Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 47 (2026), No. 2, pp. 645–672
DOI : 10.47974/JIOS-2117

TrustChain - A secure digital voting framework

Gaurav Indra [†]
Alongbar Wary *
Shiksha Nayan [§]
Bhavya Jatrana [‡]
Nameesha Chand [@]
Department of Information Technology
Indira Gandhi Delhi Technical University For Women
Kashmere Gate 110006
Delhi
India

Abstract

Electronic voting has several advantages in terms of accessibility, effectiveness, and security, it is nevertheless susceptible to ballot tampering, impersonation, and multiple voting. The main difficulty is striking a balance between voter anonymity and transparency. Blockchain based systems improve verifiability and trust, but they frequently jeopardize privacy. To overcome these limitations we propose TrustChain, a blockchain based system that combines a Merkle Root based Tracking ID, a multi-chain Ethereum architecture, and real time facial recognition in order to get around these restrictions. Ethereum guarantees immutability and tamper resistance, while a lightweight Firebase pseudo blockchain handles registration and authentication, allowing for scalable and effective verification. Performance is enhanced by this decoupled design without compromising privacy or security. Candidate specific chains reduce congestion while guaranteeing tamper proof recording, tracking IDs maintain vote integrity without disclosing identities, and facial recognition restrict

[†] *ORCID-ID:* **0000-0003-0669-1933**
* *ORCID-ID:* **0000-0001-6226-6902**
[§] *ORCID-ID:* **0009-0000-0370-3594**
[‡] *ORCID-ID:* **0009-0009-5908-1005**
[@] *ORCID-ID:* **0009-0001-1548-3231**

[†] *E-mail:* **gauravindra@igdtuw.ac.in**
* *E-mail:* **alongbarwary@igdtuw.ac.in (Corresponding Author)**
[§] *E-mail:* **shikshanayan9540@gmail.com**
[‡] *E-mail:* **bhavyaj2503@gmail.com**
[@] *E-mail:* **nameeshachand@gmail.com**

illegitimate access. Tests show a 45% faster voting process, while maintaining strong security and privacy.

Subject Classification: *68M25, 94A60, 68P27.*

Keywords: *Electronic voting, Blockchain technology, Facial recognition, Tracking ID mechanism, Merkle root hash, Two-tier architecture, Ethereum smart contracts.*

1. Introduction

The foundation of a democratic society is elections, which allow people to express their preferences and choose representatives [1]. However, ballot paper techniques are becoming perceived as antiquated and unsafe. Large-scale deployments frequently encounter logistical challenges, delays, and fraud threats [2, 3], which erode public trust in election results [4, 5]. Electronic voting, or e-voting, has become a contemporary strategy to address these issues by cutting expenses, increasing accessibility, and minimizing fraud. E-voting system is divided into two types: Ballot paper voting, which is done at a polling place and Internet-based voting, which allows ballots to be cast remotely [6, 7]. Both approaches have benefits, but because they depend on centralized infrastructure, they are susceptible to malware, identity spoofing, and denial-of-service (DoS) assaults [8, 9].

Blockchain presents a powerful way to secure e-voting. Its decentralization, fault tolerance, immutability, and privacy features reduce the risks of centralized systems [10-13]. Blockchain has also been explored for privacy-preserving smart contracts. Priyanka et al. proposed a blockchain-based cryptography model that safeguards user data and ensures privacy in smart contracts, demonstrating its relevance for secure e-voting applications [14]. By recording each vote permanently on a distributed ledger, it ensures transparency, trust, and auditability. Its proven success in finance, logistics, and healthcare highlights its potential to protect democratic processes [15]. Recent studies have explored combining blockchain with biometric authentication to improve the integrity of e-voting systems [16]. Among biometric methods, facial recognition has become popular due to its contactless nature, user convenience, and compatibility with government managed databases [8, 9]. Advances in deep learning have further increased its accuracy and scalability for voter identification. Existing blockchain based e-voting systems employ various security mechanisms such as QR codes for identity verification [10], Ethereum smart contracts to enforce tamper resistant voting logic, public key cryptography for data integrity [11], and homomorphic encryption to protect voter privacy [5]. While these approaches enhance security, they often fall short in providing full end-to-end verifiability, ensuring voter anonymity, and supporting nationwide scalability [15, 17]. Most earlier e-voting systems lack mechanisms to generate privacy preserving Tracking IDs, which would allow voters to verify their votes without compromising anonymity. Key challenges in current deployments include centralized identity storage that

threatens privacy, device dependent biometric systems vulnerable to spoofing, and scalability and latency issues in large or distributed populations [18, 19]. Achieving end-to-end verifiability while maintaining voter anonymity remains a major technical hurdle [20, 21].

This e-voting system addresses the critical limitations of traditional electronic voting by integrating a blockchain with facial recognition for robust, verifiable, and private elections. It ensures voter identity using dlib's ResNet-34 and a cryptographically linked registration blockchain [1, 4, 10]. The core of the system is a highly scalable Ethereum based multichain voting architecture that assigns a dedicated logical chain to each candidate, effectively distributing the load and preventing network bottlenecks [3, 7]. For voter anonymity, the system uses a sophisticated combination of Merkle Root hashing and XOR-based Tracking IDs, allowing vote verification without compromising identity. It further reduces the risk of ID collisions by employing binary quantization on facial encodings [1, 4, 8, 9, 17]. A lightweight hybrid architecture with a cloud-based Firebase system handles rapid user authentication, offloading computational work. This Firebase system acts as a lightweight pseudo-blockchain, emulating core blockchain concepts like sequential hashing for tamper-evidence. All core voting transactions, however, remain decentralized and immutable on the Ethereum blockchain. This dual-layer approach provides a balance of operational efficiency, robust security, and end-to-end transparency for large scale deployments [1, 2, 4, 6, 10, 11, 15, 17, 22, 23].

This paper systematically presents the proposed system. Section 2 provides a brief review of prior work, highlighting key advancements and existing gaps. Section 3 details the system's architecture and modules. Section 4 outlines our methodology for secure identity management and voting. Section 5 describes the core modules of the system. The findings of our study are presented in Section 6, supported by data analysis and graphical comparisons. Finally, we conclude our study in Section 7 and discuss future work in Section 8.

2. Related Work

Recent years have seen rapid progress in blockchain-based e-voting systems, motivated by the need for stronger voter anonymity, system integrity, and protection against manipulation [12, 13, 16]. While foundational architectures have established the basis for secure digital elections, several technical limitations persist. Our framework addresses these gaps by combining decentralized infrastructure with biometric voter authentication, cryptographic verification, and real-time status visibility. These limitations are summarized in Table 1. Bhatti et al. [4] introduced a multimodal biometric authentication model using face, iris, and fingerprint recognition with encryption and firewall safeguards. However, without blockchain integration, it lacked tamper-evident, decentralized vote recording. Peelam et al. [3] proposed DemocracyGuard, utilizing Ethereum smart contracts and public ballot storage for decentralization and auditability. Building on this direction, Singh et al. [24] and Tanwar et al. [13] developed a secure Ethereum-based voting website leveraging smart contracts to

enhance transparency and decentralization, though their approach similarly omitted biometric safeguards and real-time verification. Similarly, Spanos and Kantzavelou's EtherVote framework demonstrates the viability of secure Ethereum smart contracts for ballot recording [25], though it leaves biometric authentication and real-time status monitoring unaddressed while improving trust and transparency, it excluded biometric authentication and real-time verification. Li et al. [2] developed S3Voting, a shard-based blockchain design with homomorphic time-lock puzzles and ring signatures to enhance scalability and confidentiality. Yet, it did not incorporate biometric checks, privacy-preserving traceability, or real-time voter engagement. Prathyusha et al. [26] combined blockchain with facial recognition, but omitted Merkle Root hashing and hybrid cloud-blockchain scalability. Faruk et al. [1] integrated biometrics with Hyperledger Fabric, ensuring encrypted vote recording but limiting decentralization, auditability, and real-time traceability due to its permissioned model. Building on these foundations, our system advances beyond prior works. By adopting a public Ethereum framework, hybrid authentication with Firebase, and timestamp-based traceability, we achieve decentralized participation, secure voter authentication, and real-time transparency. A broader survey by Benabdallah et al. [27] consolidates such approaches, identifying scalability, privacy preservation, and interoperability as recurring challenges in blockchain enabled elections.These improvements align with the future research directions suggested in earlier studies, enabling scalable, auditable, and citizen-trustworthy e-voting systems.

