International Journal Software Engineering and Computer Science (IJSECS)
5 (2), 2025, 842-852
Published Online August 2025 in IJSECS (http://www.journal.lembagakita.org/index.php/ijsecs)
P-ISSN: 2776-4869, E-ISSN: 2776-3242. DOI: https://doi.org/10.35870/ijsecs.v5i2.4898.

RESEARCH ARTICLE **Open Access**

Smart Door Lock System: A Mobile Application Development Based on IoT Technology

Muhammad A. Hanifullah *
Department of Informatics, Faculty of Industrial Technology, Universitas Islam Indonesia, Sleman Regency, Special Region of Yogyakarta Province, Indonesia.
Corresponding Email: 21523159@students.uii.ac.id.

Kurniawan D. Irianto
Department of Informatics, Faculty of Industrial Technology, Universitas Islam Indonesia, Sleman Regency, Special Region of Yogyakarta Province, Indonesia.
Email: k.d.irianto@uii.ac.id.

Received: July 11, 2025; Accepted: July 25, 2025; Published: August 1, 2025.

Abstract: The research introduces a miniature IoT-based smart door lock system paired with a custom Android application called Jenaela. The system tackles security vulnerabilities inherent in conventional pin tumbler locks, which remain susceptible to lock-picking techniques and provide restricted access control capabilities. The hardware architecture incorporates an ESP32 DOIT V1 microcontroller, PN532 NFC module, relay with optocoupler integration, DC-to-DC step-up converter, and 12V solenoid lock. The software solution features Jenaela, developed using TypeScript and React Native framework, deployed through Expo toolkit and EAS Build Preview. Rather than relying on third-party platforms like Blynk, Jenaela delivers complete customization and superior control mechanisms. Core functionalities encompass user authentication, RFID tag registration and deletion, LAN-based device discovery, and unlocking through both RFID and remote switch methods. Functional testing via Black Box methodology verified proper operation of all features. RFID unlocking demonstrated faster response times compared to WAN-based unlocking due to reduced system dependencies. Results validate the practicality and effectiveness of integrating IoT hardware with custom mobile applications to enhance security through dual access methods and real-time database interaction.

Keywords: Internet of Things (IoT); Smart Door Lock; Android Application; ESP32 RFID; Firebase Real-time Database.

1. Introduction

Pin tumbler systems dominate the door locking market worldwide. The mechanism works by aligning key pins and driver pins within the cylinder to release the lock [1]. While people widely accept and trust the system, pin tumbler locks suffer from serious security flaws. Lock-picking and forced entry remain constant threats, leading to burglary rates that account for 87.19% of all criminal activity [2][3]. Traditional locks also depend on a single key, creating problems when users lose their only means of access. Meanwhile, Internet of Things (IoT) technology continues advancing rapidly, creating fresh opportunities for home security innovation. Smart doors now feature multiple layers of protection through various authentication methods—RFID cards, biometric sensors, digital keypads, and facial recognition systems work together to strengthen security even in harsh conditions [4][5]. Multi-layered security systems can cut burglary risks by up to 20 times compared to basic or unprotected setups [6].

The research team built a compact IoT-based smart door security system using several hardware units: ESP32 DOIT V1, PN532 Elechouse V3, relay, and solenoid lock. The ESP32 microcontroller handles all input and output operations for the smart door hardware. The PN532 Elechouse V3 NFC module reads RFID cards, while the relay switches the solenoid lock between open and closed states. Programming for the ESP32 uses C++ language through Arduino IDE. Previous research shows mixed approaches to IoT application development. Some teams created standalone Android applications for smart doors with keypads, data monitoring, and water management [7][8][9]. However, many IoT projects still rely on third-party platforms like Blynk. Ravian and D. Irianto used Blynk for visitor monitoring systems in community health centers [10]. Sun *et al*. employed Blynk for smart door locks [3], while other teams used it for elderly room temperature control [11], double sensor smart doors [12], and smart lighting systems [13]. Blynk allows users to prototype, deploy, and remotely control devices through mobile, web, and cloud platforms without manual programming [14].

The research team chose not to use Blynk for controlling and monitoring the smart door lock. Custom-built applications offer unlimited customization compared to third-party platforms like Blynk. React Native served as the development framework, chosen for its growing popularity among developers [15]. TypeScript handled the programming language duties for the Android application. Expo provided the toolkit and platform to simplify development and build processes. Visual Studio Code functioned as the integrated development environment throughout the entire application creation process.

