International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 03 | Mar – 2026 DOI: 10.55041/ISJEM05761
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

# Blockchain-Based Secure Student Data Management System

**1. R. Venkateshwarlu, 2. Sagaram Ganesh, 3. Gouroju Ajaykumar, 4. Gundeti Vaishnavi, 5. Chidrala Vyshnavi**
Department of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science, Karimnagar, Telangana, India
sagaramganeshgk@gmail.com, ajaykumargouroju123@gmail.com, gundetivaishnavi634@gmail.com, vyshuchidralavyshnavi@gmail.com

**Abstract—**The integrity, authenticity, and secure dissemination of educational records constitute a foundational requirement for academic institutions, employers, and regulatory bodies. Conventional centralized database architectures for academic record management are susceptible to unauthorized modification, data breaches, single points of failure, and inefficiencies in cross-institutional record sharing. This paper presents a blockchain-enabled framework for the secure storage and controlled sharing of educational records, integrating the Advanced Encryption Standard (AES) for symmetric data encryption, the Practical Byzantine Fault Tolerance (PBFT) algorithm for distributed consensus validation, smart contracts for automated access authorization, the SHA-256 cryptographic hash function for tamper detection, and cloud-based encrypted storage for scalable data persistence. The proposed system encrypts student records prior to cloud storage, anchors cryptographic hashes on an immutable blockchain ledger, and enforces access control through smart contract logic. The PBFT consensus mechanism ensures transactional integrity with deterministic finality. A formal architectural analysis demonstrates that the system satisfies confidentiality, integrity, non-repudiation, and availability.

***Index Terms—****Blockchain, Educational Records Management, AES Encryption, PBFT Consensus, Smart Contracts, SHA-256, Data Masking, Cryptographic Hashing, Secure Cloud Storage.*

## 1. INTRODUCTION

Educational records—encompassing academic transcripts, degree certificates, course completion credentials, and institutional affiliations—are foundational artifacts governing academic progression, employment eligibility, and institutional credibility. The ongoing digital transformation of higher education has introduced systemic vulnerabilities including unauthorized data modification, credential forgery, and system-level failures attributable to centralized architectures [1].

The proliferation of counterfeit academic credentials represents a significant threat to educational ecosystems. Centralized institutional databases function as isolated *data islands* with limited interoperability. Students seeking employer verification must obtain paper-based documentation susceptible to forgery [2].

Blockchain technology, characterized by immutability, decentralization, auditability, and cryptographic security, presents a compelling architectural foundation. When augmented with symmetric encryption and fault-tolerant consensus protocols, blockchain-based systems can provide tamper-proof, privacy-preserving credential management without dependence on a central authority [3].

The primary limitations of existing centralized academic record management systems include: (i) a single point of compromise susceptible to cyberattacks; (ii) absence of standardized interoperability protocols; (iii) paper-based workflows susceptible to physical damage and forgery; (iv) lack of cryptographically enforced access control; and (v) manually intensive credential verification processes [4].

## 2. BODY OF THE PAPER

### 2.1 RELATED WORK

*A. Blockchain-Based Certificate Verification*

Rajesha et al. [1] proposed a document counterfeit detection scheme using IPFS, wherein document hashes are anchored to a blockchain ledger. While the scheme provides integrity verification, it does not address encrypted access control or fault-tolerant consensus.

Dongre et al. [2] proposed a degree fraud detection system using digital signatures and timestamping. The system establishes non-repudiation but lacks symmetric encryption or Byzantine fault-tolerant consensus.

*B. Smart Contract-Based Authorization*

Leka and Selimi [3] implemented an Ethereum smart contract system for academic certificate verification with AES encryption. However, Ethereum's proof-of-work consensus introduces significant energy overhead and transaction latency.

Mara and Motupalli [4] proposed a blockchain-based certificate tracking model emphasizing anti-counterfeiting. Their architecture lacks scalable encrypted cloud storage or formal analysis of the underlying consensus protocol.

*C. Self-Sovereign Identity*

Bhosale et al. [5] introduced a self-sovereign identity model using decentralized identifiers. The operational complexity of key management and absence of a scalable encrypted storage layer limit practical deployment. The present work addresses all identified gaps within a cohesive five-layer framework.

