IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

Blockchain Based Academic Certificate Verification System

Rutuja Dudhate[1], Shubham Gadge[2], Pramod Gopale[3]

Student, Department of Computer Applications, Samarth College of Engineering & Management, Belhe, Maharashtra, India[1,2]

Professor, Department of Computer Applications, Samarth College of Engineering & Management, Belhe, Maharashtra, India[3]

Abstract: In most organizations, identity is still checked against a central database. This is convenient to run. The catch is that one party ends up owning the records, and people must keep uploading the same documents every time they sign up for something new. In this paper we propose a hybrid framework for verifying academic certificates that uses a blockchain for one job only: keeping a shared record of trust. Each issuer and holder gets a decentralized identifier (DID), and the claims go inside verifiable credentials (VCs). For every credential, a smart contract stores just its hash and its status, so nothing personal is written on the chain. We describe the architecture, follow a credential from issue to revocation, give an eight-step verification algorithm, and discuss the threats and a test plan. We have not built the prototype yet, so no performance numbers appear here; we will report them after we have measured them.

Keywords: Blockchain, decentralized identifiers, digital identity, privacy, self-sovereign identity, verifiable credentials

I. INTRODUCTION

Any online service has to establish who is on the other side. Usually this involves an identifier, some personal attributes, credentials like a degree certificate or a licence, and a history of earlier logins, and these are almost never kept together. Our work is on digital identification and verification, and we believe a person should have to submit documents only once; once verified, the same proof ought to be reusable. What actually happens is that each new service asks for the documents again. Every upload creates one more copy on one more server, and each copy can be leaked or misused.

Central databases are not wrong in principle. When a single organization handles registration, storage and recovery, it can do so consistently, and for one service this is often the sensible option. The picture changes once several unrelated organizations all have to depend on the same set of records. Whoever runs the database becomes a dependency for all of them. A breach, an outage or a bad update at that point affects every connected service together, while the extra copies held by other parties widen the exposure further.

With a shared ledger, every organization in the network reads the same record, and none of them has to take one operator's word for it. Still, identity documents should not go on a public chain. Anything stored there is open to everyone and, in practice, can never be deleted. Two W3C standards, Decentralized Identifiers [1] and the Verifiable Credentials Data Model 2.0 [2], [3], allow a more careful design. For identity proofing in particular, we also draw on the NIST digital identity guidelines [4].

Building on these standards, we design a framework in which a verifier can check a credential independently and personal data is never placed on the public ledger. What we present is a design, written so that a prototype can be built from it. Throughout the paper, the running example is an academic degree certificate: a university is the issuer, a graduate is the holder, and an employer or another institution is the verifier. We are not suggesting that today's identity systems be thrown out. In this paper we:

1) describe the three roles (issuer, holder and verifier) and the difficulty each one faces today;
2) lay out an architecture in which sensitive data never reaches the chain;
3) define how a credential moves through its life and how a verifier should check it;
4) look at the main security threats and privacy risks; and
5) write test cases that cover valid, altered, expired and revoked credentials.

II. BACKGROUND

A. Blockchain

In a blockchain, the ledger is replicated over many nodes, and these nodes use a consensus protocol to agree on what it contains. Because every block carries the hash of the block before it, an earlier entry cannot be edited without the change being noticed. What makes this useful for identity is the shared, tamper-evident record; file storage is not the point. For that reason, only the items a verifier has to trust are placed on the chain, namely keys, hashes and status flags.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

B. Cryptographic Hashing

A hash function takes input of arbitrary length and produces a short, fixed-length output. Altering even one character of the input gives a completely different result, and our design depends on exactly this behaviour. The hash of each credential is stored at the time of issue. Later, at presentation time, the verifier hashes the credential once more and compares the two values. If they agree, nothing has changed. If they differ, someone has altered the content.

C. Digital Signatures

Every credential is signed by the issuer using its private key. Anybody holding the issuer's public key can then confirm two things: that the issuer really signed it, and that no one has edited it since. A valid signature, however, says nothing about whether the issuer itself should be trusted; we come back to this issue in Section VIII.

D. Decentralized Identifiers

A DID is an identifier created and controlled by its owner, with no central body needed to register it [1]. Looking up a DID gives back a DID document, which contains the owner's public keys and service endpoints. In our framework, this is how a verifier gets hold of the issuer's key before checking the signature.