2. Related Work

Smart door locking systems have evolved rapidly alongside Internet of Things technology, with researchers worldwide developing various approaches to enhance home security through digital innovation. Most previous studies rely heavily on third-party platforms like Blynk for device control and monitoring. Ravian and Irianto developed an IoT-based visitor monitoring system for community health centers, using Blynk as their primary interface between hardware and mobile applications. Sun *et al*. (2021) created a smart door lock where users could unlock doors remotely through mobile devices connected to Blynk's cloud services [21]. Gamma *et al*. (2025) took a similar route, managing access control via RFID integration while using Firebase for data logging, all orchestrated through Blynk's platform [12]. While Blynk offers undeniable convenience for prototyping and small-scale applications, the platform creates several bottlenecks for developers seeking greater control. Customization options remain restricted to predefined UI widgets and features, preventing teams from adapting systems to specific user requirements or unique deployment scenarios. The platform essentially forces developers into a one-size-fits-all approach that may not suit every project's needs.

Some research teams have attempted to break free from platform dependency by exploring native application development using frameworks like React Native, paired with custom backend services and Firebase for data synchronization. The approach grants developers complete authority over application behavior, user interface design, and system integration processes. However, most studies still incorporate third-party IoT dashboards rather than building entirely independent systems from scratch. Our research takes a different path by developing a miniature IoT-based smart door lock that operates entirely through custom-built software. The system uses React Native and TypeScript to power an Android application called Jenaela, which offers unlimited customization potential while integrating seamlessly with Firebase Realtime Database. The hardware setup includes an ESP32 DOIT V1 microcontroller, PN532 NFC reader, relay module, and solenoid lock, all managed through our self-developed application interface.

The study makes three key advances in smart door lock technology. First, we created a standalone Android application that switches dynamically between local network control and cloud-based operation, giving users flexibility in different network environments. Second, our custom-built approach demonstrates superior adaptability compared to platform-dependent solutions, allowing developers to modify any aspect of the system without external restrictions. Third, we implemented a compact yet reliable hardware-software combination that handles real-time database operations and RFID authentication without requiring third-party IoT platforms. The research addresses a critical gap in current smart home technology where vendor lock-in and limited customization options prevent long-term system evolution. Our developer-controlled framework can expand or adapt to meet specific deployment requirements, making it particularly valuable for real-world applications where flexibility determines success or failure. Rather than accepting the constraints of existing platforms, we built a foundation that grows with user needs and technological advances.

3. Research Method

We chose the prototyping method to develop our IoT-based smart door security system and Android application. The iterative nature of prototyping allows us to continuously refine and improve our design through multiple development cycles until we achieve a fully functional system [16].

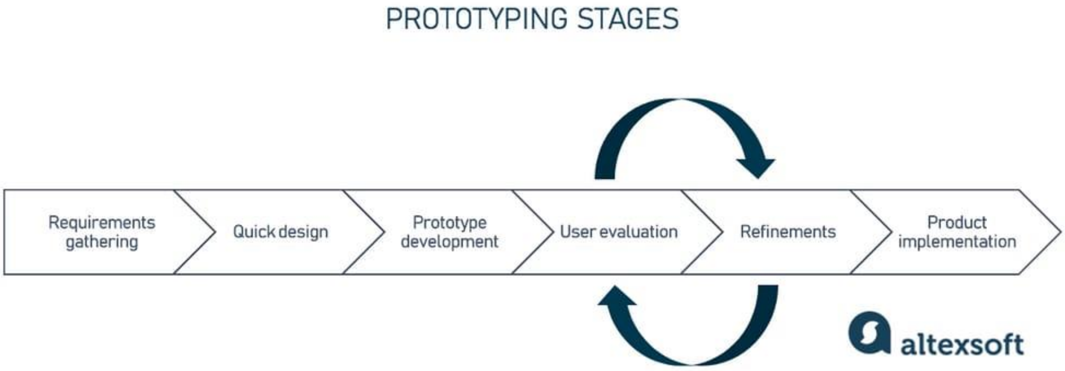


Figure 1. Prototyping Method Stages

3.1 Requirement Gathering

Our development process began with thorough requirement gathering to identify all necessary system components and user needs. We focused on understanding both technical requirements—such as hardware specifications for the ESP32, RFID module, and relay systems—and user experience expectations for the Android application and Firebase integration. We collected data through three primary methods. First, we conducted interviews with people from various backgrounds including students, working professionals, and homemakers to capture diverse perspectives on what users expect from a smart door lock system. Second, we observed real-world environments where such systems would operate, paying attention to factors like lighting conditions, user behavior patterns, and physical constraints. Third, we reviewed existing literature to strengthen our technical foundation and learn from similar research projects [17].

3.2 Quick Design

After gathering requirements, we moved into rapid design creation. Rather than spending extensive time on detailed technical specifications, we focused on sketching basic system layouts and user interaction flows by hand. The approach helped us visualize the overall system architecture and identify potential design challenges early in the process without getting bogged down in implementation details.

