

Enhancing Medical Data Security Through Blockchain Smart Contract and Decentralized Application

Rinday Zildjiani Salji¹, Herman Herman^{*2}, Herman Yuliansyah³

^{1,2} Master Program of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, 55164

³ Department of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, 55191

Article Info

Article history:

Received May 21, 2025

Revised Jul 19, 2025

Accepted Oct 15, 2025

Keywords:

Blockchain

Medical Data

Off-chain

On-chain

Smart Contract

ABSTRACT

This research studies implementation of decentralized applications (DApps) that are combined with blockchain technology and IPFS for storing patient medical data. The goal of this research is to increase the security, transparency, and access control of stored medical data to make sure only legitimate users can access the data. The proposed system uses smart contracts on the Ethereum network to handle user rights of access (doctors, patients, and admins) and ensure data integrity through the blockchain immutability feature. Patient medical records are retained in IPFS and traced using the Content Identifier (CID). Implementation outcome reveals that the system can safely process medical information, keeping patients in full control of their information, and restricting data access only to scheduled time. This system also shows the potential of blockchain and IPFS technology-based applications in achieving a more efficient health ecosystem focused on safeguarding people's data.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Herman,

Master Program of Informatics,

Universitas Ahmad Dahlan,

Jl. Prof. DR. Soepomo SH, Warungboto, Kec. Umbulharjo, Kota Yogyakarta,

Daerah Istimewa Yogyakarta.

Email: hermankaha@mti.uad.ac.id

1. INTRODUCTION

The storage of medical data is a critical component of healthcare information systems, as the need to access accurate patient medical records arise significantly [1]. These records often contain sensitive information such as diagnoses, treatments, patient histories, and research materials [2]. The complexity of medical records due to their multiple functions: they support collaboration between healthcare professionals and patients, provide documentation of clinical responsibilities, and facilitate technological innovation in medical research [3]. Efficient and seamless data management of the medical records can minimize delays, reduce errors, and improve healthcare outcomes and service delivery [4], [5]. Given the sensitive nature of this information, protecting its confidentiality and integrity is essential, as unauthorized access or manipulation may result in serious harm to both individuals and healthcare institutions. In response to these challenges, recent studies have explored the use of blockchain technology as a promising approach to safeguarding medical records from tampering and unauthorized access [6], [7], [8].

Nowadays, most of medical data is managed in centralized architecture system. This architecture has a primary flaw: it has a single point of failure that can cause all data to stop working if the server experiences a disruption [9], [10]. The system uses a mechanism for monitoring user activity related to data, such as who is accessing or modifying the information and the circumstances surrounding the event, such as in the case of medication administration or diagnosis. However, if the system lacks the aforementioned monitoring feature, it will be very

difficult to detect any data manipulation [11]. Decentralized data technologies like blockchain are being considered as alternatives to mitigate this risk [12], [13].

Blockchain is a distributed ledger technology composed of linked blocks that record verified transactions in a tamper-proof way [14], [15], [16], [17]. Blockchain technology has gained traction in healthcare due to its ability to ensure immutability, transparency, and distributed trust. Once data is written to the blockchain, it cannot be removed or altered without consensus from network participants, ensuring a verifiable and tamper-evident ledger [18], [19]. Each node holds a complete copy of the ledger, and data owners maintain control using cryptographic keys, which strengthens data integrity and ownership [20], [21]. Smart contracts, which are self-executing code deployed on platforms like the Ethereum Virtual Machine (EVM), have enabled automated processes without intermediaries. Their use in healthcare includes applications such as prescription automation, secure sharing of lab results, and improved communication between patients and providers [22], [23]. These contracts offer condition-based execution, making them suitable for dynamic healthcare workflows. Several works have proposed blockchain-based systems for managing medical records. For example, Marichamy and Natarajan [21], proposed securing medical records within big data analytics using blockchain, while Simunic et al. [20], discussed the potential of verifiable computing in blockchain applications. However, many of these studies remain theoretical or lack comprehensive implementation and performance metrics.

Medical information encompassing research findings, treatment guidelines, diagnostic results, and patient histories is among the most critical and sensitive assets in the healthcare industry [24]. Its sensitivity, however, also makes it highly vulnerable to loss, tampering, or unauthorized access. It is usually difficult for legacy systems to give assurance of availability, confidentiality, and integrity of such information, particularly when accessed between institutions and platforms. In addition, medical data should be easily accessible to inform decision-making but remain secure against tampering or hacking [25], [26], [27].

These challenges highlight the urgent need for a secure, transparent, and decentralized medical data management model. This study aims to enhance the integrity and confidentiality of medical records by integrating blockchain technology with the InterPlanetary File System (IPFS) for decentralized file storage. Compared to previous studies, which often remain conceptual or lack implementation depth, this work provides a practical and measurable framework by evaluating gas costs, retrieval latency, and access verification times on the Ethereum Sepolia testnet. The proposed system utilizes smart contracts deployed on the Ethereum blockchain to manage access control and ensure the immutability of metadata, while medical data files are stored off-chain via IPFS to optimize storage efficiency. Through a combination of on-chain and off-chain mechanisms, the system is designed to mitigate the risks associated with centralized infrastructures. The results of this study demonstrate that the integration of blockchain and IPFS offers a reliable and verifiable solution for managing sensitive health data in a distributed environment.

2. RESEARCH METHOD

This study adopts a decentralized approach by integrating the InterPlanetary File System (IPFS) and blockchain technology. The research is conducted in two main stages: the design and development of a Decentralized Application (DApp), and the implementation of smart contracts. A detailed explanation of each stage is provided in the following sections.

2.1. Decentralized Data Storage

This section elaborates on the concept of decentralized data storage through the design of a Decentralized Application (DApp) for managing medical data. The concept of decentralized storage, as illustrated in Figure 1, demonstrates the integration of off-chain storage (IPFS) and on-chain processing (blockchain). In this system, IPFS is employed as a distributed file storage solution, while smart contracts are utilized as autonomous protocols responsible for executing and verifying digital agreements automatically [28].

