Confluence University Journal of Science and Technology
Journal of the Confluence University of Science and Technology,
Osara, Kogi State
Journal Homepage: www.cujostech.com.ng
CUJoSTech

Design And Implementation of a Secure Remote Patient Monitoring System Using Advanced Encryption Standard and Meta Mask Authentication

*[1]NUHU B.K., [1]AGAJO J., [1]SOEMPIT C.E., [1]INALEGWU A.E., [2]JACK K.E., [1]DOGO E. M, [3]OLATOMIWA L. J.
[1]Department of Computer Engineering, Federal University of Technology, Minna, Niger State, Nigeria
[2]Department Mechatronics Engineering, Federal University of Technology, Minna, Niger State, Nigeria
[3]Department of Electrical/Electronics Engineering, Federal University of Technology, Minna, Niger State, Nigeria.
*nuhubk@futminna.edu.ng
DOI: https://doi.org/10.5455/CUJOSTECH.250410
Research Article

Abstract

Remote Patient Monitoring (RPM) systems enable continuous tracking of patient vital signs outside clinical settings, improving outcomes and reducing hospital visits. However, privacy and security concerns remain a major barrier to their adoption. In this paper, we present the design and implementation of a secure RPM system that uses Advanced Encryption Standard (AES) to encrypt sensor data and MetaMask (an Ethereum wallet/extension) to authenticate users. The system integrates biomedical sensors (ECG, pulse oximeter, and temperature) with an Arduino microcontroller and Wi-Fi module. Encrypted data are transmitted to a cloud database, while physicians' access and decrypt data via a web interface protected by MetaMask login. A prototype was developed and tested, it successfully collected and transmitted patient vitals (heart rate, SpO_2, temperature) to the cloud and permitted only authenticated users to view the decrypted data. Performance evaluation of the system showed that it achieved an average 981.1ms delay in rendering the collected vitals to the doctor. The AES encryption on the Arduino recorded minimal latency of <50ms per block, while network transmission delay was within 500ms per upload. The proposed approach ensured confidentiality of health data in transit and at rest, addressing data privacy challenges in RPM.

Keywords: *Remote Patient Monitoring, Internet of Things, AES Encryption, MetaMask, Data Privacy, Healthcare Security.*
Article History: *Received: 18 April 2025; Accepted: 22 June 2025; Published: 27 June 2025*

1. Introduction

The healthcare industry has experienced technological transformation through innovations like Remote Patient Monitoring (RPM), which facilitates the collection and transmission of patient-generated health data (PGHD) using mobile devices. RPM enables patients, especially those at high risk or in remote areas, to monitor vital health metrics such as blood pressure, heart rate, and oxygen levels from home and share this data with healthcare providers [1]. This system reduces the frequency of hospital visits, enhances access to healthcare services, and supports early detection of potential health issues, thereby improving overall patient outcomes [2]. RPM is now increasingly integrated into broader telehealth systems that serve diverse populations across multiple healthcare settings, including schools, prisons, and nursing homes.

RPM shifts healthcare delivery from traditional settings into everyday environments, improving patient engagement and clinical efficiency. With continuous data flow from patients, clinicians can gain a more accurate and real-time understanding of their health, enabling more responsive and personalized care [3]. The use of telehealth technologies not only improves access for patients in underserved regions but also supports chronic disease management and post-discharge care. By bringing care into the home, RPM reduces logistical burdens on patients while supporting more proactive medical interventions [3].

RPM enables patients to be monitored from home using wearable or bedside sensors, reducing hospital burden and improving patient comfort. In literature there are several works on RPM, each with its strengths and weaknesses. For example, authors in [4] described an RPM system that incorporated a microcontroller and sensors to collect human vitals (pulse oximeter, blood pressure, temperature) and transmitted them to a web interface and database. Authors in [5] note that a typical RPM framework consists of *data collection*, *processing*, *communication*, and *interpretation* stages. Authors in [6] present a prototype with wearable ECG devices, a smartphone gateway (via Bluetooth), and a cloud server with a web User Interface (UI). These works establish that modern RPM systems commonly used embedded microcontrollers, wireless modules, and web/cloud services for data management. Other works and approaches to RPM are reviewed hereunder.