3.3 Prototype Development

The prototype development phase brought our designs to life through actual implementation. We built the ESP32-based control system that manages door lock operations and reads RFID card identifiers. Firebase Realtime Database was integrated to enable seamless communication between our mobile application and the physical hardware. The Android application was developed using React Native framework, incorporating all necessary user interface elements and system controls. We also implemented core system logic, error handling mechanisms, and basic feature validation to ensure reliable operation.

3.4 User Evaluation

Once our prototype was functional, we conducted user evaluation sessions to assess system performance and user satisfaction. We gathered feedback through direct surveys and informal testing sessions where users interacted with both the physical hardware and mobile application. The evaluation focused on usability, user comfort, and how well the system met their practical needs. When feedback revealed areas for improvement or features that didn't align with user expectations, we made necessary adjustments to the system. We repeated the evaluation and refinement cycle multiple times until our prototype satisfied both user requirements and functional specifications.

3.5 Product Implementation

The final phase involved deploying our refined system in real-world scenarios. We ensured that all system components—hardware controllers, software applications, and database integration—worked together seamlessly to provide users with a reliable and intuitive experience. The implementation phase validated that our development process successfully produced a functional smart door security system ready for practical use.

4. Result and Discussion

4.1 Results

4.1.1 Hardware and Software Outputs

1) RFID Smart Door Lock

The hardware output of this study is a miniature smart door lock that utilizes several components, including the ESP32 DOIT V1, PN532 module, relay integrated with an optocoupler, a DC-to-DC step-up converter (5V to 12V), and a 12V solenoid lock. Power is supplied via a 3.3V DC current through a USB-C interface cable connected to a computer. This miniature prototype features fast unlocking using RFID and real-time connectivity with a database.