2.1 *Proposed Enhancements Over Existing Models*

To address the shortcomings of earlier systems, our architecture introduces improvements in scalability, transparency, anonymity, and voter trust. A key advancement is shifting from a permissioned to a public blockchain. While Faruk et al. [1] relied on Hyperledger Fabric, we employ Ethereum, a decentralized, publicly accessible blockchain that supports open participation and interoperability [19]. Compatibility with tools like MetaMask and Web3.js further facilitates secure interaction and transparent voter participation [19]. Our hybrid model integrates a pseudo-blockchain on Firebase for authentication and identity validation [20, 28]. This module enables lightweight, real-time login handling while maintaining cryptographic integrity through two tier registration and login blocks linked via hash pointers [1]. Login blocks include timestamps and references to registration block hashes [3], enabling immutable recordkeeping and anomaly detection. Unlike prior monolithic approaches, our model enforces blockchain-like verification even at registration, incorporating Merkle Root hashing to ensure data integrity and traceability [1, 21, 29]. Facial Recognition with facial embeddings replaces hardware-dependent methods [30, 31], building on biometric models from Faruk et al. [1] and Bhatti et al. [4] to enable cloud-backed, scalable, real-time voter authentication. A major innovation is real-time vote counting and Tracking ID generation. Each vote receives a unique Tracking ID derived from Merkle Root hashes and facial encodings

through XOR operations [17, 20]. This enables voters to verify their vote without compromising anonymity, addressing transparency-secrecy tensions [21]. Table 1 summarizes these gains in traceability, verifiability, and openness. Furthermore, segregating user identity into individual registration blocks prevents breaches from propagating across users [3, 5]. Using Merkle Root hashing across registration and login, the system introduces tamper-evident security, redressing gaps in Faruk et al.'s model and advancing towards a decentralized, auditable, and voter-verifiable e-voting framework [2, 19, 25, 32].

Table 1

Comparative Analysis of Blockchain-Based E-Voting System

System	Blockchain Platform	Biometric Authentication	Security Approach	Verifiability & Tracking ID	Anonymity & Transparency Tradeoff	System Disadvantage
Transforming online voting: a novel system utilizing blockchain and biometric verification for enhanced security, privacy, and transparency [1]	Hyperledger Fabric (Private)	Inbuilt system, Facial and fingerprint Authentication	Hyperledger Fabric, Dual Factor Authentication, Facial and fingerprint Authentication, Raft Consensus Algorithm	System Level verifiability, No Tracking Mechanism	Voter identity is separated from vote content using biometrics + Hyperledger Fabric. Transparent via blockchain, but no Tracking ID for user-side verification.	High infrastructure cost, limited to controlled environments, built-in fingerprint scanners assumption (not true for every device/ laptops)
S3Voting: A Blockchain Sharding-Based E-voting Approach with Security and Scalability [2]	Ethereum + Sharding	None	Homomorphic time-lock puzzles, ZKPs, ring signatures	Verifiability and Tracking using Homomorphic Time-lock puzzle	High anonymity and privacy but prioritises privacy and delayed verifiability over instant, user-facing transparency	No biometric or mobile interface, complex setup, or delayed transparency.
DemocracyGuard: Blockchain-based secure voting framework for digital democracy [3]	Ethereum	Facial recognition (Azure API)	Smart Contracts, Admin KYC, facial authentication, QR-based validation	Voter verified using QR verification, Results Visible to all, but No Tracking ID mechanism	QR-based anonymity, Smart Contract-based transparency, but voters can't confirm which candidate received their vote	Smart contracts lack integration with the authentication phase, The Static flag allows weak vote duplication defense
Secure Electronic Voting Machine using Multi-Modal Biometric Authentication System, Data Encryption, and Firewall [4]	None	Multimodal (Fingerprint + Iris Scan + Facial Recognition)	AES Encryption, Firewall Protection, Biometric and OTP Verification, but traceable serial ID via SMS	Simultaneous Serial ID Generation, no blockchain traceability	Limited anonymity (SMS-linked ID), limited transparency (tracks single votes only)	Lacks blockchain, highly complex, highly dependent on technology, not a hardware-bound system, lacks cross-system compatibility.

E-Voting system using Blockchain technology [10]	Ethereum	None	Firebase authentication + SHA-256 Hashing	Uses Smart Contracts to verify transactions, Tracking ID - None	Pseudo-Anonymous provides transparency through blockchain	No device-level protection, no vote correction, limited data support, lacks biometrics, and anonymity.
TrustChain (Proposed Solution)	Ethereum (Ganache + MetaMask)	Face Recognition	Merkle Root Hash verification, Face Auth, Firebase Auth, Two-tier registration-login blockchain architecture	Merkle Leaf Hash *(Not reversible, privacy-preserving)*	Face Auth + Merkle Login + Vote Tracking + Per-Candidate Blockchains	No critical disadvantage observed under current testing conditions. Combines privacy, verifiability, scalability, and ease of deploymentâ€"the most balanced and robust architecture.

One of the biggest contributions of the system proposed here is real-time vote counting and Tracking ID creation. Every cast vote is tagged with a specific Tracking ID from Merkle Root hashes and facial encoding of the user through an XOR operation [17, 32]. This functionality gives voters the capability to check that their vote has been cast safely and counted securely without disclosing their identity. This mechanism settles the long-standing tension between system transparency and vote secrecy once and for all–a problem not fully solved in previous systems [25]. A comparative overview of these improvements with previous approaches is given in Table 1, which shows gains in traceability, openness, and real-time verifiability across models. Furthermore, the system provides increased trust by segregating each user's identity data into individual registration blocks so that any anomaly or breach on one user cannot extend to others [3, 5]. The modular architecture gives increased security from the first interaction and has significant benefits over shared-ledger-based systems. By using Merkle Root hashing in both registration and login, our system adds a potent layer of tamper-evident security. This directly addresses a key gap in Faruk et al.'s solution, advancing a truly decentralized, auditable, and voter-verifiable e-voting framework [1, 2, 20, 27, 33, 34].

3. System Architecture

The proposed electronic voting system combines blockchain technology with Facial Recognition authentication in a layered architecture as shown in Figure 1.

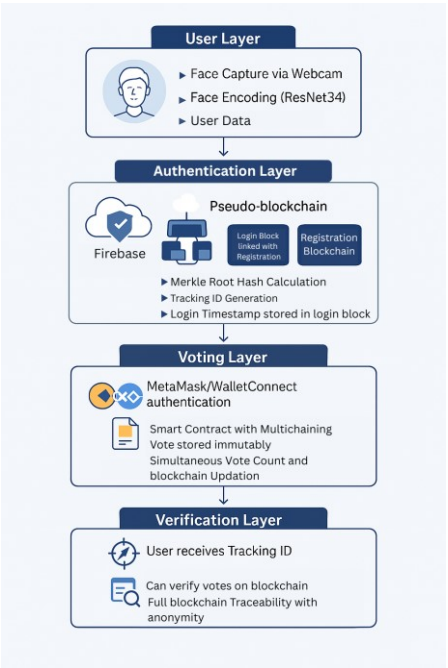

Figure 1
Layered System Architecture Combining Facial Recognition and Blockchain-Based Voting

3.1 *Facial Authentication Layer*

To prevent voter impersonation and ensure only legitimate users vote, the first layer uses biometric authentication via Facial Recognition [20, 27]. It employs dlib's ResNet-34 model, trained on the Labeled Faces in the Wild (LFW) dataset for high real-world accuracy. The model generates 128-dimensional facial encodings as unique voter identifiers, matched with stored templates during login. Figure 1 shows the sequential process from face detection to final identity confirmation.

3.1.1 Voter Registration Phase

The voter's facial image is captured, converted from BGR to RGB, and processed through the Face Recognition pipeline. A 128-Dimension feature vector (face encoding) is then obtained using face_recognition.face_encoding() on the preprocessed image. Mathematically, Equation (1) defines the process, where E denotes the face encoding, I the preprocessed image, and f the ResNet-34 embedding function. Finally, the encoding is saved as an array in Firebase within the voter's registration block, ensuring secure and efficient retrieval. The workflow is detailed in Algorithm 1.

$$E = f(I) \tag{1}$$

Algorithm 1: Facial Recognition Process for Voter Registration

Input: Captured face image I, Voter ID
Output: Successful face registration (1) or failure (0)
Preconditions:
– A valid face image must be uploaded
– The image must contain exactly one face
Save the face image locally using Voter ID-based filename
Convert I from RGB to BGR using OpenCV
Compute face encoding $E \leftarrow$ `face_recognition.face_encodings`(I, model="cnn")
if *no face detected in* I **then**
  Delete saved image;
  return Failure (0);
end
Check for Existing Face Registrations:
foreach *stored encoding* E_i *in database* **do**
  Convert E_i to NumPy array;
  Compute Euclidean distance $d(E, E_i) = \sqrt{\sum (E - E_i)^2}$;
  if $d(E, E_i) \leq$ *FACE_DISTANCE_THRESHOLD* **then**
    return Failure ("Face already registered") (0);
  end
end
return Successful Face Registration (1);
End

