Secure Electronic Voting System Using Blockchain Technology

Harshitha H B[1] Vijay Kumar S[2] Yeshaswini Bhandari K R[3]
[1,2,3]Department of Computer Applications
[1,2,3]ATME College of Engineering, Mysore, Karnataka, India

Abstract — Conventional paper-based and electronic voting systems suffer from centralised control, vulnerability to tampering, high logistical cost and limited accessibility, all of which weaken voter confidence. This paper presents a secure electronic voting system built on blockchain technology. Voter identity is verified through an Aadhaar-number check followed by an e-mail one-time password (OTP); each ballot is encrypted and recorded as a transaction in a hash-linked distributed ledger using SHA-256 and Merkle trees, and smart-contract logic enforces one vote per voter and automates tallying. The prototype is a Django web application with separate portals for voters, candidates and the Election Commission, a configurable voting period and a live statistics dashboard. Tests on a sample election showed votes being recorded, counted and displayed without duplicate or altered entries. Limitations concerning scalability, digital-identity security and legal acceptance are discussed, and directions for future work are outlined.

Keywords: Blockchain, Electronic Voting, Smart Contract, SHA-256, Merkle Tree, OTP Authentication, Distributed Ledger.

I. Introduction

Elections are the foundation of democratic governance, yet the mechanisms used to conduct them still struggle with fraud, manipulation, high cost and low public trust. Paper ballots require heavy logistics and slow manual counting, while electronic voting machines and early online voting portals usually rely on centralised servers. A centralised design creates a single point of failure and exposes the election to insider tampering, phishing, data breaches and denial-of-service attacks. Accessibility is a further concern, because rural voters, expatriates and persons with disabilities often cannot reach a polling booth.

Blockchain, first introduced as the basis of the Bitcoin currency [1], is a distributed ledger in which records are grouped into blocks, each linked to the previous block by a cryptographic hash. Because the ledger is replicated across many nodes and altering one block invalidates every block after it, recorded data becomes tamper-evident and effectively immutable. These properties match the requirements of an election: integrity of every ballot, transparency of the process and independence from a single trusted authority [2].

This paper presents the design and prototype implementation of a secure electronic voting system based on blockchain. The main contributions are: (i) a layered architecture that separates voter registration, authentication, vote casting, ledger storage and result generation; (ii) a two-step authentication scheme combining Aadhaar-number verification with an e-mail OTP; (iii) a ledger design in which every vote is stored as a SHA-256-hashed transaction inside a Merkle-tree block, with smart-contract rules preventing double voting and automating the count; and (iv) a working web prototype with an Election Commission dashboard. The remainder of the paper reviews related work, describes the system, explains the cryptographic foundations, reports implementation and results, and concludes with future work.

II. Related Work

Ohize et al. [2] reviewed blockchain frameworks, trends and solutions for electronic voting, analysing scalability, infrastructure and security concerns. Their discussion of deployments in Estonia and Switzerland highlights auditability as a clear benefit and high implementation cost as a recurring drawback. Atik [3] examined how decentralisation and cryptography can resist vote manipulation, coercion and hacking, and summarised current achievements and open opportunities.

Several authors propose concrete architectures. Singh [4] describes a decentralised Ethereum-based internet voting system using smart contracts; simulations demonstrated transparency and accuracy, and biometric authentication was suggested to improve security and scalability. Romano et al. [5] use linkable ring signatures together with smart contracts to prevent double voting while keeping voters anonymous. Kim et al. [6] combine homomorphic encryption with blockchain so that voter data can be stored securely and still support statistical analysis without exposing individual choices. The SHARVOT model [10] applies Shamir's secret sharing and a circle-shuffle step to anonymise votes, relying on smart contracts for transparency and immutability.

Spanos and Kantzavelou [7] present an Ethereum-based internet voting scheme that removes dependence on central databases, and discuss accessibility, privacy, affordability and scalability for national elections. Panja [8] addresses ballot stuffing through strong voter registration and authentication so that results remain verifiable. Obulu [9] reports a university-level deployment at the University of Nigeria, Nsukka, identifying weaknesses of traditional electronic voting and proposing blockchain as a secure replacement.

