APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 1 (2025) 382-400
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Intruder Alert System with Firebase Cloud for Syndicate Store and Service

Muhammad Haziq Badderol[1], Nurul Azma Abdullah[1*]

[1] *Fakulti Sains Komputer dan Teknologi Maklumat*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: azma@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.01.021

Article Info

Received: 8 May 2024
Accepted: 18 June 2025
Available online: 30 june 2025

Keywords

Access control, Alert system, Firebase Cloud, Notification, Prototype methodology

Abstract

The alert system is key for protecting the privacy and safety of any workplace. Syndicate Store and Service Sdn Bhd currently uses an outdated alert system, which lacks real-time notification capabilities and poses significant security vulnerabilities. Therefore, the system is to develop a modern alert system integrated with Firebase Cloud. This new system aims to enhance security by providing real-time notifications, thus significantly improving the efficiency and reliability of the alert mechanism in the workplace. This system integrates various functionalities such as real-time alert notifications and user monitoring by utilizing a web application, Firebase Realtime Database, and Arduino Uno Wi-Fi Rev2. The development of this system follows the prototype methodology, which involves iterative cycles of requirement gathering, design, build prototype, user evaluation, refine prototype and maintenance. The system is anticipated to reduce security vulnerabilities by providing real-time notifications, allowing for prompt responses to unauthorized access attempts. Additionally, the user-friendly interface and automated processes are expected to streamline the management of users and access control.

1. Introduction

In such an environment, where staff spend almost all their week working and may even stay overnight to meet deadlines, an alert system becomes essential to ensuring continuous safety and security for both staff and valuable assets.

The Syndicate Store and Service Sdn Bhd is a small business located in Ampang Selangor, specializing in cleaning and restoring shoes to keep the longevity and aesthetic appeal of customers' footwear. The business focuses on providing comprehensive care for shoes of various styles and materials, including leather, suede, and canvas, offering specialized cleaning techniques and restoration services tailored to each pair's unique characteristics. Specialists utilize a variety of materials and methods to replace worn parts, experiment with finishing compositions, and recreate specific effects like wear, abrasion, or patina to give shoes a renewed look [1].

Currently, the staff rely on manual key-based systems for access control. These methods, while functional, are time-consuming and lead to inefficiencies of potential security risks. However, they also lack real-time alert monitoring and access control capabilities, which leaves the company vulnerable to unauthorized access and

security breaches. This impacts on the business's ability to promptly respond to potential breaches, risking customer assets and the integrity of the restoration process.

The project focuses on designing, developing, and testing the functionality of an Intruder Alert System integrated with Firebase Cloud to enhance the security measures of Syndicate Store and Service's office environment. The design phase ensures that all necessary functionalities and security protocols are incorporated. Following the design, the development phase involves building the system with Firebase Cloud integration for real-time data synchronization, user management, and alert notifications. The final phase includes alpha and beta testing to ensure the system's reliability and effectiveness, utilizing feedback from administrators and users to refine and optimize performance before full deployment. The web application will feature several modules for both admin and user functionalities; administrators will manage user access, user registration, real-time intrusion detection, while users will have a streamlined interface for registration process for user account and pin code for access system and update password. The implementation of a keypad access control system will offer real-time intrusion detection capabilities. The system will be configured to trigger immediate alerts via a Telegram mobile application upon detecting unauthorized access attempts. Integration with Firebase Cloud will establish a centralized and secure platform for storing access logs and unauthorized activity data.

2. Literature Review

This section provides a comprehensive review of the literature related to Intruder Alert System with Firebase Cloud, highlighting the key components and comparisons of the existing system which can be used as a guide to develop a better system.

2.1 Key-Lock Mechanism

Digital technologies are increasingly important in modern security systems, providing advanced features like alerts and remote access control [2]. Key locks, essential for physical security, range from simple mechanical locks to sophisticated biometric and electronic systems. These technologies aim to provide secure and reliable access control while addressing the limitations of traditional lock systems [3],[4],[5],[6],[7]. Businesses like Syndicate Store and Service Sdn Bhd need robust security measures to protect assets and ensure safety, especially in the face of sophisticated security breaches. Advanced door lock systems, including RFID card, smart locks and keypad pin code offer secure and reliable access control. Below is section 2.1.1 2.1.2 and 2.1.3.

2.1.1 RFID Card

Keypad locks or electronic locks offer a convenient and secure way to control access without physical keys. Users enter a unique PIN code on a keypad to unlock the door. These systems have lower implementation and maintenance costs compared to biometric systems or key card technologies. Installing and configuring a PIN code access system is straightforward, making it suitable for various environments. It uses the logical link control protocol (LLCP) exchange and a timestamp to match the user's password data for access verification [8]. They are harder to pick or force open, reducing the risk of lost or stolen keys. PIN codes can be used alone or alongside other security measures like key cards or biometrics.

2.1.2 Smart Locks

Smart locks have become an essential component in modern security systems, combining advanced technology with enhanced usability to meet today's security needs. These locks offer a high level of security and convenience by allowing remote access control and utilizing features like facial recognition, fingerprint recognition, and Bluetooth communication [10], [11], [12]. It is one of the most secure systems because a person's biometrics never match others', making it highly resistant to forgery and duplication [13]. Furthermore, smart locks have seen innovations such as hand gesture recognition-based key management systems, providing a unique and secure way of accessing properties [14]. The implementation of smart locks enhances security by generating detailed access logs and simplifying permission management, making them beneficial for both residential and commercial applications.

2.1.3 Keypad Pin-Code

Keypad locks or electronic locks provide a convenient and secure way to control access without the need for physical keys. Instead, users enter a unique PIN code into a keypad to unlock the door. Implementation and maintenance costs are often lower compared to more advanced biometric systems or key card technologies. Installing and configuring a PIN code access system is relatively straightforward, making it suitable for various environments. They are more difficult to pick or force open and eliminate the risk of a lost or stolen key. PIN codes can be used as a standalone security measure, or they can be used in conjunction with other security measures,

such as key cards or biometrics. To enhance the security of keypad PIN codes, recent research has proposed various innovative approaches. In summary, while PIN code access systems offer simplicity, cost-effectiveness, and user-friendliness, they come with security considerations that must be carefully managed to ensure a balance between convenience and protection against unauthorized access.

2.2 Access Data Management Cloud

Access data management in the cloud has revolutionized how security systems handle and store data, offering a centralized and scalable solution for real-time monitoring and control. The integration of cloud computing with the Internet of Things (IoT) has garnered interest as well. Frameworks that merge cloud platforms with sensor networks have been created to manage and control real-world sensing devices in large-scale deployments effectively [14].

Public cloud computing environments, such as Amazon AWS, Microsoft Azure, and Google Cloud Platform, have achieved remarkable improvements in computational performance in recent years and are also expected to be able to perform massively parallel computing [14]. One significant area of research focuses on implementing access control mechanisms to safeguard sensitive data stored in the cloud. Cloud-based platforms, such as Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP) provide robust data storage capabilities, allowing secure access logs and system activities to be stored and retrieved efficiently.

2.2.1 Amazon Web Services (AWS)

AWS is a managed cloud service that enables connected devices to securely and reliably interact with cloud applications and other devices. Cloud platforms offer access control techniques as part of cloud services (e.g., AWS Identity and Access Management (IAM)) provided by them [15]. Services such as Amazon RDS provide a secure, scalable database solution for storing access logs and user data, ensuring data integrity and availability. AWS's real-time monitoring capabilities through Amazon CloudWatch allow for continuous oversight and quick response to security incidents. Overall, AWS's robust infrastructure and advanced security features make it a powerful platform for developing and maintaining secure, efficient access control systems.

2.2.2 Microsoft Azure

Azure Monitor is a cloud-based monitoring and analytics service provided by Microsoft Azure, designed to scale seamlessly with the growth of a cloud environment. It serves as an indispensable tool for monitoring and managing Azure deployments, offering a comprehensive toolkit for data collection, analysis, and visualization. Additionally, studies have highlighted the impact of process allocation strategies on high-performance cloud computing on the Azure platform, emphasizing the necessity for efficient resource allocation and management [16]. However, integrating Azure Monitor involves certain limitations. Web applications require a stable internet connection, which can be problematic in areas with unreliable connectivity. Web interfaces may also be less accessible from mobile devices compared to dedicated mobile apps, potentially hindering remote monitoring capabilities. Implementing Azure Monitor as the user interface requires significant development effort to extract and present relevant access control data, increasing development time and complexity. This is especially important if the system generates a large volume of security events and access data, necessitating careful budgetary planning for cost-effectiveness. Considering these factors is essential to effectively leverage the benefits of Azure Monitor.