Authors in [7] presented a remote patient monitoring system that uses Android devices and wireless sensors to track patients' vital signs and GPS location in real time. Health data is sent securely to a server, and emergency alerts are triggered if abnormal readings are detected. Doctors can access patient data through mobile or web apps. The system ensures patient mobility, data security via AES encryption, and efficient data storage using compression techniques. However, a single authentication scheme was employed, which limits the security depth.

In a similar vein, the works of [8] and [9], presented a design that focused on remote electrocardiogram (ECG) data collection and transmission using the MQTT protocol for lightweight and reliable communication. The design allowed doctors

to monitor patients' heart activity in real time via a mobile or desktop application with zero packet loss and minimal delay. However, the security of the collected data was not considered.

To ensure an intuitive and real-time diagnosis, [10] presented an Android-based visualization system for remote patient monitoring using Internet of Medical Things (IoMT) devices. It allows healthcare providers to view real-time health data through intuitive dashboards, improving monitoring efficiency. Usability tests showed the system was significantly more user-friendly and effective than traditional methods, offering better system quality, information clarity, and user satisfaction. The solution supports improved patient care and reduces healthcare costs. However, it lacks security for the patients' data.

In terms of data security in RPM, [11] proposed blockchain-based architecture where health data are encrypted (using RSA, ECC, or AES) before being written to a distributed ledger. This ensures confidentiality even though the ledger is public. The AES algorithm is widely used in IoT applications for its efficiency. AES-128 offers 128-bit security. The AES cipher works by applying multiple rounds of SubBytes, ShiftRows, MixColumns, and AddRoundKey operations to a 128-bit data block file. It is efficient with microcontrollers and provides strong confidentiality for transmitted data [12]. However, AES alone does not provide authentication or integrity, so secure transmission typically also requires MACs or signatures file.

For user authentication and access control, blockchain and cryptocurrency wallets offer innovative solutions. MetaMask is a browser extension that manages Ethereum keys; it can prompt a user to sign messages or transactions with their private key [13]. In an RPM context, integrating MetaMask means that clinicians authenticate by signing a nonce with their wallet, no central server stores passwords. Once the signature is verified (i.e. the user controls a known public address), the system grants access. This decentralized login approach removes single points of credential storage, reducing breach risk. MetaMask offers increased security through blockchain and cryptographic keys and decentralization, whereas password schemes are vulnerable to reuse and server attacks.

Despite its potential, RPM adoption faces significant challenges, particularly in the areas of data privacy and security. Patients often hesitate to fully engage with remote monitoring technologies due to fears that their sensitive health information may be exposed or misused [14]. These concerns are heightened in cloud computing environments where data is dispersed across numerous devices and platforms, making it harder to secure. To address these risks, healthcare providers must implement comprehensive cybersecurity measures, including the use of Advanced Encryption Standard (AES) for data protection, two-factor authentication, role-based access control, and advanced infrastructure monitoring to maintain trust and regulatory compliance [14]. Thus, this paper combines AES for *data confidentiality* with MetaMask for *strong authentication*, as advocated by recent studies as [15].

2.0 Materials and Methods

The proposed secure remote patient monitoring system integrates hardware and software components to collect, encrypt, transmit, and display patient health data. Hardware elements include sensors for measuring SpO_2, heart rate, temperature, and ECG, along with a keypad, display, Arduino Nano microcontroller, and Wi-Fi module for internet connectivity. The system encrypts the collected data and sends it to a Firebase real-time database as shown on the block diagram presented in Figure 1. On the software side, a React-based web application retrieves and decrypts the data for monitoring by authorized medical personnel, using MetaMask authentication to ensure privacy and security.

From the block diagram, the proposed RPM system consists of three main parts: (1) a sensor node (patient side), (2) a cloud database, and (3) a web interface (hospital side).