3.1.2 Voter Authentication: Login Phase

The voter undergoes live facial authentication during the login process. To standardize, a real face image is captured and encoded into RGB space. After that, the picture undergoes the encoding procedure used during registration to produce a 128-dimensional feature vector. Mathematical encoding takes place in Equation (2), where E' is the face encoding of the newly captured image, and I' represents the newly captured face image.

$$E' = f(I') \tag{2}$$

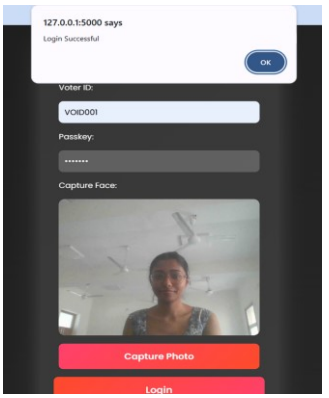

Figure 2
Face Authentication Successful (Euclidean distance ≤ 0.38)

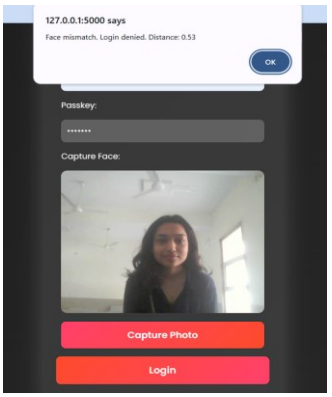

Figure 3
Face Authentication Failed (Euclidean distance > 0.38).

Algorithm 2: Facial Authentication Process During Voter Login

Input: Captured face image I', Voter ID
Output: Authentication success (1) or failure (0)
Preconditions:
    – A valid face image must be uploaded
    – The image must contain exactly one face
Convert I' from BGR to RGB using OpenCV
Compute face encoding: $E' \leftarrow$ face_recognition.face_encodings(I', model="cnn")
if *no face detected in* I' **then**
    return Failure ("No face detected") (0)
end
Retrieve stored face encoding E from Firebase for the given Voter ID
Compute Euclidean distance: $D = \sqrt{\sum_{i=1}^{128}(E_i - E'_i)^2}$
if $D \leq$ *FACE_DISTANCE_THRESHOLD (0.38)* **then**
    return Authentication Success (1)
end
else
    return Authentication Failure (0)
end
End

To check identity, the system retrieves the earlier stored face encoding E corresponding to the registration of a voter and computes the Euclidean distance D between E and E' as given by Equation (3):

$$D = \sqrt{\sum_{i=1}^{128} (E_i - E_{i'})^2} \tag{3}$$

If the distance calculated is less than or equal to 0.38, the system authenticates the user as a voter, and the authentication is considered successful. Else, access is refused. The choice of threshold is made cautiously to ensure that there is high accuracy in distinguishing between real voters and impostors. The result of this authentication is visually illustrated in Figure 2 and Figure 3, depicting successful and unsuccessful facial verifications, respectively. A detailed breakdown of the facial authentication steps can be found in Algorithm 2.

3.2 *Blockchain-Linked Voter Registration and Login*

Our approach employs a cryptographically chained two-tier structure where each registration block is linked to its corresponding login block [1]. During registration, we securely record key user data like Voter ID, Aadhaar, phone number, passkey, and face encoding into a user-specific registration block. After successful registration, this data is used to generate a Merkle Root Hash. For every login attempt, a new login block is created with a timestamp and a cryptographic reference to the corresponding registration block's Merkle Root Hash [20, 25]. This two-level block structure as described in Figure 4 ensures cryptographic traceability and data integrity. Although our system isn't a fully decentralized blockchain, it effectively emulates key properties such as sequential hashing and tamper evidence within a centralized framework built on Firebase

[33]. This hybrid configuration provides blockchain-like verification in a lightweight, responsive, and scalable infrastructure, making it ideal for prototyping and contexts where full blockchain deployment isn't yet feasible [3]. Unlike existing research that focuses solely on the voting process, our system integrates blockchain security concepts at the user onboarding stage. This ensures the voter's identity is immutable and tamper-proof, effectively preventing duplicate or phony registrations. The use of Merkle Root Hash verification between registration and login provides tamper-proof timestamp tracking, enabling accurate auditing of user behavior and protecting against unauthorised access and replay attacks. Furthermore, Firebase enhances real-time performance and scalability, providing a clear path for future migration to a fully decentralized system.

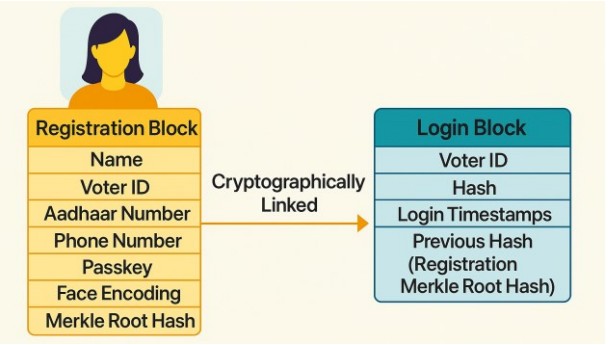

Figure 4
Two-tier Cryptographically Linked Registration and Login Blocks

3.3 *Multichain Voting Architecture*

In the proposed system, each competing candidate is given a logically separate vote chain on the blockchain, which uses a multichain blockchain architecture [2, 20, 33]. Because every vote cast for a candidate is only recorded on that candidate's vote chain, the system can separate vote data and handle transactions on its own. By spreading vote entries across several chains, this method improves speed and scalability while lowering network congestion and increasing processing efficiency [33, 35]. In addition to enhancing efficiency, this vote record separation by candidate minimises the risk of fraud or data manipulation by creating auditable, tamper-evident transaction logs [20, 34]. The technology ensures data integrity, enhances transparency, and increases the overall legitimacy of the election process by maintaining independently verified ledgers for every participant [17].

3.4 *Merkle Root Hash Computation*

The Merkle Root Hash is computed hierarchically by grouping individual leaf nodes or data entries into a binary tree structure [25]. Each leaf node originally signifies the cryptographic hash of a single data element, and these

hashes are paired up and combined using a secure cryptographic function like SHA-256 [20]. In situations where an odd number of nodes is present at a level, the last node is copied to ensure a complete binary structure. This recursive aggregation is illustrated pictorially in Figure 5. The system keeps hashing the nodes of the tree structure recursively until it gets a single unique hash value.

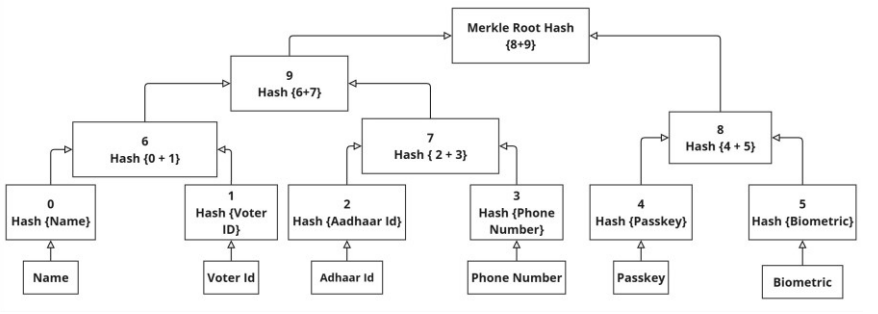

Figure 5
Merkle Tree Construction from User Credentials for Root Hash Generation

The basic operation employed to build every internal node of the Merkle tree can be represented mathematically as illustrated in Equation (4), where H_i represents the parent node hash, h_{2i-1} and h_{2i} are the two child hashes being combined, and P denotes concatenation. This recursive hashing continues up the tree until a single unique hash value (Merkle Root) is obtained, securely representing the integrity of all the underlying voter credentials.

$$H_i = \text{Hash}(h_{2i-1} || h_{2i}) \tag{4}$$

3.5 *Tracking ID Mechanism*

To have secure, anonymous, and user verifiable voting, the system creates a separate Tracking ID for every voter. Using this ID, voters can be assured their vote is captured on the blockchain without revealing their identity [1, 3, 25]. The system determines the minimum number of bits b for a unique ID for a population of $N = 1.438 \times 10^9$ voters using the following formula as shown in Equation (5):

$$b = \lfloor \log_2(N) \rfloor \tag{5}$$

Substituting N, we get the following Equation (6):

$$\log_2(1.438 \times 10^9) \approx 30.47 \tag{6}$$

This results in approximately 31 bits. However, a buffer of 34-40 bits is added to minimize collision risk, as collisions become likely near $2^{b/2}$. For instance, 34 bits can represent roughly 10^{10} unique values, comfortably covering over 1.438 billion voters. This allows for the assignment of a compact 10-digit identifier (e.g., “1234567890”), balancing uniqueness, scalability, and usability. Each voter’s Tracking ID securely tracks their vote without revealing identity by integrating Merkle Root hashing, quantized facial encodings, and XOR-based

encoding to produce a tamper-proof verification token. A Merkle Root Hash (MRH) is first generated from registration data using SHA-256 as shown in Equation (7) below:

$$\text{MRH} = \text{SHA} - 256(\text{Voter\_Data}) \tag{7}$$

Since the MRH is long, further optimization is achieved using quantized face embeddings. Facial Recognition models produce a 128-dimensional floating-point vector $E = \{e_1, e_2, \dots, e_{128}\}$, which is binarized based on its sign to a 128-bit binary representation Q which is shown in Equation (8):

$$Q_i = \begin{cases} 1, & \text{if } e_i > 0 \\ 0, & \text{otherwise} \end{cases} \text{ for } i = 1,2, \dots ,128 \tag{8}$$