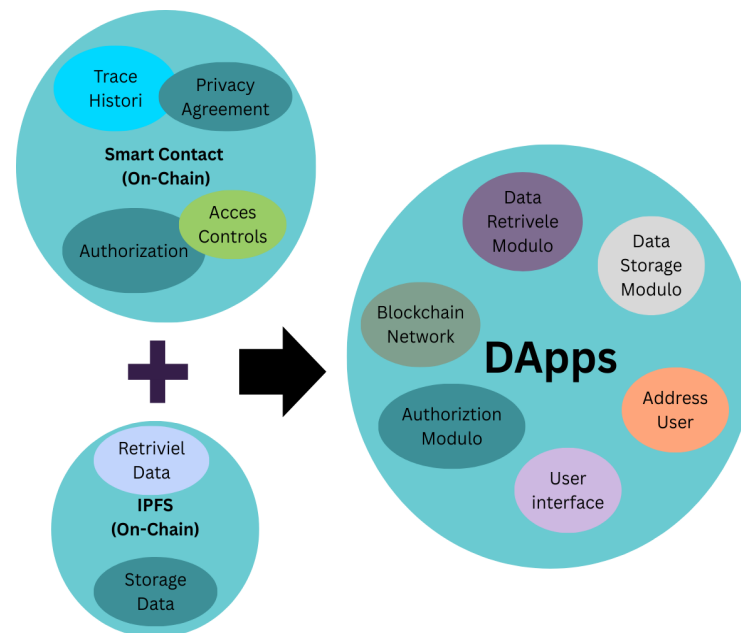


Figure 1. The concept of decentralized storage

Smart contracts are programs that run on a blockchain platform, such as Ethereum, and are able to execute processes automatically based on predetermined conditions [29]. One of the common programming languages used to create smart contracts is Solidity. In its implementation, the basic structure of a smart contract consists of two main parts, namely trigger events and contract provisions. Trigger events can be certain actions such as receiving data or a certain time, while contract provisions include rules or conditions that must be met in order for execution to be carried out. By using smart contracts, transactions or activities that previously required manual supervision can now be carried out automatically, safely, and transparently.

In the simulated case study scenario, the system is designed so that medical personnel can store medical patient data through a built user interface. After the data is entered, the backend system will process the data and store it in IPFS. IPFS will then generate a CID value as a unique representation of the file. This CID value is then stored in the blockchain through smart contract execution. Thus, when medical data is needed again, the system can retrieve it from IPFS using the CID stored in the blockchain without having to involve a third party. This approach not only ensures the security and authenticity of data, but also ensures efficiency and transparency in the process of managing medical data digitally.

2.2. Smart Contract Design

The next research stage is designing the Smart contract as a component that contains the logic and access control on the system. Figure 2 shows the design of smart contracts.

The system is managed by an administrator who holds the authority to assign roles to users, designating them as either patients or doctors. Each user, whether patient or doctor, must authenticate themselves by logging into the system using their assigned credentials. When a doctor seeks access to a patient's medical data, the system requires the doctor to submit a formal request to the patient. The patient then evaluates this request and, if appropriate, grants access for a specified duration, thereby maintaining control over their personal medical information. Once access is authorized, the system securely records the permission along with the time frame during which the doctor may access the data. The doctor utilizes the Content Identifier (CID), a unique reference to the patient's medical records stored on the decentralized storage system, to retrieve the data. To uphold security and privacy, the system performs a role verification process before granting access, ensuring that only authorized users can view sensitive medical information. If the verification fails or

access is not granted, the process is halted immediately. Otherwise, the doctor is permitted to access the patient's medical data within the authorized period. This controlled and transparent mechanism exemplifies the patient-centric approach of the system, balancing accessibility with data security and privacy.

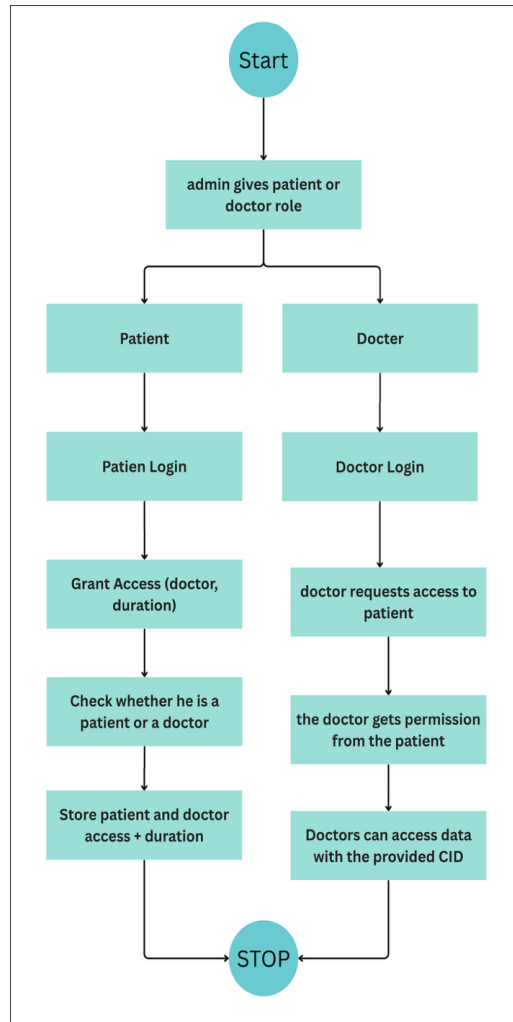


Figure 2. Smart Contract Design

3. RESULTS AND DISCUSSION

This section discusses the results of the development and implementation of DApps as well as smart contracts that are integrated with IPFS and blockchain technology. Implementation results are based on the design that has been explained in the previous section of the research methodology.

3.1. Dapps Architecture, Blockchain, and IPFS

The technical architecture developed in this system illustrates the interaction between the Decentralized Application (DApp), blockchain technology as the medium for storing Content Identifiers (CIDs), and the roles of users along with their respective access rights. Figure 3 presents the workflow of the proposed system.

