iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: https://doi.org/10.22214/ijraset.2026.85480

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

AI-Driven Voter Access and Authentication in a Blockchain-Based Secure Voting System

Bindu R.[1], HemaShree K.[2], M S Nishamya.[3], Thanushree H R.[4], Dr. Sreenivasa Murthy V.[5]
[1, 2, 3, 4]*Department of Information Science and Engineering, Rajarajeswari College of Engineering, Bengaluru, Karnataka, India*
[5]*Professor and Head, Department of Information Science and Engineering, Rajarajeswari College of Engineering, Bengaluru, Karnataka, India*

Abstract: ***Democratic participation depends on voting; however, conventional voting systems remain vulnerable to voter impersonation, unauthorized entry, repeated voting, and ballot manipulation. This paper proposes an AI-driven voter authentication and access system that combines blockchain technology to strengthen voting security and transparency. Facial recognition is applied to confirm voter identities and permit access only to eligible voters. RetinaFace performs face detection, whereas FaceNet512 produces facial feature embeddings for identity validation. Facial embeddings are compared using Euclidean distance, with FAISS enabling efficient matching of facial features. Ethereum blockchain and Solidity smart contracts store voting records and help prevent duplicate voting. Data integrity is supported through SHA-256 hashing, while Ganache serves as a local blockchain platform for development and testing. The backend is built with Java and Spring Boot, and Web3j connects the application to the Ethereum blockchain. The proposed platform is intended to deliver secure, transparent, and dependable electronic voting. Its performance, however, relies on accurate implementation and additional systematic testing.***
Keywords: Blockchain Technology, Voter Authentication, Face Recognition, Artificial Intelligence, Smart Contracts, Secure Electronic Voting.

I. INTRODUCTION

In democratic societies, voting enables citizens to choose representatives and take part in public decision-making. For a voting system to be secure, all eligible voters must be able to vote without risks of unauthorized entry, identity impersonation, or vote manipulation. Nevertheless, both conventional voting approaches and current electronic voting systems encounter issues involving voter authentication, transparency, data protection, and repeated voting.

Advances in Artificial Intelligence (AI) and blockchain technology create opportunities to build voting systems that are more secure and dependable. AI-based facial recognition can be used to confirm a voter's identity, whereas blockchain offers a distributed and tamper-resistant approach for recording transactions. When combined, these technologies can strengthen voter verification and preserve the integrity of stored votes [1], [2], [5].

The proposed system, titled AI-Driven Voter Access and Authentication in a Blockchain-Based Secure Voting System, integrates facial recognition, and blockchain technology. RetinaFace detects faces, while FaceNet512 produces facial embeddings for identity verification. Facial features are compared through Euclidean distance, and FAISS enables efficient matching [1] – [3].

Votes are recorded through the Ethereum blockchain and Solidity smart contracts, which also help prevent duplicate voting. SHA-256 hashing maintains data integrity, and Ganache offers a local blockchain environment for development and testing. Backend operations are handled through Java and Spring Boot, while Web3j supports communication with the Ethereum blockchain. In accordance with the project presentation, the database is set up for local-server use [4] – [6].

This work seeks to create a secure, transparent, and dependable voting platform that limits access to authorized voters while reducing the risk of impersonation and duplicate voting. By combining AI-based identity verification with blockchain-based vote recording, the proposed method aims to increase the security and trustworthiness of the voting process.

II. LITERATURE REVIEW

The security of electronic voting systems has increasingly been examined through the use of blockchain technology, artificial intelligence, and biometric authentication. Prior studies investigate methods for verifying voters, safeguarding vote records, avoiding repeated voting, and increasing transparency. Nevertheless, concerns regarding authentication accuracy, transaction expenses, privacy, usability, and reliance on centralized registration systems continue to require attention.

©IJRASET: All Rights are Reserved

A. Blockchain-Based Electronic Voting Systems

Distributed ledger technology is used in blockchain-based voting systems to store voting transactions. Through smart contracts, predetermined rules can be applied to confirm voter eligibility and stop an individual from voting more than once in the same election. Ethereum supports the implementation of these functions through Solidity smart contracts [5], [6].