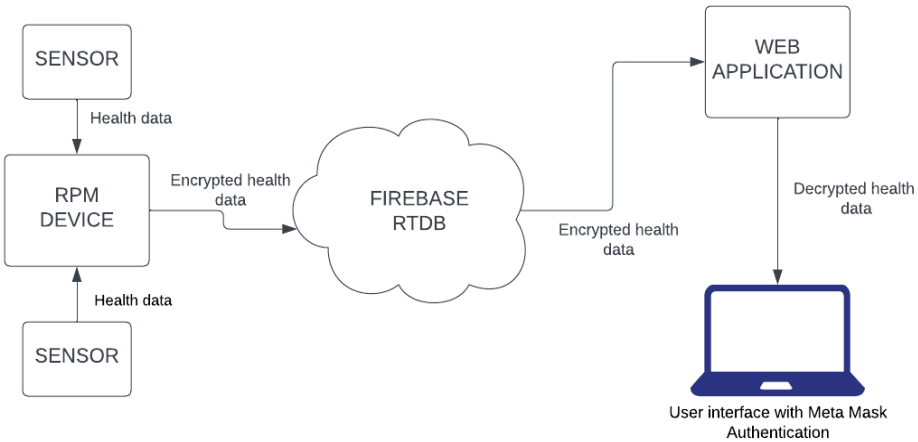


Figure 1: Block diagram of the system

2.1 Sensor Node/Data Acquisition

The design of the system was around an Arduino microcontroller (ATmega328P) equipped with biomedical sensors. One of the sensors is the AD8232 ECG sensor which was used to measure the patient's cardiac waveform. The choice of AD8232 was because it was designed to amplify and filter tiny ECG signals from the human body. Another sensor is the MAX30102 pulse oximeter/heart-rate sensor which provides blood oxygen saturation (SpO_2) and heart rate measurements by shining red and IR LEDs through the human skin and detecting the reflected light as shown in Figure 2. A LM35 temperature

sensor was interfaced to measure body temperature; it outputs a linear voltage proportional to temperature (10 mV/°C). These analog/digital signals are read by the Arduino and converted to numerical values.

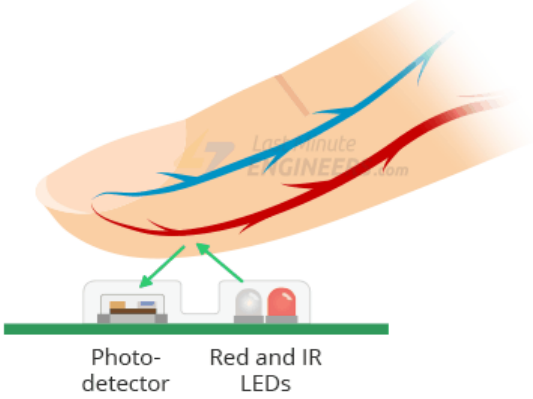


Figure 2: MAX30102 Oximeter with sensor placement

The microcontroller runs an encryption routine, before transmission, each sensor reading is encrypted using AES-128 in CBC mode. We use a pre-shared 128-bit key known to both the device and the web server. AES was chosen for its efficiency on embedded platforms and proven security. The implementation was done using AES.Lib library. This library provides functions for both AES encryption and decryption. The encrypted data is transmitted via a Wi-Fi module (ESP8266) to the cloud. The ESP8266 module includes a Tensilica 32-bit CPU and integrated Wi-Fi antenna, allowing the Arduino to post data to online services over TCP/IP. Algorithm 1 provides the steps ensured. Key aspects of this algorithm are key generation, key exchange/storage.

Algorithm 1: AES Implementation Steps
1. *Install the AESLib library on your Arduino IDE (Integrated Development Environment).* 2. *Generate a 128-bits AES key using a key generation tool.* 3. *Initialize the AES encryption using the AES_init() function from the AESLib library. The function takes the key and key size as input parameters.* 4. *Encrypt the data using the AES_cipher() function from the AESLib library. The function takes the data and data length as input parameters and returns the encrypted data.* 5. *Transmit the encrypted data over the network.*

a. Key Generation: The AES key was generated Generate using a key generation tool, this key size was 128-bit. On the Arduino IDE, the encryption of the data was done using the AESLib library, the library was initialized and the key passed in with the function aes_init(), the data was then encrypted with the aes_cipher() function, and the returned value was a cypher text. This cypher text was converted to a string value and sent to the cloud storage via the firebase's APIs, this meant that all the data on the cloud server was pure encrypted data.

b. Key Exchange/Storage: The generated key is securely stored in the frontend's environmental variables. The encrypted data and the key are passed into a CyptoJs library function, this function decrypts the data and displays it on the user's interface.