2.2.3 Google Cloud Platform (GCP)

Google Cloud Platform (GCP) offers a robust framework for developing and managing sophisticated intruder access systems. With Google Cloud's Identity and Access Management (IAM), administrators can enforce detailed security policies and control user access to resources efficiently. The integration of Google Cloud Functions allows for real-time responses to access events, enhancing the system's agility in addressing potential security threats. The Google Cloud Platform access system is designed to meet the need of access control in the era of cloud computing by providing flexibility for supporting policies in multi-tenant environments, network-independence that decouples access control from the network, and scalability to handle hundreds of thousands of servers and users [17]. Additionally, Google Cloud's BigQuery provides a scalable solution for storing and analysing access logs, facilitating comprehensive security audits and data analysis. The real-time monitoring and alerting capabilities of Google Cloud Monitoring ensure that security incidents are promptly detected and addressed.

2.3 Intrusion Alarm Systems

The intrusion alarm system is a crucial element of security measures in various environments, designed to detect and respond to unauthorized access attempts effectively [18]. Nuisance alarms, often triggered by environmental or man-made sources, present a challenge in intrusion detection systems, underscoring the necessity for effective alarm suppression techniques [19]. Systems typically include sensors, alarms, and monitoring capabilities to ensure immediate detection and response to potential security breaches. Motion sensors detect

movement within a designated area and can trigger alarms if unexpected activity is detected. Mobile applications enable remote monitoring and control of the alarm system, allowing users to receive alerts and manage security settings from their smartphones. Wireless alarms provide flexibility in installation and can communicate seamlessly with other system components to notify security personnel and send alerts to designated mobile devices.

2.3.1 Motion Sensors

The motion sensor for intrusion alert is a critical component of security systems, designed to detect unauthorized access attempts effectively [20],[21]. These sensors work by detecting changes in the environment, such as motion or heat, and trigger alarms when unusual activity is sensed. This capability allows for immediate alerts to be sent to security personnel through integrated mobile applications, ensuring swift responses to potential security breaches.

2.3.2 Mobile Applications

Mobile applications leverage embedded sensors and wireless connectivity to empower users with portable computations and context-aware communication, making them ideal for receiving intrusion alerts on the go [22]. Third-party services play a crucial role in the mobile ecosystem, enabling features such as analytics and social network integration, which can enhance the functionality of intrusion alarm systems [23]. Mobile applications heavily rely on sensor inputs, making them well-suited for receiving and processing data from motion sensors in intrusion alarm systems. When an intrusion is detected, whether through motion sensors, door/window sensors, or other detection devices, the system sends an immediate alert to the user's mobile device. This instant communication enables users to take swift action, such as contacting security personnel, triggering additional alarms, or remotely accessing surveillance cameras to assess the situation. The integration of a motion sensor with a mobile application enhances the functionality of intrusion alarm systems by providing real-time alerts and notifications directly to users' smartphones [24]. The use of mobile apps in conjunction with motion sensors for intrusion alert systems represents a promising approach to enhancing security measures and ensuring timely responses to security incidents [25].

2.3.3 Wireless Alarm Systems

The wireless intrusion alarm system is a crucial component of security systems, leveraging wireless sensor network technology to detect unauthorized access attempts effectively. These systems are composed of sensor nodes, including door magnetic switches, infrared sensors, fog sensors, gas leakage sensors, and glass-breaking alarms, forming a robust wireless sensor network system. The wireless alarm system aims to generate intrusion alerts when analysed wireless traffic deviates from the normal utilization profile of wireless resources, ensuring timely detection of potential security breaches.

2.4 Comparison of the Existing System with the System

Table 1 shows the comparative analysis reveals distinctive features among four different security systems. The "Creating an Immutable Database for Secure Cloud Audit Trail and System Logging" system emphasizes a secure cloud-based approach with an immutable database, ensuring high data integrity and security. The "SmartHome" Fingerprint Security System Using Arduino" focuses on integrating with Smart Home systems, utilizing fingerprint-based authentication for enhanced security. The "Room Security System Design Using ESP32 CAM with Fuzzy Algorithm" implements a fuzzy algorithm for heightened security in room environments. The system stands out by being designed with Arduino Uno Wi-Fi Rev2 and a web app, offering a robust and secure solution for access control. This system combines a keypad PIN code for user authentication with an Arduino Uno Wi-Fi Rev2 for hardware communication. Data storage and management are handled through Firebase Cloud, while a user- friendly web application provides a central interface for system monitoring and administration. This combination of technologies allows for real-time monitoring capabilities, enabling immediate response to security threats.

Table 1 *Comparison on Existing System*

Features/System	Securing Database for Cloud Audit Trail and System Logging	"SmartHome" Fingerprint Security System Using Arduino	Room Security System Using ESP32 CAM with Fuzzy Algorithm	The proposed System
Authentication method	None	Fingerprint-based authentication	Facial recognition authentication	Keypad Pin-Code

Table 1 *(cont.)*

Features/System	Securing Database for Cloud Audit Trail and System Logging	"SmartHome" Fingerprint Security System Using Arduino	Room Security System Using ESP32 CAM with Fuzzy Algorithm	The proposed System
System approach	Web-Based Application	Arduino, Fingerprint Sensor	ESP32 CAM	Arduino Uno Wi-Fi Rev 2 and Web-Based Application
Notification/System alert	None	None	Telegram	Telegram
Application Area	Cloud Audit Trail and System Logging	Home	Room	Office Room
Database	MySQL	Not Specified	Not Specified	NoSQL
Cloud-Based Service	None	None	None	Google Firebase
Security	High but relies on cloud	Medium and may not be suitable for some environments	Highly secure but with limited coverage area	Highly secure with alert system and real- time monitoring

3. Methodology

This section outlines the methodology used to develop the system, relying on a systematic approach to accomplishing the project's goals. It describes all the necessary steps and information required to obtain the study's results. The subheadings will correspond to the different phases of the methodology used.

3.1 Prototyping Model

Prototyping is an effective way to reduce design errors and eliminate failure factors in the initial design phase. The process, which involves the test-refinement-completion of designs using prototypes, is known as "prototyping" [26]. The prototype model in software development consists of building, testing, and reworking a prototype until it meets quality standards and serves as the basis for the final software or system. This method is particularly useful when project requirements are not well-defined, as it is an iterative, trial-and-error process between the developer and client [27]. Prototyping was chosen for this project because it is ideal for scenarios where system needs are not well specified. The iterative nature of the model allows developers to refine the system if it fails to meet user expectations. Prototyping involves eight phases: requirement gathering, design, building the prototype, user evaluation, refining the prototype, developing the prototype, testing, and maintenance, as shown in Fig. 1. The following subsections will provide further details on each phase of the prototype model.

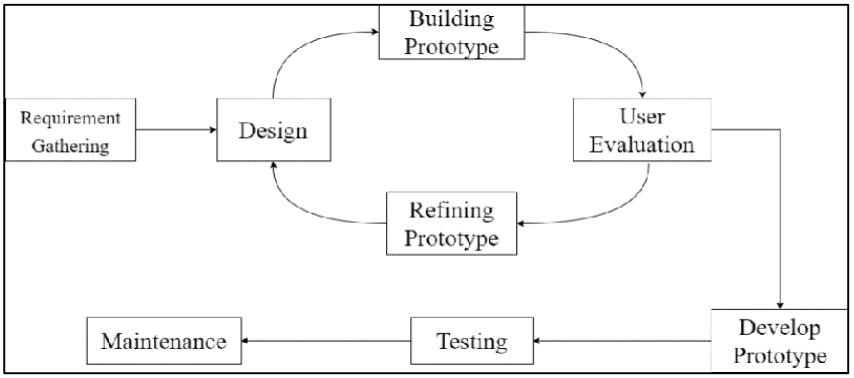


Fig. 1 *Prototyping Model Phases [28]*

Fig. 1 shows the prototype development methodology follows a phased approach, ensuring a structured and iterative process. This process is repeated until the prototype meets the needs of the users. Each phase plays a crucial role in developing the project. In the requirements gathering phase involves gathering and analysing requirements from users. Once the requirements have been gathered and analysed, a quick design of the prototype is created. This design should focus on the core features of the system and should be easy to understand for both technical and non-technical users. The prototype may be a simple mock-up or a fully functional system on the

building prototype phase. Based on the feedback from the initial user evaluation, the prototype is then refined. This may involve adding new features, changing existing features, or fixing bugs. The goal of this phase is to create a prototype that is more usable and meets the needs of the users. Once the prototype is finalized, the system is implemented based on the prototype. The system is then maintained and updated as needed. The implementation and maintenance phase may involve fixing bugs, adding new features, and improving the performance of the system.

