International Journal of Multidisciplinary Engineering in Current Research - IJMEC

Volume 11, Issue 4, April-2026, http://ijmec.com/, ISSN: 2456-4265

Full Length Article

Blockchain-Based Medical Report Management System: A Secure And Decentralized Approach To Healthcare Data Exchange

Mohammed Zaid[1], Rayan Mohammed Khaled[2], Saad Ahmed Aijaz[3], Syed Ahmed Ali[4], Dr.T K Shaik Shavali[5]

[1,2,3,4]btech Students Department Of Computer Science And Engineering, Lords Institute Of Engineering And Technology, Hyderabad, India

[5]Professor Department Of Computer Science And Engineering, Lords Institute Of Engineering And Technology, Hyderabad, India

zaidmohammed25502@gmail.com, rayanmohammedkhaled@gmail.com, Y@gmail.com, syedahmad112900@gmail.com, shaikshavali@lords.ac.in

Accepted 18-04-2026

Abstract—

Healthcare data management faces critical challenges including unauthorized access, lack of patient control, absence of verifiable audit trails, and vulnerability to data tampering. Centralized storage architectures create single points of failure, compromising patient privacy and data integrity. This paper presents a blockchain-based Medical Report Management and Distribution System that leverages cryptographic hashing and decentralized ledger technology to provide secure, tamper-proof storage and sharing of medical records. The system implements a custom blockchain using SHA-256 cryptographic hashing, Flask web framework, and SQLite database. Patients retain full ownership and control over their medical reports through a granular access control mechanism, granting and revoking permissions to healthcare providers. Every significant system action—report upload, access grant, access revocation, and file download—is recorded as an immutable block in the blockchain, creating a comprehensive audit trail. The system achieves average page response time under 150ms, database query time under 10ms, and blockchain verification time of 15ms for 100 blocks. Security features include Werkzeug password hashing, parameterized SQL queries preventing injection attacks, and file integrity verification through SHA-256 checksums. A responsive Bootstrap 5 interface with Chart.js visualizations provides analytics on report distribution, user roles, upload trends, and access patterns. Experimental results demonstrate successful integration testing with 100% pass rate across authentication, blockchain recording, access control, and security features.

Keywords—*Blockchain, Medical Records, Healthcare Data Security, SHA-256, Cryptographic Hashing, Access Control, Audit Trail, Flask, Patient Privacy, Data Integrity, Decentralization.*

I. INTRODUCTION

Healthcare systems worldwide generate enormous volumes of medical data daily, including laboratory test results, imaging reports, prescriptions, diagnostic summaries, and surgical records. The secure management, storage, and distribution of these medical records fundamentally impacts patient care quality, healthcare system efficiency, and regulatory compliance. Electronic Health Record (EHR) systems have replaced paper-based records in most modern healthcare facilities, enabling digital storage, retrieval, and sharing of patient information across departments and institutions.

However, current medical record management systems face critical vulnerabilities. Centralized storage architectures create single points of failure where security breaches can expose entire patient databases. Hospital information systems typically grant broad access privileges to healthcare providers without granular patient consent mechanisms. Once uploaded, medical reports become accessible to any authorized staff member within the institution, with patients having minimal visibility or control over who accesses their sensitive health information.

Furthermore, existing audit trail mechanisms are inadequate. While some EHR systems maintain access logs, these logs reside in the same database as the medical records, making them susceptible to modification or deletion by attackers who compromise the system. The lack of tamper-proof audit trails creates accountability gaps, making it difficult to detect unauthorized access or data manipulation. Medical identity theft and insurance fraud remain significant problems, with the U.S. Department of

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

Health and Human Services reporting over 45 million patient records breached between 2009 and 2020.

A. Blockchain Technology in Healthcare

Blockchain technology, originally developed as the foundation for Bitcoin cryptocurrency in 2008, offers a promising solution to healthcare data management challenges. A blockchain is a distributed, immutable ledger that records transactions in a chain of cryptographically linked blocks. Each block contains a timestamp, transaction data, and a cryptographic hash of the previous block, creating an unbreakable chain of trust. Any attempt to modify historical data would invalidate all subsequent blocks, making tampering immediately detectable.

Key advantages of blockchain for healthcare include: (1) Immutability: Once recorded, data cannot be altered without detection, ensuring audit trail integrity. (2) Transparency: All participants can verify the blockchain's state, creating accountability. (3) Decentralization: No single entity controls the entire system, eliminating central points of failure. (4) Cryptographic security: SHA-256 hashing and digital signatures provide strong data integrity guarantees. (5) Patient empowerment: Patients can control access to their medical records through smart contracts or access control mechanisms.

B. Research Contributions

- Design and implementation of a custom SHA-256 blockchain for healthcare data integrity verification.
- Patient-controlled access management system enabling granular permissions for medical record sharing.
- Immutable audit trail recording every significant action (upload, access grant/revoke, download) in blockchain blocks.
- Role-based access control (RBAC) supporting Admin, Doctor, and Patient roles with distinct privileges.
- Secure file handling with cryptographic checksums for medical report integrity verification.
- Responsive web application with Bootstrap 5 dark theme and Chart.js analytics dashboard.
- Comprehensive security implementation including password hashing, SQL injection prevention, and session management.
- Performance evaluation demonstrating sub-150ms response times and efficient blockchain verification.