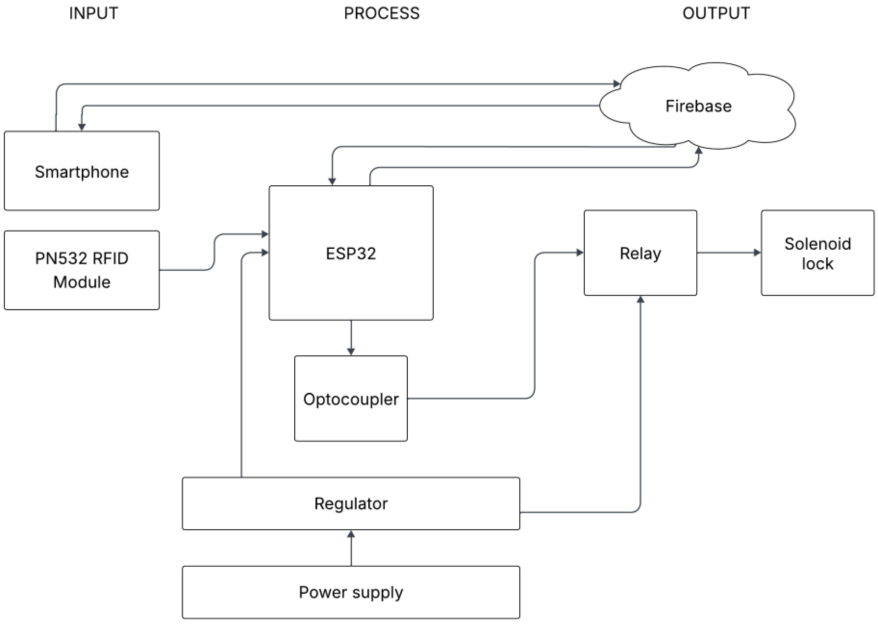


Figure 2. Smart Door Block Diagram

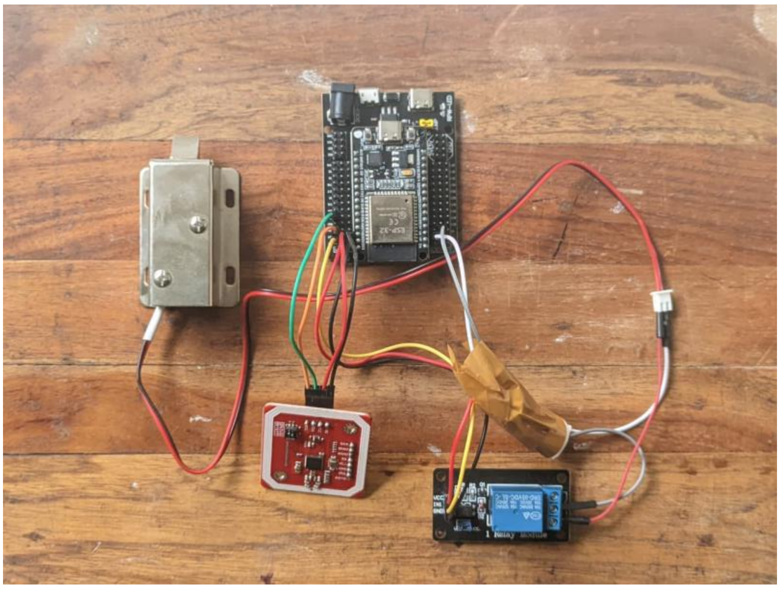


Figure 3. Smart Door Lock with RFID

2) Jenaela Android Application

The software output of this study is an Android application named "Jenaela." The application was developed using the TypeScript programming language and the React Native framework, integrated with the Expo build environment, and deployed using EAS Build Preview. The application includes several key features, such as user authentication, real-time communication with a database, support for WAN-only communication mode, and a hybrid communication mode.

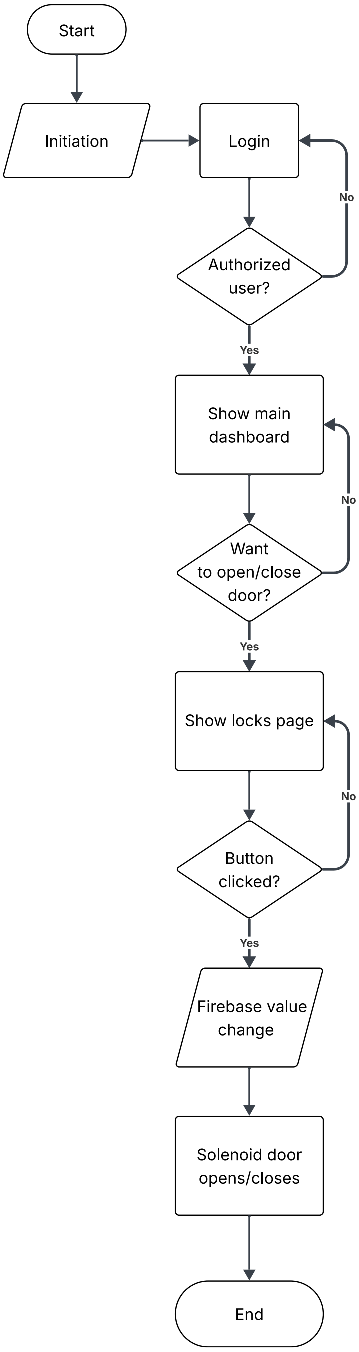


Figure 4. Jenaela Application Block Diagram: Switch via WAN

The Home page of the Jenaela application is shown in Figure 5, while the Account page is displayed in Figure 6. These pages are only accessible after the user has successfully logged in. On the Home page, users are greeted with a modern and minimalist user interface, including personalized greeting cards that display "Good morning/afternoon/evening" messages based on the local time. The page also presents device statistics, showing the total number of devices owned by the user, how many are currently unlocked, and how many are locked. Additionally, the page displays the most frequently used RFID/lock device, along with the device name—in this study, the smart door device is named "SmartLock." The Account page displays the user's identity, join date, user ID, and several other personal details related to the user.

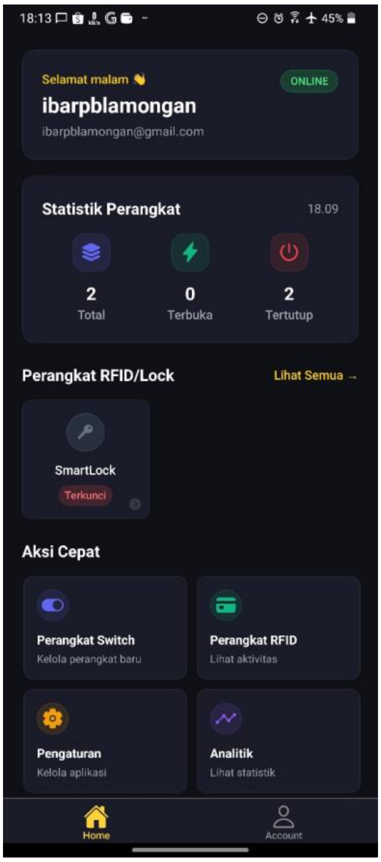
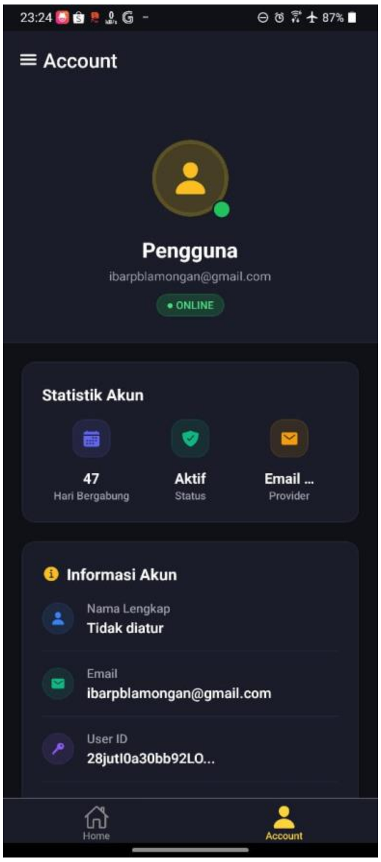