As listed in the project presentation, the 2025 research Blockchain Based E-Voting System by R. K. Rithanya, H. S. A. Harshitha, S. M. Sathya, et al. presents a blockchain-based voting method that uses Ethereum, Solidity, Truffle, Ganache, and Keccak-256 hashing. Its design emphasizes reliable voter authentication, vote integrity, transparency, and the prevention of duplicate voting. Despite these benefits, transaction fees and the challenge of correcting errors in deployed smart contracts remain limitations.

B. Biometric Authentication in Voting Systems

Voter verification can be strengthened through biometric authentication, which relies on distinctive physical traits such as fingerprints or facial characteristics. Rather than depending only on passwords, biometric systems compare newly captured biometric data against information collected during registration. When supported by suitable eligibility checks, this process may reduce identity misuse. The project presentation attributes the 2025 study Blockchain-Based Biometric Voting System Using Face and Fingerprint Authentication for Secure and Transparent Elections to Girish S. C., Nayana M., Priya B. R., Srujana S., and Sushmitha C. The study combines blockchain technology with biometric verification. Its PPT-described methodology includes a CNN algorithm, Flask backend, MetaMask wallet, Ganache, and blockchain-based vote recording. The proposed approach seeks to enhance voter authentication, privacy, and controlled access to voting. However, biometric verification by itself cannot stop voter coercion or outside influence over a voter's choice.

C. Tamper-Resistant Electronic Voting

The project presentation also refers to the 2026 study Blockchain-Based Electronic Voting System for Tamper-Proof Elections by Senthilkumar S., Harish V., Pandian M. A., and Harish B. L. This approach applies a Practical Byzantine Fault Tolerance (PBFT) consensus algorithm, cryptographic techniques, authenticated voting, and auditing to enable secure vote processing.

The system is designed to preserve vote integrity, protect privacy, and automate vote counting. However, the presentation notes limitations involving voter authentication, dependence on centralized registration, and usability. These concerns suggest that blockchain-based vote storage must be paired with dependable voter verification and an appropriately designed registration process.

D. Facial Recognition and Smart Contract Integration

Smart contracts can be combined with facial recognition to link identity confirmation with voting eligibility. Facial features may be generated from a face image and compared with the features of a registered voter to verify the voter's identity before granting access to the voting system. Once the identity is confirmed, smart contracts can validate voting eligibility and prevent duplicate voting by checking whether the voter has already cast a vote. This integration aims to strengthen voter authentication, improve security, and maintain the integrity of electronic voting [1], [5], [6].

E. Research Gap

The studies reviewed in the project presentation reveal several unresolved issues in current electronic voting systems. Conventional voting methods may be vulnerable to identity misuse and provide limited transparency, whereas electronic systems often rely strongly on centralized databases and standard authentication approaches. Although blockchain systems can strengthen the integrity of recorded transactions, they cannot by themselves confirm that the voter is the authorized individual.

Blockchain-based voting methods also face concerns involving transaction costs, smart-contract maintenance, registration dependencies, and accessibility for users. Biometric voting can improve identity verification, but its performance depends on accurate biometric matching and a secure registration procedure. Moreover, biometric authentication alone cannot prevent coercion or ensure ballot secrecy. These findings support the need for an integrated system that combines AI-based facial recognition, voter eligibility validation, and blockchain-based vote recording. The proposed approach applies RetinaFace, FaceNet512, Euclidean distance, and FAISS for facial verification, along with Solidity smart contracts to enforce voting rules and prevent duplicate voting [1] – [3], [5], [6]. This project addresses the gap of combining several voter-authentication methods with blockchain-based vote recording within one voting workflow. The system is intended to allow only authorized voters to cast votes, minimize identity misuse, prevent duplicate voting, and strengthen the integrity of stored voting records. Additional experimental assessment is needed to evaluate authentication accuracy, performance, scalability, privacy protection, and resilience against attacks.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. System Overview

The proposed system, entitled "AI-Driven Voter Access and Authentication in a Blockchain-Based Secure Voting System," aims to enhance the security, dependability, and transparency of electronic voting through the integration of artificial intelligence, biometric authentication, and blockchain technology. It is intended to verify voter identities, limit unauthorized access, prevent repeated voting, and preserve the integrity of voting transactions. Facial recognition methods and blockchain-based voting rules are combined to establish a controlled and secure voting procedure.