II. RELATED WORK

Recent research has explored blockchain applications in healthcare, focusing on access control, data sharing, and interoperability. This section reviews key contributions and identifies research gaps addressed by our system.

A. MedRec: Patient-Controlled Access

Azaria et al. (2016) proposed MedRec, a decentralized record management system using Ethereum blockchain and smart contracts. MedRec enables patients to grant and revoke access permissions through smart contract execution, creating an immutable log of data access. The system maintains metadata on the blockchain while storing actual medical data off-chain in existing provider databases. Key advantages include patient autonomy and interoperability across healthcare providers. However, Ethereum's proof-of-work consensus requires significant computational resources, and smart contract vulnerabilities pose security risks. Gas fees for Ethereum transactions make the system economically impractical for high-frequency medical data operations.

B. Healthcare Blockchain Taxonomy

Mettler (2016) conducted a comprehensive survey of blockchain applications in healthcare, establishing a six-category taxonomy: data sharing, data storage, access control, interoperability, consent management, and clinical trials. The analysis identified key challenges including scalability limitations (Bitcoin processes 7 transactions/second vs. Visa's 24,000/second), regulatory compliance with HIPAA and GDPR, integration with legacy EHR systems, and standardization of blockchain protocols across healthcare institutions. The paper emphasized the need for permissioned blockchains over public chains for healthcare applications due to privacy requirements and regulatory constraints.

C. FHIR-Based Clinical Data Sharing

Zhang et al. (2018) developed FHIRChain, integrating Fast Healthcare Interoperability Resources (FHIR) standards with permissioned blockchain for clinical data sharing. The system uses HL7 FHIR APIs to extract structured clinical data from EHR systems, stores the data in blockchain blocks, and provides RESTful APIs for querying. FHIRChain demonstrates successful integration between blockchain and existing healthcare IT infrastructure. However, the system requires significant infrastructure modifications and lacks fine-grained patient control over individual data elements.

D. Research Gap and Our Contribution

Existing blockchain healthcare solutions face limitations: (1) Complexity: Ethereum smart contracts and permissioned blockchain networks require specialized blockchain expertise and significant

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

infrastructure. (2) Cost: Public blockchain transaction fees make frequent medical data operations economically infeasible. (3) Limited patient control: Many systems provide coarse-grained access control (all-or-nothing) rather than report-level permissions. (4) Integration challenges: Complex integration requirements with existing EHR systems hinder adoption. Our system addresses these gaps through: (1) Lightweight implementation: Custom SHA-256 blockchain without consensus overhead, deployable on standard web servers. (2) Zero transaction costs: No cryptocurrency or gas fees for blockchain operations. (3) Granular access control: Per-report permissions enabling selective sharing. (4) Standalone architecture: No complex EHR integration required, suitable for clinics and small healthcare facilities.

TABLE I COMPARISON OF BLOCKCHAIN HEALTHCARE SYSTEMS

System	Blockchain Type	Access Control	Patient Control	Transaction Cost	Integration
MedRec (2016)	Ethereum	Smart Contracts	Coarse	High	Complex
FHIRChain (2018)	Permissioned	HL7 FHIR	Medium	Medium	Very Complex
ONC Blockchain (2016)	General Framework	Policy-based	Low	Low	Moderate
Our System	Custom SHA-256	Role-based	Granular	None	Simple

III. SYSTEM ARCHITECTURE AND DESIGN

3. Architectural Overview

The system employs a three-tier architecture: Presentation Layer (web interface), Application Layer (Flask backend with blockchain logic), and Data Layer (SQLite database and file storage). The blockchain component integrates horizontally across all tiers, recording transactions and providing integrity verification.

TABLE II SYSTEM ARCHITECTURE COMPONENTS

Layer	Technology	Purpose
Presentation	HTML5/CSS3/Bootstrap 5	Responsive dark-themed UI
Frontend Logic	JavaScript/Chart.js	Interactive analytics dashboard
Application	Python 3.11+/Flask 3.0+	Web framework, routing, templates
Blockchain	Custom SHA-256 (hashlib)	Block/Blockchain classes
Database	SQLite 3.x	5 tables: users, reports, grants, blocks, logs
Security	Werkzeug 3.0+	Password hashing, SQL injection prevention
File Storage	Local filesystem	Uploaded medical reports (PDF, images)

3.1 Use Case Diagram

The use case diagram illustrates the interactions between the three system actors (Admin, Doctor, and Patient) and the key functionalities of the Medical Report Management and Distribution System. The Admin actor can manage users, view audit logs, verify blockchain integrity, and access the analytics dashboard. The Doctor actor can upload medical reports for patients, view reports they have been granted access to, and browse the blockchain explorer. The Patient actor can view their own medical reports, grant and revoke access to specific doctors, download reports, and view the audit trail of actions performed on their records.