A bitwise XOR operation between the Merkle Root Hash (MRH) and the quantized facial encoding vector Q blends cryptographic identity with biometric data to produce a hybrid output X as shown in below Equation (9):

$$X = MRH \oplus Q \tag{9}$$

From the resulting hybrid value X, a 40-bit segment is extracted as shown in Equation (10):

$$X_{40} = \text{Truncate}(X, 40 \text{ bits}) \tag{10}$$

This is then converted into a 10-character alphanumeric Tracking ID using Base62 encoding as shown in Equation (11):

$$TrackingID = \text{Base62}(X_{40}) \tag{11}$$

Although collisions are highly unlikely, the system performs a collision check against existing records. If a match is found, the system modifies the XOR output and regenerates a new ID using a retry counter B_i which are shown in Equation (12) and Equation (13):

$$\begin{aligned} &\text{If } TrackingID \in \text{Firebase, then} \\ &X \leftarrow \text{Modify}(X) \Rightarrow \text{Repeat} \end{aligned} \tag{12}$$

$$X_{\text{new}} = X \oplus B_i \tag{13}$$

This ensures that each voter receives a unique, collision-free, and anonymous Tracking ID. The final functional mapping is summarized as shown in Equation (14):

$$TrackingID = f(\text{MerkleRoot}) \tag{14}$$

The complete process for generating the Tracking ID is presented in Algorithm 3 and visually illustrated in Figure 6, which consolidates the mathematical formulation into a practical workflow. The Tracking ID combines cryptographic hashing with facial feature encoding to create a verifiable, anonymous, and collision-resistant identifier. Prevent identity leakage, traceability loopholes, and duplicate registrations, moving beyond existing schemes which is demonstrated in Table 1. A comparative evaluation of the proposed tracking ID mechanism against existing schemes is presented in Table 2. The results highlight improvements in collision resistance, privacy preservation, and traceability over previous models. This integration provides a

robust, privacy-preserving, and verifiable identity layer essential for secure e-voting.

Algorithm 3: Tracking ID Generation Algorithm

Input: Merkle Root Hash (MRH) of registration block, Face encoding E
Output: Unique 10-character alphanumeric Tracking ID
Preconditions:
– Face encoding E must be stored securely
– Merkle Root Hash (MRH) must be unique
Compute 128-bit binary representation from E
Compute XOR: $T \leftarrow MRH \oplus E$
Extract 40-bit segment from T
Convert 40-bit segment to 10-character alphanumeric ID using Base62 encoding
Check Firebase for collision with existing Tracking IDs
if *Tracking ID exists in Firebase* **then**
    Modify XOR output using retry counter
    Repeat the extraction and Base62 conversion steps
end
Map unique Tracking ID to MRH for secure linkage
return Tracking ID
End

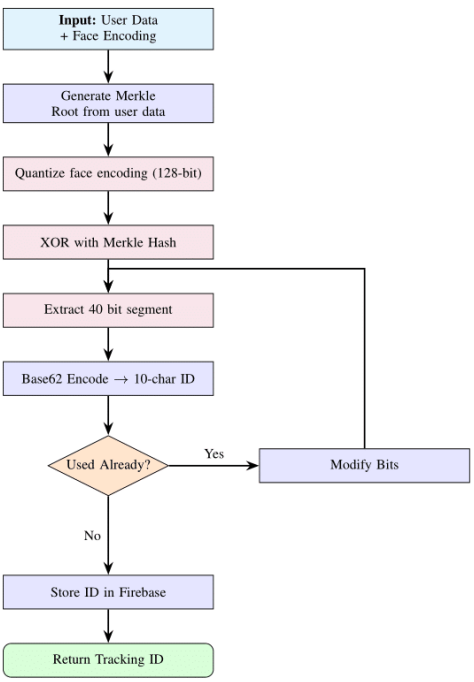

Figure 6
Flowchart for Tracking ID Generation from Merkle Hash and Face Encoding.

Table 2

Tracking ID Mechanisms Used in Blockchain-Based E-Voting Systems and Their Limitations

Paper	Tracking ID Mechanism	Limitation / Disadvantage
Transforming Online Voting [1]	Traceable via blockchain	Traceable but no user-side verification
S3Voting [2]	End-to-end verifiability using Homomorphic Time-Lock Puzzle	Complex verification is indirect and requires computation, no readable Tracking ID
DemocracyGuard [3]	QR-based anonymity, no Tracking ID system	Provides anonymity but no traceability to the selected candidate
Secure Electronic Voting Machine [4]	Serial ID generation	No blockchain linkage; tracking is local and non-auditable
Basic Blockchain E-Voting [10]	None	Public key traceable to voter identity; leaks partial anonymity
TrustChain (Proposed Solution)	Merkle Root Hash and Face Encoding	No significant disadvantages; verifiable, anonymous, and collision-resistant with full blockchain traceability

3.6 *Security Layer Stack*

The proposed e-voting system employs a multi-layered security architecture to ensure privacy, integrity, and verifiability across all user actions and blockchain operations which is depicted in Figure 7. At the foundation, Firebase storage security ensures that personal data, including registration blocks with facial encodings and login blocks, is hashed with SHA-256-based Merkle Roots and securely stored, preventing unauthorized access [20]. Building on this, face recognition authentication employs deep learning with 128-dimensional encodings and strict Euclidean thresholds to verify voter identity and prevent impersonation [4, 30]. Next, Merkle Root hash comparison strengthens login integrity by generating a fresh Merkle Root for each attempt, which is validated against the registered root to detect tampering and guarantee fast integrity checks [20, 25]. In addition, timestamped login hashes create a tamper evident audit trail, mitigating replay attacks through precise temporal linking of login events. Finally, blockchain ledger security records voting actions on immutable, per candidate chains, providing decentralized transparency and reinforcing system trustworthiness [2, 33]. Collectively, this layered security stack ensures that every critical operation within the e-voting framework is independently safeguarded, providing end-to-end protection against vote fraud, data manipulation, and identity theft without compromising auditability or anonymity.

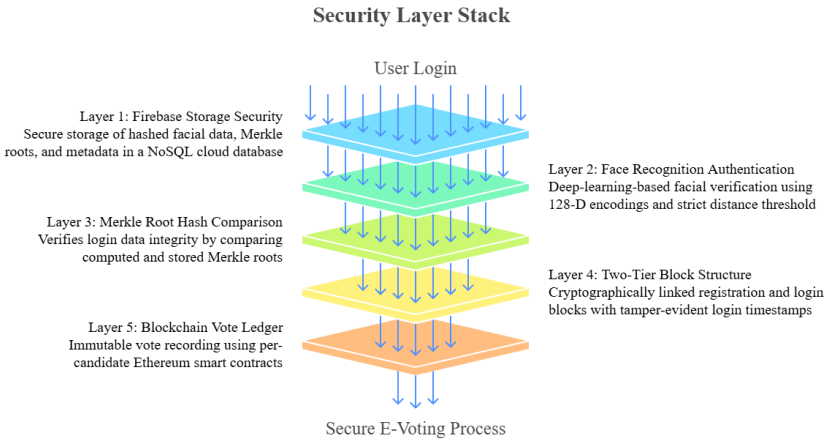

Figure 7
Five Layered Security Stack implemented across the e-voting framework

4. Methodology

To provide a safe, transparent, and tamper-evident voting process, the system being proposed applies blockchain technology, Facial Recognition, Merkle Root Hash verification, and full blockchain traceability without tampering with privacy. Through a multi-step process, the system maintains voter anonymity and security while permitting only genuine and distinct individuals to vote. Figure 8 illustrates the overall architecture and workflow of the system.