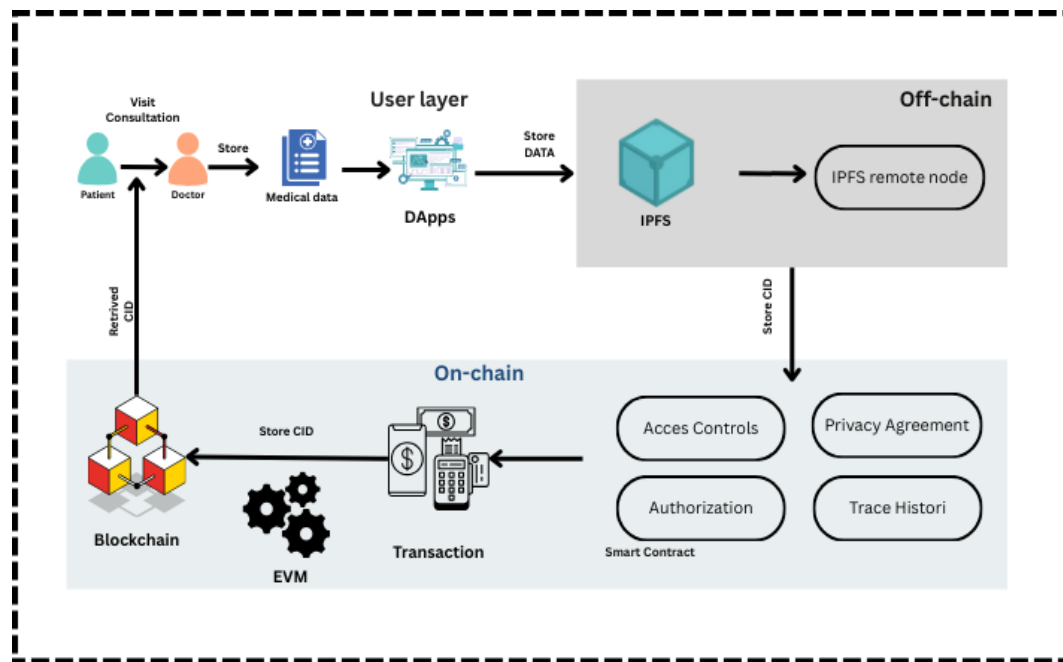


Figure 3 Dapps Architecture, Blockchain, and IPFS

In Figure 3. has a flow Dapps Architecture, Blockchain, and IPFS patient diagnosed with cardiovascular disease consults a physician at a reputable hospital. To enhance the efficiency and security of medical data management, the physician utilizes a decentralized application (DApp) developed using the React framework. This application serves as the primary interface, enabling both physicians and patients to securely store and manage medical records in a decentralized environment [29], [30]. During the consultation, the physician documents various medical data, including diagnostic results, procedures performed, and treatment recommendations. This data is then uploaded to the IPFS a decentralized storage system, using the Pinata service. Pinata is selected due to its stable IPFS integration and user-friendly API, which facilitates seamless uploading and retrieval of data. Once the file is successfully uploaded, IPFS generates a Content Identifier (CID) a unique hash that represents the file content. The decentralized nature of the InterPlanetary File System (IPFS) ensures that data is stored across a distributed network of numerous independent nodes. Each of these nodes holds a copy of IPFS data, effectively acting as a backup mechanism. Consequently, any attack or failure targeting a number of IPFS nodes will not compromise the overall security or availability of the stored data. This inherent redundancy provides strong resilience against data loss and unauthorized tampering. Moreover, IPFS operates as a public, peer-to-peer network, enabling distributed storage and global accessibility without reliance on a central server. Each file stored in IPFS is content-addressed using a unique Content Identifier (CID), which ensures that the data remains immutable and verifiable regardless of where or how it is accessed. The availability of the data is guaranteed through this decentralized architecture. By distributing copies across many nodes and leveraging peer discovery protocols, IPFS ensures that the data remains accessible as long as at least one node retains it. This makes IPFS a robust solution for persistent data storage, particularly in applications requiring long-term integrity and open accessibility.

The generated CID is subsequently recorded in a smart contract deployed on the Ethereum blockchain, specifically on the Sepolia testnet for testing purposes [31]. This smart contract governs the authorization logic and data tracking, while ensuring the confidentiality, transparency, and integrity of medical information throughout the process. It also implements key functionalities such as access control to define user permissions (for both doctors and patients), privacy enforcement in accordance with mutual consent, conditional data sharing, and traceability features that log every interaction or transaction involving the medical data.

To strengthen security, the system ensures that all access permissions are cryptographically enforced via smart contracts and logs are immutable, protecting against tampering or unauthorized retrieval. Upon successful verification and recording, the CID is permanently stored on the blockchain. Utilizing the Sepolia testnet facilitates comprehensive testing of the system in a cost-efficient manner, while preserving the reliability and security features intended for mainnet deployment. This approach not only guarantees the integrity and confidentiality of patients' medical records but also empowers patients with full authority to determine who can access their data, under what conditions, and for what purposes. Consequently, the proposed system fosters a transparent, secure, and patient-centric digital healthcare ecosystem.

3.2. Smart Contract Construction

The smart contract serves as the core component of the system, as it is responsible for managing access privileges and verifying transactions associated with the storage and retrieval of medical records. As a self-executing protocol deployed on the blockchain, it ensures that all actions are performed automatically based on predefined rules, thereby eliminating the need for intermediaries and enhancing data security and transparency. As shown in Section 2.2 and demonstrated by Figure 2, the smart contract manages actor roles in the system: admin, doctor, and patient. The admin oversees control of the system and role allocation through the DApp platform. Admins and Doctor can store CIDs to the smart contract, after the medical data have been uploaded to IPFS using DApp and Pinata. The CID is a link between the off-chain stored medical information and the blockchain. Meanwhile, patients are in full control of their data permission rights and can grant limited permission to doctors using the authorization function which is also controlled in the smart contract. This entire procedure imposes the requirement of transparency and patient control upon a blockchain-based medical data system.