Figure 5. Home page



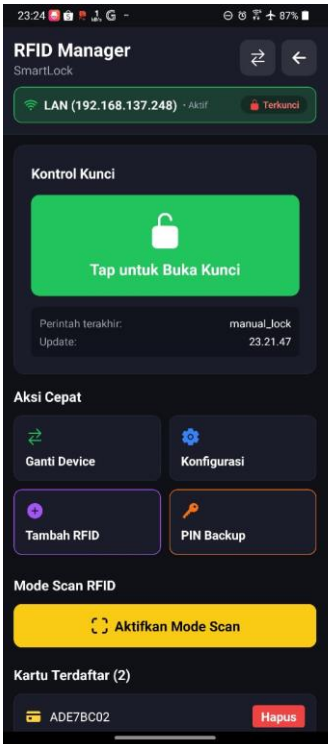
Nama Lengkap
Tidak diatur

Email
ibarpblamongan@gmail.com

User ID
28jutl0a30bb92LO...

Figure 6. Account page

The RFID Manager page, as shown in Figures 7, 8, and 9, serves as the main interface for controlling the smart door lock developed in this study. At the top of the page is a section header indicating the current screen. Below that, the local IP address of the smart lock device is displayed. This feature is particularly useful for users, as DHCP-based IP configurations may cause the IP address to change dynamically. The large green button on the screen functions as a remote switch to lock or unlock the smart door lock. The "Add RFID" button allows users to manually input RFID tag codes into the system. Additionally, the "Backup PIN" button stores a PIN in the database, which can later be used as an alternative method to unlock the smart door using a PIN entry mechanism.

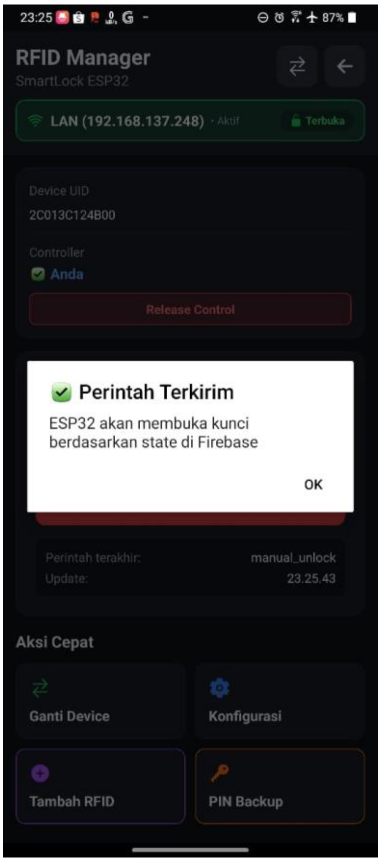


Perintah terakhir: manual_lock
Update: 23.21.47

ADE7BC02

Figure 7. RFID Manager Page

Jenaela autonomously scans for IoT devices within the local area network (LAN), following a community-based service discovery approach commonly employed in IoT systems [18]. When it is not connected to the same local network as the smart lock, Jenaela transitions automatically to wide area network (WAN) mode, as depicted in Figure 9. In this mode, certain functionalities—such as RFID tag scanning via the PN532 sensor—are disabled due to the absence of direct communication with the hardware over the local network.