4.1 *Voter Registration Process*

At registration, every voter submits name, Voter ID, Aadhaar ID, phone number, passkey, and a live facial photograph for biometrics verification. The face encoding is pulled from the photograph by a previously trained CNN model (ResNet-34 via dlib) [30, 31]. Duplicate registrations are avoided by a Euclidean distance criterion of 0.38 [4], which establishes a duplicate flag on each match and also thwarts submissions where Voter ID, Aadhaar, phone number, or face encoding are already present. Once authenticated, a registration block is established with voter information and a Merkle Root Hash (MRH) for data integrity [20]. Each voter is provided a distinct Tracking ID which is the output from the MRH and face encoding (see Section 3.5) [25], allowing tracing of ballots without voter ID exposure. The registration block is securely linked to a login block to establish a tamper-evident chain from registration to authentication [1]. The blocks comprise a pseudo-blockchain structure that displays decentralized integrity in ledgers while allowing responsiveness in real-world elections. This registration-login chaining to ensures transparency and verifiability without comprising voter privacy. Algorithm 4 outlines biometric verification, Merkle Hashing, and Tracking ID generation for secure voter onboarding for later authentication.

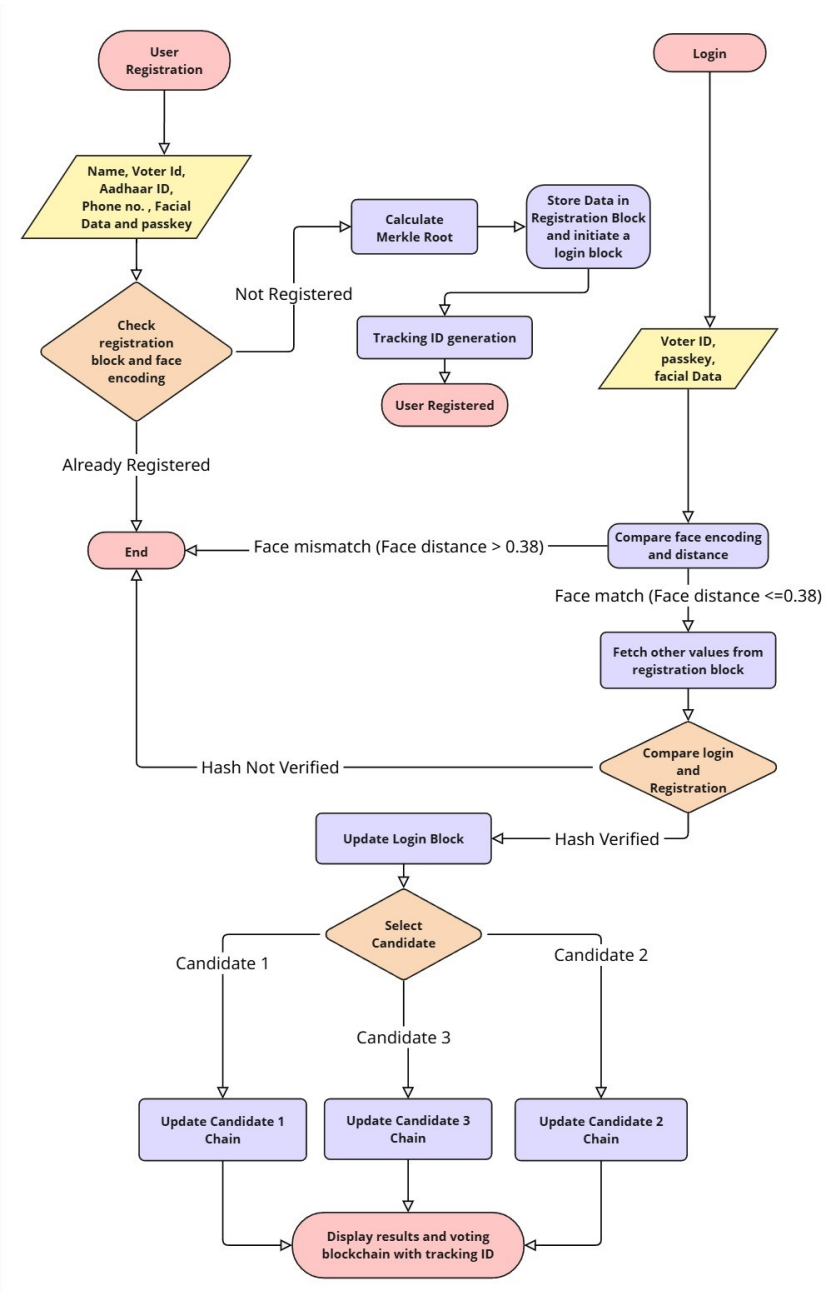

Figure 8
Workflow of the TrustChain E-Voting System with Facial Recognition and Blockchain Integration

Algorithm 4: Registration Process in a Pseudo-Blockchain Architecture

Input: Name, VoterID, AadhaarID, PhoneNumber, Passkey, Face Image I
Output: Registration Success (1) or Failure (0)
Preconditions:
  – *VoterID*, *AadhaarID*, and *PhoneNumber* must be unique
  – Facial encoding must not match existing encodings within $d \leq 0.38$
Convert I to RGB format
Compute facial encoding: $E \leftarrow$ face_encoding(I)
foreach *stored encoding* E_i *in database* **do**
  Compute Euclidean distance: $d(E, E_i) = \sqrt{\sum (E - E_i)^2}$
  if $d(E, E_i) \leq 0.38$ **then**
    return Reject Registration; Abort Procedure
  end
end
Compute Merkle Root Hash (MRH) from voter data
Convert E to 128-bit binary representation
Compute XOR: $T \leftarrow MRH \oplus E$
Extract 40-bit segment from T
Convert to 10-character alphanumeric Tracking ID
Store registration block containing:
  Name, VoterID, AadhaarID, PhoneNumber, Passkey, Face Encoding (E), MRH, Tracking ID
Initiate cryptographically linked login block
return Registration Success (1)
End

4.2 *Voter Login Process*

The voter login authenticates using Facial Recognition and Merkle Root Hash (MRH) verification. Voters are prompted for their Voter ID, passkey, and a live face image I'. Without a Voter ID, login is refused. Otherwise, the image is matched against the encoded image stored with a strict Euclidean limit ≤ 0.38 [4]. On success, Aadhaar ID, phone, and encoding are accepted and a new MRH calculated and matched against stored MRH [20]. On mismatch, tampering is reported [25]. On verification, authentication succeeds and a new login block with timestamp and verification hash is built [3]. Algorithm 5 formalizes the same.

4.3 *Voting Process*

Upon successful authentication, voters enter the ballot via a multi-chain blockchain procedure that delivers maximum security and visibility [2]. Each aspirant is provided with a unique vote chain (see Section 3.3), keeping votes isolated, tamper-evident, and autonomously verifiable [2, 20, 33]. Upon authentication, the voter attaches a MetaMask wallet, chooses a candidate, and submits a vote. Prior to approval, the system validates if the voter has already cast a ballot. If not, the vote is added to the candidate's chain. Each vote block is signed with Keccak-256, hashing candidate ID, tracking ID, timestamp, and the prior block hash. The whole election procedure is executed on Ethereum via Smart Contracts, facilitating decentralized, automated, and tamper-proof voting

[33, 35]. MetaMask is both wallet and gateway to Ethereum [33], providing for verified identity and hassle-free interactivity. There is a stout locking system re-verifying voter ID, allowing one vote per person. Every ballot is captured as a blockchain transaction, facilitating real-time counting and obviating the necessity for a separate count [24, 34]. This ensures that post-vote tampering is avoided by the maintenance of immutable time-stamped records. Figure 9 illustrates the voting procedure, while Algorithm 6 details the corresponding steps. They together illustrate how decentralization and automation guarantee anonymity, auditability and transparency for voters and integrity to the system while improving fairness, efficiency, and credibility.