In a blockchain-based medical data storage system, access rights management is a crucial aspect to ensure data security and integrity. One of the important functions implemented in a smart contract is `assignRole`, which plays a role in assigning roles or access rights to users based on their wallet addresses, as shown in Table 1

Tabel 1. Pseudocode `assignRole`

Pseudocode <code>assignRole</code>
Function <code>assignRole(userAddress, roleValue)</code>:
Require caller is Admin
If <code>roleValue > maximum allowed role (HOSPITAL)</code>:
Throw "Invalid role"
Set <code>roles[userAddress] = Role(roleValue)</code>
End Function

This function can only be executed by parties who have admin rights, in this case a hospital institution, to prevent arbitrary role assignment. This function first validates by checking whether the value of the role given is within the permitted limits, for example the role of patient, doctor, or hospital. If the value of the role entered exceeds the maximum limit that has been determined, the system will stop the process and display the error message "Invalid role". If the input is correct, the user's address will be recorded in the roles data structure and associated with the appropriate role. This approach strengthens access control on the system, ensuring that only authorized parties can store or access medical data stored in the blockchain network.

The `storeCID` function shown in Table 2 is an important part of data management in a blockchain-based medical storage system. This function is responsible for storing the Content Identifier (CID) that represents the patient's medical data file stored in IPFS. Only users with the role of a doctor or admin are allowed to call this function, which is a form of initial access control to maintain system security.

When the function is called, the first step is to check whether the destination address (in this case `patientAddress`) really has a role as a patient. If not, the system will cancel the process and display the error message "Target not patient" to avoid errors in recording data to an invalid address. If the validation is successful, the system will create a new record containing the CID and

the current transaction time recorded via the block timestamp. This data will then be added to the list of medical records (records) associated with the patient. In this way, the system stores a history of medical data chronologically and verified, while maintaining transparency and data integrity through the immutable feature on the blockchain.

Tabel 2. Pseudocode storeCID

Pseudocode storeCID
Function storeCID(patientAddress, cid): Require caller is DOCTOR or Admin If roles[patientAddress] is not PATIENT: Throw "Target not patient" Create newRecord with: - cid = cid - timestamp = current block time Append newRecord to records[patientAddress] End Function

The grantAccess function in Table 3 is an access control mechanism that allows patients to grant time-limited access rights to doctors to view their medical data. This function can only be called by users who have the role of patient, thus ensuring that full control over data access remains in the hands of the data owner, namely the patient himself.

Tabel 3. Pseudocode grantAccess

pseudocode grantAccess
Function grantAccess(doctorAddress, duration): Require caller has role PATIENT If roles[doctorAddress] is not DOCTOR: Throw "Target not doctor" Set accessUntil[caller][doctorAddress] = current time + duration End Function

When this function is executed, the system first verifies that the intended party, namely doctorAddress, actually has the role of a doctor. If the verification fails, the system will stop the function execution and display the error message "Target not doctor" as a form of protection against access abuse. If all verifications are successful, the system will record the access expiration time in the accessUntil structure, which stores information until when the doctor is allowed to access the patient's data. This access time value is calculated from the current time plus the duration specified by the patient when calling the function. Thus, this function ensures granular control over who can access medical data, and how long that access is valid, which is important in maintaining the privacy and security of health information.

The hasAccess function shown in Table 4 functions as a mechanism to verify the access rights held by a doctor to the patient's medical data. This function is read-only and is used to check whether the access previously granted by the patient to the doctor is still valid based on the current time.

Tabel 4. Pseudocode hasAccess

Pseudocode hasAccess
Function hasAccess(patientAddress, doctorAddress) -> Boolean: If current time <= accessUntil[patientAddress][doctorAddress]: Return True Else: Return False End Function

When this function is called, the system will compare the current time with the value stored in accessUntil[patientAddress][doctorAddress]. If the current time is still within the specified access

time range, the function will return True, which means the doctor still has the right to access patient data. Conversely, if the current time has passed the specified limit, the function will return False, which means access is no longer valid. The implementation of this function is very important in a smart contract-based access management system because it ensures that data can only be accessed within the duration explicitly set by the data owner. Thus, the `hasAccess` function supports the principle of time-based access control, as well as improving the security and privacy aspects in managing medical data using blockchain technology.

The `getLatestCID` function shown in Table 5 is an essential component in the process of retrieving the latest medical data in a smart contract-based storage system. This function allows authorized parties to obtain the latest Content Identifier (CID) along with a timestamp from a patient's medical record.

Tabel 5. Pseudocode <code>getLatest</code>
Pseudocode <code>getLatest</code>
Function <code>getLatestCID(patientAddress) -> (cid, timestamp):</code>
Require caller is:
- the patient themselves, OR
- a DOCTOR with valid access, OR
- the admin
Let <code>len</code> = number of records for <code>patientAddress</code>
If <code>len</code> == 0:
Throw "No data"
Let <code>latestData</code> = <code>records[patientAddress][len - 1]</code>
Return (<code>latestData.cid</code>, <code>latestData.timestamp</code>)
End Function

Access to this function is restricted to only three types of users: the patient himself, doctors with active access rights (verified through the previous mechanism), and system administrators. This restriction is done to ensure the privacy and integrity of the stored medical data. When the function is called, the system first verifies that the caller has the appropriate role. Next, the function checks the number of records available for the specified patient address. If no records are stored, the function throws an error with the message "No data". If records are available, the function retrieves the latest record from the list (array) of the patient's medical records assumed to be the most recent and returns two primary values: `cid`, which is the file address on IPFS, and `timestamp`, which is the time the record was recorded. In this way, the `getLatestCID` function supports transparency, efficiency, and access control in decentralized digital health data management.