2.2 Cloud Database

Encrypted patient records are stored on a secure cloud database. We assume a standard IoT backend (Firebase) that supports secure REST or MQTT communication. The data remains encrypted at rest (with AES) so that the cloud provider cannot read them. Firebase Realtime Database stores and syncs patient data in real time using web sockets, ensuring all users see updates instantly. It enforces security rules to control access, allowing only authorized users to read or write data based on authentication and ownership.

2.3 Web Interface and MetaMask Authentication

Healthcare providers use a web application to view patient data. The web app first requires the doctor to connect their MetaMask wallet as seen in the homepage (Figure 3). When the user clicks "Login with MetaMask", the browser extension prompts them to sign an authentication message using their Ethereum private key (Figure 4). The application verifies this signature against a list of approved public addresses. Upon successful verification, the user is marked as authorized. The web app then queries the cloud database for encrypted values, applies AES decryption using the shared key, and displays the cleartext vital signs on the dashboard (Figure 5). Because the private key never leaves MetaMask, and no plaintext credentials are stored on the server, this adds a strong security layer.

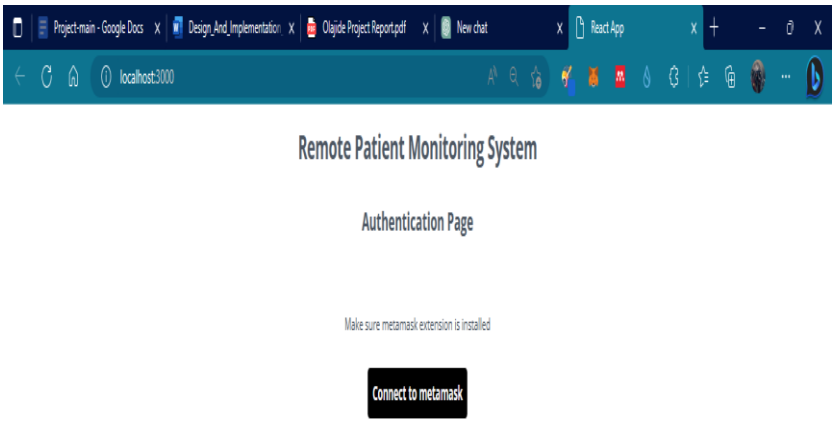