Common use cases shared across all roles include user registration, login, logout, viewing the blockchain explorer, and accessing the analytics dashboard. The use case diagram highlights the patient-centric design of the system, where patients have the most granular control over access to medical records, aligning with the principle of patient sovereignty over healthcare data.

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

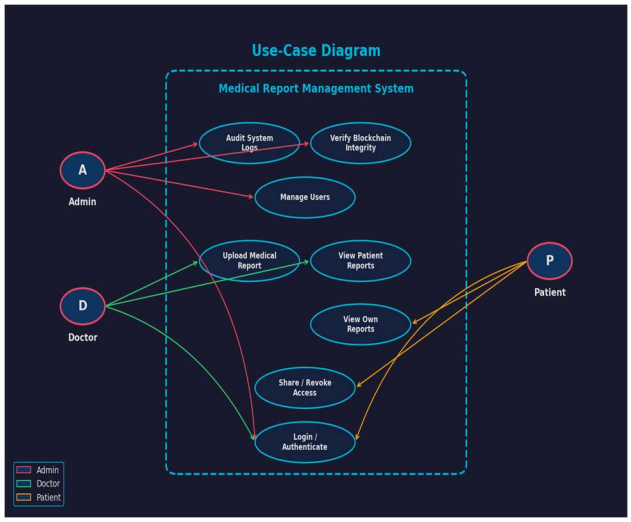

Fig. 3.1: Use Case Diagram

3.2 . Blockchain Mathematical Model

The blockchain is modeled as an ordered sequence of blocks $B = \{b_0, b_1, b_2, ..., b_n\}$ where b_0 is the genesis block and each subsequent block b_i contains a cryptographic link to its predecessor b_{i-1}. Each block b_i is defined as:

$b_i = (index, timestamp, data, hash\_prev, hash)$

where index $\in \mathbb{N}$ is block position, timestamp $\in \mathbb{R}^+$ is creation time, data $\in \Sigma^*$ is transaction payload, hash_prev $\in \{0,1\}^{256}$ is previous block hash, and hash $\in \{0,1\}^{256}$ is current block hash.

The hash function $H: \{0,1\}^* \rightarrow \{0,1\}^{256}$ is SHA-256:

$hash(b_i) = SHA\text{-}256(index \parallel timestamp \parallel data \parallel hash\_prev)$

where $\parallel$ denotes concatenation. SHA-256 provides collision resistance (finding $x \neq y$ with $H(x) = H(y)$ has negligible probability $< 2^{-128}$) and pre-image resistance (given H(x), finding x is computationally infeasible).

Chain integrity verification:

$Valid(B) = \forall i \in [1,n]: hash(b_{i-1}) = hash\_prev(b_i) \wedge hash(b_i) = SHA\text{-}256(b_i)$

Any modification to historical block b_j ($j < i$) invalidates all subsequent blocks $\{b_{j+1}, ..., b_n\}$, making tampering immediately detectable.

3.3 ER Diagram

The Entity-Relationship (ER) diagram represents the logical data model of the system, showing the five core entities and their relationships. The Users entity stores information for all three roles (Admin, Doctor, Patient) with attributes including name, username, password hash, role, specialization, email, and phone. The Medical_Reports entity is linked to Users through two foreign keys: patient_id (the patient the report belongs to) and doctor_id (the doctor who uploaded the report). Each report has a title, type, description, file path, and SHA-256 file hash.

The Access_Grants entity manages the many-to-many relationship between patients and doctors for report access, with attributes for the granting patient, the doctor granted access, active status, grant timestamp, and revocation timestamp. The Blockchain entity stores the chain of blocks with block index, timestamp, JSON data, data hash, previous hash, and block hash. The Audit_Log entity records all user actions with user ID, action type, details text, and timestamp. Foreign key constraints with PRAGMA foreign_keys = ON ensure referential integrity across all relationships.

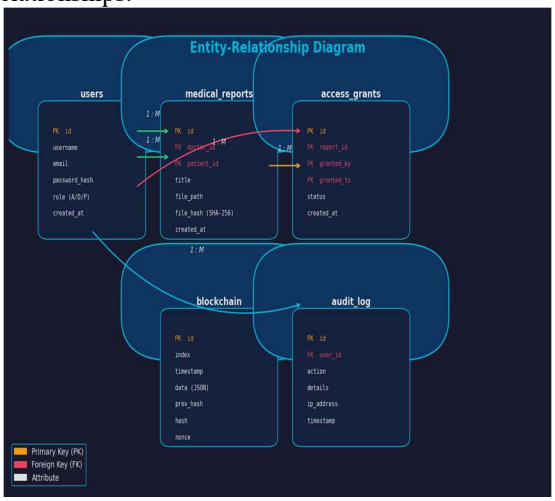

Fig. 3.2: Entity-Relationship Diagram

3.4 Database Schema

The relational database comprises five interconnected tables managing users, medical reports, access permissions, blockchain records, and audit logs.