The process starts with voter registration, during which the application collects voter details, a voter identification number, and a facial image. The facial information obtained during registration is processed and retained as a reference for future authentication. When a voter accesses the application, a facial image is captured or submitted and then processed using OpenCV and RetinaFace. FaceNet512 produces a 512-dimensional facial embedding, which is compared with the stored representation through Euclidean distance. FAISS assists with similarity searching and matching among facial representations [1] – [3], [7].

The Java backend, implemented with Spring Boot, manages authentication activities and handles requests from the frontend. Access control and authenticated operations are supported by Spring Security and JSON Web Tokens (JWT). The local server stores and manages the application data and voter information required by the system [8], [9].

Following successful authentication, the application verifies the voter's eligibility and current voting status. Eligible voters are granted access to the candidate-selection interface and may submit their votes. The voting component communicates with a Solidity smart contract deployed on the Ethereum network. Web3j links the Java backend with the blockchain, whereas Ganache supplies a local blockchain environment for development and testing purposes. SHA-256 hashing is included in the proposed transaction process to support integrity verification of vote-related information.

The smart contract manages the required voting state and applies configured rules to stop duplicate voting. After a transaction is processed successfully, the system can issue voting confirmation to the user. Through the combined use of biometric verification, backend access restrictions, and blockchain transaction recording, the system seeks to reduce common electronic voting risks and establish a more dependable voting process [4] – [6].

B. System Architecture

The proposed architecture includes several connected elements: the voter interface, application backend, authentication component, biometric recognition component, local server, blockchain integration layer, and Ethereum smart contract. Each component has a defined function and interacts with other modules to complete the overall voting procedure.

The frontend is developed using HTML5, CSS3, and JavaScript. It presents the interfaces required for voter registration, authentication, candidate selection, and vote confirmation. MetaMask and JavaScript support blockchain wallet interactions when these are required by the application.

Java and Spring Boot are used in the backend layer to handle requests, coordinate voting activities, and communicate with authentication, biometric, and blockchain services. Spring Security and JWT are responsible for controlling access to protected functions [8].

OpenCV and RetinaFace are used in the biometric authentication layer for facial-image processing and face detection. FaceNet512 produces facial embeddings, while Euclidean distance and FAISS enable comparison between the captured face and registered facial representations. The result of this verification is then supplied to the wider application workflow [2], [3], [7].

The data-management layer relies on a local server to handle the information necessary for voter registration and authentication. It retrieves registered voter data and performs the checks required before voting. The local server and blockchain serve different functions: application-level information is maintained by the local server, whereas the blockchain records transactions and maintains the voting state specified by the smart contract.

Ethereum, Solidity smart contracts, Ganache, and Web3j form the blockchain layer. The smart contract applies the voting rules and maintains the required voting status. Communication between the Java backend and Ethereum is enabled by Web3j, while Ganache supports local deployment and testing. In the proposed process, SHA-256 hashing is applied to vote-related data before blockchain submission [4] – [6].

Communication among these components enables the system to confirm voter identity, evaluate eligibility, process voting requests, and record transactions. A vote can be processed only when the required authentication and voting-state conditions have been fulfilled.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