3.2 Requirement and Gathering Phase

During the requirement gathering and analysis phase, staff members of the Syndicate Store and Service company undergo interviews about the system to obtain information, using a questionnaire to collect both qualitative and quantitative data. Examples of qualitative data include the procedure, data flow, database structure, operational and non-operational aspects of the manual access system, and the staff attributes that need to be entered into the database. Quantitative data will assist the system in producing automated messages such as intruder alerts and notifications of access via the Telegram application. This phase also addresses the hardware and software requirements for the Intruder Alert System with Firebase Cloud. Since the Prototyping Model is dynamic and iterative, these requirements are carefully considered and linked with the growing understanding of the stakeholder's needs. Feedback from stakeholders obtained throughout prototype iterations is crucial in defining these requirements. The hardware and software requirements are mentioned in the next paragraph.

The hardware requirements for the Intruder Alert System with Firebase Cloud were identified following the completion of the analysis phase. The system employs several essential hardware components to ensure its functionality and reliability. These components include a laptop for programming and monitoring, an Arduino board as the central processing unit, a buzzer for audible alerts, an LCD display for visual feedback, a keypad for user input, a servo for mechanical operations, and LEDs to indicate system status. Each of these components plays a critical role in the overall operation of the system, contributing to its effectiveness in providing security alerts and controlling access.

The software requirements for the project are equally important in supporting the system's functionality and performance. The development process utilizes the Arduino IDE for programming the Arduino board, and Visual Studio Code IDE for additional coding and integration tasks. XAMPP is used to set up a local server environment, facilitating database management and web interface development. The Telegram Bot is implemented to handle the notification system, providing real-time alerts to users. Google Firebase serves as the backbone for data storage and retrieval, offering a robust and scalable cloud-based solution for managing access logs and other critical information. These software tools collectively ensure the system operates smoothly, delivering the necessary security functions effectively.

3.3 Quick Design

During the design phase, the system architecture outlines the high-level structure, illustrating how components like the Arduino Uno Wi-Fi Rev2, keypad, Google Firebase, web application, and network infrastructure interact. The keypad inputs user PIN codes, which Arduino processes and verifies with Firebase, updating the real-time database accordingly. The web application retrieves this data to provide administrators with a comprehensive interface for monitoring and managing the system, while Firebase's cloud functions handle event-triggered alerts and notifications.

The user interface design aims to be intuitive and user-friendly, featuring an administrator dashboard with a secure login page, an overview home page, detailed access logs, user management capabilities, alert notifications, and various settings configurations. Wireframes illustrate the layout, with the login page providing secure authentication, the home page displaying system status and recent activity, and the access logs presenting sortable data entries. User management allows for adding, editing, or removing permissions, while alerts and notifications keep administrators informed of any critical events.

3.4 Build Prototype

Afterward, the focus shifts to developing a functional version of the system by implementing and integrating hardware and software components and conducting initial testing to ensure cohesive operation. The hardware implementation begins with setting up the Arduino Uno Wi-Fi Rev2 by connecting all the components correctly and ensuring a stable power supply. Network configuration follows, involving connecting the Arduino to the office Wi-Fi and implementing error handling to manage potential network disruptions. On the software side, the Arduino IDE is used to write and upload code that reads keypad inputs, connects to Firebase, and sends data for authentication and logging. Essential libraries for Wi-Fi, Firebase, and keypad handling are utilized to streamline development. Firebase configuration includes setting up the Realtime Database to store access logs and user data, implementing Firebase Authentication to secure user access, and developing Firebase Cloud Functions to automate responses to specific events, such as sending alerts for unauthorized access attempts.

The web application is developed using Visual Studio Code, integrating Firebase services for data retrieval and management. The administrator dashboard is designed to provide a user-friendly interface for monitoring access

logs, managing user permissions, and viewing system status, with real-time updates ensuring administrators are informed of the latest system activities. XAMPP is used to create a local development environment, hosting web applications and enabling testing of database interactions, while PHP scripts handle server-side operations. Integration and testing begin with ensuring the Arduino, keypad, and Firebase components are correctly integrated, testing communication between the Arduino and Firebase to verify PIN authentication and logging. The web application is also integrated with Firebase to retrieve and display data accurately. Functional testing verifies that each component operates as intended, followed by system testing to ensure seamless operation of all components together. By completing the Building Prototype Phase, the project transitions from design to a functional prototype, ensuring that the hardware and software components are correctly implemented and integrated, providing a working model of the intruder access system for further refinement and testing.

3.5 User Evaluation

The user evaluation phase begins with gathering feedback from end-users to assess the system's usability, functionality, and overall effectiveness. The target end-users, including staff members and other stakeholders in the office, are shown the prototype at this point. During the evaluation process, users are introduced to the system and provided with training to familiarize themselves with its features. They are then given the opportunity to interact with the system independently, performing tasks such as entering PIN codes, accessing logs, and managing users. Their interactions are observed, and feedback is collected through surveys, interviews, and observations.

Following data collection, feedback is analysed to identify areas for improvement, including usability issues, functionality gaps, performance optimizations, security concerns, and overall user satisfaction. A detailed report summarizing the evaluation process, participant feedback, and analysis results is prepared, highlighting key findings and actionable recommendations for system enhancements. An action plan is developed to prioritize and implement improvements, with scheduled follow-up evaluations to assess the effectiveness of implemented changes. Through this evaluation process, the User Evaluation Phase ensures that the intruder access system is refined based on real user feedback, resulting in a more user-friendly, functional, and secure solution that meets the needs and expectations of its intended users.

3.6 Refining Prototype

This phase begins with a thorough analysis of the feedback, identifying patterns and recurring themes among user responses. Based on this analysis, priorities are established, and an action plan is developed to address the identified areas for improvement. Usability enhancements focus on redesigning interface elements and optimizing workflows to improve user experience, while functionality improvements address reported bugs and incorporate additional features requested by users. Performance optimization involves refining code and conducting stress testing to ensure smooth system operation under varying conditions. Security enhancements are made to address any vulnerabilities identified during user testing, reinforcing data protection measures and access controls. After users evaluate the prototype through iterative testing and modifications, this phase ensures a more user-friendly final product, minimizes potential problems, and saves both time and resources in the long run. This process will continue until the system meets user requirements.

3.7 Implementation and Maintenance

Finally, in the testing and maintenance phase, the focus shifts to ensuring the reliability, performance, and security of the system through testing procedures and ongoing maintenance activities. This phase begins with various testing procedures, including functional testing to validate system features, integration testing to ensure seamless hardware and software integration, performance testing to assess responsiveness and scalability, and security testing to identify and mitigate vulnerabilities. This ongoing process ensures smooth system operation, maximizes its value, and allows for adaptation to user requirements.

4. System Analysis and Design

This section will cover the Analysis and Design phase. The focus is on understanding the system requirements and creating a detailed plan for its architecture and functionality. This phase is critical for defining system components, and establishing the overall workflow, ensuring a clear and coherent path from concept to implementation.

4.1 Functional Requirements

The functional requirements of the system are designed to ensure comprehensive and efficient access control within office environments. Table 2 shows the functional requirements of Intruder Alert System with Firebase Cloud. The system must allow users to authenticate themselves using a keypad to enter a PIN code, with the entered code being verified against user credentials stored in Firebase Authentication. For access control, the system must grant access by unlocking the door when a valid PIN is entered and deny access while logging failed attempts for unauthorized users. Real- time data synchronization is crucial, ensuring that access logs and user data are updated

instantaneously using Firebase Realtime Database, allowing all system components, including the Arduino, web app, and Firebase, to access the most current information. Access logging requires the system to record all events, including user identity, timestamps, and access status, storing this data for at least 90 days. Additionally, the system must generate alerts for unauthorized access attempts and notify administrators via Telegram application. A web application interface should be available for administrators to monitor access logs and system activities in real time, with functionalities to manage user permissions. For user management, administrators must be able to add, modify, or delete user credentials, ensuring only authorized personnel can perform these tasks.

Table 2 *Functional requirements*

No	Modules	Functionalities
1	User Access	Upon entering the PIN code, the system verifies its validity against the stored user credentials in the Firebase Authentication service.
2	Intruder Alert	The system is configured to generate alerts for specific events, such as repeated failed access attempts. These alerts are sent to designated administrators via Telegram application, enabling timely response and intervention.
3	User Registration	Integrate a step for administrator approval to verify and authorize user registrations, enhancing security and control over access.
4	Registration user ID	Automatically generate a unique registration ID for each user upon successful registration, aiding in identification and access management.
5	User Authentication	Users should be able to authenticate their identity through assigned PIN code using the keypad for authentication.
6	Application Notification	Administrators should receive instant alert notifications via Telegram in case of security breaches or suspicious activities.
7	System Monitoring	The system continuously monitors access events and system status in real-time, updating the access logs and system data stored in the Firebase Realtime Database accordingly.
8	Web-Based Application	The Administrator can access the web application interface to view real-time access logs, monitor system activity, and manage user permissions

4.2 Use Case Diagram

The use case diagram illustrates the various interactions between the administrator and the Intruder Alert System. It highlights the key functionalities and actions performed by the admin within the system. Fig. 2 shows the use case diagram.