Across these works the common themes are immutability, anonymity and verifiability, whereas cost, scalability and identity management remain open issues. The present work targets a practical middle ground: a low-cost web prototype that combines familiar identity checks (Aadhaar number and OTP) with hash-linked storage and automated counting.

III. Problem Statement and Objectives

The problem addressed is the lack of a voting platform that is simultaneously secure, transparent, private and easy to access. The objectives are to: (a) authenticate voters so that only eligible persons participate; (b) guarantee that each voter casts at most one vote; (c) record every ballot immutably on a distributed ledger; (d) preserve voter anonymity while keeping the process verifiable; (e) support remote voting from web or mobile devices; and (f) tally and publish results

All rights reserved by www.ijsrd.com

automatically in real time, reducing cost, delay and human error. The design is intended to be adaptable to organisational polls as well as larger elections.

IV. PROPOSED SYSTEM

A. System Overview

In the proposed system every vote is a transaction that is encrypted, hashed and appended to a blockchain maintained by several nodes, instead of being stored in one database. Casting a ballot is therefore similar to making a ledger entry that every node can verify but none can silently change. Smart contracts check eligibility, reject repeat votes and keep the running tally, so no external intermediary is needed to count or announce the result.

B. System Modules

The system consists of seven modules. The *User Authentication* module validates voter identity through credentials, Aadhaar-number verification and an OTP to prevent impersonation. The *Voter Registration* module maintains the voter list and eliminates duplicate or fraudulent entries. The *Voting* module offers a simple interface for selecting a candidate and encrypting the vote. The *Blockchain* module stores each encrypted vote as part of an immutable block. The *Smart Contract* module automates counting and enforces election rules. The *Result Generation and Verification* module tallies results in real time while preserving privacy. Finally, the *Administration and Monitoring* module lets authorised officials manage registrations, set the voting period and monitor activity.

C. System Architecture

Fig. 1 shows the architecture. Voters and the voter-registration service communicate with the application over HTTPS. The admin service layer (application server, file server, database and administrator) works with the election-preparation services, which hold the voter list and election management functions. Two blockchain nodes and an independent audit node are reached through REST APIs, and the authority publishes the election result. Because several nodes hold the ledger, the failure or compromise of one node does not stop the election or erase the record.

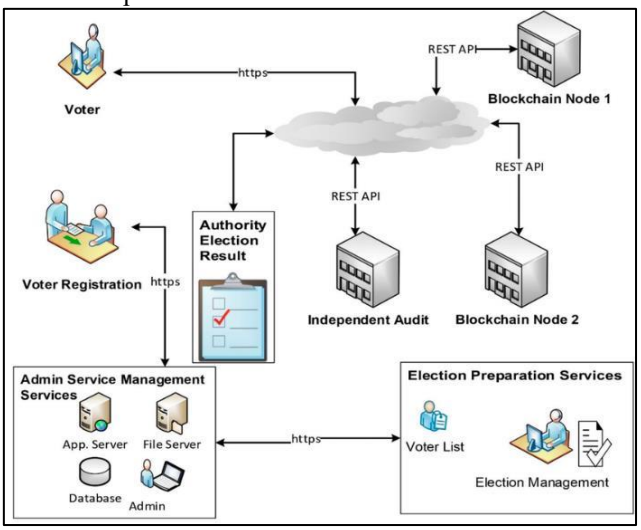


Fig. 1: System architecture of the blockchain voting system.

D. Authentication and Voting Workflow

A new user registers with a username, name, e-mail address, phone number and password. The entered Aadhaar number is verified; if verification fails the user is marked unauthorised and cannot proceed, otherwise the user is authorised. An OTP is then generated and sent to the registered e-mail address through the Simple Mail Transfer Protocol (SMTP), and the voter must enter it to be admitted to the ballot page. After the voter selects a candidate, the system creates a block, computes its hash values using SHA-256 and Merkle trees, records the vote, and returns a vote identifier to the voter. When the voting period ends, the results are declared. The message flow among the voter, authentication unit, blockchain, vote-casting unit and crypto server is shown in Fig. 2.