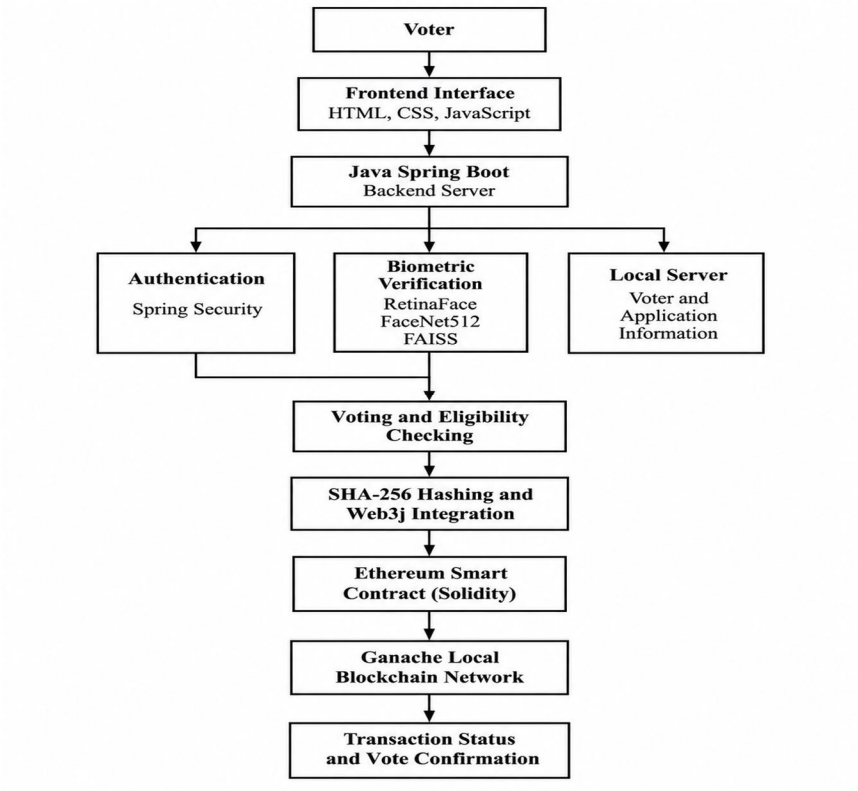


Fig. 1. Overall architecture of the Proposed System.

C. Voter Registration Module

The voter registration module creates the initial identity record needed for a person to participate in the electronic voting process. During registration, the voter submits the required information, including the voter identification number and other details requested by the application. A facial image is also collected to support biometric verification during later login and voting activities.

The backend processes the submitted data and stores it on the local server according to the application's data-management design. The biometric module processes the facial image to create the facial representation needed for future comparisons. This stored representation serves as the reference when the voter later attempts to access the voting application.

Registration is significant because the reliability of later authentication depends on accurate registered information and a sufficiently clear facial image. Required fields must be validated, and the voter identifier must comply with the system's registration requirements. The registration procedure should also stop the same voter identifier from being registered more than once.

After registration is completed, the system retains the information necessary for authentication and eligibility checking. The registration module and blockchain module have separate responsibilities: voter information is managed at the application level by the local server, while the smart contract maintains the voting state used to prevent repeated voting.

D. Face Detection and Preprocessing

Biometric verification begins with face detection. When a voter submits or captures a facial image through the application, the system processes the image to locate the face region. OpenCV supports image handling and preprocessing, whereas RetinaFace performs face detection [2], [7].

The identified face region is then prepared for feature extraction. By isolating the facial area and preparing it for additional analysis, preprocessing produces suitable input for the recognition model. Recognition reliability can be affected by input-image quality. Detection and matching may be influenced by poor lighting, blurred images, unsuitable camera angles, or partial facial obstruction.

The purpose of this module is to locate the facial area that will later be examined by the recognition model. Face detection and face recognition are different processes: RetinaFace locates the face within an image, while FaceNet512 transforms the detected face into a numerical representation for comparison with registered facial data [1], [2].

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

E. Facial Feature Extraction Using FaceNet512

After the face has been detected, FaceNet512 is used to obtain facial features. The model converts the detected face into a 512-dimensional numerical embedding. Instead of comparing only original image pixels, the system represents facial characteristics through this embedding, which is appropriate for similarity comparison [1].

During the registration stage, a facial embedding is produced for the voter and retained as the reference representation. During authentication, another embedding is generated from the newly captured facial image. The current representation is subsequently compared with the registered representation to determine their similarity.

Facial embeddings allow registered and newly captured images to be compared consistently. However, the resulting accuracy depends on the model, image quality, preprocessing method, and matching threshold. Consequently, representative test images should be used to assess the recognition module before it is deployed in a real election setting [1].

F. Facial Matching Using Euclidean Distance and FAISS

The facial matching component determines whether a newly captured face belongs to a registered voter. It compares the embedding produced during authentication with the reference embedding saved at registration. Euclidean distance measures the difference between these numerical representations [1], [3].

For two facial embeddings, (A) and (B), each containing 512 numerical components, Euclidean distance is calculated as:

$$d(A,B) = \sqrt{\sum_{i=1}^{512}(A_i - B_i)^2}$$

Here, (A_i) and (B_i) represent the (i)-th components of the two embeddings. A lower distance value shows that the embeddings are closer in feature space, while a higher value shows a greater difference. A predefined matching threshold is then used by the system to decide whether the embeddings represent a match.