Algorithm 5: Voter Login Algorithm

Input: Voter ID, Passkey, Captured face image I'
Output: Authentication success (1) or failure (0)
Preconditions:
–Face encoding E is securely stored in the blockchain
– Threshold distance must satisfy $d \leq 0.38$
Convert I' to RGB format and Compute face encoding: $E' \leftarrow$ face_encoding(I')
Compute Euclidean distance with stored encoding: $d = \sqrt{\sum(E - E')^2}$
if $d > 0.38$ **then**
  return Reject Authentication; Abort Procedure
end
Retrieve stored Merkle Root Hash (MRH) and other detals from blockchain
Compute new hash: $MRH' =$ SHA-256($VoterID \parallel Passkey \parallel E' \parallel ...$)
if $MRH' \neq MRH$ **then**
  return Reject Authentication; Abort Procedure
end
Update and timestamp login block
Compute new login block hash:
$NewHash =$ SHA-256(timestamp$_1$ $\parallel$ timestamp$_2$ $\parallel$ timestamp$_3$ $\parallel$ ...)
return Authentication Success (1)
End

4.4 *Vote Display*

One of the most crucial elements of the e-voting system put forward is the assignment of a Tracking ID to every voter at the point in time their vote is cast successfully. This Tracking ID is an anonymous, secure identifier enabling the voter to verify in private whether their vote has been properly registered on the blockchain without exposing their identity or any other sensitive personal details [1, 3, 25]. The employment of Tracking IDs is a bedrock of ensuring transparency, accountability, and voter confidence within the electronic voting process. After casting a vote, the voter can access a simple, easy to use blockchain viewer interface that displays immutable vote records with metadata such as block number, candidate ID, timestamp, and anonymous tracking ID. Voters can search their unique ID to confirm which candidate their vote has been cast for. The retrieval and mapping process is detailed in Algorithm 7. These records, rendered in a card-style design which is shown in Figure 10, update in real time to

show the latest transactions, making the system transparent, readable, and user-friendly even for non-technical individuals. When users choose "View Blockchain", the system connects to Ethereum via MetaMask [33], and Web3.js bridges the front-end with smart contracts [33, 35], retrieving real-time vote data so the viewer always reflects the current ledger state. Concurrently with blockchain viewing, the system provides a live vote count dashboard which is shown in Figure 11. Each chain grows with new votes, and valid counts are computed by excluding the genesis block, which is only a placeholder. Algorithm 8 outlines this process, ensuring only legitimate votes are tallied and clearly presented.

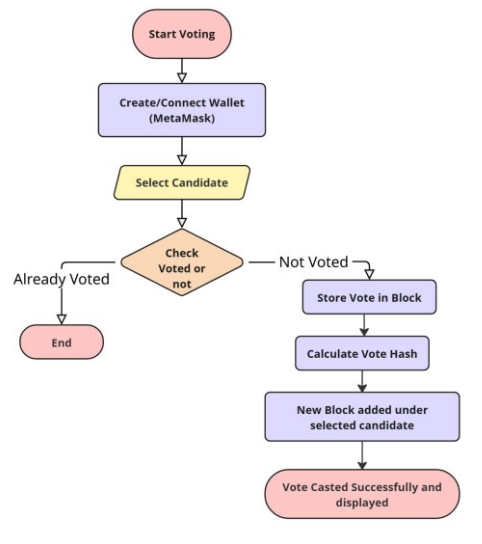

Figure 9
Flowchart for Voting Process

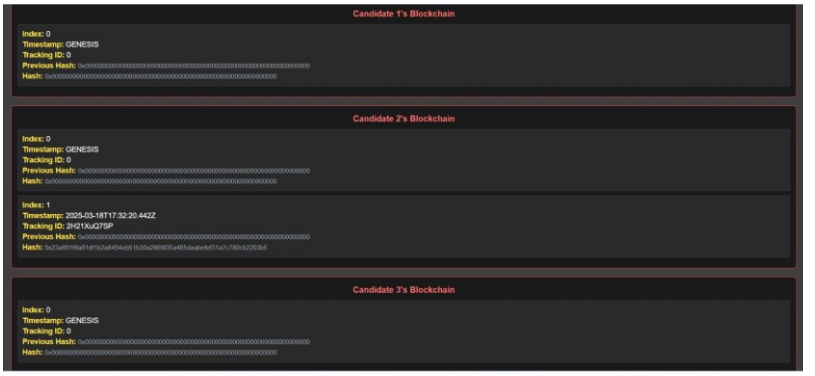

Figure 10
Visual Display of Voting Blockchain with Tracking ID after casting Vote to Candidate 2

Algorithm 6: Vote Casting Algorithm

Input: Voter authentication status, Candidate ID, Tracking ID, Smart Contract Instance

Output: Vote successfully cast (1) or failure (0)

Preconditions:
- Voter must be authenticated via secure login (MetaMask verified)
- Voter must not have already cast a vote
- Smart contract must be deployed and connected

if *Authentication == Success* **then**
  if *Vote has not been cast by this Tracking ID* **then**
    Retrieve hash of previous block from candidate's blockchain
    Create vote block with:
      - Candidate ID
      - Tracking ID
      - Timestamp
      - Previous Block Hash
    Current Block Hash = `Keccak256(CandidateID || TrackingID || Timestamp || PrevHash)`
    Call `castVote(CandidateID, TrackingID, CurrentHash)` on Smart Contract
    MetaMask prompts transaction confirmation
    if *Transaction confirmed on Ethereum* **then**
      Record vote block with Current Hash in candidate's blockchain
      return Vote successfully cast (1)
    end
    else
      return Vote failed due to transaction error (0)
    end
  end
  else
    return Vote rejected: User has already voted (0)
  end
end
else
  return Vote rejected: Authentication failed (0)
end
End

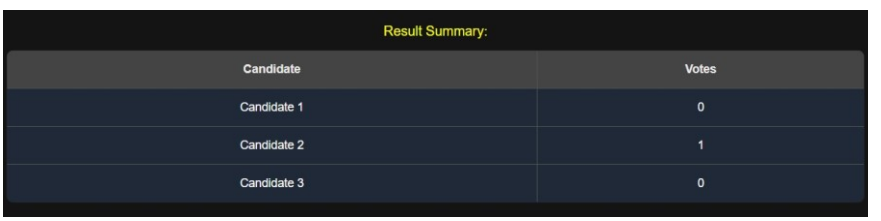

Figure 11
Blockchain Table Display Post Vote Cast to Candidate 2

Algorithm 7: View Blockchain Votes Using Tracking ID

Input: Smart contract instance, Candidate IDs (1 to N), user-provided Tracking ID
Output: Highlight user's Tracking ID if found in any blockchain

Preconditions:
- Voter must be authenticated (logged in)
- Blockchain network must be connected via MetaMask
- Candidate blockchains must be deployed and accessible

for *CandidateID* $\leftarrow 1$ **to** N **do**
  Retrieve `chainLength` $\leftarrow$ getChainLength(CandidateID)
  for $i \leftarrow 1$ **to** *chainLength* - *1* **do**
    Retrieve block: $B \leftarrow$ getBlock(CandidateID, i)
    Display block details in Candidate-specific chains: block number, Candidate ID, timestamp, Tracking ID
  end
end
End

5. Core Modules and Their Functional Overview

The proposed e-voting system integrates a set of core modules that collectively reinforce security, privacy, anonymity, scalability, and verifiability, ensuring a trustworthy, open and transparent process. It begins with facial authentication using the ResNet-34 model of dlib, allowing secure and contactless verification by comparing facial encodings with those stored securely in Firebase to ensure that only the legitimate user can access the system [4, 30, 31]. Access control is further strengthened by a pseudo-blockchain mechanism supported by Firebase, where registration generates a pseudo-block containing facial data and metadata, and each login produces a timestamped block hash-linked to the registration record. These are chained via Merkle Root hashing, creating a tamper-evident and auditable authentication history with anomaly detection capabilities [1, 3, 20, 25]. The system adopts a hybrid blockchain model where registration and login are handled on Firebase's pseudo-blockchain and the voting process is executed on Ethereum through candidate-specific smart contracts, thereby avoiding congestion common in single-chain systems [2, 33]. Privacy-preserving vote verification is achieved through a unique Tracking ID, derived from the Merkle Root Hash of registration data and binarized facial encodings using XOR and Base62 encoding, which ensures collision resistance and blockchain traceability without revealing biometric information [20]. Finally, a layered security and modular architecture enables voters to confirm their ballots through a blockchain explorer displaying hashes, tracking IDs, and timestamps, with Ethereum integration secured via MetaMask and smart contract communication through Web3.js. Thus providing real-time transparency, auditability, and end-to-end protection.

Algorithm 8: Vote Tally Table Update Algorithm

Input: Candidate IDs $C = \{C_1, C_2, \ldots, C_n\}$, smart contract instance
Output: Updated vote counts in the tally table

Preconditions:
- All candidate blockchains must be successfully deployed
- Smart contract instance must be connected and active
- The system must have read access to each candidate's blockchain

foreach *candidate* $C_i \in C$ **do**
  Retrieve chain length: $L_i \leftarrow \text{getChainLength}(C_i)$
  Compute valid votes: $V_i \leftarrow L_i - 1$; `// Exclude genesis block`
  Update tally table with V_i for candidate C_i
end
End

6. Results

The suggested blockchain-based electronic voting system was tested through registration, login, casting of votes, and verification, highlighting the cryptographic tracking ID and multichain approach. Action Frequency Analysis which is shown in Figure 12 emphasizes efficient and balanced voter interactions such as registration, authentication, vote casting, and verification through the blockchain without redundancy or peak, confirming that the Tracking ID mechanism discourages duplicate actions and maintains workflow integrity [25]. Latency evaluation which is shown in Figure 13 and 14 validates responsiveness, with the mean latency for all operations remaining minimal even when under load, having a close to linear trendline to reflect that the multichain blockchain structure distributes computational load effectively without bottlenecks [2].