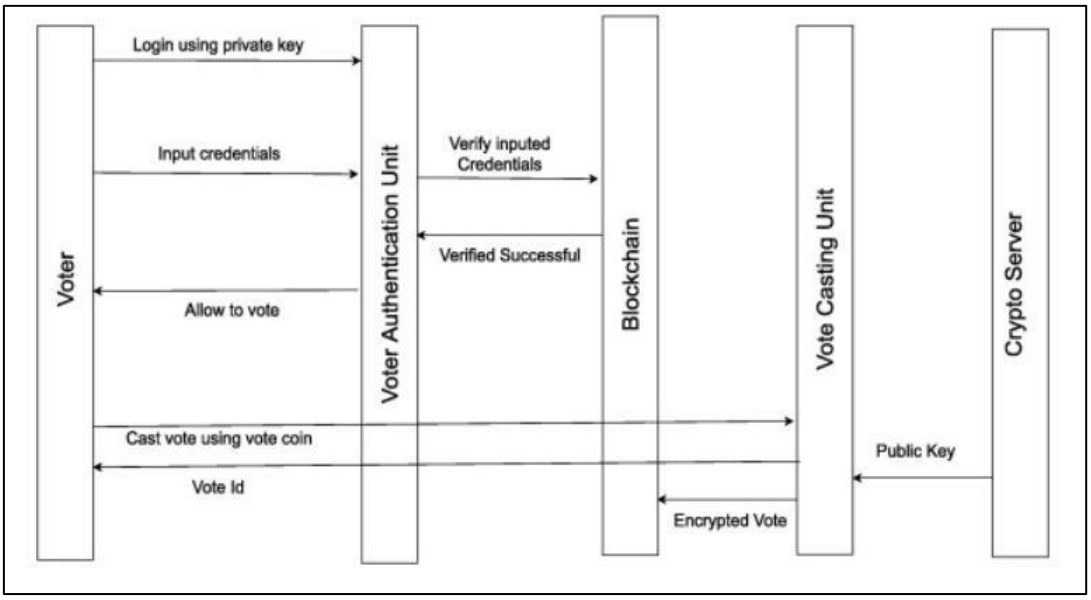


Fig. 2: Sequence diagram of voter authentication and vote casting.

All rights reserved by www.ijsrd.com

E. Vote Counting

Each registered voter is issued a single "vote coin". Casting a vote transfers this coin to the wallet of the chosen candidate, so a voter who has spent the coin cannot vote again. After the end time, the encrypted votes are forwarded to the vote-counting unit, decrypted with the private key held by the crypto server, and counted. As shown in Fig. 3, the tally procedure reads the balance of every candidate wallet, returns the list of candidates with party symbol, seat number and balance, selects the candidate holding the highest number of vote coins, and adds that candidate to the list of winners for the region.

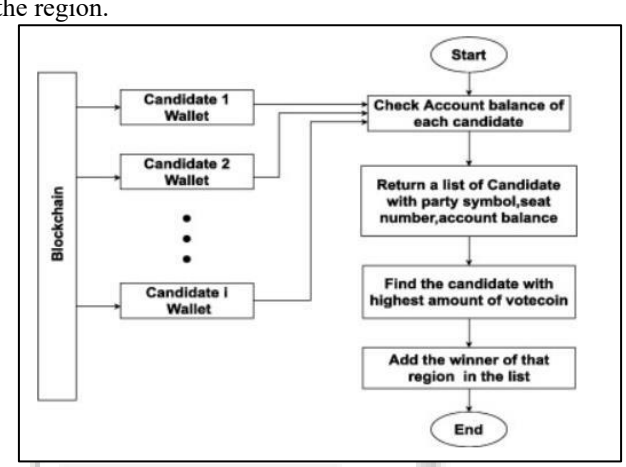


Fig. 3: Flowchart of vote tallying from candidate wallet balances.