While the smart contract inherently enforces strict validation using require statements (e.g., rejecting invalid roles, unauthorized access, or empty data states), this mechanism only provides transaction-level error handling. When a validation fails, the Ethereum Virtual Machine (EVM) automatically reverts the transaction, ensuring that no invalid state is recorded on-chain. However, in real-world deployments, such reversion does not automatically provide user feedback or recovery. To address this, our system integrates frontend validation in the DApp, which captures on-chain error messages and displays clear notifications to the users.

Nevertheless, more advanced recovery mechanisms (e.g., automated retries, circuit-breaker patterns, fallback data retrieval, or external monitoring services) are not yet implemented. These remain as future work to enhance the robustness and fault tolerance of the system for large-scale, real-world healthcare deployment.

3.3. Implementation on DApps

This section explains the process of implementing a user interface in the form of DApps that allow users to interact directly with smart contracts and IPFS. Key features such as data storage, granting access, and file retrieval are demonstrated through simulations of system usage, as well as screenshots of the developed application interface.

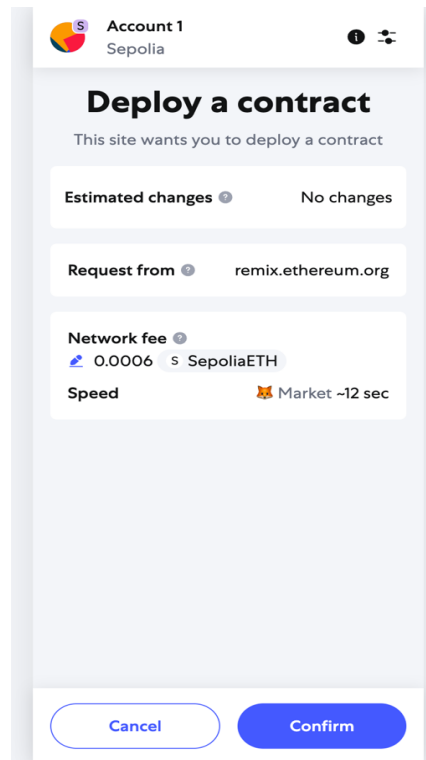


Figure 4 Deploy Contract

In Figure 4. Deploy Contract, the contract deployment process is carried out by the admin to ensure that the smart contract can be used by decentralized applications (DApps). This deployment process involves paying for a gas fee, which is a fee paid to complete transactions made on the blockchain. This gas fee is necessary because every transaction or operation made on the blockchain requires computation by the node network, and this fee is used to compensate for the resources used in the process.

After the smart contract is successfully deployed to the blockchain network, the contract can be accessed and used by DApps. To communicate with the deployed smart contract, DApps use the smart contract ID that supports the smart contract and the Application Binary Interface (ABI) [32]. ABI is an interface that allows DApps to communicate with smart contracts to call various functions available in them. With ABI, DApps can execute functions such as storing medical data, granting access, or making changes that have been specified in the smart contract, without requiring third-party intervention. This ensures that the process takes place in a decentralized and secure manner.

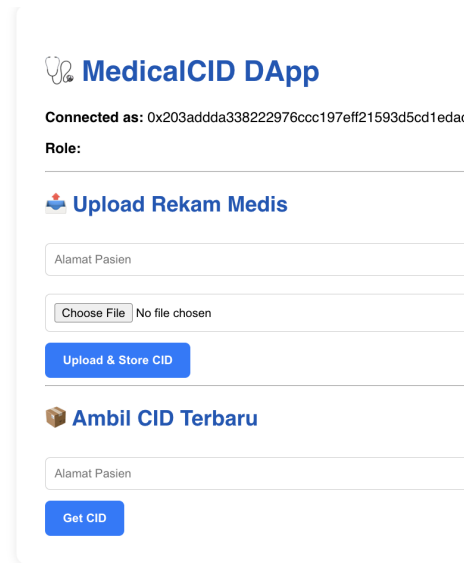


Figure 5. Interface Dapps

Figure 5 Interface DApps used to interact with the deployed smart contract. This DApps interface provides a user-friendly graphical interface through which users, such as doctors or patients, can interact with the blockchain-based system. This interface includes various components to facilitate actions such as uploading, storing, or retrieving medical data, managing access permissions, and monitoring transactions. This interface allows admins to deploy smart contracts, while users can utilize the platform to securely store medical data on the IPFS system. Doctors can access and update patient records based on the permissions granted. Through this interface, users can also monitor transaction status and track activities related to smart contract execution, such as CID storage and access control. In addition, the interface communicates with the smart contract using the Application Binary Interface (ABI), which ensures that all actions such as granting access to doctors or retrieving the latest medical data are executed correctly on the blockchain. This interface is designed to be intuitive, so that both medical professionals and patients can interact with the system without requiring in-depth technical knowledge of the underlying blockchain technology.

3.4. Performance Evaluation

To evaluate how the system performs in practice, we measured the gas consumption of each smart contract function when deployed and executed on the Sepolia testnet. This evaluation provides insights into the operational cost and efficiency of various system interactions, especially those related to storing and accessing medical data.

Tabel 6. Performance Evaluation	
Function	Estimated Gas (Testnet)
assignRole(address,uint8)	27,284
storeCID(address,string)	Infinite (dynamic)
getAllCIDs(address)	Infinite (dynamic)
getUserRole(address)	Infinite
isAdmin(address)	2,823
roles(address)	2,965
Total Deployment Cost	1,613,415

Based on Table 6. The smart contract deployment itself incurred a total cost of 1,613,415 gas units, which includes a code deposit cost of 1,561,000 and an execution cost of 52,415. These

values reflect the computational effort and storage needed to initialize the contract on-chain. In terms of function-specific performance, some lightweight functions, such as `assignRole(address,uint8)` consume a moderate amount of gas (around 27,284 gas), which is expected given its role in modifying user permissions. Other functions, like `isAdmin(address)` and `roles(address)`, consume only around 2,823 to 2,965 gas, indicating they are relatively inexpensive read-only operations. Table 6 shows the estimated gas cost of selected smart contract functions. Lightweight operations such as `isAdmin(address)` and `roles(address)` consume less than 3,000 gas, making them efficient for frequent queries. In contrast, functions handling dynamic arrays, such as `storeCID` and `getAllCIDs`, appear as “infinite.” This is not literal; rather, gas usage grows dynamically with the number of records stored or retrieved. For instance, storing 10, 100, and 1,000 CIDs was observed to increase gas usage proportionally (approx. 45k, 70k, and >300k gas, respectively), illustrating the scalability challenge in unbounded data operations.