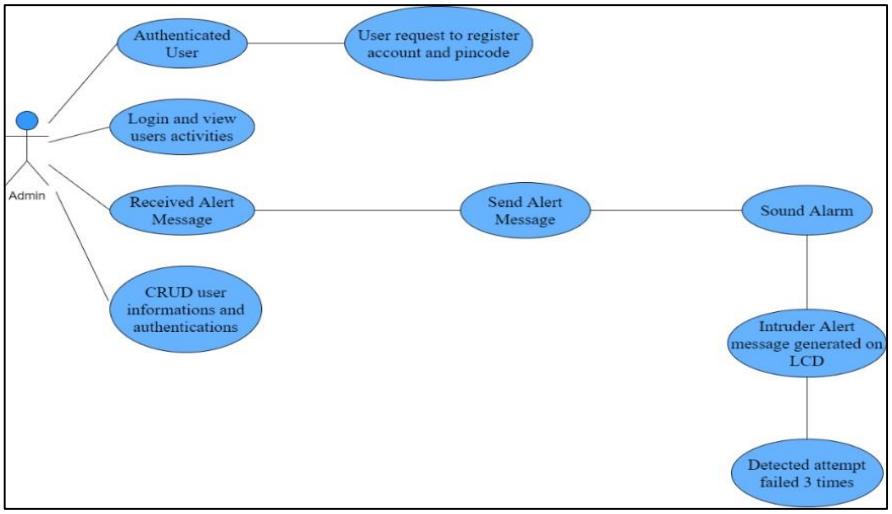


Fig. 2 *Use Case Diagram*

Fig. 2 is the use case diagram for the system that involves two primary actors: "User" and "Admin". Users can register for the system through a web app by creating an account and setting a PIN code. This PIN code is then stored securely in a cloud database like Firebase Realtime Database. Once registered, users can log in by entering their PIN code on the keypad. The system verifies the PIN code against the stored credentials, granting access or denying it based on the verification result. Valid PIN codes grant access and might provide feedback through an LED

or a message on the web app. Admin can manage user accounts using CRUD operations (Create, Read, Update, Delete). This allows them to add new users, view existing users, modify user information, and potentially remove users from the system. Additionally, they can manage user authentication credentials (likely PIN codes) through this function. Moreover, an "Intruder Alert Message Generated on LCD" functionality. This indicates the system's potential to display an alert on an LCD screen in case of unauthorized access attempts (multiple failed PIN entries) and will send notification to Telegram Application and store the intruder access details into Firebase Cloud.

4.3 Class Diagram

The class diagram provides a detailed view of the structural components of the Intruder Alert System, outlining the key classes and their relationships. It features four main classes: User, Access Control, Intruder Alert, and Notification System. Each class contains its attributes and methods, illustrating their responsibilities within the system. Fig. 3 shows the class diagram of the system.

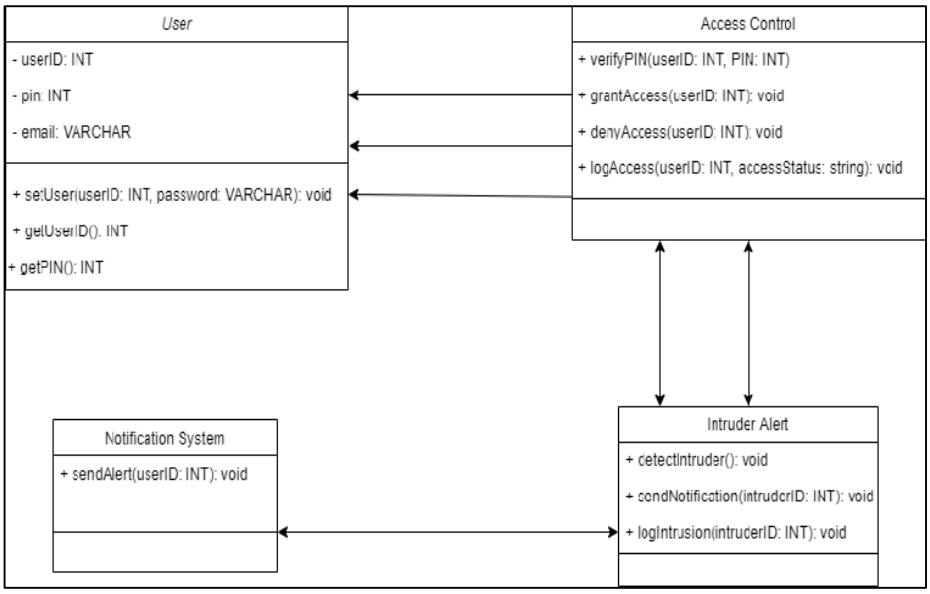


Fig. 3 *Class Diagram*

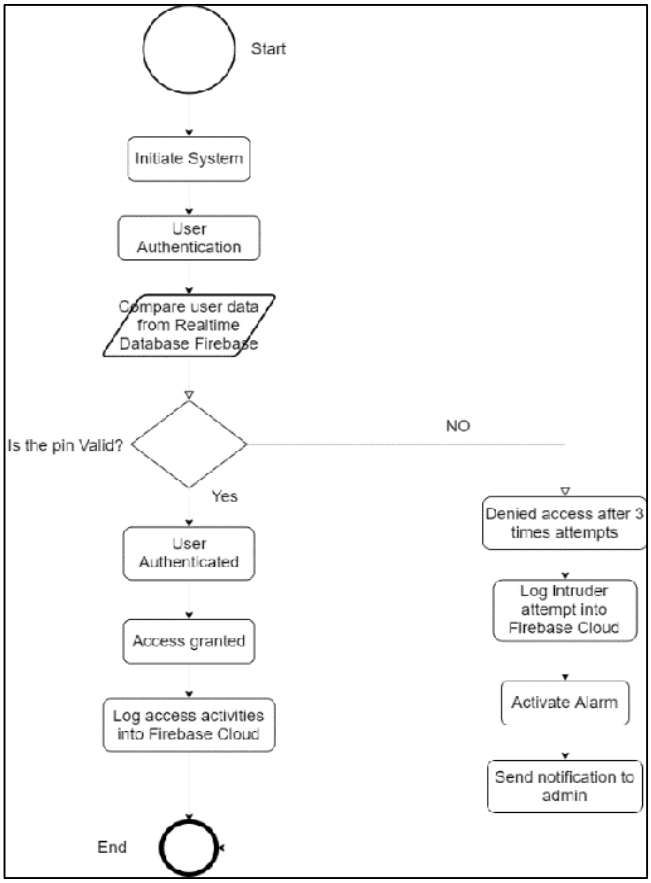
There are several entities that play distinct roles in facilitating access control and intruder alert functionalities. Firstly, the AccessControl class depends on the User class for user authentication, indicating that it relies on user information to verify access. Similarly, the IntruderAlert class also relies on the User class to log intrusion attempts and send notifications, showcasing its dependency on user data for security monitoring. Additionally, the NotificationSystem class utilizes the User class to send alerts, emphasizing its reliance on user information for effective communication. Furthermore, the IntruderAlert class collaborates with the AccessControl class to detect intrusions and log intrusion attempts, indicating a mutual interaction between security-related functionalities. Lastly, the NotificationSystem class interacts with the IntruderAlert class to send alerts to administrators, highlighting the interconnectedness of security monitoring and alerting mechanisms. These relationships underscore the interconnected nature of the system components, ensuring seamless operation and robust security management.

4.4 Flowchart Diagram

The flowchart illustrates the user authentication process within the Intruder Alert System, detailing each step from system initiation to access logging. Fig. 4 is a flowchart diagram for the system. It begins with user authentication, where users input their PIN on the keypad. Subsequently, the system verifies the validity of the entered PIN. If the PIN is deemed valid, the system proceeds to access control. Here, it checks whether the access attempt is successful. If successful, the door unlocks via the servo motor, access is logged, and Firebase is updated with the access status. Concurrently, real-time monitoring updates Firebase with access log data, which is also displayed on the web application. Conversely, if the PIN is invalid, the access attempt is logged, the user is notified of access denial, and an alert is sent to the administrator via the notification system. Administrators have the capability to manage user credentials, including addition, modification, or deletion, all reflected in Firebase. In case of unauthorized access attempts or system anomalies, alerts are dispatched to administrators via Telegram. Administrators utilize the web application interface to monitor access logs, system activity, and manage user

permissions. The system also conducts regular backups of access logs and user data, alongside routine maintenance and updates for optimal performance. Ultimately, the process concludes upon completion.

Fig. 4 *Flowchart Diagram*



4.5 Data Dictionary

The data dictionary serves as a comprehensive reference for understanding the structure, attributes, and relationships of the data elements used within the Intruder Alert System. It provides detailed descriptions of each data entity, including field names, data types, constraints, and any relationships to other data entities. Table 3 and Table 4 will provide the user table and intruder table.