Latency Distribution and Stability which is shown in Figure 15, 16 and 17 also support reliability, the boxplot presents tight interquartile ranges with no extreme outliers, the histogram presents a left-skewed distribution with mostly low latency and no long-tail spikes, and the scatter plot presents evenly distributed response times over long runs, affirming predictable system behavior and the strength of the blockchain monitoring infrastructure. Lastly, the cryptographic tracking which is shown in Figure 18 depicts a KDE unimodal

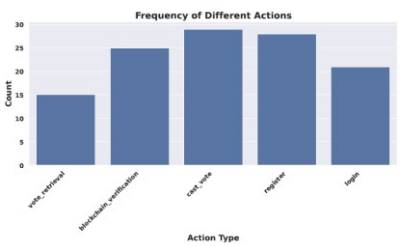

Figure 12
Action Frequency: Frequency of User Interactions (Registration, Login, Voting, and Verification)

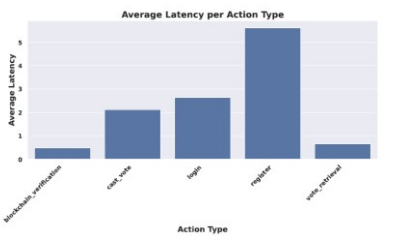

Figure 13
Average Latency Across Core System Operations

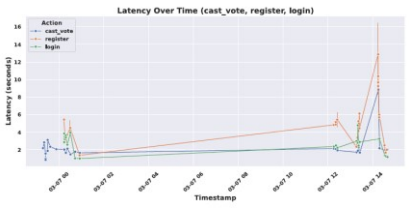

Figure 14
Latency Trend Over Time Indicating System Stability

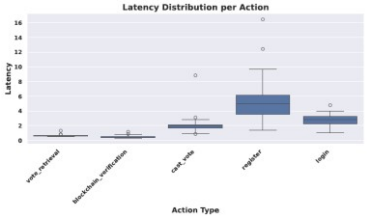

Figure 15
Boxplot of Latency Showing Low Variance and Minimal Outliers

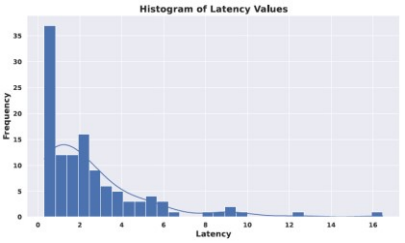

Figure 16
Histogram of Latency Values Across Sessions

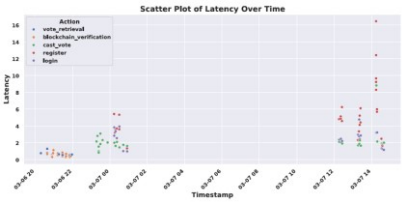

Figure 17
Scatter Plot of Login/Vote Latency per Session

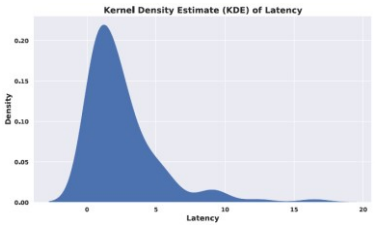

Figure 18
KDE Plot of Latency: Single Peak Near 2 Seconds Confirms Low Delay

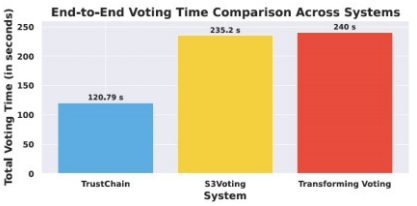

Figure 19
Phase-wise Latency Comparison Across Voting Systems

latency peak, confirming that most transactions are processed at an almost constant rate and establishing that the cryptographic tracking ID mechanism does not cause processing overhead, thus providing stable, efficient, and reliable system behavior.

Comparison with Existing Works: The TrustChain system proposed was tried with 20 users who finished registration, biometric login, vote casting, and blockchain verification. The simulated environment was subjected to realistic scenarios of live face detection, Ethereum integration, and Merkle Root Hashing for integrity. The result was an average end-to-end latency around 2 minutes per user, and the linear nature of the system proves its scalability to maintain time below 2.5 minutes per user even at 100 voters. Table 3 contrasts TrustChain against current blockchain systems. M. Li et al.'s S3Voting [2] takes 39.2 minutes

for 10 aspirants at an average 3.92 minutes (235 seconds) per voter, while Faruk et al. [1] records a 4-minute (240-second) per-user performance. As against those Figures, TrustChain records 2 minutes 10 seconds (130 seconds), thus saving voting time by more than 40% compared to S3Voting and around 45% compared to Transforming Online Voting, as Figure 19 illustrates. The performance gains are a result of a pseudo-blockchain for rapid registration or login, swift face encoding for biometrics, and parallelised Ethereum chains for simultaneous writes per aspirant that eliminate single-chain bottlenecks.

TrustChain therefore offers faster voting and performs biometric verification, maintaining vote integrity and blockchain verifiability. The combination of Tracking ID, Merkle Roots, per-module chain partitioning, and localized facial authentication balances efficiency and scalability and reliability, and thus makes TrustChain both faster and more robust for practical e-voting.

Table 3

End-to-End Voting Time Comparison Across E-Voting Systems

System	Total Time (seconds)	Time Description
Transforming Online Voting [1]	240	Reported total time from accessing the voting portal to vote submission
S3Voting [3]	235	Estimated duration including ballot generation, casting, and tallying in a sharded setup
TrustChain (Proposed System)	130	Measured time including registration, biometric login, vote casting, and blockchain view (empirical observation)

7. Conclusion

This work presents *TrustChain*, a scalable and secure e-voting platform designed to overcome persistent issues in electronic voting. By combining blockchain, cryptographic tracking, and facial recognition, the system enables concurrent voting, reduces congestion, and ensures transparent real-time tabulation. A central feature is the *Tracking ID*, a cryptographically generated token that follows each voter from registration to verification. Security is reinforced through pseudo-blockchain login, Firebase support, and Merkle Root hashing, which preserve voter anonymity while preventing double voting. Tests demonstrated low latency, stable performance, and strong scalability. With Ethereum smart contracts and MetaMask integration, votes are recorded immutably in a decentralized way. Looking ahead, TrustChain aims to adopt post-quantum cryptography, mobile voting, and broader accessibility, shaping a secure, inclusive, and resilient foundation for digital democracy.

8. Future Scope

The proposed system establishes a strong foundation for future growth in security, accessibility, and scalability. A key direction is extending *TrustChain* to mobile-based voting, ensuring safe participation for citizens in rural or remote

areas. Improving accessibility for disabled voters, senior citizens, and those with mobility challenges is equally important to guarantee inclusivity. Biometric modules can be enhanced with deep learning-based facial recognition and AI-driven liveness detection to counter spoofing attacks. Additional safeguards such as Two-Factor Authentication (2FA) with biometric keys or time-sensitive OTPs will further strengthen login security. At the national level, integration with electoral databases will require policy standardization and collaboration. For scalability, blockchain sharding and backend optimizations will enable millions to vote reliably. With quantum computing on the horizon, post-quantum cryptography will be essential, ensuring *TrustChain* evolves as a secure and future-proof digital democracy solution.

References

[1] M. J. H. Faruk, F. Alam, M. Islam, and A. Rahman, "Transforming online voting: a novel system utilizing blockchain and biometric verification for enhanced security, privacy, and transparency," *Cluster Computing*, vol. 27, no. 4, pp. 4015–4034 (Apr. 2024), doi: 10.1007/s10586-023-04261-x.

[2] M. Li et al., "S3voting: A blockchain sharding based e-voting approach with security and scalability," *IEEE Transactions on Dependable and Secure Computing*, vol. 22, no. 2, pp. 1–16 (Jan. 2024), doi: 10.1109/tdsc.2024.3446392.

[3] M. S. Peelam, G. Kumar, K. Shah, and V. Chamola, "Democracyguard: Blockchain-based secure voting framework for digital democracy," *Expert Systems*, vol. 42, no. 2, p. e13694 (Aug. 2025), doi: 10.1111/exsy.13694.

[4] B. Jasdev, C. Satvik, W. Ansh, and V. Abhishek, "Secure electronic voting machine using multi-modal biometric authentication system, data encryption, and firewall," *International Journal of Performability Engineering,* vol. 15, no. 10, pp. 2570–2577 (Jan. 2019), doi: 10.23940/ijpe.19.10.p2.25702577.

[5] S. Zhang, L. Wang, and H. Xiong, "Chaintegrity: blockchain-enabled large-scale e-voting system with robustness and universal verifiability," *International Journal of Information Security*, vol. 19, no. 3, p. 323–341 (Sep. 2019), doi: 10.1007/s10207-019-00465-8.