V. Cryptographic and Algorithmic Foundations

A. Hashing and Block Linking

The Secure Hash Algorithm 256 (SHA-256) produces a fixed 256-bit digest for each vote transaction. The digests of the transactions in a block are combined pairwise into a Merkle tree whose root, R_i, summarises the whole block. The hash of block i is

$$H_i = \text{SHA-256}(H_{i-1} \| R_i \| t_i) \quad (1.1)$$

where H_{i-1} is the hash of the previous block, t_i is the block timestamp and ‖ denotes concatenation. If any vote is altered, its digest, the Merkle root and every later block hash change, so tampering is detected immediately by the other nodes.

B. Consensus

All nodes must agree on the order and validity of transactions. Energy-intensive Proof of Work is unsuitable for an election, so lighter protocols such as Practical Byzantine Fault Tolerance (PBFT) [11] and Proof of Authority (PoA) are preferred. They give faster finality and lower energy use, and PoA fits a setting where the validating nodes are known institutions such as the election authority and independent auditors.

C. Encryption and Digital Signatures

To protect privacy, a vote is encrypted before it leaves the voter's device, using public-key schemes such as RSA or elliptic-curve cryptography (ECC); only the holder of the decryption key can read it. The Elliptic Curve Digital Signature Algorithm (ECDSA) [12] is used to sign transactions, so that a ballot can be verified as genuine and cannot later be repudiated or modified. In the prototype, the public key is supplied by the crypto server at vote time and the private key is used only after the voting period closes.

D. Smart Contracts

Smart contracts are programs stored on the blockchain that execute deterministically. In this system they check voter eligibility, record the one-time use of each vote coin, reject duplicate ballots and maintain the tally. Since the rules run identically on every node, a transaction is either valid or rejected, which removes ambiguity and the need for intermediaries.

VI. Implementation

A. Technology Stack

The prototype is a web application developed in Python and served locally during development. Table I lists the minimum hardware and software environment.

Item	Specification
Processor / RAM	Intel Core i3 or above / 4 GB
Storage	500 GB hard disk
Operating system	Windows 10 or later
Language	Python 3, Django web framework
Security	SHA-256, Merkle tree, SMTP-based OTP
Tools	Anaconda, Spyder, Visual Studio Code

Table I: Minimum Requirements

B. User Interfaces

The application provides separate login portals for voters, candidates and the Election Commission. The home page presents the main features (voter privacy, verified voters and transparent results) and offers a light/dark display mode. The voter signup page collects registration details, and the verification page sends an OTP to the voter's e-mail address, displaying it in masked form. The Election Commission portal contains a dashboard with the total number of voters, candidates and votes cast, a pie chart of the vote distribution by party, and a form for setting the voting start and end times (Fig. 4).

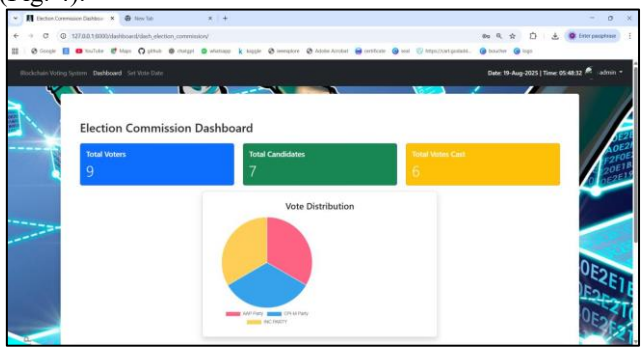


Fig. 4: Election Commission dashboard with vote statistics and voting-period settings.