E. Verifiable Credentials

A verifiable credential groups one or more claims (for example, that a student received a B.Sc. degree in 2024) together with a cryptographic proof added by the issuer. The holder normally stores it in a digital wallet and presents it to a verifier on request. These three roles, and the exchanges between them, are specified in the VC Data Model 2.0 [2].

F. Self-Sovereign Identity

In self-sovereign identity (SSI), people carry their own credentials and choose who gets to see them, and when. SSI does not let users certify themselves; whether a degree was awarded is still decided by the university. Only the place where the proof is kept moves, from the university's server to the student's wallet. Prior work identifies identifiers, claims, credentials and governance as the core components of an SSI system [5], [6].

III. Problem Statement and Research Gap

Today, most verification depends on central databases. A person hands the same document to several organizations, and each one keeps its own copy. This multiplies the copies of personal data and the points at which it can leak, and it leaves the user dependent on databases they cannot inspect or control. Table I sums these problems up. The question this paper tries to answer is a simple one: can a shared trust layer handle verification without any sensitive data going on-chain?

TABLE I
Issues With Conventional Verification

Issue	Current Problem	Proposed Response
Trust dependency	Verification depends on a central provider	Shared registry that any verifier can query
Tampering	Stored files need extra integrity controls	Issuer signatures and on-chain integrity references
Interoperability	Providers use different formats	Standard DID and VC data models

Standards are not what is missing; the difficulty lies in combining them. DIDs and VCs have already been specified [1], [2], and several surveys cover SSI and blockchain-based identity [5]–[8]. Far fewer studies explain, at a level detailed enough to implement, what data should go on the chain, what should stay off it, and how a verifier ought to combine signature, hash and status checks. This paper addresses that gap. We do not claim that a blockchain is always preferable; for many use cases a properly managed central service is sufficient.

IV. Literature Review

Researchers broadly agree that a ledger on its own does not solve identity management. The studies that informed this work are listed in Table II. Mühle et al. [5] identify the essential components of SSI, while Schardong and Custódio [6] survey the field and highlight problems that are still unresolved. Kuperberg [7] examines blockchain-based identity management from the viewpoint of enterprises and ecosystems. Yan et al. [8] adopt an interdisciplinary approach and propose a research agenda that treats technical, organizational and governance issues jointly.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

TABLE II
SUMMARY OF RELATED WORK

Reference	Focus	Observation Used Here
[5] Mühle et al.	Survey of SSI components	Core building blocks of an SSI system
[6] Schardong & Custódio	SSI review and taxonomy	Breadth of SSI research and open concerns
[7] Kuperberg	Blockchain identity management	Enterprise and ecosystem perspective
[8] Yan et al.	Decentralized identity review	Technical, organizational and governance issues together

These studies show that no single mechanism is enough on its own. We therefore focus on one problem, namely how to check a signed credential while keeping personal data off a public chain. Keeping the scope narrow makes the system easier to build and also easier to evaluate fairly.

V. PROPOSED ARCHITECTURE

The framework has five parts, shown in Fig. 1:

- *Issuer:* an organization such as a university or licensing body that verifies a person's identity, issues credentials, and publishes its DID and public key.
- *Holder:* the individual described by the credential, who stores it in a wallet and chooses when to present it.
- *Verifier:* an employer, bank or other service that relies on the claims in the credential.
- *Blockchain trust layer:* a smart contract that acts as a registry. It keeps issuer keys, credential hashes and the status of each credential.
- *Off-chain data layer:* full records and supporting documents sit here, in a database protected by access control.

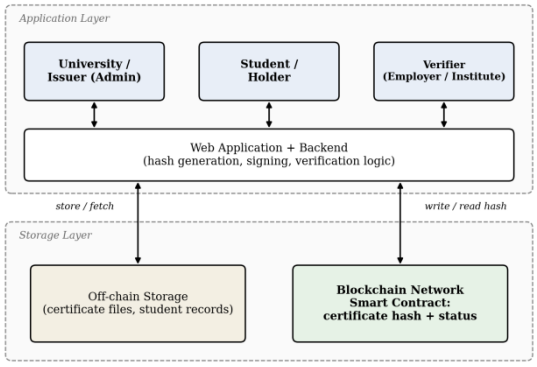


Fig. 1. System architecture of the proposed blockchain-based certificate verification system.