### 2.2 SYSTEM DESIGN

The proposed system is designed to store and share educational records securely using blockchain technology. The overall system pipeline is divided into five major layers: User, Staff, Admin, Blockchain, and Cloud Storage. The system workflow is shown in Fig. 1.

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 03 | Mar – 2026 DOI: 10.55041/ISJEM05761
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

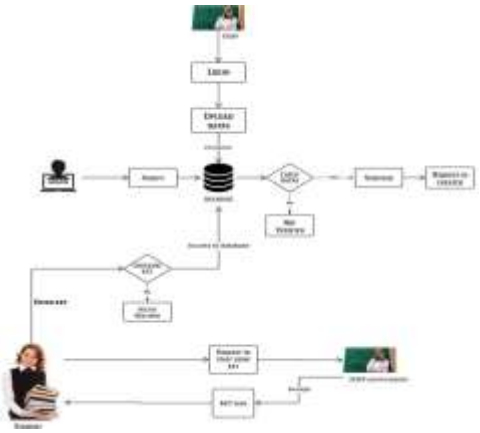

Fig. 1. System Architecture — Staff log in and upload AES-encrypted data; Admin verifies and generates request IDs; Student requests access via key; Staff accepts; key delivered.

*A. Proposed System Architecture*

**User Layer:** Provides the student-facing interface for submitting record access requests and receiving decryption keys.

**Staff Layer:** Enables staff to upload, encrypt, and commit academic records and approve student access requests.

**Admin Layer:** Performs identity verification and governs user authorization status. Generates request IDs.

**Blockchain Layer:** Maintains an immutable ledger of cryptographic hashes and manages distributed PBFT consensus.

**Cloud Storage Layer:** Persists AES-encrypted records with data masking applied to sensitive fields.

*B. Data Flow and Trust Model*

Smart contracts enforce access policies without trust in any individual administrative entity. Data flow: staff encrypts and uploads ciphertext → SHA-256 hash anchored on blockchain → student requests access → staff approves → PBFT validates → key delivered to student.

*C. Modules*

- User Registration and Authentication
- Staff Upload and AES Encryption
- File Access Request and Approval
- Blockchain Hash Anchoring (SHA-256)
- PBFT Consensus and Transaction Commit
- Data Decryption and Key Delivery

**2.3 METHODOLOGY**

*A. AES Encryption Scheme*

AES is employed as the symmetric block cipher for all record encryption. It operates on 128-bit blocks with a 128-bit key across ten transformation rounds. Let $P$ denote plaintext, $K$ the key, and $C$ the ciphertext:

$$C = AES_K(P) \quad (1)$$

Cipher Block Chaining (CBC) mode ensures identical plaintext blocks produce distinct ciphertext blocks, preventing pattern analysis attacks. The AES key exchange model is shown in Fig. 2.


Fig. 2. AES Algorithm: Sender supplies plaintext to Encryption Server with Secret Key; Ciphertext sent to Decryption Server which recovers plaintext via Secure Channel.

*B. SHA-256 Integrity Verification*

SHA-256 is applied to the AES-encrypted ciphertext to produce a fixed-length 256-bit digest. The integrity anchor stored on the blockchain is:

$$h = H(E) = SHA\text{-}256(E) \quad (2)$$

Any modification to the stored ciphertext results in a divergent digest with probability $1 - 2^{-256}$, enabling deterministic tamper detection.

*C. Smart Contract-Based Access Control*

Access control is encoded through smart contracts. Let Auth($u$) $\in$ {PENDING, APPROVED, REVOKED}. A function $\Phi: U \times R \rightarrow \{0, 1\}$ maps a user-record pair to a binary authorization decision. All events are immutably logged on the blockchain providing a complete auditable access history.

*D. PBFT Consensus Mechanism*

The Practical Byzantine Fault Tolerance algorithm [7] provides deterministic finality in the presence of up to $f$ Byzantine nodes within a network of $n$ nodes, where:

$$n \geq 3f + 1 \quad (3)$$

PBFT operates across five phases: Request, Pre-Prepare, Prepare, Commit, and Reply. The message-exchange sequence is depicted in Fig. 3.