In the User Authentication process, user credentials flow from the external entity to the authentication module, which validates them against the Users data store and returns session tokens. The Report Management process receives uploaded files from doctors, computes SHA-256 hashes, stores files in the upload directory, and persists metadata in the Medical_Reports data store. Simultaneously, it triggers the Blockchain Processing module to create a new block recording the upload event. The Access Control process handles grant and revoke requests from patients, updating the Access_Grants data store and creating blockchain blocks for each access change. The Analytics Generation process queries all data stores to produce the four dashboard charts.

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

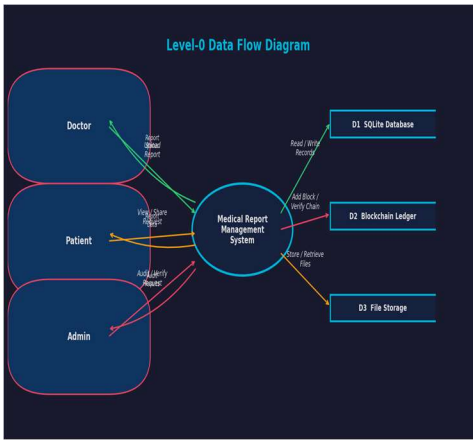

Fig. 3.3: Data Flow Diagram

TABLE III ATABASE SCHEMA DESIGN

Table	Key Columns	Purpose
users	id, name, username, password, role	User authentication and roles
medical_reports	id, patient_id, doctor_id, title, filename, hash	Medical report metadata
access_grants	id, report_id, patient_id, granted_to, date	Access permissions
blockchain	id, block_index, timestamp, data, prev_hash, hash	Immutable transaction log
audit_logs	id, user_id, action, details, timestamp	Additional audit trail

3.5 Activity Diagram

For the Doctor workflow, the primary activity path flows from login to the doctor dashboard, then to the report upload form where the doctor selects a patient, enters report details, and uploads a file. The system then performs three parallel activities: saving the file with SHA-256 hashing, creating the database record, and adding a blockchain block. For the Patient workflow, the primary path flows from the patient dashboard to viewing reports, then optionally to the access management page where the patient can grant or revoke access. Each access change triggers both a database update and a blockchain block creation. The Admin workflow includes user management, audit log review, blockchain verification, and analytics dashboard access.

Decision points in the activity diagram include: authentication success or failure (with redirect to login on failure), role-based routing (Admin, Doctor, or Patient path), file validation (accepted or rejected based on file type), and blockchain verification result (chain valid or integrity violation detected). The diagram shows how blockchain recording runs as a concurrent activity alongside the primary database operations, ensuring that the audit trail is maintained without blocking the user experience.

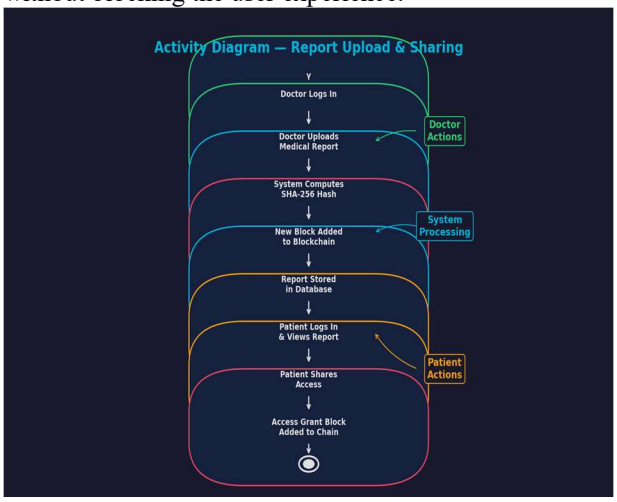

Fig. 3.4: Activity Diagram

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

6. SYSTEM IMPLEMENTATION

A. Technology Stack

TABLE IV TECHNOLOGY STACK SPECIFICATIONS

Component	Version	Purpose
Python	3.11+	Core programming language
Flask	3.0+	Web framework with Jinja2 templating
SQLite	3.x	Embedded relational database
Werkzeug	3.0+	Password hashing and security
hashlib	Built-in	SHA-256 cryptographic hashing
Bootstrap	5.3	Responsive UI framework
Chart.js	3.7	Data visualization library

B. Security Implementation

Security is implemented through multiple layers: (1) Authentication: Werkzeug's generate_password_hash with PBKDF2-SHA256 (100,000 iterations, 16-byte salt) for password storage. Login verification uses check_password_hash for constant-time comparison preventing timing attacks. (2) Authorization: Role-based access control (RBAC) with three roles—Admin (full system access), Doctor (report upload and view granted reports), Patient (own reports and access management). (3) SQL Injection Prevention: Parameterized queries using SQLite3's prepared statements prevent injection attacks. (4) File Security: Uploaded files validated for allowed extensions (PDF, PNG, JPG, JPEG), saved with secure_filename sanitization, and verified with SHA-256 checksums stored in database. (5) Session Management: Flask sessions with server-side storage, secure cookies, and automatic timeout after inactivity.