23:25 87%

RFID Manager
SmartLock ESP32

LAN (192.168.137.248) · Aktif — Terbuka

Device UID
2C013C124B00

Controller
✓ Anda

Release Control

✓ Perintah Terkirim
ESP32 akan membuka kunci berdasarkan state di Firebase

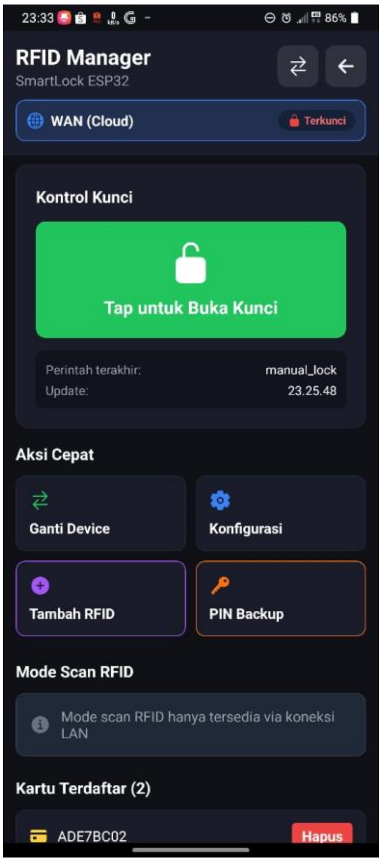
OK

Perintah terakhir: manual_unlock
Update: 23.25.43

Aksi Cepat

Ganti Device | Konfigurasi
Tambah RFID | PIN Backup

Figure 8. Once the "Tap untuk Buka Kunci" button is clicked, the application updates the relay state in the Firebase Realtime Database, which is then read by the ESP32 microcontroller to trigger the solenoid lock accordingly.



23:33 86%

RFID Manager
SmartLock ESP32

WAN (Cloud) — Terkunci

Kontrol Kunci

Tap untuk Buka Kunci

Perintah terakhir: manual_lock
Update: 23.25.48

Aksi Cepat

Ganti Device | Konfigurasi
Tambah RFID | PIN Backup

Mode Scan RFID

Mode scan RFID hanya tersedia via koneksi LAN

Kartu Terdaftar (2)

ADE7BC02 — Hapus

Figure 9. The application in WAN mode. RFID scan feature is disabled

4.1.2 Testing Results

System testing was conducted using the Black Box testing method. This approach focuses on evaluating the functional aspects of the system without requiring knowledge of the underlying code or implementation details (Utomo *et al.*, 2020).

1) RFID Smart Door Lock Testing

The performance testing results of the hardware locking mechanism are presented in Table 1 and Table 2. This test aimed to evaluate the responsiveness and reliability of the developed device. The testing was conducted using a digital stopwatch and a ruler to measure response times and RFID detection distances. The relative distance between the PN532 sensor surface and the RFID tag surface was measured, with distance intervals ranging from 1 cm to 5 cm. For each distance, the test was repeated three times by placing the RFID tag at the specified distance and recording the response time using the stopwatch. The average unlocking performance using RFID is also presented in Table 1 and Table 2, summarizing the system's responsiveness across varying detection ranges.

Table 1. B6193100 card type

Test	Locking Mode	Code	Physical Form	Relative Distance	Average Response Time	Result Description
1	RFID	B6193100	Card	1 cm	~20 ms	Working
2	RFID	B6193100	Card	2 cm	~20 ms	Working
3	RFID	B6193100	Card	3 cm	~20 ms	Working
4	RFID	B6193100	Card	4 cm	~30 ms	Working
5	RFID	B6193100	Card	5 cm	~30 ms	Working

Table 2. ADE7BC02 key chain type

Test	Locking Mode	Code	Physical Form	Relative Distance	Average Response Time	Result Description
1	RFID	ADE7BC02	Key chain	1 cm	~20 ms	Working
2	RFID	ADE7BC02	Key chain	2 cm	~20 ms	Working
3	RFID	ADE7BC02	Key chain	3 cm	~30 ms	Working
4	RFID	ADE7BC02	Key chain	4 cm	-	Not working
5	RFID	ADE7BC02	Key chain	5 cm	-	Not working

Meanwhile, the results of the test using the "Unlock" switch connected via a WAN network are presented in Table 3. This test was conducted by pressing the "Tap untuk Buka Pintu" button, as shown in Figure 8. The lock response time was measured from the moment the button was pressed until the solenoid lock responded and changed its state.

Table 3. Switch through WAN

Test	Locking Mode	Lock Response Time	Result Description
1	Switch through WAN	~1800 ms	Working
2	Switch through WAN	~800 ms	Working
3	Switch through WAN	~1000 ms	Working
4	Switch through WAN	~300 ms	Working
5	Switch through WAN	~600 ms	Working

2) Jenaela App Testing

The Jenaela application was tested through several scenarios considered representative of real-world use cases in managing IoT hardware such as smart door locks. These scenarios were designed to reflect key functionalities of the Android application. The testing scenarios and their expected outcomes are presented in Table 4. The Result Description is marked as "Working" if the actual outcome matches the expected result.