Table 3 *User table*

No	Field name	Data type	Required	Unique	PK/FK	Ref. table
1	UserId	Int	True	True	PK	User
2	Email	Varchar	True	True	-	-
3	RegDate	TIMESTAMP	True	True	-	-
4	Pincode	Int	True	False	-	-

Table 3 shows the User Table stores information about the users registered in the system. This table includes the following fields: UserId, which is an integer that serves as the primary key and uniquely identifies each user; Email, which is a varchar data type and must be unique and is required for user identification and communication; RegDate, a timestamp that records the date and time of user registration, ensuring that each entry is unique and required; and Pincode, an integer used for user authentication, which is required but does not need to be unique. The UserId field serves as the primary key (PK) for this table.

Table 4 *Intruder table*

No	Field name	Data type	Required	Unique	PK/FK	Ref. table
1	IntruderID	Int	True	True	PK	-
2	IntruderDateTime	TIMESTAMP	True	False	-	-
3	Status	Bool	True	False	-	-

Table 4 shows the Intruder Table logs information about potential security breaches or unauthorized access attempts. This table includes the following fields: IntruderID, which is an integer that uniquely identifies each intruder incident and serves as the primary key; IntruderDateTime, a timestamp that records the date and time of the intruder alert, ensuring it is required but not unique; and Status, a boolean value that indicates the current status of the intruder alert (e.g., verified or not verified), which is also required. This table does not have any foreign key references to other tables.

4.6 Logical Design

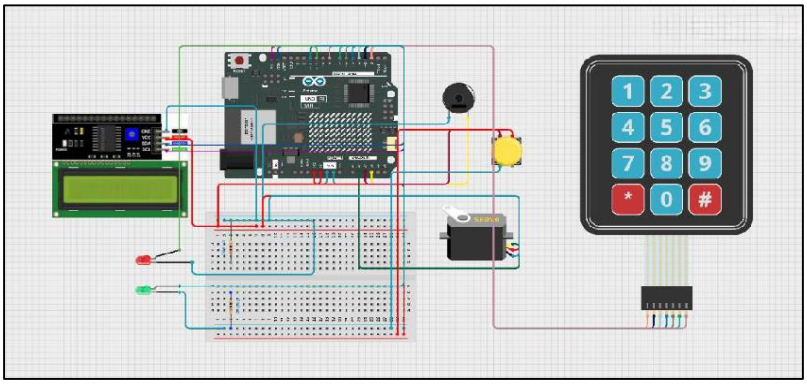


Fig. 5 *Logical Design of the System*

The logical design illustrated in Fig. 5, the logical design of the Arduino system integrates various components to create a versatile and functional access control mechanism. At the core of the system lies the Arduino Uno Wi- Fi Rev2, serving as the central processing unit and facilitating communication with other devices. The 4x3 Keypad Membrane Matrix enables users to input their access credentials securely, providing a user-friendly interface for PIN entry. Once the credentials are entered, the Arduino processes the input using its onboard logic, verifying the authenticity of the access attempt. Simultaneously, the I2C 16x2 LCD Display provides visual feedback to users, displaying system status and access prompts in real-time. Upon successful authentication, the 9g SG90 Servo motor triggers the physical unlocking of the door, ensuring seamless access control. Additionally, the system incorporates a 5V Piezo Buzzer to provide audible feedback on access status, alerting users of successful or denied access attempts. Furthermore, the inclusion of green and red LEDs enhances the visual indication of system status, with the green LED signaling successful access and the red LED indicating access denial. Finally, the Arduino Push Button Module B3F adds an additional input option for system administrators, facilitating manual control or override functionalities if required. Together, these components form a robust and efficient access control system, offering both security and user convenience in various environments, such as offices or restricted areas.

5. Implementation and Testing

This section focuses on developing the hardware and software components, integrating them, and conducting rigorous testing to validate their performance. Implementation involves writing and deploying code, configuring hardware, and setting up network connections. Once the system is operational, various testing procedures are carried out, including functional testing to ensure all features work correctly, integration testing to verify seamless interaction between components, performance testing to assess system responsiveness, and security testing to identify and address vulnerabilities.

5.1 User Registration Module

The user registration module is designed to streamline the process of new users creating their accounts on the system. This module allows users to create accounts by providing essential information such as their email address and a secure password.

Step 1:	Display the User Registration Form with fields for email.
Step 2:	User enters their email address in the provided input field.
Step 3:	Provide a "Send OTP to Email" button to initiate the registration process
Step 4:	Check if the entered email ends with '@gmail.com'. If not, display an error message asking the user to enter a valid Gmail address.
Step 5:	Use the fetch API to send a POST request to send_otp_email.php with the user's email and the generated OTP.
Step 6:	Generate a 6-digit OTP (One-Time Password).
Step 7:	If the OTP is sent successfully, redirect the user to the OTP verification page (user_verify_otp.php)
Step 8:	User enters the OTP received via email in the provided input field.
Step 9:	If the OTP is valid, push the user's email and verification status to the pending_registrations node in Firebase Realtime Database.
Step 10:	If the database update is successful, display a success message and redirect the user to the login page (user_login.php).

Fig. 6 *Algorithm of User Registration for Create account on web application*

Fig. 6 shows the algorithm of the user registration process. The user registration process begins by displaying a form with a field for the email address. The user enters their email address into this field and clicks the "Send OTP to Email" button to initiate the registration process. The system checks if the entered email ends with '@gmail.com'. If not, it displays an error message asking the user to enter a valid Gmail address. Upon a valid email entry, the system generates a 6-digit OTP (One-Time Password) and uses the fetch API to send a POST request to send_otp_email.php with the user's email and the generated OTP. If the OTP is sent successfully, the user is redirected to the OTP verification page (user_verify_otp.php). On this page, the user enters the OTP received via email in the provided input field. The system then validates the OTP. If the OTP is valid, it pushes the user's email and verification status to the pending_registrations node in the Firebase Realtime Database. If the database update is successful, a successful message is displayed, and the user is redirected to the login page (user_login.php).

Step 1:	Check if the user is logged in by verifying the session variable user_logged_in.
Step 2:	Retrieve the user email and Firebase-generated ID (user_key) from the session.
Step 3:	Provide a "Request OTP" button and an OTP input field for OTP verification.
Step 4:	Store the OTP request details (email, user ID, request time) in Firebase under otp_requests/{userId}.
Step 5:	Check if an OTP has already been generated for the user by querying otp_generated/{userId} in Firebase.
Step 6:	Users enters the OTP received from Administrator
Step 7:	If the OTP is valid, display the page with a form to enter a 4-digit pincode.
Step 8:	Validate that the pincode is a 4-digit number using a regular expression.
Step 9:	Set the pincode and other user details (email, user ID, enroll registered time, status) in Firebase under registered_pincode/{userId}.

Fig. 7 *User Registration Module of Create Pin Code for Access System on web application*

Fig. 7 shows the process begins by checking if the user is logged in by verifying the session variable user_logged_in. If logged in, the system retrieves the user email and Firebase-generated ID (user_key) from the session. The user is provided with a "Request OTP" button and an OTP input field for OTP verification. When the user requests an OTP, the system stores the OTP request details (email, user ID, request time) in Firebase under otp_requests/{userId}. The system checks if an OTP has already been generated for the user by querying otp_generated/{userId} in Firebase. The user then enters the OTP received from the administrator. If the OTP is valid, the system displays a page with a form to enter a 4-digit pincode. The entered pincode is validated to ensure it is a 4-digit number using a regular expression. Finally, the pincode and other user details (email, user ID, enroll registered time, status) are set in Firebase under registered_pincode/{userId}. The OTP is sent to the administrator rather than via email for security reasons. This approach ensures that only authorized personnel (administrators) can generate and distribute OTPs, reducing the risk of unauthorized access attempts. It provides an additional layer of security by involving a human element in the verification process, making it harder for potential intruders to intercept or manipulate the OTP. This method ensures tighter control over the distribution of access codes and enhances the overall security of the system.