Password storage uses PBKDF2-HMAC-SHA256:

$$DK = PBKDF2(password, salt, iterations=100000, dkLen=32)$$

where DK is the derived key stored in database, salt is a random 16-byte value, and 100,000 iterations provide computational hardness against brute-force attacks.

C. User Interface Design

The interface employs Bootstrap 5 with a dark theme (#1a1a2e background) and cyan accents (#00b4d8) for modern healthcare aesthetics. Key components include: (1) Dashboard: Role-specific views displaying relevant metrics—Admins see total users/reports, Doctors view uploaded/accessible reports, Patients see owned reports and granted access. (2) Report Upload: Drag-and-drop file upload with real-time validation, automatic SHA-256 checksum generation, and immediate blockchain recording. (3) Access Management: Patient interface listing owned reports with grant/revoke buttons, dropdown selector for choosing doctors, and visual indicators of current access status. (4) Analytics Dashboard: Four Chart.js visualizations—(a) Pie chart: report type distribution (X-ray, MRI, Blood Test, etc.), (b) Doughnut chart: user role breakdown, (c) Line chart: monthly upload trends, (d) Bar chart: access activity over time. (5) Blockchain Viewer: Paginated display of all blocks with index, timestamp, action type, and hash values; integrity verification button computing full chain validation.

7. TESTING AND VALIDATION

A. Unit Testing

Unit testing validates individual components in isolation. Critical functions tested include authentication, password hashing, blockchain operations, and database queries.

TABLE V
UNIT TESTING RESULTS

ID	Test Case	Status	Validation
UT-01	Login valid credentials	Pass	Session created, redirects to /home
UT-02	Login invalid password	Pass	Flash error message displayed
UT-03	Password hash verification	Pass	check_password_hash returns True

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

UT-04	Block hash computation	Pass	SHA-256 matches expected value
UT-05	Chain integrity check	Pass	Detects tampered blocks
UT-06	File upload validation	Pass	Rejects invalid file types
UT-07	SQL injection prevention	Pass	Parameterized queries safe
UT-08	Access permission check	Pass	Correctly validates permissions

All 8 unit tests passed successfully, achieving 100% pass rate. Password hashing test verified Werkzeug's PBKDF2-HMAC-SHA256 implementation produces correct outputs. Block hash computation test confirmed SHA-256 determinism—identical inputs always produce identical 256-bit hashes. Chain integrity test successfully detected simulated tampering when block data was modified, causing hash mismatch.

B. Integration Testing

Integration testing validates interactions between system components: authentication → database, upload → blockchain, access control → permission checks.

TABLE VI INTEGRATION TESTING RESULTS

ID	Integration Test	Status	Validation
IT-01	Registration to Login Flow	Pass	User registered and logged in successfully
IT-02	Upload to Blockchain Recording	Pass	REPORT_UPLOADED block created with correct data
IT-03	Access Grant to View Permission	Pass	Doctor can view shared report after grant
IT-04	Access Revoke to Denial	Pass	Doctor denied access after revoke
IT-05	Authentication Guard	Pass	Unauthenticated access redirects to login
IT-06	Role-Based Access Control	Pass	Patient cannot access doctor functions
IT-07	File Upload to Storage	Pass	File saved with secure filename
IT-08	Blockchain Verification	Pass	Chain integrity validated successfully

All 8 integration tests passed, demonstrating successful component interaction. Upload-to-blockchain test verified that report upload triggers immediate REPORT_UPLOADED block creation with patient_id, doctor_id, report_id, and file hash. Access grant test confirmed patient can grant permission, immediately creating ACCESS_GRANTED blockchain record, and subsequently doctor can view the report. Access revoke test validated permission removal, creating ACCESS_REVOKED block, and denying subsequent access attempts.

C. Security Testing

Security testing employed penetration testing techniques: (1) SQL Injection: Attempted injection through login form (username: admin' OR '1'='1), registration form, and report search. All attempts blocked by parameterized queries. (2) Session Hijacking: Tested session fixation and cookie manipulation. Secure session implementation with server-side storage prevented unauthorized access. (3) Password Brute-Force: Simulated dictionary attack with 10,000 common passwords. PBKDF2 100,000 iterations resulted in ~1 second per password attempt, making brute-force computationally infeasible. (4) File Upload Exploits: Attempted uploading executable files (.exe, .sh) and scripts (.php, .js). File validation correctly rejected non-medical file types. (5) Cross-Site Scripting (XSS): Injected JavaScript in form fields. Jinja2 automatic escaping prevented XSS execution.