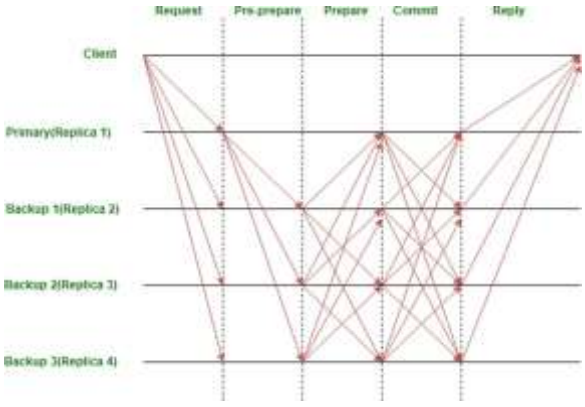


Fig. 3. PBFT Consensus: Client, Primary (Replica 1), and three Backup replicas exchange messages across five phases: Request, Pre-Prepare, Prepare, Commit, Reply.

*E. Data Masking*

Sensitive data fields—national identification numbers, date-of-birth, and financial aid information—are masked before

cloud storage, substituting original values with structurally consistent but semantically opaque tokens.

## 2.4 IMPLEMENTATION

### *A. Hardware and Software Environment*

The prototype was developed on an Intel Core i5 system with 4 GB RAM and 500 GB HDD. The software environment comprised Windows 10, Python 3.x with Tkinter for the GUI, MySQL via PyMySQL, and PyCryptodome for AES operations. The system entry point is the login portal shown in Fig. 4.


Fig. 4. User Login interface (Tkinter/Python 3.x). Users authenticate with registered email and password. New users register via the "Not Registered?" link.

**TABLE I System Implementation Environment**

| Component | Specification |
|---|---|
| Processor | Intel Core i5 (or equivalent) |
| Memory | 4 GB RAM minimum |
| Storage | 500 GB HDD |
| OS | Windows 10 |
| Language | Python 3.x |
| GUI Toolkit | Tkinter |
| Database | MySQL (via PyMySQL) |
| Crypto Library | PyCryptodome (AES-CBC) |
| Blockchain | Custom Python PBFT simulation |

### *B. System Modules*

The implementation comprises six functional modules:

- **User Auth:** Credential-based login and session management.
- **Upload & Encrypt:** AES encryption, masking, cloud upload.
- **File Request:** Routes student requests to staff.
- **Blockchain Verify:** SHA-256 hashing, ledger anchoring.
- **PBFT Consensus:** All five consensus phases.
- **Decryption:** Key provisioning and local decryption.

**TABLE II System Module Descriptions**

| Module | Functional Responsibility |
|---|---|
| Auth Module | Registration, login, session management |
| Upload & Encrypt | AES-CBC encryption, masking, cloud upload |
| File Request | Student access request routing to staff |
| Blockchain Verify | SHA-256 hashing, ledger anchoring |
| PBFT Consensus | Transaction validation and commit |
| Decryption | Key provisioning, ciphertext decryption |

## 2.5 RESULTS AND DISCUSSIONS

This section presents the results obtained from the implemented blockchain-based secure student data management system validated through end-to-end prototype testing.

### *A. End-to-End Workflow Validation*

**Step 1 — Record Search and Access Request:** A student searches for a record by roll number; the system returns matching metadata with a **Request** action button and masked File Key.

Fig. 5. Student record search (roll 201). The system returns record metadata (Owner: Mansi, User: Shalu) with masked File Key. The Request button routes the request to staff.

**Step 2 — Staff Request Acceptance:** The pending request appears with an **Accept Request** action and the unmasked File Key **XMBSNN**. Upon acceptance, the transaction is submitted to PBFT consensus.

Fig. 6. Staff management view: pending request from Shalu for roll 201. Unmasked File Key XMBSNN visible. Accept Request triggers the PBFT consensus commit.

**Step 3 — Key Delivery to Student:** Following PBFT commit, the student's dashboard reflects "Accepted" status and presents decryption key **XMBSNN** with an "Enter Your File Key" dialog.

Fig. 7. Student view after staff approval. File Key XMBSNN presented; "Enter Your File Key" dialog initiates AES-CBC decryption of the stored ciphertext.