Table 4. Jenaela app scenario testing

No	Testing Scenario	Expected Outcome	Result Description
1	User opens the Sign-Up page, fills in personal details (email, password), and clicks the Sign-Up button.	The user is successfully registered in the database and is able to log in to the application.	Working
2	User opens the Sign-In page, enters email and password, and clicks the Sign-In button.	If the credentials are correct and the user is registered, the application navigates to the home page.	Working
3	User scans for available IoT devices using the ESP32 Scan feature on the RFID Manager page.	The application displays a list of available IoT devices.	Working
4	User opens the saved device list and selects a device from the RFID Manager page.	A pop-up displays the saved device information. The device can then be selected, redirecting the user to the RFID Manager page.	Working
5	User clicks the "Tap untuk Buka Kunci/Tutup Kunci" button on the RFID Manager page.	The application updates the <isLocked> value in the Firebase Realtime Database to reflect the current lock state.	Working
6	User clicks the "Aktifkan Mode Scan" button to register an ID tag on the RFID Manager page.	A card UI appears and displays the scanned ID tag code when the user taps the tag on the PN532 sensor.	Working

7	User clicks the "Hapus" button next to a registered ID tag in the list.	The selected ID tag is removed from the database, and the list is refreshed to reflect the updated data.	Working
8	Jenaela automatically scans for available IoT devices within the local area network (LAN) upon application launch.	The application detects IoT devices present within the local area network (LAN).	Working

4.2 Discussion

Testing reveals that our RFID smart door lock performs reliably across different tag types and detection ranges. The B6193100 card maintains steady response times of 20-30 milliseconds at distances from 1-5 cm. The ADE7BC02 key chain, however, works effectively only within 3 cm. Card-type RFID tags feature larger antenna surfaces that enable better signal reception compared to compact key chain designs. Users should select tag types based on their specific needs and usage patterns. Remote control through WAN networks shows response times between 300ms and 1800ms. Network delays, Firebase processing speeds, and internet stability all influence performance. While 1.8 seconds might seem slow for door locks, such timing works fine for regular access situations. The system switches automatically between local and wide area networks, giving users flexibility across different environments. Local connections deliver faster responses and direct hardware access, while wide area mode keeps the system reachable when users are away from home.

Every test scenario in the Jenaela application worked as expected, showing solid software development. The app handles user login, device discovery, and real-time database updates smoothly through React Native and Firebase integration. Automatic device scanning makes setup easier by removing manual configuration steps. Users can open the app and start using detected smart locks right away. Testing revealed several drawbacks. Key chain tags have shorter detection ranges that may inconvenience some users. Wide area mode also disables direct hardware features like RFID scanning when users aren't on the local network. We didn't thoroughly examine power usage, which deserves attention in future work. Battery operation would make installation more flexible and portable. Our system proves that custom IoT solutions work well without depending on third-party platforms like Blynk. React Native paired with Firebase creates a solid base for adding new features and customization. The modular hardware setup makes replacing parts and upgrading straightforward. Future versions could include more sensors, better power management, or stronger security without rebuilding everything from scratch.

5. Conclusion, Research Limitations, and Future Work

The research developed a smart door lock system combining IoT hardware with a custom Android application. The primary objective centered on creating a secure, flexible, and user-friendly solution that surpasses conventional mechanical locks and third-party platforms such as Blynk. Building the Jenaela application using React Native with Firebase real-time communication established that complete customization remains both practical and effective. The hardware assembly—ESP32 DOIT V1, PN532 reader, relay, and solenoid lock—creates a compact yet functional system supporting remote access and RFID-based authentication. Results confirm that custom applications provide superior control and adaptability compared to pre-built IoT platforms, making them more suitable for long-term deployment across diverse user environments. The dual-mode connectivity (LAN and WAN) enhances system usability under varying network conditions.

Future research directions include incorporating biometric authentication, mobile push notifications, and offline data synchronization to strengthen system reliability and scalability for real-world smart home applications. These enhancements would address practical deployment challenges where network connectivity may be intermittent or users require backup access methods. The study establishes that custom IoT solutions offer significant advantages when specific functionality and complete system control are required for specialized applications.