Figure 3: Homepage of the RPM System Interface

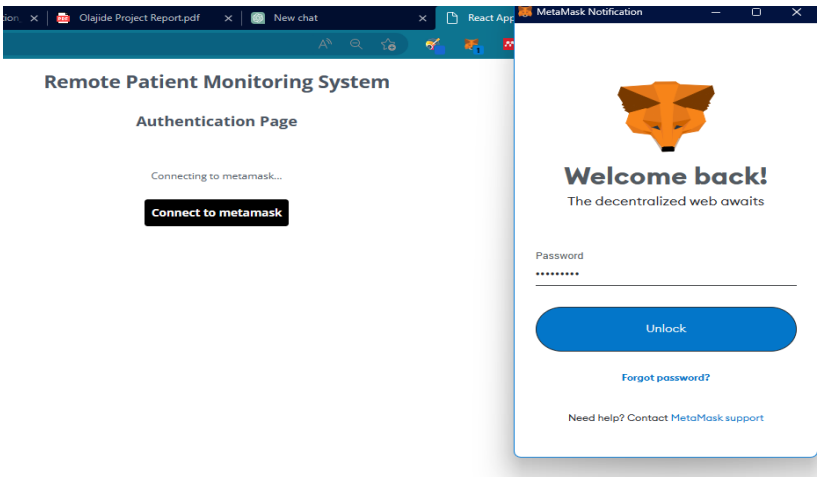


Figure 4: Login Interface of the RPM System Interface

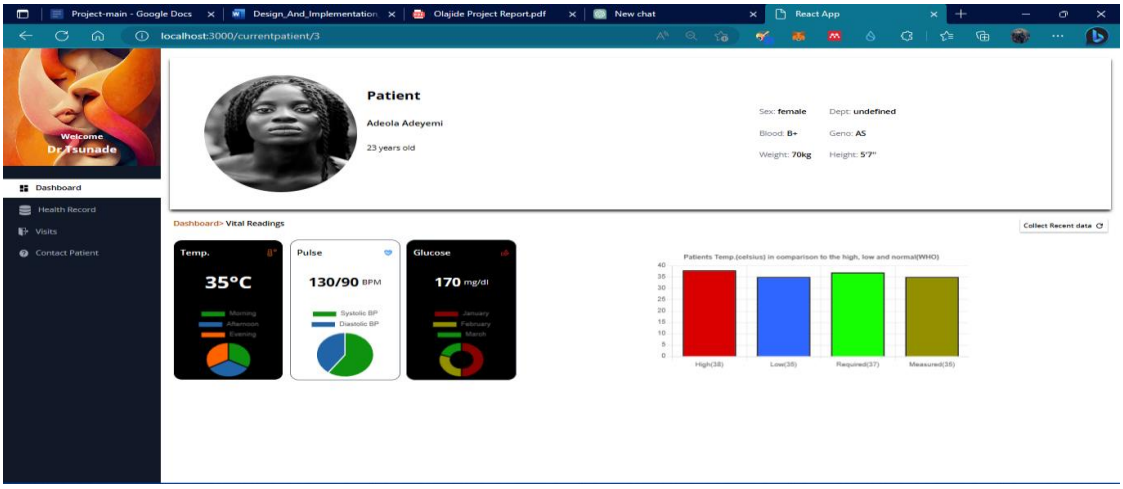


Figure 5: Doctor's View of the Patient Dashboard

Additionally, a Liquid Crystal Display (LCD) and keypad are provided on the designed local prototype device as seen in Figure 6, to allow local interaction (threshold setting and alerts) with the patient. A buzzer functionality is also used to notify the patient when any vital sign crosses a danger threshold.

Figure 6: Final Developed Prototype RPM System

3.0 Results and Discussion

The integration of MetaMask was seamless, only users with the private key could complete the login handshake. Results were obtained for response time and data security analysis.

3.1 Response Time Analysis Result

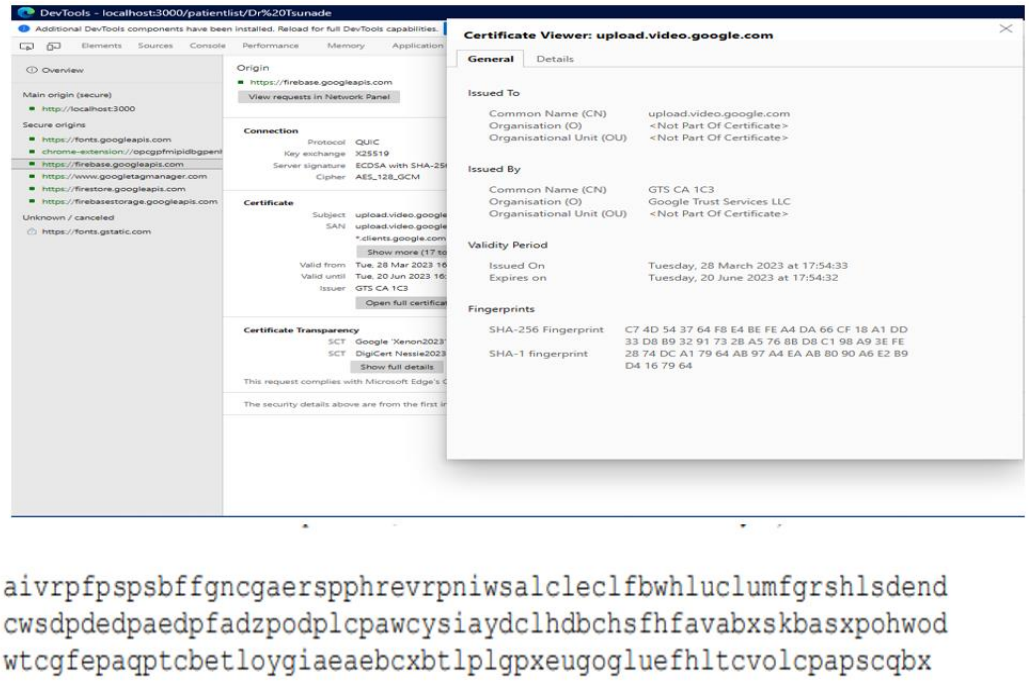
The time it takes for the encrypted sensor readings in the firebase to get decrypted and rendered to an authenticated doctor was evaluated as presented in Table 1. It can be observed that the test was done from 10 trials, and the average was obtained to represent the true response time of the system. The highest was 1206 milliseconds (ms) obtained in the first trial, while the lowest was 843ms from the third trial. The average response time is 981.1ms which shows a quick response, maintaining "availability" property of a secured information.