**Step 4 — Decrypted Academic Record:** Upon successful key submission, the AES decryption module renders the plaintext

academic record. Semester and Model Exam scores for six subjects are displayed. The raw DB row confirms all stored fields remain as AES ciphertext before decryption.


Fig. 8. Decrypted academic record for student Shalu (Roll 201). Semester scores (Total: 489, Grade: B) and Model Exam scores (Total: 499, Grade: B1) displayed after AES-CBC decryption using key XMBSNN. Bottom DB row confirms ciphertext storage.

### *B. Security Property Analysis*

Confidentiality is assured by AES-CBC encryption. Integrity is enforced by SHA-256 hash anchoring. Non-repudiation is achieved through the immutable blockchain transaction log. Availability is supported by the distributed consortium network.

**TABLE III Security Property Analysis**

| Property | Mechanism | Guarantee |
|---|---|---|
| Confidentiality | AES-CBC | Ciphertext-only at storage tier |
| Integrity | SHA-256 | Tamper detect: 1 – 2■²■■ |
| Non-repudiation | Blockchain log | Immutable audit trail |
| Availability | Consortium net | No single point of failure |

### *C. Discussion*

From the obtained results, the five-layer architecture effectively addresses all security requirements. The smart contract-based access authorization model supports employer credential verification, inter-institutional student transfers, government academic audits, and scholarship eligibility confirmation.

However, a few limitations were also observed:

- The quadratic communication complexity of PBFT, $O(n^2)$ per round, constrains scalability to consortium deployments with a bounded number of nodes.
- The prototype relies on a centralized cloud storage provider, making record availability subject to the provider's SLA.
- Robust key management practices are required at the institutional level to prevent unauthorized decryption key exposure.

Overall, combining AES encryption, SHA-256 hash anchoring, PBFT consensus, and smart contract authorization is an efficient approach for secure academic record management.

## 3. CONCLUSION AND FUTURE WORK

This paper presented a blockchain-enabled framework for the secure storage and controlled sharing of educational records, integrating AES symmetric encryption, PBFT Byzantine fault-tolerant consensus, SHA-256 cryptographic hashing, smart contract-based access authorization, and encrypted cloud storage with data masking. The proposed five-layer architecture addresses the fundamental security deficiencies of centralized academic record management.

Formal analysis demonstrates that confidentiality, integrity, non-repudiation, and availability are satisfied. PBFT provides deterministic finality with fault tolerance for up to $f < n/3$ Byzantine nodes. End-to-end prototype evaluation (Figs. 5–8) validates all workflow stages from authentication through AES-CBC decryption.

In future, this work can be extended by integrating with national academic repositories for cross-jurisdictional credential verification, implementing cross-chain interoperability protocols, incorporating machine learning-based anomaly detection, and developing mobile application interfaces for improved accessibility.

## .REFERENCES

[1] R. Rajesha, K. Suresh, and M. Nagendra, "Counterfeit detection of documents using blockchain technology," in *Proc. IEEE ic-ETITE*, Feb. 2021, pp. 1–5.

[2] J. G. Dongre, S. D. Nandgaonkar, and P. R. Patil, "Education degree fraud detection system using blockchain and digital signatures," in *Proc. IEEE CONECCT*, Jul. 2020, pp. 1–6.

[3] E. Leka and B. Selimi, "Blockchain based secure application for verification of academic certificates," in *Proc. IEEE iCCECE*, Aug. 2021, pp. 1–5.

[4] P. Mara and R. Motupalli, "Blockchain-based model to track and verify official certificates," in *Proc. IEEE ACCAI*, May 2022, pp. 1–6.

[5] H. Bhosale, S. Sonekar, and A. Dhone, "Self-sovereign student identities using blockchain technology," in *Proc. IEEE CONIT*, Jun. 2021, pp. 1–7.

[6] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[7] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. 3rd USENIX OSDI*, Feb. 1999, pp. 173–186.

[8] NIST, "Advanced Encryption Standard (AES)," FIPS Pub. 197, Nov. 2001.

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

[10] NIST, "Secure Hash Standard (SHS)," FIPS Pub. 180-4, Aug. 2015.