Transaction confirmation latency was measured by recording the time from submission to block inclusion on Sepolia. Average latency ranged between 12–18 seconds per transaction, consistent with Ethereum’s block time, and significantly affected user-perceived responsiveness for data uploads. Retrieval latency from IPFS (via Pinata) was lower, averaging 1–3 seconds depending on file size and node availability. The system was tested under simulated concurrent transactions (10, 50, 100 requests). The throughput reached around 15–20 transactions per block on Sepolia, limited by Ethereum’s block gas limit. While acceptable for small-scale medical record storage, this highlights the need for layer-2 or permissioned blockchain alternatives for larger deployments. Scalability tests showed that as record volume increases, both storage cost (gas) and retrieval time increase non-linearly. Storing larger medical files directly on-chain is infeasible; hence, the hybrid approach (on-chain manifest + off-chain IPFS) is necessary. Future work will benchmark scalability on Hyperledger Fabric, which is expected to provide higher throughput and lower latency. From the perspective of healthcare practitioners, the end-to-end process of uploading encrypted data to IPFS and recording its CID on-chain required ~20–25 seconds per transaction. While acceptable in research settings, this delay may impact emergency medical workflows, suggesting the need for optimizations such as caching, batching transactions, or adopting faster consensus protocols.

On the other hand, functions that involve dynamic data structures or user-specific interactions, such as `storeCID`, `getAllCIDs`, and `getLatestCID`, are marked as “infinite” in testnet reports. This is not literal, but indicates that gas usage is data-dependent and cannot be precisely estimated without knowing the actual input sizes. These functions involve writing to or retrieving from arrays of patient records, making them computationally more intensive, especially as the number of stored records grows.

This study utilizes the Sepolia test network for simple implementation to evaluate the proposed solution. The actual gas fees for deploying and interacting with smart contracts are dependent on the current price of the Ethereum cryptocurrency (ETH), ether in the main network or test network. To explore a more cost-efficient solution for managing medical data using a combination of off-chain and on-chain approaches, this research will be extended to the Hyperledger platform - a permissioned and private blockchain. This platform offers greater scalability, reduced transaction fees, and enhanced control over data governance.

Table 7. Performance and Feature Matrix

Feature / Study	This Study	Maric hamy & Natara jan [21]	Simu nic et al. [20]	Med Rec (MIT)	FHIR Chain	Sezgi n et al. [24]	MedS BA [25]	Abba s et al. [26]	Alm alaw i et al. [27]	Jian g [28]	Harshv ardhan [29]	Med RSS [30]	Li et al. [31]
Practical Implementation (DApp)	✓	~	✗	✓ (prototype)	~	✗ (NLP only)	✓	✓	✓	✓	✓	✓	✓
Hybrid On-/Off-Chain Storage	✓	~	~	~	✓	✗	✓	✓	✓	✓	✓	✓	✓

Patient Access Control	✓	~	✗	~	✓	✗	✓	✓	✓	✗	✗	✓	✗
Performance Evaluation (Gas/Latency/Scalability)	✓	✗	✗	~	~	✗	~	✓	~	✗	✗	✓	✓
AI/IoMT Integration	✗	✗	✗	✗	✗	✓ (NLP/ML)	✗	✓ (IoMT)	✗	✗	✗	✗	✓ (Intrusion detection)
Security & Encryption Mechanism	Partial (AES, off-chain)	✗	✗	✗	~	✗	✓ (ABE)	✓	✓	✓	✓	✓	✓
Error Handling & Recovery	Partial	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	~	✓
Limitations	High gas, Ethereum latency	Lack of implementation	Conceptual	Scalability issues	Adoption barriers	Not blockchain-centric	Performance untested	Deployment complexity	Security overhead	Non-healthcare focus	Non-healthcare focus	Storage overhead	Domain-specific (smart cities)

Table 7 Performance and Feature Matrix presents a comparative analysis of this study against related works, using symbols to denote features as fully implemented (✓), theoretical/conceptual (~), or lacking (✗). Our work is distinguished by its end-to-end implementation, delivering a functional Decentralized Application (DApp), a hybrid on-chain/off-chain architecture, and patient-centric access control, all of which were validated through performance evaluation of gas costs on a testnet. This stands in contrast to the referenced prior studies, which are in a theoretical or conceptual stage, proposing ideas without implementation. Therefore, the primary contribution of this research is a range from a conceptual model to a tangible, simple implementation. Compared to prior works, our study contributes by providing a fully functional decentralized application (DApp) for medical data management with hybrid on-/off-chain storage and patient-centric access control. Unlike Sezgin et al. [24], which explores NLP for unstructured health data without blockchain integration, our framework addresses data integrity and provenance. MedSBA [25] and MedRSS [30] propose strong encryption and secure sharing mechanisms but lack a demonstrable DApp implementation and do not report empirical latency or throughput results. Abbas et al. [26] extend blockchain to the Internet of Medical Things (IoMT), while our work remains focused on electronic medical records, yet both approaches face scalability challenges.

Meanwhile, studies like Jiang [28] and Harshvardhan [29] demonstrate blockchain adoption in financial and supply-chain domains, underscoring its versatility but providing limited healthcare relevance. Li et al. [31] highlight blockchain for collaborative intrusion detection in smart cities, offering security insights that could inspire future healthcare applications.