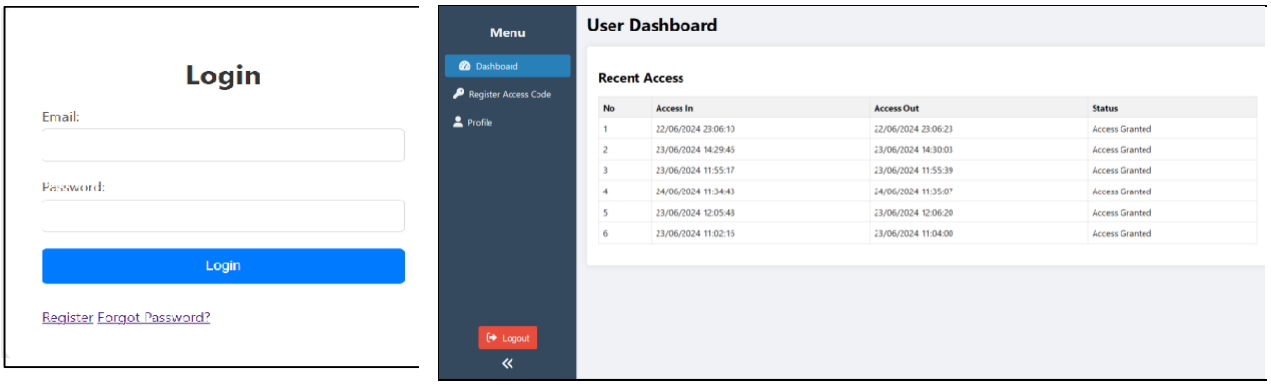
5.2 User Login Module

The user login module is a critical component of the intruder alert system, designed to manage user authentication and access control efficiently.

Step 1:	Present the login form with fields for email and password.
Step 2:	Include a "Login" button for form submission.
Step 3:	Retrieve the email and password entered by the user.
Step 4:	Query the Firebase Realtime Database to find a user with the matching email.
Step 5:	Iterate through the results to get user data and user key.
Step 6:	Check if the password entered matches the stored password
Step 7:	If it matches, store the user key and email in sessionStorage.
Step 8:	Send a POST request to user_session.php to set the session variables.
Step 9:	If the email or password is incorrect, display an error message.
Step 10:	If there is an error fetching user data from Firebase, log the error and display a generic error message.

Fig. 8 *User Login Module on web application*

Fig. 8 shows the user login process begins by checking if the user is already logged in by verifying the session variable user_logged_in. If the user is logged in, they are redirected to the dashboard. If not, the login form is displayed, prompting the user to enter their email and password. The Firebase configuration is initialized with the provided credentials. An event listener is added to the "Login" button to handle form submission, preventing the default behavior. The system retrieves the email and password entered by the user and queries the Firebase Realtime Database to find a user with the matching email. It then iterates through the results to get the user data and user key. The system checks if the entered password matches the stored temporary password; if it does, the user key and email are stored in sessionStorage, and the user is redirected to create_new_password.php. If the password matches the stored password, the user key and email are stored in sessionStorage, and a POST request is sent to user_session.php to set the session variables. If the session setup is successful, the user is redirected to the dashboard; if not, an error message is displayed. If the email or password is incorrect, an error message is shown, and any errors fetching user data from Firebase are logged, with a generic error message displayed to the user as displays in Fig. 9 (a) and (b).



(a) **(b)**

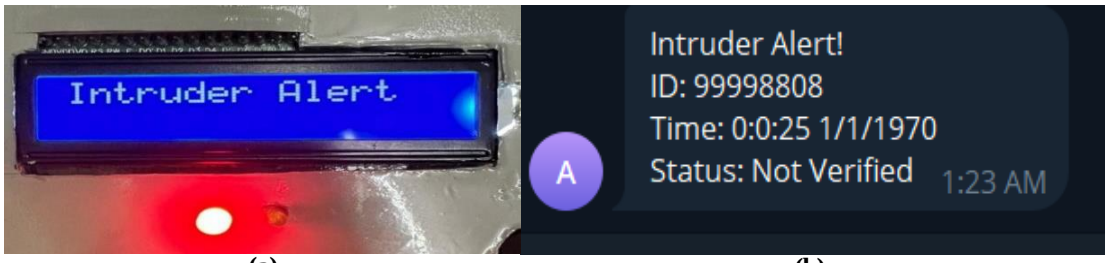
Fig. 9 *(a) User Login Page (b) User Dashboard Page*

5.3 Intruder Alert Module

The intruder module is a critical component of the security system designed to enhance the protection of premises against unauthorized access. Fig. 10 shows the intruder alert process starts by increasing the count of failed attempts each time an incorrect PIN code is entered. When the attempt count reaches 3, the system triggers an intruder alert. It displays "Access Denied" and the remaining attempts on the LCD and turns on the red LED. The system generates a unique intruder ID, gets the current time, and stores the intruder data (ID, time, and status) in Firebase under intruders/{intruderId}. If the data storage is successful, it displays "Intruder Alert!" on the LCD, activates the buzzer, and sends an intruder alert notification to Telegram with the intruder details.

Step 1: Initialize the LCD and display "ACCESS DOOR".
Step 2: Wait for the user to press the '#' key to start the PIN code entry.
Step 3: LCD display "Enter Pincode:".
Step 4: Capture the user's input for a 4-digit PIN code.
Step 5: Retrieve the registered PIN codes from Firebase.
Step 6: If the pin code is not found from the Firebase, show "Access Denied" on the LCD.
Step 7: Activate the red LED to indicate a failed access attempt and display the number of attempts left on the LCD.
Step 8: Each time an incorrect PIN code is entered, increment the count of failed attempts.
Step 9: If the number of failed attempts reaches 3, proceed to trigger an intruder alert.
Step 10: Show "Intruder Alert" on the LCD.
Step 11: Turn on the buzzer to sound an alarm.
Step 12: Create a unique intruder ID by appending a random number to "9999".
Step 13: Store the intruder ID, intruder time, and status ("Not Verified") in Firebase under intruders/{intruderId}.
Step 14: Send an intruder alert notification to Telegram with the intruder details.

Fig. 10 *Intruder Alert Module*



(a) **(b)**

Fig. 11 *(a) Intruder Alert Message on LCD Display; (b) Intruder Notification on Telegram*

5.4 User Management Module

The User Management Module is an integral part of the admin dashboard, designed to allow administrators to efficiently manage user accounts by providing functionalities for viewing detailed user information, performing CRUD operations (Create, Read, Update, Delete), and monitoring user and unauthorized activities to ensure smooth and secure system operations.

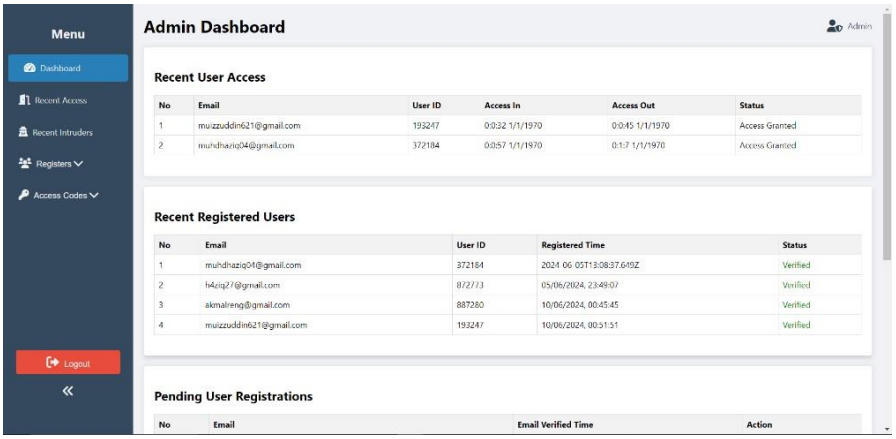


Fig. 12 *Admin Dashboard*

Fig. 12 shows the user management module within the admin dashboard which enables administrators to efficiently manage user accounts and monitor various user activities. Upon logging in, authenticated admins can access the dashboard through session validation, ensuring secure access. The sidebar navigation menu contains links to different sections such as Recent User Access, Recent Intruders, Registered Users, and Pending Registrations. The main content area displays several tables, including Recent User Access, which lists recent access attempts by users with details like Email, User ID, Access In, Access Out, and Status; Recent Registered Users, displaying newly registered users with their Email, User ID, Registration Time, and Status; and Pending User Registrations, listing users who have initiated the registration process but are yet to be verified, including details such as Email and Verified Time. The data for these tables is fetched from Firebase and updated in real- time, allowing admins to keep track of user activities and manage accounts effectively.