Table 1: System Response Time

Trials	Response Time (ms)
1	1206
2	900
3	843
4	930
5	1002
6	957
7	893
8	1102
9	988
10	990
Average	981.1

3.2 Security Analysis Result

The performance of the security aspect of this paper was adequate. AES encryption on the Arduino introduced minimal latency (<50 ms per block), and network transmission was on the order of 500 ms per upload (including Wi-Fi delay). The use of MetaMask added security with negligible user inconvenience (a one-time signature). Our results show that the system can continuously record and send vital parameters to the cloud securely. All functionalities (encryption/decryption, MQTT/HTTP posting, MetaMask login) worked as intended. No data breaches occurred during testing, demonstrating that AES + MetaMask effectively protects RPM data. Figure 7 shows the legitimacy of the backend after integration with the web-app. Firebase provides secure authentication using industry-standard protocols including OAuth 2.0 and OpenID Connect. This ensured that only authorized users can access the application and its resources. The AES cyphertext after the sensor reading was encrypted is also depicted in Figure 6.




```
aivrpfpspsbffgncgaerspphrevrpniwsalcleclfbwhluclumfgrshlsdend
cwsdpdedpaedpfadzpodplcpawcysiaydclhdbchsfhfavabxskbasxpohwod
wtcgfepaqptcbetloygiaeaebcxbtlplgpxeugogluefhltcvolcpapscqbx
```

Figure 7: Firebase Web-certificate Plus AES cyphertext after encryption

3.3 Scalability Analysis

The system, initially tested with one user, can scale effectively to support hundreds or even thousands of patients and physicians with proper backend planning. Each patient generates approximately 42 MB of daily data, so 100 users would require around 4.2 GB/day, which can effectively be managed with Firebase's Blaze plan. Real-time updates using Firebase's WebSocket infrastructure is capable of supporting tens of thousands of concurrent connections, and MetaMask-based authentication scales independently with minimal server load. With optimized data structuring and cloud resources, the system can handle growing user demand while maintaining real-time performance and security.

3.4 Discussions

The implemented system addresses the core challenge of data privacy in RPM. By encrypting all medical readings with AES before they leave the patient's device, we ensure that intercepted or stored data cannot be understood without the key. This guards against eavesdropping and unauthorized cloud access. In addition, requiring doctors to authenticate with MetaMask means that stolen database credentials or phishing of passwords are ineffective. The adversary would need the doctor's Ethereum private key, which is infeasible to guess. This decentralized authentication reduces single points of failure in the security design.

However, certain limitations remain. AES, while very secure against brute force, is known to be vulnerable to *side-channel* attacks such as power analysis on embedded devices. In future hardware versions we could use masking or hardware AES modules to mitigate this. AES also provides no integrity check, so an active attacker could tamper with ciphertext without detection; adding an HMAC or using an authenticated encryption mode such as AES-GCM is a likely consideration in the future version. On the authentication side, MetaMask usability depends on browser support and key management by users. Loss of the private key would lock out the doctor, so key recovery or multi-sign schemes would be considered.

Despite these limitations, our solution compares favorably to previous systems. Unlike systems that rely on centralized passwords, our MetaMask approach eliminates a vulnerable credential store. Compared to systems that do not encrypt sensor data, our AES layer greatly reduces the risk of privacy breach. In practice, healthcare providers can deploy this RPM prototype to gain patient metrics with confidence that confidentiality is maintained.

4.0 Conclusion

This Paper presented a design and implemented a secured remote patient monitoring system that integrates AES encryption and MetaMask-based authentication. The system leverages IoT sensors (AD8232 ECG, MAX30102 SpO_2, LM35 temperature) connected to an Arduino and Wi-Fi module, sending encrypted data to the cloud. A web interface requires MetaMask login, ensuring only authorized clinicians can decrypt and view the data. Our prototype proved that vital signs could be transmitted and retrieved securely: heart rate, oxygen saturation, and temperature readings were successfully recorded, encrypted, transmitted, and decrypted for display. This approach directly addresses data privacy challenges in RPM. Future work will extend the system with message authentication, scalability testing, and evaluation in clinical environments bearing in mind low bandwidth, packet loss, or sensor failures scenarios.