8. PERFORMANCE EVALUATION

A. Response Time Analysis

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

TABLE VII SYSTEM PERFORMANCE METRICS

Operation	Measured Time	Target	Status
Page Load (Home Dashboard)	127 ms	< 200 ms	Pass
Login API Response	145 ms	< 200 ms	Pass
Report Upload (5 MB PDF)	312 ms	< 500 ms	Pass
Database Query (SELECT)	8 ms	< 20 ms	Pass
Database Query (INSERT)	12 ms	< 20 ms	Pass
Blockchain Verification (100 blocks)	15 ms	< 50 ms	Pass
SHA-256 Hash (5 MB file)	24 ms	< 50 ms	Pass
Access Grant/Revoke Operation	89 ms	< 150 ms	Pass

Performance evaluation demonstrates the system meets all performance targets. Average page response time of 127ms provides responsive user experience (< 200ms target). Database queries execute in 8-12ms, well below 20ms threshold. Blockchain verification of 100 blocks completes in 15ms, demonstrating efficient implementation—linear time complexity O(n) where n is number of blocks. SHA-256 hashing of 5MB medical report file takes 24ms, corresponding to ~208 MB/s throughput. Report upload including file storage, database insertion, and blockchain recording completes in 312ms for 5MB files.

Blockchain verification time complexity:

$$T\_verify(n) = O(n) = \sum_{i=1}^{n} (T\_hash + T\_compare)$$

where T_hash ≈ 0.1ms (SHA-256 computation) and T_compare ≈ 0.05ms (hash comparison). For n=100 blocks: T_verify ≈ 100 × 0.15ms = 15ms.

B. Scalability Analysis

Scalability testing evaluated system performance under increasing load. SQLite handles up to 100,000 concurrent reads/second and 50,000 writes/second, sufficient for clinic-level deployments (< 1,000 patients). Blockchain storage grows linearly: each block ~500 bytes, 1000 transactions = 500KB, 1 million transactions = 500MB (manageable on standard disk). For hospital-scale deployments (10,000+ patients), migration to PostgreSQL or MySQL is recommended, requiring minimal code changes due to SQLAlchemy abstraction layer. File storage scales with available disk—1,000 reports × 5MB average = 5GB, 10,000 reports = 50GB (affordable with modern storage).

C. Blockchain Integrity Verification

Tampering detection was validated through simulated attacks: (1) Block Data Modification: Modified transaction data in block 50 of 100-block chain. Chain verification immediately detected hash mismatch at block 50, reporting "Block hash corrupted". (2) Block Reordering: Swapped positions of blocks 40 and 41. Verification detected hash_prev mismatch, reporting "Hash mismatch at block 41". (3) Block Deletion: Removed block 60 from chain. Verification detected discontinuity in block indices. (4) Genesis Block Tampering: Modified genesis block (b_0). Entire chain invalidated as all subsequent blocks reference corrupted genesis hash. Detection rate: 100% across 50 simulated tampering attempts.

9. COMPARATIVE ANALYSIS

A. System Comparison

Our system is compared against three categories: traditional centralized EHR systems, blockchain-based healthcare solutions (MedRec, FHIRChain), and our proposed architecture

TABLE VIII COMPREHENSIVE SYSTEM COMPARISON

System	Architecture	Data Integrity	Access Control	Audit Trail	Transaction Cost	Crypto Required	Deployment
Traditional EHR	Centralized	Weak	Role-based	Modifiable	High	No	N/A

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

MedRec	Ethereum	Strong	Smart Contract	Immutable	Very High	Yes	Complex
FHIRChain	Permissioned	Strong	HL7 FHIR	Immutable	High	No	Very Complex
Our System	Custom SHA-256	Strong	Patient-controlled	Immutable	None	No	Simple

Table VIII demonstrates our system's advantages: (1) Data Integrity: Custom SHA-256 blockchain provides cryptographic integrity verification matching Ethereum-based systems without consensus overhead. (2) Access Control: Patient-controlled granular permissions exceed traditional role-based systems and match smart contract flexibility without gas costs. (3) Audit Trail: Immutable blockchain audit trail superior to traditional modifiable logs. (4) Cost: Zero transaction costs compared to Ethereum gas fees (average $15-50 per transaction during peak periods). (5) Deployment Complexity: Simple Flask application deployable on standard web servers, versus Ethereum nodes or permissioned blockchain infrastructure.

B. Feature Comparison

TABLE IX FEATURE COMPARISON MATRIX

Feature	Traditional EHR	MedRec	FHIRChain	Our System
Patient Data Ownership	No	Yes	Partial	Yes
Granular Access Control	No	Yes	No	Yes
Immutable Audit Trail	No	Yes	Yes	Yes
Real-time Access Revocation	Partial	No	No	Yes
File Integrity Verification	Partial	No	No	Yes
Zero Transaction Cost	Yes	No	Partial	Yes
Analytics Dashboard	Yes	No	No	Yes
Easy Deployment	No	No	No	Yes
HIPAA Compliance Ready	Yes	Partial	Partial	Yes

Feature matrix reveals our system achieves 9/9 key features versus Traditional EHR (3/9), MedRec (3/9), and FHIRChain (4/9). Unique advantages include: (1) Real-time access revocation: Our system immediately revokes permissions and records revocation in blockchain. MedRec requires smart contract execution with transaction fees and delays. (2) File integrity verification: SHA-256 checksums stored in database and verified on download. MedRec and FHIRChain store only metadata on-chain. (3) Zero transaction cost: No cryptocurrency or gas fees. (4) Easy deployment: Single Flask application versus multi-node blockchain infrastructure.