[6] M. V. Vladucu, Z. Dong, J. Medina, and R. Rojas-Cessa, "E-voting meets blockchain: A survey," *IEEE Access*, vol. 11, pp. 23 293–23 308 (Jan. 2023), doi: 10.1109/ACCESS.2023.3253682.

[7] R. Fatih, S. Arezki, and T. Gadi, "A review of blockchain-based e-voting systems: Comparative analysis and findings," *International Journal of Interactive Mobile Technologies* (iJIM), vol. 17, no. 23, pp. 49–67 (Dec. 2023), doi: 10.3991/ijim.v17i23.45257.

[8] N.-M. Botezatu, "Electronic voting system using blockchain technology," *International Journal of Advanced Statistics IT & Communication and Economics Life Sciences*, vol. 14, no. 1, pp. 205–211 (2024), doi: 10.2478/ijasitels-2024-0008.

[9] H. V. Patil, K. G. Rathi, and M. V. Tribhuwan, "A study on decentralized e-voting system using blockchain technology," *International Research Journal of Engineering and Technology*, vol. 5, no. 11, pp. 48–53 (Nov. 2018). [Online]. Available: https://www.irjet.net/archives/V5/i11/IRJETV5I1109.pdf

[10] A. Indapwar, "E-voting system using blockchain technology," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 3, pp. 2775–2779 (Jun. 2020), doi: 10.30534/ijatcse/2020/45932020.

[11] A. Alshehri, M. Baza, G. Srivastava, W. Rajeh, M. Alrowaily, and M. Almusali, "Privacypreserving e-voting system supporting score voting using blockchain," *Applied Sciences,* vol. 13, no. 2, p. 1096 (Jan. 2023), doi: 10.3390/app13021096.

[12] D. D. Kumar, D. V. Chandini, and D. Reddy, "Secure electronic voting system using blockchain technology," *International Journal of Smart Home*, vol. 14, no. 2, pp. 31–38 (Oct. 2020), doi: 10.21742/ijsh.2020.14.2.04.

[13] S. Tanwar, N. Gupta, P. Kumar, and Y.-C. Hu, "Implementation of blockchain-based e-voting system," *Multimedia Tools and Applications,* vol. 83, no. 1, pp. 1449–1480 (May 2023), doi: 10.1007/s11042-023-15401-1.

[14] Priyanka, B. Keswani, and R. Hussain, "Blockchain-based cryptography model to preserve privacy for smart contracts," *Journal of Discrete Mathematical Sciences and Cryptography,* vol. 24, no. 8, pp. 2279–2289 (2021), doi: 10.1080/09720529.2021.2015091.

[15] C. V. Kumar and P. Selvaprabhu, "An examination of distributed and decentralized systems for trustworthy control of supply chains," *IEEE Access,* vol. 11, pp. 137 025–137 052 (Jan. 2023), doi: 10.1109/ACCESS.2023.3338739.

[16] J. I. Ademola, A. M. Mustapha, and T. E. Abioye, "An improved e-voting system using blockchain technology," Avaialable at SSRN: https://ssrn.com/abstract=4733290 (2024), doi: 10.2139/ssrn.4733290.

[17] M. H. Berenjestanaki, H. R. Barzegar, N. E. Ioini, and C. Pahl, "Blockchain-based e-voting systems: A technology review," *Electronics,* vol. 13, no. 1, p. 17 (Dec. 2023), doi: 10.3390/electronics13010017.

[18] K. M. u. Mannonov and S. Myeong, "Citizens' perception of blockchain-based e-voting systems: Focusing on tam," *Sustainability,* vol. 16, no. 11, p. 4387 (May 2024), doi: 10.3390/su16114387.

[19] H. O. Ohize, A. O. Olumide, I. O. Akinyemi et al., "Blockchain for securing electronic voting systems: A survey of architectures, trends, solutions, and challenges," *Cluster Computing*, vol. 28, no. 2, p. 132 (Nov. 2024), doi: 10.1007/s10586-024-04709-8.

[20] D. Kumari, N. Veni, P. Kumar, M. H, and H. Purohit, "Votereum: Blockchain based secure voting system," in 2024 4th International Conference on Pervasive Computing and Social Networking (ICPCSN). Salem, India: *IEEE*, pp. 580–584 (2024), doi: 10.1109/ICPCSN62568.2024.00097.

[21] G. Pranitha, T. Rukmani, T. N. Shankar, N. Kumar, and S. Padhy, "Utilization of blockchain in e-voting system," in 2022 2nd International Conference on Intelligent Technologies (CONIT). Hubli, India: *IEEE*, pp. 1–6 (, Jun. 2022), doi: 10.1109/conit55038.2022.9847995.

[22] F. Abo-Akleek, M. Mowafi, E. S. Taqieddin, and A. S. Shatnawi, "Leveraging blockchain for robust and transparent e-voting systems," *Cyber Security and Applications,* vol. 3, p. 100086 (Feb. 2025), doi: 10.1016/j.csa.2025.100086.

[23] A. S. Yadav, Y. Urade, A. Thombare, and A. A. Patil, "E-voting using blockchain technology," *International Journal of Engineering Research & Technology,* vol. 9, no. 7 (Jul. 2020), doi: 10.17577/IJERTV9IS070183.

[24] A. Singh, R. Ganesh, V. Patil, M. Ali, and A. Razaque, "Secure voting website using ethereum and smart contracts," *Applied System Innovation,* vol. 6, no. 4, p. 70 (Aug. 2023), doi: 10.3390/asi6040070.

[25] A. Marouan, M. Badrani, A. Zannou, N. Kannouf, and A. Chetouani, "E-voting system based on blockchain for enhanced university elections," *SN Computer Science*, vol. 6, no. 3, p. 204 (Feb. 2025), doi: 10.1007/s42979-025-03671-5.

[26] Rahul, P. Gulia, and N. S. Gill, "Articulation of blockchain enabled e-voting systems: A systematic literature review," *Peer-to-Peer Networking and Applications,* vol. 18, no. 3, p. 142 (Apr. 2025), doi: 10.1007/s12083-025-01956-3.

[27] S. T. Alvi, A. Singh, A. Razaque, and M. Tanweer, "Dvtchain: A blockchain-based decentralized mechanism to ensure the security of digital voting system," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 9, pp. 6855–6871 (2022), doi: 10.1016/j.jksuci.2022.06.014.

[28] D. González, D. Frias Mena, A. Massó Muñoz, O. Rojas, and G. Sosa-Gómez, "Electronic voting system using an enterprise blockchain," *Applied Sciences,* vol. 12, no. 2, p. 531 (Jan. 2022), doi: 10.3390/app12020531.

[29] K. M. Khan, J. Arshad, and M. M. Khan, "Secure digital voting system based on blockchain technology," *International Journal of Electronic Government Research,* vol. 14, no. 1, pp. 53–62 (Jan. 2018), doi: 10.4018/IJEGR.2018010103.

[30] V. S. Preiya, R. Prabhu, S. Rani et al., "Blockchain-based e-voting system with face recognition," *Fusion: Practice and Applications,* vol. 12, no. 1, pp. 53–63 (Jan. 2023), doi: 10.54216/FPA.120104.

[31] N. Prathyusha, P. Pooja, and A. Vijay Vasanth, "Blockchain based e-voting system with facial recognition," in 2023 International Conference on Inventive Computation Technologies (ICICT). *IEEE*, pp. 1203–1210 (Apr. 2023), doi: 10.1109/icict57646.2023.10134227.

[32] S. C. Pancholi, A. Nakadi, S. N. Chinta, and R. Mathew, "Integrating blockchain and facial recognition for a decentralized and tamper-proof voting system," in 2024 5th International Conference on Data Intelligence and Cognitive Informatics (ICDICI). Tirunelveli, India: *IEEE,* pp. 95–101 (2024), doi: 10.1109/ICDICI62993.2024.10810799.

[33] A. Spanos and I. Kantzavelou, "Ethervote: a secure smart contract-based e-voting system," *Wireless Networks*, vol. 31, no. 2, pp. 1279–1299 (Aug. 2024), doi: 10.1007/s11276-024-03818-x.

[34] A. Benabdallah, A. Audras, L. Coudert, N. El Madhoun, and M. Badra, "Analysis of blockchain solutions for e-voting: A systematic literature review," *IEEE Access,* vol. 10, pp. 70 746–70 759 (Jan. 2022), doi: 10.1109/ACCESS.2022.3187688.

[35] P. Rani, V. Kumar, I. Budhiraja, A. Rathi, and S. Kukreja, "Deploying electronic voting system use-case on ethereum public blockchain," in 2022 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). Gandhinagar, Gujarat, India: *IEEE,* pp. 1–6 (2022), doi: 10.1109/ANTS56424.2022.10227805.

Received April, 2025