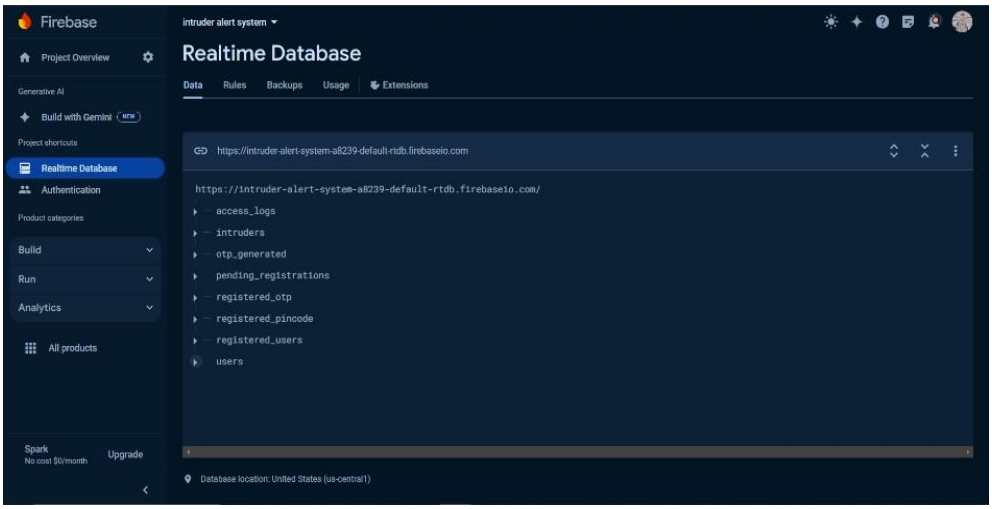


Fig. 13 *Intruder Alert System with Realtime Database Firebase*

Fig. 13 shows the user management module in the Firebase Realtime Database for the intruder alert system which is essential for handling all user-related data and functionalities. This module is organized into several key nodes: users, pending_registrations, registered_users, and registered_pincode, each serving a specific purpose in the user management workflow. The user's node contains detailed information about each registered user, including user IDs, email addresses, registration dates, and other necessary metadata for managing user accounts, ensuring that each user has a unique identifier and secure storage credentials. The pending_registrations node stores information of users who have initiated but have not completed the registration process, including email addresses and verification statuses. Once users verify their email and complete the registration process, their data moves to the registered_users node, which tracks active users and manages their access privileges. The registered_pincode node stores the PIN codes set by users for accessing secure areas, including user ID, email, pincode, and registration time, ensuring secure storage and verification of PIN codes during access attempts. Additionally, the access_logs node, while not directly part of user management, is crucial for tracking user activities by logging each access attempt with details such as user ID, access time, and attempt status (granted or denied). The otp_generated node handles the storage of one-time passwords (OTPs) generated for user authentication, which are essential for verifying user identities during registration and access processes. The intruder's node logs any unauthorized access attempts, including information about the attempt, its time, and status, vital for security monitoring and alerts. By organizing user data into these specific nodes, the Firebase Realtime Database ensures efficient management of user registration, authentication, and access control processes, with real-time capabilities enhancing the system's security and responsiveness.

5.5 Functional Testing

Functional testing of the system involves verifying that each feature of the software application operates in conformance with the requirement specifications, ensuring that the user registration, login, access control, and alert functionalities work as intended and seamlessly interact with Firebase for real-time data updates and notifications.

Table 5 *Functional testing for Admin*

No	Description	Expected Result	Actual Result
1	Admin able to login and logout to the system	Admin successfully logs into the system and logs out, ensuring session management functions correctly.	Pass
2	View Recent User Access	Admin should see a table with recent user access details (Email, User ID, Access In, Access Out, Status).	Pass
3	View Recent Registered Users	Admin should see a list of recently registered users with details (Email, User ID, Registration Time, Status).	Pass
4	View Pending User Registrations	Admin should see a list of users pending registration with details (Email, Verified Time).	Pass

Table 5 *(cont.)*

No	Description	Expected Result	Actual Result
5	View Pending Access Code Requests	Admin should see a list of pending access code requests with details (Email, User ID, Request OTP Time).	Pass
6	View Recent Intruders	Admin should see a list of recent intruders with details (Intruder ID, Intruder Time, Status).	Pass
7	Generate and View OTP for User Registration Requests	Admin should be able to generate and view OTP for pending user registration requests.	Pass
8	Received Alert Notification	Admin should receive alert notification from Telegram	Pass

Table 6 *Functional testing for User*

No	Description	Expected Result	Actual Result
1	Login with Temporary Password	User should be able to log in with a temporary password and be redirected to create a new password.	Pass
2	Login with Permanent Password	User should log in successfully with their permanent password and be redirected to the user dashboard.	Pass
3	User Registration	User should successfully register with their email and receive an OTP.	Pass
4	OTP Verification	User should be able to enter the received OTP and verify their registration.	Pass
5	Request OTP for Access Code	User should be able to request an OTP for registering their access code.	Pass
6	Submit OTP for Access Code	User should successfully submit the OTP and be redirected to set a pin code.	Pass
7	Register Pin code	User should be able to set a 4-digit pin code and have it stored in Firebase.	Pass
8	Reset Password	User should be able to request a password reset and successfully change their password via the link received.	Pass
9	View Dashboard	User should be able to see their dashboard with relevant details and options.	Pass

5.6 Test Plan

The test plan for the Intruder Alert System with Firebase Cloud aims to ensure the system's functionality, security, and overall performance. This test plan focuses on different aspects of the system, addressing various user roles and scenarios. The primary user roles for the Intruder Alert System are the admin and the general user.

Table 7 *Test Plan for Security*

No	Description	Expected Result	Actual Result
1	Verify user registration with email and OTP	User should be able to register and receive an OTP	Pass
2	Verify login with valid credentials	User and admin should log in successfully	Pass
3	Verify login with invalid credentials	User should receive an error message	Pass
4	Verify admin access control	Only admin users should access the admin dashboard	Pass
5	Verify role-based access control	Users should only access functionalities according to their roles	Pass
6	Verify data encryption in transit	Data should be encrypted using HTTPS during transmission	Pass

Table 7 *(cont.)*

No	Description	Expected Result	Actual Result
7	Verify data encryption at rest	Sensitive data should be encrypted in Firebase	Pass
8	Verify data integrity	Data should not be altered during transmission or storage	Pass
9	Verify intruder alert on multiple failed attempts	System should trigger an intruder alert and notify the admin via Telegram	Pass
10	Verify logging of unauthorized access attempts	Unauthorized access attempts should be logged in Firebase	Pass
11	Verify session fixation	Session IDs should be regenerated after login	Pass
12	Verify session timeout	User sessions should expire after a period of inactivity	Pass

6. Conclusion and Future Work

The development and implementation of the intruder alert system using Arduino and Firebase Realtime Database have successfully addressed the critical needs of the organization. The system provides a robust framework for real-time monitoring, user management, and secure access control, significantly enhancing the overall security posture. By leveraging advanced technologies and automated processes, the system has improved operational efficiency and reduced administrative burdens, while ensuring a high level of security for both personnel and assets. However, there are areas that can be further improved and expanded upon in future work. Integrating multi-factor authentication, enhancing data storage scalability, and developing a dedicated mobile application for administrators are just a few potential enhancements. These improvements will ensure that the system remains resilient, scalable, and user-friendly, adapting to the evolving security landscape and user needs. Conducting regular security audits and incorporating user feedback will also be crucial in refining the system and maintaining its effectiveness over time.

6.1 Result of the Develop System

The development of the intruder alert system has yielded significant improvements in both security and operational efficiency. The system successfully integrates real-time monitoring and alerting capabilities, providing administrators with immediate notifications of access attempts and potential security breaches. This has enhanced the responsiveness to security incidents, allowing for swift action to mitigate risks. The user- friendly interface of the admin dashboard has streamlined user management, making it easier for administrators to handle registrations, monitor access logs, and manage security alerts. The implementation of secure PIN code access has reduced the vulnerabilities associated with traditional physical keys, ensuring that only authorized personnel can gain entry. Additionally, the use of Firebase Realtime Database has provided a centralized and secure platform for data storage, enabling efficient retrieval and analysis of access logs and user activities. Automated processes, such as OTP verification and real-time notifications via Telegram, have further improved the efficiency of the system, reducing administrative overhead. Overall, the developed system has proven to be a robust, efficient, and user-friendly solution for managing and securing access, significantly enhancing the security posture of the organization.

6.2 Advantages of the Intruder Alert System with Firebase Cloud

The intruder alert system developed using Arduino and Firebase Realtime Database offers several significant advantages that enhance security, user management, and operational efficiency. Firstly, it provides real-time monitoring of access attempts and immediate alerts for any unauthorized access, ensuring prompt responses to security breaches and minimizing potential risks and damage. The user-friendly interface of both the admin dashboard and the user portal allows for easy management of user registrations, monitoring of access logs, and handling of security alerts, while users can smoothly navigate through registration and access processes. Enhanced security is achieved through the integration of a keypad access system with secure PIN codes, reducing the risk of unauthorized access compared to traditional physical keys or key cards. The use of OTP (One-Time Password) for user registration adds an extra layer of security, ensuring that only verified users gain access. Storing all user data, access logs, and security events in Firebase Realtime Database provides a centralized, secure, and scalable solution for data management, facilitating efficient data retrieval, analysis, and auditing. The system's flexibility and scalability allow it to be easily adapted to different environments and requirements, with new features and functionalities added without significant overhauls. Utilizing open-source hardware (Arduino) and cloud services (Firebase) makes the system cost-effective compared to traditional security systems, with lower implementation and maintenance costs being particularly beneficial for small to medium-sized businesses. Automated processes

such as real-time notifications via Telegram, user authentication, and access logging enhance operational efficiency, reducing administrative overhead and allowing staff to focus on more critical tasks. Additionally, comprehensive access logs provide detailed insights into user activities, allowing for thorough security audits and investigations, helping to identify potential security issues and take preventive measures. By integrating these features, the intruder alert system significantly improves the security posture of any organization, providing a robust, efficient, and user-friendly solution to manage and monitor access.