A. On-Chain/Off-Chain Rule

The boundary between on-chain and off-chain data must be fixed before implementation begins, and we apply a simple rule. Data is allowed on the chain only if a verifier needs it and it discloses nothing personal; this covers DIDs, public keys, credential identifiers, hashes and status flags. All other data is kept in controlled storage, with a hash connecting the two sides so that the record itself is never exposed. Keeping data off-chain does not automatically make it secure, and the off-chain store still requires encryption, access control, key management and a retention policy.

B. Credential Lifecycle

A credential passes through five stages (Fig. 2):

1) *Issuance.* Once the person's identity has been proofed, the issuer creates and signs the credential and records its hash with an "active" status in the registry.
2) *Storage.* The holder stores the credential in a wallet. The issuer keeps its own copy of the record, off-chain.
3) *Presentation.* When asked, the holder sends the verifier the credential or a presentation derived from it.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

4) *Verification.* The verifier runs the algorithm in Section V-C.
5) *Status change.* The issuer can suspend or revoke a credential at any point, and every credential also expires on its own date. Only the updated status is written to the chain; the reason for the change, which may be sensitive, is kept off-chain.

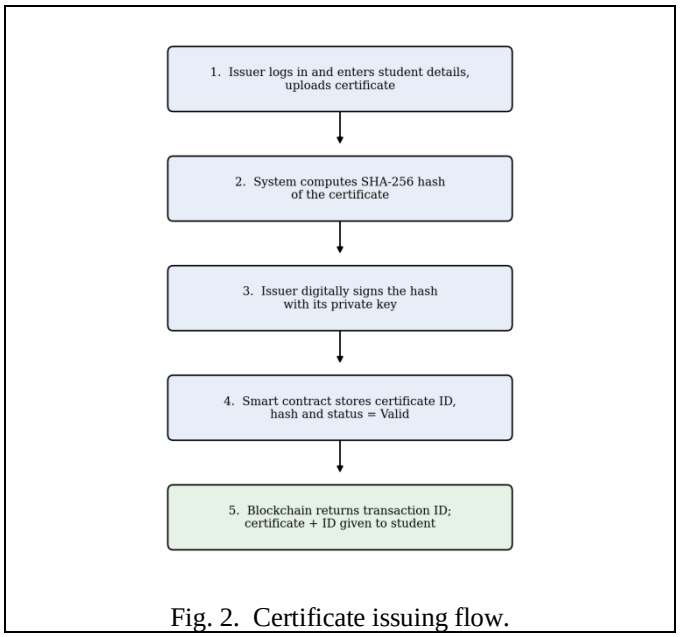


Fig. 2. Certificate issuing flow.

C. Verification Algorithm

The verifier carries out the following steps in order:

1) Receive the credential or verifiable presentation.
2) Identify the issuer from its DID and retrieve its verification method.
3) Make sure the credential is well formed and no required field is missing.
4) Check the issuer's signature (or other proof) with that key.
5) Hash the credential again and see whether the result matches the value in the registry.
6) Look at the issue date and the expiry date.
7) Ask the status registry for the current status; if the credential is suspended or revoked, reject it.
8) Accept only if every step above has passed. Otherwise reject, and log the step that failed.

Recording which step failed is useful during evaluation, since it lets us confirm that each fault in the test set was detected by the check designed for it.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

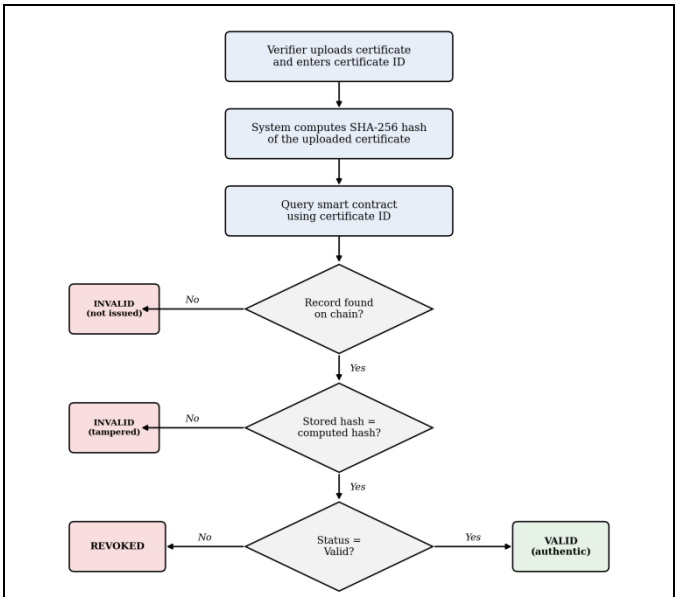


Fig. 3. Certificate verification flow.