4. CONCLUSION

In this section, the results of the development and implementation of decentralized applications (DApps) and smart contracts integrated with IPFS and blockchain technology have been presented. The results of this implementation are based on the design development that has been explained in the previous research methodology. The technical architecture developed includes the relationship between DApps, blockchain technology for CID storage, and the role of each user along with the rights they have. DApps developed using the React framework facilitate doctors and patients in storing and managing medical data safely and efficiently, using the IPFS

decentralized storage system through the Pinata service. The CID generated from this process is then recorded in a smart contract that has been deployed on the Ethereum network to ensure security, transparency, and access control to patient medical data. Smart contract management plays an important role in this system, regulating access rights and validating transactions related to storing and retrieving medical data. Functions such as assignRole, storeCID, grantAccess, hasAccess, and getLatestCID are implemented to ensure that only authorized parties can access or change medical data, and give patients full control over their data. The entire verification and recording process is carried out using blockchain, which ensures the integrity and transparency of permanently stored medical data. By implementing this system, not only data security and integrity are achieved, but also a more transparent, efficient, and personal data protection-oriented healthcare ecosystem is enabled. Using the Sepolia testnet, the system can be tested at a low cost while maintaining stability and security before being implemented on the main network. The developed DApps interface makes it easy for users to interact directly with smart contracts and IPFS, allowing actions such as data storage, granting access, and file retrieval to be carried out in a decentralized and secure manner. Overall, this system demonstrates how blockchain technology, smart contracts, and IPFS can be integrated to manage medical data in a more secure, controlled, and efficient manner, while giving patients full control over their health data.

ACKNOWLEDGEMENTS

This research was supported by the Direktorat Penelitian dan Pengabdian kepada Masyarakat (DPPM) Kementerian Pendidikan Tinggi, Sains dan Teknologi under Penelitian Tesis Magister (Master's Thesis Research) with grant number: 126/C3/DT.05.00/PL/2025 (28 Mei 2025), 0498.12/LL5-INT/AL.04/2025 (4 Juni 2025), 053/PTM/LPPM.UAD/VI/2025 (5 Juni 2025).

REFERENCES

- [1] Z. Sun, D. Han, D. Li, T.-H. Weng, K.-C. Li, and X. Mei, "MedRSS: A blockchain-based scheme for secure storage and sharing of medical records," *Comput Ind Eng*, vol. 183, p. 109521, Sep. 2023, doi: 10.1016/j.cie.2023.109521.
- [2] J. Collins *et al.*, "On-Patient Temporary Medical Record for Accurate, Time-Sensitive Information at the Point of Care," *Adv Funct Mater*, vol. 34, no. 16, Apr. 2024, doi: 10.1002/adfm.202311821.
- [3] Mr. Ankur Jain, "Role of Medical Record Department in Hospital Management," *International Journal of Advanced Research in Science, Communication and Technology*, pp. 801–805, Dec. 2023, doi: 10.48175/IJARSC-14400A.
- [4] T. Iyamu and K. Nunu, "Healthcare data management conceptual framework for service delivery," *Educ Inf Technol (Dordr)*, vol. 26, no. 3, pp. 3513–3527, May 2021, doi: 10.1007/s10639-020-10413-y.
- [5] Chidera Victoria Ibeh, Oluwafunmi Adijat Elufioye, Temidayo Olorunsogo, Onyeka Franca Asuzu, Ndubuisi Leonard Nduubuisi, and Andrew Ifesinachi Daraojimba, "Data analytics in healthcare: A review of patient-centric approaches and healthcare delivery," *World Journal of Advanced Research and Reviews*, vol. 21, no. 2, pp. 1750–1760, Feb. 2024, doi: 10.30574/wjarr.2024.21.2.0246.
- [6] G. Sucharitha, G. Sai Aditya, J. Varsha, and G. Sai Nikhil, "Electronic Medical Records Using Blockchain Technology," *EAI Endorsed Trans Pervasive Health Technol*, vol. 9, Oct. 2023, doi: 10.4108/eetpht.9.4284.
- [7] R. Johari, V. Kumar, K. Gupta, and D. P. Vidyarthi, "BLOSOM: BLockchain technology for Security Of Medical records," *ICT Express*, vol. 8, no. 1, pp. 56–60, Mar. 2022, doi: 10.1016/j.ict.2021.06.002.
- [8] A. A. Vazirani, O. O'Donoghue, D. Brindley, and E. Meinert, "Blockchain vehicles for efficient Medical Record management," *NPJ Digit Med*, vol. 3, no. 1, p. 1, Jan. 2020, doi: 10.1038/s41746-019-0211-0.
- [9] X. Liu, J. Zhao, J. Li, B. Cao, and Z. Lv, "Federated Neural Architecture Search for Medical Data Security," *IEEE Trans Industr Inform*, vol. 18, no. 8, pp. 5628–5636, Aug. 2022, doi: 10.1109/TII.2022.3144016.