6.3 Disadvantages of the Intruder Alert System with Firebase Cloud

Despite its numerous advantages, the intruder alert system developed using Arduino and Firebase Realtime Database has some disadvantages. One significant drawback is the reliance on a stable internet connection for real-time updates and notifications. Any disruption in the internet service can lead to delays in alerts and access data synchronization, potentially compromising security. Additionally, while the system is cost-effective, the initial setup and configuration of hardware and software components may require technical expertise, which could be a challenge for non-technical users or small businesses without dedicated IT support. Another limitation is the reliance on a single platform (Firebase) for data storage and management, which might pose scalability and data redundancy concerns as the user base grows. The Arduino-based system also has limited memory storage for storing and fetching data; if the memory is exhausted, the system may freeze and auto-restart, leading to potential interruptions in monitoring. Furthermore, the system's security heavily depends on the robustness of the PIN codes and OTPs; if these are not managed properly, it could lead to vulnerabilities. Lastly, while automated processes enhance efficiency, they may also reduce human oversight, potentially allowing sophisticated intrusion attempts to go undetected if not complemented by regular manual audits and updates.

6.4 Future Work

Future work on the intruder alert system can focus on several key areas to enhance its functionality, security, and user experience. One primary area of improvement is the integration of multi-factor authentication (MFA) to further secure access, combining PIN codes with biometric verification methods such as fingerprint recognition. Additionally, expanding the system to support multiple data storage platforms could address scalability and redundancy concerns, ensuring the system remains robust as the user base grows. Developing a mobile application specifically for administrators could provide more convenient access to real-time alerts and system controls, improving responsiveness. Furthermore, regular updates and improvements to the user interface can ensure the system remains user-friendly and accessible to non-technical users. Lastly, conducting comprehensive security audits and incorporating feedback from users can help in continually refining the system, ensuring it meets evolving security standards and user needs effectively.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design****: M.H. Badderol, N.A. Abdullah;* ***data collection****: M.H. Badderol, N.A. Abdullah;* ***analysis and interpretation of results****: M.H. Badderol, N.A. Abdullah;* ***draft manuscript preparation****: H M.H. Badderol, N.A. Abdullah. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] A. BABYCH, T. Lypskyi, and A. RADIKOVA, "Modern Methods and Ancient Technologies of Restoration and Restoration of Shoes," *Herald of Khmelnytskyi National University*. 2022. doi: 10.31891/2307-5732-2022-311-4-204-208.

[2] F. Wahyu Wibowo and M. Habib, "A LOW-COST ENTRY DOOR USING DATABASE BASED ON RFID AND MICROCONTROLLER," vol. 12, no. 17, 2017, [Online]. Available: www.arpnjournals.com

[3] B. Qi, P. Lougovski, R. C. Pooser, W. P. Grice, and M. Bobrek, "Generating the Local Oscillator 'Locally' in Continuous-Variable Quantum Key Distribution Based on Coherent Detection," *Physical Review X*. 2015. doi: 10.1103/physrevx.5.041009.

[4] S.-J. Song and H. Nam, "Visible Light Identification System for Smart Door Lock Application With Small Area Outdoor Interface," *Current Optics and Photonics*. 2017. doi: 10.3807/copp.2017.1.2.090.

[5] M. Derbali, "Toward Secure Door Lock System: Development IoT Smart Door Lock Device." 2023. doi:

10.22541/au.168055385.54003954/v1.
[6] S. Bhowmick, A. R. Kumar, and M. Sangworasil, "Fingerprint Door Lock Using Arduino UNO R3," *International Journal for Multidisciplinary Research*. 2023. doi: 10.36948/ijfmr.2023.v05i03.3819
[7] S. M. Abdullah, "Design Secured Smart Door Lock Based on Jaro Winkler Algorithm," *Tikrit Journal of Pure Science*. 2023. doi: 10.25130/tjps.v21i6.1095.
[8] O. A. Simon, U. I. Bature, K. I. Jahun, and N. M. Tahir, "Electronic doorbell system using keypad and GSM," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 9, no. 3, p. 212, Dec. 2020, doi: 10.11591/ijict.v9i3.pp212-220.
[9] M.K.Islam and A. Rajee, "FINGERPRINT DOOR LOCK USING ARDUINO," 2022, doi: 10.13140/RG.2.2.30005.04329.
[10] M. A. K, "Access Control Using Facial Recognition," *Interantional Journal of Scientific Research in Engineering and Management*. 2024. doi: 10.55041/ijsrem30467.
[11] S. Jeong, "Design on Novel Door Lock Using Minimizing Physical Exposure and Fingerprint Recognition Technology," *Joiv International Journal on Informatics Visualization*. 2022. doi: 10.30630/joiv.6.1.858.
[12] P. Zhang, Y. Li, and H. Chi, "An Elliptic Curve Signcryption Scheme and Its Application," *Wireless Communications and Mobile Computing*. 2022. doi: 10.1155/2022/7499836.
[13] S. Mamonov and R. Benbunan-Fich, "Unlocking the Smart Home: Exploring Key Factors Affecting the Smart Lock Adoption Intention," *Information Technology and People*. 2020. doi: 10.1108/itp-07-2019-0357.
[14] A. Botta, W. de Donato, V. Persico, and A. Pescapè, "Integration of Cloud Computing and Internet of Things: A Survey," *Future Generation Computer Systems*. 2016. doi: 10.1016/j.future.2015.09.021.
[15] H. A. Hassan, A. I. Maiyza, and W. M. Sheta, "Impact of Process Allocation Strategies in High Performance Cloud Computing on Azure Platform," *Scalable Computing Practice and Experience*. 2017. doi: 10.12694/scpe.v18i2.1288.
[16] A. Sankaran, P. K. Datta, and A. Bates, "Workflow Integration Alleviates Identity and Access Management in Serverless Computing." 2020. doi: 10.1145/3427228.3427665.
[17] M. Ohue, K. Aoyama, and Y. Akiyama, "High-Performance Cloud Computing for Exhaustive Protein-Protein Docking." 2020. doi: 10.48550/arxiv.2006.08905.
[18] Y. Duan and K. Han, "Research on Access Control Security in Cloud Computing Environment," *Destech Transactions on Computer Science and Engineering*. 2017. doi: 10.12783/dtcse/cst2017/12596.
[19] T. Vaiyapuri and A. Binbusayyis, "Deep Self-Taught Learning Framework for Intrusion Detection in Cloud Computing Environment," *Iaes International Journal of Artificial Intelligence (Ij-Ai)*. 2024. doi: 10.11591/ijai.v13.i1.pp747-755.
[20] S. A. Mahmoud, Y. S. Visagathilagar, and J. Katsifolis, "Nuisance Alarm Suppression Techniques for Fibre-Optic Intrusion Detection Systems." 2012. doi: 10.1117/12.915950.
[21] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and Survey of Collaborative Intrusion Detection," *Acm Computing Surveys*. 2015. doi: 10.1145/2716260.
[22] M. A. Baballe, "Detection of Crossed Walls Security Alarm System Against Invasion," *Review of Computer Engineering Research*. 2021. doi: 10.18488/journal.76.2021.81.14.26.
[23] B. Aljedaani, A. Ahmad, M. Zahedi, and M. A. Babar, "An Empirical Study on Developing Secure Mobile Health Apps: The Developers' Perspective." 2020. doi: 10.1109/apsec51365.2020.00029.
[24] A. Razaghpanah *et al.*, "Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem." 2018. doi: 10.14722/ndss.2018.23353.
[25] G. He, B. Xu, L. Zhang, and H. Zhu, "Mobile App Identification for Encrypted Network Flows by Traffic Correlation," *International Journal of Distributed Sensor Networks*. 2018. doi: 10.1177/1550147718817292.
[26] D. Patel, "Comments on 'An Agile-Based Integrated Framework for Mobile Application Development Considering Ilities,'" *Ieee Access*. 2020. doi: 10.1109/access.2020.3005376.
[27] D. Divyanshu, R. Kumar, D. Khan, S. Amara, and Y. Massoud, "Logic Locking Using Emerging 2t/3t Magnetic Tunnel Junctions for Hardware Security," *Ieee Access*. 2022. doi: 10.1109/access.2022.3208650.
[28] O. A. Simon, U. I. Bature, K. I. Jahun, and N. M. Tahir, "Electronic doorbell system using keypad and GSM," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 9, no. 3, p. 212, Dec. 2020, doi: 10.11591/ijict.v9i3.pp212-220.