All rights reserved by www.ijsrd.com

VII. Results and Discussion

A. Prototype Results

The prototype was exercised with a sample election. The dashboard in Fig. 4 reports 9 registered voters, 7 candidates and 6 votes cast, with the pie chart showing the distribution among the participating parties. In these runs each vote was recorded once, a voter who had already voted could not vote again, and the counts on the dashboard were consistent with the stored transactions. The voting window set by the Election Commission was enforced by the system. These results come from a small functional test and are not a measure of large-scale performance.

B. Testing

Testing followed three levels. *Unit testing* checked individual components such as registration, Aadhaar-number validation, OTP generation and expiry, hash computation and the vote-coin transfer. *Integration testing* verified that the flow from authentication through block creation to tallying works when the modules are connected, and that the dashboard reflects the ledger. *Validation testing* confirmed that the complete system meets the stated requirements: only authorised voters can vote, each voter votes once, results match the recorded votes, and the interface is usable by non-technical voters.

C. Comparison with Existing Systems

Table II compares the proposed system with paper ballots and conventional centralised electronic voting.

Feature	Paper ballot	Central e-voting	Proposed
Data storage	Physical boxes	Single server	Distributed ledger
Tamper resistance	Low to medium	Medium	High (hash-linked)
Counting	Manual, slow	Automatic, central	Smart contract, real time
Remote voting	No	Limited	Yes
Single point of failure	Yes	Yes	No
Cost	High	Medium	Low per election

Table II: Comparison Of Voting Approaches

D. Limitations

The system has not yet been evaluated with the load of a national election, and transaction throughput of the underlying consensus method must be measured before such use. Digital identities remain a sensitive point, since Aadhaar-number and e-mail checks can be weaker than biometric authentication. Voters on the wrong side of the digital divide may still be excluded, and legal and regulatory frameworks for blockchain voting are not yet established. Finally, coercion and vote selling in unsupervised remote settings are not eliminated by the technology alone.

VIII. Conclusion and Future Work

This paper presented a secure electronic voting system that uses a hash-linked distributed ledger, OTP-based authentication and smart-contract logic to make elections secure, transparent, accessible and efficient. The prototype showed that eligible voters can register, authenticate, cast a single vote and see results updated on an administrator dashboard without manual counting. With further testing, scaling and regulatory approval, blockchain voting can become a trustworthy basis for modern elections.

Future work includes improving transaction speed so that millions of votes can be processed concurrently, adding biometric verification and stronger cryptography, providing mobile applications with multi-language support, using machine learning for fraud detection, and aligning the platform with legal requirements so that governments and organisations can adopt it.

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.
[2] H. O. Ohize et al., “A review of frameworks, trends, solutions, and challenges in the use of blockchain technology to secure electronic voting systems,” Springer, 2025.
[3] M. A. U. K. Atik, “A comprehensive analysis of blockchain voting,” 2024.
[4] I. Singh, “Enhancing transparency and security in internet voting,” SN Comput. Sci., 2024.
[5] S. P. Romano, A. Russo, M. I. González Vasco, and A. Fernández Anta, “Chirotonia: A secure and scalable e-voting system,” arXiv preprint, 2021.
[6] H. Kim et al., “A reliable e-voting framework with homomorphic encryption and blockchain,” arXiv preprint, 2021.
[7] A. Spanos and I. Kantzavelou, “EtherVote: Decentralized internet voting,” arXiv preprint, 2023.
[8] S. Panja, “An auditable blockchain e-voting system,” 2021.
[9] B. Obulu, “Secure voting with blockchain at the University of Nigeria, Nsukka,” 2025.
[10] “SHARVOT: A secure blockchain voting system,” unpublished.
[11] M. Castro and B. Liskov, “Practical Byzantine fault tolerance,” in Proc. 3rd USENIX Symp. Operating Systems Design and Implementation (OSDI), 1999, pp. 173–186.
[12] D. Johnson, A. Menezes, and S. Vanstone, “The elliptic curve digital signature algorithm (ECDSA),” Int. J. Inf. Secur., vol. 1, no. 1, pp. 36–63, 2001.

All rights reserved by www.ijsrd.com