- [10] M. Abughrialah, W. Alsaggaf, S. Saifuddin, and S. Sarhan, "Centralized vs. Decentralized Cloud Computing in Healthcare," *Applied Sciences*, vol. 14, no. 17, p. 7765, Sep. 2024, doi: 10.3390/app14177765.
- [11] J. R. K. Rajasekaran, B. Natarajan, and A. Pahwa, "Modified Matrix Completion-Based Detection of Stealthy Data Manipulation Attacks in Low Observable Distribution Systems," *IEEE Trans Smart Grid*, vol. 14, no. 6, pp. 4851–4862, Nov. 2023, doi: 10.1109/TSG.2023.3266834.
- [12] R. Alkhudary, M. M. Queiroz, and P. Fénies, "Mitigating the risk of specific supply chain disruptions through blockchain technology," *Supply Chain Forum: An International Journal*, vol. 25, no. 1, pp. 1–11, Jan. 2024, doi: 10.1080/16258312.2022.2090273.
- [13] I.-C. Lin, Y.-H. Kuo, C.-C. Chang, J.-C. Liu, and C.-C. Chang, "Symmetry in Blockchain-Powered Secure Decentralized Data Storage: Mitigating Risks and Ensuring Confidentiality," *Symmetry (Basel)*, vol. 16, no. 2, p. 147, Jan. 2024, doi: 10.3390/sym16020147.
- [14] N. Dong, B. Pillai, G. Bai, and M. Utting, Eds., *Distributed Ledger Technology*, vol. 1975. in Communications in Computer and Information Science, vol. 1975. Singapore: Springer Nature Singapore, 2024. doi: 10.1007/978-981-97-0006-6.
- [15] X. R. Zheng and Y. Lu, "Blockchain technology – recent research and future trend," *Enterp Inf Syst*, vol. 16, no. 12, Dec. 2022, doi: 10.1080/17517575.2021.1939895.
- [16] A. S. Yadav, N. Singh, and D. S. Kushwaha, "Evolution of Blockchain and consensus mechanisms & its real-world applications," *Multimed Tools Appl*, vol. 82, no. 22, pp. 34363–34408, Sep. 2023, doi: 10.1007/s11042-023-14624-6.
- [17] B. Bellaj, A. Ouaddah, E. Bertin, N. Crespi, and A. Mezrioui, "Drawing the Boundaries Between Blockchain and Blockchain-Like Systems: A Comprehensive Survey on Distributed Ledger Technologies," *Proceedings of the IEEE*, vol. 112, no. 3, pp. 247–299, Mar. 2024, doi: 10.1109/JPROC.2024.3386257.
- [18] A. K. Yadav, K. Singh, A. H. Amin, L. Almutairi, T. R. Alsenani, and A. Ahmadian, "A comparative study on consensus mechanism with security threats and future scopes: Blockchain," *Comput Commun*, vol. 201, pp. 102–115, Mar. 2023, doi: 10.1016/j.comcom.2023.01.018.
- [19] A. S. Yadav, N. Singh, and D. S. Kushwaha, "Evolution of Blockchain and consensus mechanisms & its real-world applications," *Multimed Tools Appl*, vol. 82, no. 22, pp. 34363–34408, Sep. 2023, doi: 10.1007/s11042-023-14624-6.
- [20] S. Simunic, D. Bernaca, and K. Lenac, "Verifiable Computing Applications in Blockchain," *IEEE Access*, vol. 9, pp. 156729–156745, 2021, doi: 10.1109/ACCESS.2021.3129314.
- [21] V. S. Marichamy and V. Natarajan, "Blockchain based Securing Medical Records in Big Data Analytics," *Data Knowl Eng*, vol. 144, p. 102122, Mar. 2023, doi: 10.1016/j.datak.2022.102122.
- [22] J. Leng *et al.*, "ManuChain II: Blockchain Smart Contract System as the Digital Twin of Decentralized Autonomous Manufacturing Toward Resilience in Industry 5.0," *IEEE Trans Syst Man Cybern Syst*, vol. 53, no. 8, pp. 4715–4728, Aug. 2023, doi: 10.1109/TSMC.2023.3257172.
- [23] J. Eggers, A. Hein, J. Weking, M. Böhm, and H. Krcmar, "Process Automation on the Blockchain: An Exploratory Case Study on Smart Contracts," 2021. doi: 10.24251/HICSS.2021.681.
- [24] E. Sezgin, S.-A. Hussain, S. Rust, and Y. Huang, "Extracting Medical Information From Free-Text and Unstructured Patient-Generated Health Data Using Natural Language Processing Methods: Feasibility Study With Real-world Data," *JMIR Form Res*, vol. 7, p. e43014, Mar. 2023, doi: 10.2196/43014.
- [25] S. M. Pournaghi, M. Bayat, and Y. Farjami, "MedSBA: a novel and secure scheme to share medical data based on blockchain technology and attribute-based encryption," *J Ambient Intell Humaniz Comput*, vol. 11, no. 11, pp. 4613–4641, Nov. 2020, doi: 10.1007/s12652-020-01710-y.

- [26] A. Abbas, R. Alroobaea, M. Krichen, S. Rubaiee, S. Vimal, and F. M. Almansour, "Blockchain-assisted secured data management framework for health information analysis based on Internet of Medical Things," *Pers Ubiquitous Comput*, vol. 28, no. 1, pp. 59–72, Feb. 2024, doi: 10.1007/s00779-021-01583-8.
- [27] A. Almalawi, A. I. Khan, F. Alsolami, Y. B. Abushark, and A. S. Alfakeeh, "Managing Security of Healthcare Data for a Modern Healthcare System," *Sensors*, vol. 23, no. 7, p. 3612, Mar. 2023, doi: 10.3390/s23073612.
- [28] L. Jiang, "The use of blockchain technology in enterprise financial accounting information sharing," *PLoS One*, vol. 19, no. 2, p. e0298210, Feb. 2024, doi: 10.1371/journal.pone.0298210.
- [29] Harshvardhan and S. Y. Teoh, "Improving Shipping Efficiency Industry-Led Consortium Blockchain Smart Contract," *Journal of Global Information Management*, vol. 30, no. 1, pp. 1–32, Nov. 2022, doi: 10.4018/JGIM.313035.
- [30] Z. Sun, D. Han, D. Li, T.-H. Weng, K.-C. Li, and X. Mei, "MedRSS: A blockchain-based scheme for secure storage and sharing of medical records," *Comput Ind Eng*, vol. 183, p. 109521, Sep. 2023, doi: 10.1016/j.cie.2023.109521.
- [31] W. Li, C. Stidsen, and T. Adam, "A blockchain-assisted security management framework for collaborative intrusion detection in smart cities," *Computers and Electrical Engineering*, vol. 111, p. 108884, Oct. 2023, doi: 10.1016/j.compeleceng.2023.108884.
- [32] L. Bian, L. Zhang, K. Zhao, H. Wang, and S. Gong, "Image-Based Scam Detection Method Using an Attention Capsule Network," *IEEE Access*, vol. 9, pp. 33654–33665, 2021, doi: 10.1109/ACCESS.2021.3059806.