VI. Methods and Materials

The study proceeds in five stages. Stage one defines the problem and the three roles. Stage two develops the architecture together with the on-chain/off-chain rule. Stage three specifies the credential lifecycle and the verification algorithm. In stage four, we will build a prototype consisting of a web application, a blockchain test network, smart contracts and off-chain storage. Stage five will run the prototype against the test cases in Section X and measure correctness, tamper detection, response time and overhead.

VII. Implementation Stack

The planned tools are listed in Table III. They were chosen mainly because they are well documented and widely used, which makes the prototype simpler to build and easier for others to reproduce. The exact versions will be given in the final report.

TABLE III
PLANNED TECHNOLOGY STACK

Layer	Technology	Purpose
Frontend	React.js, HTML, CSS, JavaScript	Issuer, holder and verifier interfaces
Backend	Node.js with Express.js	Application APIs and business logic
Smart contracts	Solidity	Trust and status registry
Blockchain	Ethereum-compatible test network	Hosts the registry contract
Off-chain DB	MongoDB	Application and non-public data
Wallet	MetaMask (browser)	Test accounts and transaction signing

VIII. Security Analysis

A valid signature shows only that a certain key was used to sign the credential; it does not show that the key belongs to a trustworthy issuer. The verifier also needs a dependable way to map the issuer's name to its correct public key, and in our design this role is played by the DID registry. Even so, the ledger cannot stop key theft, phishing, a staff member issuing credentials without permission, or poor access control. Table IV lists each major threat alongside the control that mitigates it.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

TABLE IV
THREAT ANALYSIS

Threat	Attack or Failure	Control
Credential modification	Claim altered after issuance	Digital signature and integrity check
Issuer key compromise	Unauthorized credential signing	Protected key storage and rotation
Replay	Old proof reused	Fresh challenge (nonce) per request
Revocation failure	Invalid credential accepted	Status and revocation check
Data leakage	Sensitive attributes exposed	Data minimization and off-chain access control
Registry manipulation	Unauthorized status or key change	Authenticated contract functions and governance
Off-chain compromise	Private data accessed	Encryption, access control and monitoring

IX. PRIVACY CONSIDERATIONS

The design follows data minimization: a verifier receives only what it needs to make its decision. For instance, an employer verifying a degree must know that the degree is authentic and has not been revoked, but has no reason to see the holder's home address. Because a public blockchain is readable by anyone, personal documents and unnecessary attributes are never stored on it. Some credential formats also support selective disclosure, where the holder shows only the claims that are needed. Consent, data retention, access control and account recovery must be handled by the application, as the ledger cannot provide them.

X. EXPERIMENTAL DESIGN AND TEST CASES

Here we explain the evaluation plan for the prototype. No performance figures are given, as nothing has been measured so far. We will create a controlled set of valid credentials and deliberately altered ones, and every credential will go through the same verification procedure. Every test will be done on one Windows computer, and we will not change its hardware or network settings between runs. Table V lists the test cases.

TABLE V
TEST CASES

ID	Scenario	Expected Outcome	Metric
TC01	Valid signed credential	Accepted	Correctness, response time
TC02	Modified credential	Rejected	Tamper detection
TC03	Wrong issuer key	Rejected	Authentication correctness
TC04	Expired credential	Rejected or flagged	Lifecycle correctness
TC05	Revoked credential	Rejected or flagged	Status correctness
TC06	Replay of a previous proof	Rejected when freshness is required	Replay resistance
TC07	Unknown credential	Rejected	Registry validation
TC08	Unauthorized data access	Denied	Privacy and access control



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026
DOI: 10.17148/IJARCCE.2026.151060

A. Evaluation Metrics

- *Verification accuracy:* the number of correct decisions divided by the total number of test cases, with the total always reported. False acceptances and false rejections are counted separately, because accepting a bad credential is much more serious than rejecting a good one.
- *Average verification time:* total measured time divided by the number of valid timing runs, on stated hardware and network.
- *Transaction overhead:* the transactions and gas used for issuance and status changes on the chosen network.
- *Storage overhead:* the number of bytes written to the chain compared with the size of the data kept off-chain, which indicates how much the hybrid design keeps away from the ledger.
- *Scalability:* how response time changes as the number of credentials or requests grows, with request volume, hardware, network and software stated.