The paper lacks comparisons on the use of MetaMask and AES for security with other methods. Future work will also explore alternatives like OAuth 2.0, biometrics, and MFA for authentication, and compare AES with RSA or ECC for encryption. This would help identify the best options based on usability, performance, and device limitations in remote healthcare settings.

Acknowledgements

The authors are grateful to the Department of Computer Engineering, Federal University of Technology for providing the Lab space and supporting materials and components that make this research feasible.

Funding

This research was supported by the Tertiary Education Trust Fund under Institution Based Research (IBR) Grant with ID: TETFUND/FUTMINNA/2024/051.

Conflict of interest:

The authors declare no conflicting interest

Authors' contributions:

All authors contributed equally to the research from ideation to the final draft production of this manuscript and gave their approval.

References

1. Boikanyo K, Zungeru AM, Sigweni B, Yahya A, Lebekwe C. Remote patient monitoring systems: Applications, architecture, and challenges. Scientific African. 2023 Jul 1;20:e01638.
2. Adeghe EP, Okolo CA, Ojeyinka OT. A review of wearable technology in healthcare: Monitoring patient health and enhancing outcomes. OARJ of Multidisciplinary Studies. 2024;7(01):142-8.
3. Serrano LP, Maita KC, Avila FR, Torres-Guzman RA, Garcia JP, Eldaly AS, Haider CR, Felton CL, Paulson MR, Maniaci MJ, Forte AJ. Benefits and challenges of remote patient monitoring as perceived by health care practitioners: a systematic review. The Permanente Journal. 2023 Sep 22;27(4):100.
4. Sebastian S, Jacob NR, Manmadhan Y, Anand VR, Jayashree MJ. Remote patient monitoring system. International Journal of Distributed and Parallel Systems. 2012 Sep 1;3(5):99.
5. Malasinghe LP, Ramzan N, Dahal K. Remote patient monitoring: a comprehensive study. Journal of Ambient Intelligence and Humanized Computing. 2019 Jan 29;10:57-76.
6. Hossain MS, Muhammad G. Cloud-assisted industrial internet of things (iiot)–enabled framework for health monitoring. Computer Networks. 2016 Jun 4; 101:192-202.
7. Kamel M, George L. Remote patient tracking and monitoring system. International journal of computer science and mobile computing. 2013 May;2(12):88-94.
8. Archip A, Botezatu N, Şerban E, Herghelegiu PC, Zală A. An IoT based system for remote patient monitoring. In 2016 17th international carpathian control conference (ICCC) 2016 May 29 (pp. 1-6). IEEE.
9. Maqbool S, Iqbal MW, Naqvi MR, Arif KS, Ahmed M, Arif M. IoT based remote patient monitoring system. In2020 international conference on decision aid sciences and application (DASA) 2020 Nov 8 (pp. 1255-1260). IEEE.
10. Khan MA, Din IU, Kim BS, Almogren A. Visualization of remote patient monitoring system based on internet of medical things. Sustainability. 2023 May 16;15(10):8120.
11. de Moraes Rossetto AG, Sega C, Leithardt VR. An architecture for managing data privacy in healthcare with blockchain. Sensors. 2022 Oct 29;22(21):8292.
12. Rosillo García J. *Implementation of the AES algorithm using a microcontroller with 3 cores* (Bachelor's thesis, Universitat Politècnica de Catalunya).
13. NFTnow. *How to Set Up and Use MetaMask's Crypto Wallet: A guide [Internet]*. NFTNow, 2024 [Cited 2025 June 14]. Available from: https://nftnow.com/guides/how-to-set-up-metamask-wallet/
14. Houser SH, Flite CA, Foster SL. Privacy and security risk factors related to telehealth services–a systematic review. Perspectives in health information management. 2023 Jan 10;20(1):1f.
15. Brijwani GN, Ajmire PE, Junaid MA, Charasia SA, Bhende D. Revolutionizing Healthcare Record Management: Secure Documentation Storage and Access through Advanced Blockchain Solutions. arXiv preprint arXiv:2503.00742. 2025 Mar 2.