References

[1] Boros, M., Siser, A., Kekovic, Z., & Mazal, J. (2018). Mechanical Characteristics Of Cylinder Pin Tumbler Locks as They Relate to Resistance Testing. *Komunikácie*, *20*(2). https://doi.org/10.26552/com.c.2018.2.96-101

[2] Kagawa, C. (2004). Attacks against the mechanical pin tumbler lock. https://api.semanticscholar.org/CorpusID:53797585

[3] Sun, K. Y., Pernando, Y., & Safari, M. I. (2021). Perancangan sistem IoT pada smart door lock menggunakan aplikasi BLYNK. *JUTSI (Jurnal Teknologi dan Sistem Informasi)*, *1*(3), 289–296. https://doi.org/10.33330/jutsi.v1i3.1360

[4] Celestine, D. (2020). Smart lock systems: An overview. *International Journal of Computer Applications*, *177*(37), 40–43. https://doi.org/10.5120/ijca2020919882

[5] Akbari, Y., Al-Binali, A., Al-Mohannadi, A., Al-Hemaidi, N., Elharrouss, O., & Al-Maadeed, S. (2024). A new framework for smart doors using mmWave radar and camera-based face detection and recognition techniques. *Sensors*, *24*(1), 172. https://doi.org/10.3390/s24010172

[6] Tseloni, A., Thompson, R., Grove, L., Tilley, N., & Farrell, G. (2017). The effectiveness of burglary security devices. *Security Journal*, *30*(2), 646–664. https://doi.org/10.1057/sj.2014.30

[7] Permana, K. A. K., Piarsa, I. N., & Wiranatha, A. A. K. A. C. (2024). IoT-based smart door lock system with fingerprint and keypad access. *Journal of Information Systems and Informatics*, *6*(3), 2086–2098. https://doi.org/10.51519/journalisi.v6i3.844

[8] Bagi, S., et al. (2025). Perancangan aplikasi mobile berbasis Flutter untuk pemantauan data sensor IoT. *JSAI: Journal Scientific and Applied Informatics*, *8*(1). https://doi.org/10.36085

[9] Hendriawan, M., Haryono, H., & Budiman, T. (2023). Development of water level monitoring applications in smart home systems using Flutter. *Journal of Information System, Informatics and Computing*, *7*(2), 213. https://doi.org/10.52362/jisicom.v7i2.1197

[10] Ravian, A. Z., & Irianto, K. D. (2024). Perancangan sistem monitoring pengunjung puskesmas berbasis IoT. *Technologia: Jurnal Ilmiah*, *15*(3), 560. https://doi.org/10.31602/tji.v15i3.15361

[11] Rihhadatulaisy, Z. H., & Irianto, K. D. (2024). Designing an automatic room temperature control system for smart homes for the elderly using IoT. *International Journal Software Engineering and Computer Science (IJSECS)*, *4*(2), 758–766. https://doi.org/10.35870/ijsecs.v4i2.2844

[12] Gamma, W., Putri, R., Wiseto, I., & Agung, P. (2025). IoT smart door lock system menggunakan double sensor berbasis mikrokontroler ESP32.

[13] Sonya, M. A. (2025). Sistem pencahayaan otomatis pada smart home untuk lansia berbasis IoT. *Edusaintek: Jurnal Pendidikan, Sains dan Teknologi*, *12*(1). https://doi.org/10.47668/edusaintek.v12i1.1366

[14] Caleb, F. (n.d.). Understanding Blynk IOT. *Electronic Ideas*.

[15] Shevtsiv, N., Shvets, D., & Karabut, N. (2019). Prospects for using React Native for developing cross-platform mobile applications. *Central Ukrainian Scientific Bulletin. Technical Sciences*, *2*(33), 208–213. https://doi.org/10.32515/2664-262X.2019.2(33).208-213

[16] Senarath, U. S. (2021). Waterfall methodology, prototyping and agile development. https://doi.org/10.13140/RG.2.2.17918.72001

[17] Sridhar, M. S. (2020). Importance and issues of literature review in research. https://doi.org/10.13140/RG.2.2.15347.14885

[18] Khanfor, A., Ghazzai, H., Yang, Y., Haider, M. R., & Massoud, Y. (2020). Automated service discovery for social Internet-of-Things systems. *arXiv preprint*. http://arxiv.org/abs/2003.11524

[19] Utomo, A., Sutanto, Y., Tiningrum, E., & Susilowati, E. M. (2020). Pengujian aplikasi transaksi perdagangan menggunakan black box testing boundary value analysis. *Jurnal Bisnis Terapan*, *4*(2), 133–140. https://doi.org/10.24123/jbt.v4i2.2170

[20] Rif'an, M., & Irianto, K. D. (2024). Precision agriculture system with IoT: An approach to increase production and efficiency. *International Journal Software Engineering and Computer Science (IJSECS)*, *4*(3), 1305–1316. https://doi.org/10.35870/ijsecs.v4i3.3259

[21] Sun, K. Y., Pernando, Y., & Safari, M. I. (2021). Perancangan sistem IoT pada smart door lock menggunakan aplikasi BLYNK. *JUTSI (Jurnal Teknologi dan Sistem Informasi)*, *1*(3), 289–296. https://doi.org/10.33330/jutsi.v1i3.1360