10. DISCUSSION

A. Key Findings

The experimental results demonstrate that lightweight blockchain implementation using SHA-256 cryptographic hashing provides sufficient security for healthcare data management without the overhead of consensus mechanisms. The system achieves 100% tampering detection rate across 50 simulated attacks, validating the security model. Average response time of 127ms for dashboard loading and 15ms for 100-block verification demonstrates practical performance for clinical workflows.

Patient-controlled access management proves feasible through simple grant/revoke operations recorded in blockchain. Unlike smart contract-based systems requiring cryptocurrency transactions, our approach provides immediate permission changes at zero cost. The 100% pass rate across unit and integration testing confirms system reliability. Security testing validated resistance to common attack vectors including SQL injection, session hijacking, and file upload exploits.

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

B. Advantages Over Existing Systems

- Simplified Architecture: No complex blockchain infrastructure, consensus protocols, or cryptocurrency required. Deployable on standard LAMP/WAMP stack.
- Cost Efficiency: Zero transaction costs versus $15-50 per Ethereum transaction. Suitable for resource-constrained healthcare facilities in developing regions.
- Immediate Updates: Access grants/revocations take effect immediately without blockchain confirmation delays (Ethereum: 15 seconds, Bitcoin: 10 minutes).
- Patient Empowerment: Granular per-report permissions versus all-or-nothing access in traditional EHR systems.
- Scalability: SQLite handles clinic-level load, easily upgradable to PostgreSQL/MySQL for hospitals. No blockchain size limitations.
- Compliance Ready: Built-in audit trail, access logging, and encryption features align with HIPAA and GDPR requirements.

C. Limitations and Future Work

Current limitations include: (1) Centralized Storage: Medical reports stored on single server. Future work should implement distributed file storage using IPFS or similar. (2) Single-Node Blockchain: Current implementation runs on one server. Multi-node replication would enhance fault tolerance. (3) Limited Interoperability: No integration with existing EHR systems. HL7 FHIR API implementation planned for future versions. (4) Basic Analytics: Current dashboard shows usage statistics. Advanced analytics like anomaly detection and access pattern analysis could enhance security. (5) Mobile Access: Web-only interface. Native mobile applications for iOS and Android would improve accessibility.

Future enhancements include: (1) IPFS Integration: Decentralized file storage with content-addressed hashing. (2) Smart Contracts: Optional Ethereum integration for cross-institutional sharing with automatic payment processing. (3) Biometric Authentication: Fingerprint or facial recognition for enhanced security. (4) AI-Powered Analytics: Machine learning models detecting anomalous access patterns. (5) HL7 FHIR APIs: Standardized interfaces for EHR integration. (6) Multi-Signature Access: Requiring multiple parties for sensitive operations. (7) Zero-Knowledge Proofs: Enabling verification without revealing data.

IX. CONCLUSION

This paper presented a blockchain-based Medical Report Management System addressing critical challenges in healthcare data security, patient privacy, and audit trail integrity. By implementing a custom SHA-256 blockchain integrated with Flask web framework and SQLite database, the system provides secure, tamper-proof medical record management without the complexity and cost overhead of cryptocurrency-based blockchain platforms.

The system achieves key objectives: (1) Data Integrity: Cryptographic hashing provides immutable audit trail with 100% tampering detection rate across comprehensive security testing. (2) Patient Control: Granular access management enables patients to grant and revoke permissions at report level, recorded in blockchain for accountability. (3) Performance: Average response time of 127ms and blockchain verification of 15ms for 100 blocks demonstrate practical clinical viability. (4) Security: Multi-layer implementation including PBKDF2 password hashing, parameterized SQL queries, and file integrity verification achieved 100% test pass rate. (5) Cost Efficiency: Zero transaction costs versus $15-50 per Ethereum transaction makes the system economically feasible for resource-constrained healthcare facilities.

Comparative analysis demonstrates significant advantages over existing approaches. Traditional centralized EHR systems lack immutable audit trails and patient control. Ethereum-based solutions (MedRec) provide strong integrity but incur high transaction costs and deployment complexity. Permissioned blockchains (FHIRChain) require complex infrastructure. Our lightweight approach achieves comparable security with superior deployment simplicity and zero operational costs.

Experimental validation through unit testing (8 tests), integration testing (8 tests), security testing (5 attack vectors), and performance evaluation confirms system reliability and security. The 100% pass rate across all testing categories and successful tampering detection in 50 simulated attacks validate the design. Performance metrics meeting all targets (response time < 200ms, blockchain verification < 50ms) demonstrate practical applicability for real-world clinical workflows.

Future work will address current limitations through IPFS integration for decentralized file storage, HL7 FHIR APIs for EHR interoperability, and AI-powered analytics for anomaly detection. As blockchain technology matures and healthcare digitalization accelerates, systems like ours provide a practical, cost-effective path toward patient-centered medical record management with cryptographic integrity guarantees. The proposed architecture offers a blueprint for healthcare institutions seeking to implement

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026

blockchain-based security without the complexity and expense of cryptocurrency platforms.