B. Result Reporting Rule

In the final report, every value will be accompanied by its test environment, sample size, network configuration, software versions and date of measurement. Anything that has not been measured will be marked as planned. Our goal is a study that others can reproduce and that does not claim more than the evidence supports.

XI. Discussion and Limitations

The framework handles two distinct questions. The first is whether the credential has been altered since it was issued, which is answered by the signature and hash checks. The second is whether it is still valid at the time of verification, which is answered by the status registry. Because the two checks are kept apart, a failed verification points straight to the part that caused it.

Keeping personal data off the ledger reduces what the public can see. The off-chain database, though, still has to stay online, be encrypted and backed up, be audited, and be reachable only through secured APIs. Key management is a second concern. Suppose an issuer loses its private key or has it stolen: every credential it ever signed is now in doubt, and the blockchain gives no warning, because from its side nothing has gone wrong. For this reason, key recovery and rotation will be tested in the prototype instead of being taken for granted.

Identity proofing, the very first step, is our biggest concern. Before a certificate is issued, the university must confirm that the student is the person named in its own records. A mistake here cannot be fixed later, since every check after it would be protecting the wrong person's identity.

Lastly, the ledger stores whatever the issuer writes and has no way to tell true from false. A wrong claim made at issue time will be kept just as carefully as a correct one.

XII. Future Work

Several directions remain open. With zero-knowledge proofs, a holder could show they are over 18 without revealing their date of birth. We also want to test key recovery and rotation by causing failures on purpose. Performance can be compared across different blockchain networks and with larger numbers of credentials. We would also like a few real issuers, holders and verifiers to try the workflow, so we can see where people without technical training get stuck. Further work is also needed on interoperability between DID methods, credential formats and existing institutional systems.

XIII. Conclusion

This paper has presented a framework in which the blockchain acts as a shared record of trust and integrity rather than a repository for personal data. Identity data and public trust metadata are kept apart throughout. Because of this, a verifier can rely on signatures, hashes and status records alone, and people should not have to hand over the same documents again and again. Our next step is to implement the prototype, evaluate it using the test cases defined here, and report only results that have actually been measured.

References

[1] W3C, "Decentralized Identifiers (DIDs) v1.0," W3C Recommendation, 19 Jul. 2022. [Online]. Available: https://www.w3.org/TR/did/

[2] W3C, "Verifiable Credentials Data Model v2.0," W3C Recommendation, 15 May 2025. [Online]. Available: https://www.w3.org/TR/vc-data-model/

[3] W3C, "The Verifiable Credentials 2.0 family of specifications is now a W3C Recommendation," W3C News, 15 May 2025. [Online]. Available: https://www.w3.org/news/2025/the-verifiable-credentials-2-0-family-of-specifications-is-now-a-w3c-recommendation/

[4] D. Temoshok et al., "NIST SP 800-63-4: Digital Identity Guidelines," National Institute of Standards and Technology, 2025, doi: 10.6028/NIST.SP.800-63-4.



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.151060

[5] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A survey on essential components of a self-sovereign identity," *Computer Science Review*, vol. 30, pp. 80–86, 2018, doi: 10.1016/j.cosrev.2018.10.002.

[6] F. Schardong and R. Custódio, "Self-Sovereign Identity: A Systematic Review, Mapping and Taxonomy," *Sensors*, vol. 22, no. 15, Art. 5641, 2022, doi: 10.3390/s22155641.

[7] M. Kuperberg, "Blockchain-Based Identity Management: A Survey From the Enterprise and Ecosystem Perspective," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1008–1027, 2020, doi: 10.1109/TEM.2019.2926471.

[8] Z. Yan, X. Zhao, Y. Liu, and X. Luo, "Blockchain-driven decentralized identity management: An interdisciplinary review and research agenda," *Information & Management*, vol. 61, no. 7, Art. 104026, 2024, doi: 10.1016/j.im.2024.104026.