as this system effectively fulfills the need for security and immediate alerts.

The presented system provides a comprehensive solution for security monitoring in a company by effectively incorporating motion sensors and opening/closing sensors. These sensors, utilizing proven technologies such as PIR and Hall sensors, deliver accurate and real-time information about the status of different installations, including doors and windows. A notable feature of the system is its master-slave configuration, which optimizes data transfer and reduces server load. By employing a slave module for communication and message transmission, the system ensures efficient operation and minimizes the number of requests reaching the server. The interface circuit between the ESP module and the sensors further enhances reliability by facilitating precise timing and reliable triggering of sensor events.

The mobile application developed alongside the system plays a crucial role in providing users with convenient access to security information. Through direct notifications on their mobile devices, users can easily stay informed about the status of various installations in their company. Secure login credentials, such as email and password, personalize the user experience and ensure authorized access to the application. In today's digitally-driven society, where mobile technologies and high-speed internet have become prevalent, mobile applications have become an indispensable part of everyday life. This mobile application aligns with the prevailing trend of mobile apps dominating individuals' digital routines. Offering convenience, interactivity, and compatibility, these apps serve as ideal platforms for disseminating crucial information, such as security updates. Furthermore, the growing market for mobile apps provides opportunities for revenue generation and drives innovation across various industries.

Compared to existing solutions reliant on video surveillance, the described system boasts distinct advantages. By delivering instant and accurate alerts directly to employees' mobile phones, it eliminates the need for constant monitoring. This feature significantly enhances operational efficiency, enabling prompt responses to security events without requiring physical presence in the monitored area. The system successfully fulfills the objectives of security and instant alert delivery, effectively meeting the demands of modern security requirements. The system and mobile application exemplify the integration of reliable sensor technologies, a well-designed master-slave configuration, and a user-friendly mobile app interface. These advancements in security monitoring systems, particularly the inclusion of mobile applications, underscore the growing significance of technology in enhancing security measures and providing real-time information to users.

## 4. CONCLUSION

This research has resulted in a practical and effective security solution that follows technical procedures and industry standards. While various options exist for physical security, such as video surveillance, they have limitations compared to the solution proposed here, which is specifically tailored to meet the needs of sub-Saharan countries. Considering the importance of physical security in the broader context of cybersecurity, this solution has the potential to add significant value. The research process involved in designing this security system has led to the development of a product that fulfills the requirements of companies. By following a comprehensive methodology, all necessary steps were taken to ensure the system's reliability and efficiency. It is important to recognize the growing impact of security-related social issues in companies, which presents a major concern for our future society. Incidents continue to pose significant challenges in numerous societies today. This research holds great importance as it aligns with the evolving landscape of software systems, where reliance on human judgment is decreasing and computational intelligence is playing a greater role. The integration of hardware and software components, combined with the utilization of UML language and server operating systems, exemplifies the modern approach to designing security systems. These elements are crucial in addressing the complexities and requirements associated with contemporary security solutions. This research contributes to the field by offering a comprehensive and well-designed security system that specifically caters to the needs of sub-Saharan countries. The findings emphasize the significance of combining technical expertise, adherence to standards, and an understanding of societal challenges to develop effective solutions that enhance physical security in the digital age.

## REFERENCES

[1] K.A.M. Zeinab, and S.A.A. Elmustafa, "Internet of things applications, challenges and related future technologies," *World Scientific News*, vol. 67, no. 2, pp. 126-148, 2017.

[2] H. Koç, A.M. Erdoğan, Y. Barjakly, and S. Peker, "UML diagrams in software engineering research: a systematic literature review,' In *Proceedings*, vol. 74, no. 1, pp. 1-13. 2021.

[3] D. Grimaldi, and V. Fernandez, "Performance of an internet of things project in the public sector: The case of Nice smart city," *The Journal of High Technology Management Research*, vol. 30, no. 1, pp. 27-39, 2019.

[4] W.H. Hassan, "Current research on Internet of Things (IoT) security: A survei," *Computer networks*, vol. 148, pp. 283-294, 2019.

[5] M. Alsaffar, S. Aljaloud, B.A. Mohammed, Z.G. Al-Mekhlafi, T.S. Almurayziq, G. Alshammari, and A. Alshammari, "Detection of Web Cross-Site Scripting (XSS) Attacks," *Electronics*, vol. 11, no. 14, pp. 2212, 2022.

[6] H. Bersini, "Uml for abm," *Journal of Artificial Societies and Social Simulation*, vol. 15, no. 1, pp. 9, 2012.

[7] H.Koç, A.M. Erdoğan, Y. Barjakly, and S. Peker, "UML diagrams in software engineering research: a systematic literature review," In *Proceedings*, vol. 74, no. 1, pp. 13, 2021.

[8] S.I. Ayon, and A.S.B. Shahadat, "Smart Security Box using Arduino and GSM Module," In *2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things,* pp. 155-159, 2019.

[9] A. Zifarelli, G. Menduni, M. Giglio, A. Elefante, A. Sukhinets, A. Sampaolo, P. Patimisco, S. Fangyuan, W. Chongwu, Q.J. Wang, and V. Spagnolo, "Compact and versatile QEPAS-based sensor box for simultaneous detection of methane and infrared absorber gas molecules in ambient air," *Frontiers in Environmental Chemistry*, vol. 3, pp.1-12, 2022.

[10] S. Bourekkache, O. Kazar, and A. Aloui, "Computer and Network Security: Ontological and Multi-agent System for Intrusion Detection," *J. Digit. Inf. Manag.*, vol. 17, no. 3, pp. 133, 2019.

[11] J. Yoon, and S. Lee, "A method and tool to recover data deleted from a MongoDB," *Digital Investigation*, vol. 24, pp. 106-120, 2018.

[12] A.S. Ismailov, and Z.B. Jo'Rayev, "Study of arduino microcontroller board," *Science and Education*, vol. 3, no. 3, pp. 172-179, 2022.

[13] B. Brahara, D. Syamsuar, and Y. Kunang, "Analysis of Malware Dns Attack on the Network Using Domain Name System Indicators", *J. Inf. Syst. Informatics*, vol. 2, no. 1, pp. 131-153, Mar. 2020.

[14] T.K. Woodstock, and R.F. Karlicek, "RGB color sensors for occupant detection: An alternative to PIR sensors," *IEEE Sensors Journal*, vol. 20, no. 20, pp. 12364-12373, 2020.

[15] W. Wang, A. Sadeqi, H.R. Nejad, and S. Sonkusale, "Cost-effective wireless sensors for detection of package opening and tampering," *IEEE Access*, vol. 8, pp. 117122-117132, 2020.

[16] S. Kim, and T.H. Baek, "Examining the antecedents and consequences of mobile app engagement," *Telematics and Informatics*, vol. 35, no. 1, pp.148-158, 2018.

[17] A. Balapour, H.R. Nikkhah, and R. Sabherwal, "Mobile application security: Role of perceived privacy as the predictor of security perceptions," *International Journal of Information Management*, vol. 52, pp. 102063, 2020.

[18] K. Dhenuvakonda, and A. Sharma, "Mobile apps and internet of things (IoT): A promising future for Indian fisheries and aquaculture sector," *Journal of Entomology and Zoology Studies*, vol. 8, no. 1, pp. 1659-1669, 2020.

[19] Y.C. Hsu, and Y.H. Ching, "Mobile app design for teaching and learning: Educators' experiences in an online graduate course," *International Review of Research in Open and Distributed Learning*, vol. 14, no. 4, pp. 117-139, 2013.