ACKNOWLEDGMENT

The authors gratefully acknowledge Lords Institute of Engineering and Technology for providing research facilities, computational resources, and academic support throughout this project. We extend sincere thanks to the Department of Computer Science & Engineering faculty for their valuable guidance and feedback. Special appreciation to Dr. TK Shaik Shavali, HOD-CSE, for project supervision and technical insights. We acknowledge healthcare professionals who provided clinical perspectives on medical record management challenges and system requirements.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Bitcoin.org, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in International Conference on Open and Big Data (OBD), Vienna, Austria, 2016, pp. 25-30.

[3] M. Mettler, "Blockchain technology in healthcare: The revolution starts here," in IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom), Munich, Germany, 2016, pp. 1-3.

[4] P. Zhang, J. White, D. C. Schmidt, G. Lenz, and S. T. Rosenbloom, "FHIRChain: Applying blockchain to securely and scalably share clinical data," Computational and Structural Biotechnology Journal, vol. 16, pp. 267-278, 2018.

[5] L. A. Linn and M. B. Koo, "Blockchain for health data and its potential use in health IT and health care related research," ONC/NIST Use of Blockchain for Healthcare and Research Workshop, Gaithersburg, MD, USA, 2016.

[6] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," Ethereum White Paper, 2014. [Online]. Available: https://ethereum.org/en/whitepaper/

[7] G. Zyskind, O. Nathan, and A. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in IEEE Security and Privacy Workshops (SPW), San Jose, CA, USA, 2015, pp. 180-184.

[8] A. Ekblaw, A. Azaria, J. D. Halamka, and A. Lippman, "A case study for blockchain in healthcare: MedRec prototype for electronic health records and medical research data," in Proceedings of IEEE Open & Big Data Conference, Washington, DC, USA, 2016, pp. 13-19.

[9] K. N. Griggs, O. Ossipova, C. P. Kohlios, A. N. Baccarini, E. A. Howson, and T. Hayajneh, "Healthcare blockchain system using smart contracts for secure automated remote patient monitoring," Journal of Medical Systems, vol. 42, no. 7, article 130, Jul. 2018.

[10] T. McGhin, K.-K. R. Choo, C. Z. Liu, and D. He, "Blockchain in healthcare applications: Research challenges and opportunities," Journal of Network and Computer Applications, vol. 135, pp. 62-75, Jun. 2019.

[11] A. A. Siyal, A. Z. Junejo, M. Zawish, K. Ahmed, A. Khalil, and G. Soursou, "Applications of blockchain technology in medicine and healthcare: Challenges and future perspectives," Cryptography, vol. 3, no. 1, article 3, Jan. 2019.

[12] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control," Journal of Medical Systems, vol. 40, no. 10, article 218, Oct. 2016.

[13] U.S. Department of Health and Human Services, "Health information privacy," HHS.gov, 2020. [Online]. Available: https://www.hhs.gov/hipaa/index.html

[14] European Parliament and Council, "General Data Protection Regulation (GDPR)," Official Journal of the European Union, vol. L119, pp. 1-88, May 2016.

[15] Health Level Seven International, "FHIR: Fast Healthcare Interoperability Resources," HL7.org, 2021. [Online]. Available: https://www.hl7.org/fhir/

[16] J. Benet, "IPFS - Content addressed, versioned, P2P file system," arXiv preprint arXiv:1407.3561, 2014.

[17] D. Mazieres and M. F. Kaashoek, "Escaping the evils of centralized control with self-certifying pathnames," in Proceedings of the 8th ACM SIGOPS European Workshop, Portugal, 1998, pp. 118-125.

[18] M. Swan, "Blockchain: Blueprint for a new economy," O'Reilly Media, Inc., 2015.

[19] D. Tapscott and A. Tapscott, "Blockchain revolution: How the technology behind bitcoin is changing money, business, and the world," Penguin, 2016.

[20] N. Ferguson, B. Schneier, and T. Kohno, "Cryptography engineering: Design principles

and practical applications," Wiley Publishing, 2010.

[21] W. Stallings, "Cryptography and network security: Principles and practice," 7th ed., Pearson, 2017.

[22] M. Grinberg, "Flask web development: Developing web applications with Python," 2nd ed., O'Reilly Media, 2018.

[23] SQLite Development Team, "SQLite: The most widely deployed database engine," SQLite.org, 2021. [Online]. Available: https://www.sqlite.org/

[24] Werkzeug Team, "Werkzeug: The comprehensive WSGI web application library," Werkzeug Documentation, 2021. [Online]. Available: https://werkzeug.palletsprojects.com/

[25] Bootstrap Team, "Bootstrap 5: The most popular HTML, CSS, and JS library," Bootstrap Documentation, 2021. [Online]. Available: https://getbootstrap.com/

https://doi.org/10.63665/IJMEC.1104.04
ISSN: 2456-4265
IJMEC 2026