FAISS supports similarity searching across stored facial representations. This can assist in retrieving nearby matches when the number of registered voters increases. The final determination must also account for the configured threshold and the voter identifier or registration record being checked. A successful biometric result does not by itself confirm voting eligibility; registration status and prior voting status must also be verified [3].

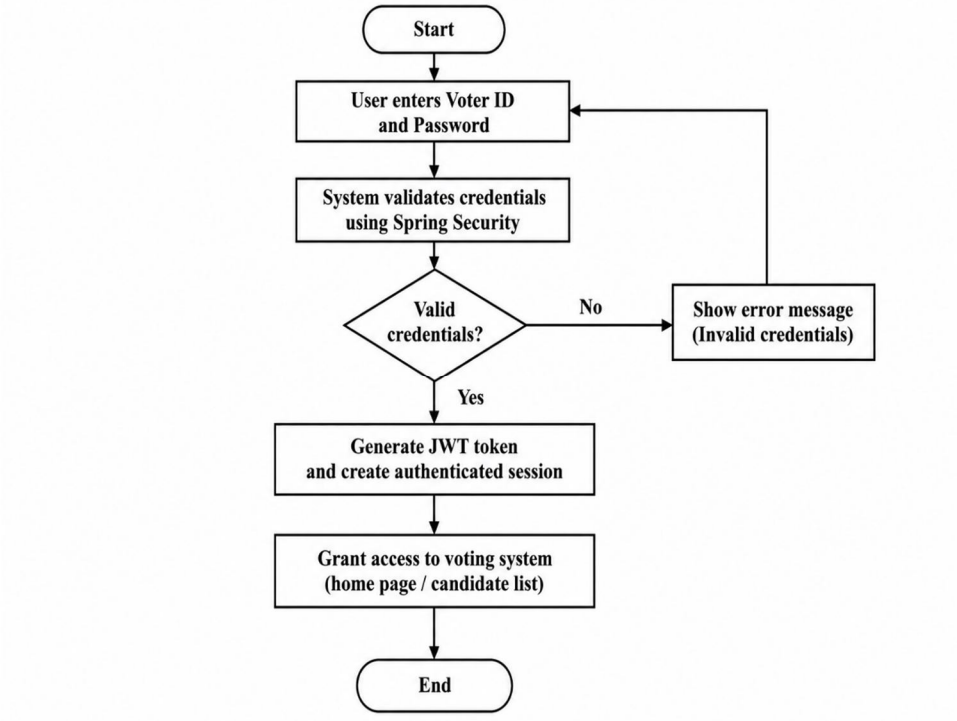


Fig. 2. Workflow of Authentication and Access Control

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

G. Blockchain Integration and Smart Contract

The blockchain module records voting transactions and maintains the voting state required by the application. Ethereum functions as the blockchain platform, while voting rules are implemented in Solidity smart contracts. The smart contract specifies the actions allowed during voting and stores the state required to determine whether a voter has already participated [5], [6].

Ganache provides a local Ethereum network for development and testing. Developers can deploy smart contracts, submit test transactions, and inspect voting operations without relying on a live election network. Web3j connects the Java backend to the Ethereum blockchain and enables the backend to communicate with the deployed smart contract [5].

When an authenticated and eligible voter submits a vote, the backend evaluates the relevant conditions before calling the required smart-contract operation. The contract checks the voting state and rejects attempts that violate configured rules, including another vote from an identifier already recorded as having voted. The blockchain network then processes the transaction and records it according to the smart contract's logic.

The proposed workflow applies SHA-256 hashing to vote-related information before that information is submitted to the blockchain. A cryptographic hash can indicate whether the data represented by the hash has been modified. Nevertheless, hashing alone cannot ensure ballot secrecy, establish voter eligibility, or secure the entire voting system. These properties depend on the overall system design, the data sent to the contract, and the correct implementation of authentication and voting rules [4].

H. Duplicate Voting Prevention

Preventing duplicate voting is a central requirement of the proposed system. Before accepting another vote, the voting module checks whether the authenticated voter has already participated in the election. The local server can perform application-level checks, while the smart contract retains the voting state required for blockchain-level enforcement [5], [6].

When a voter tries to cast a ballot, the backend checks the voter identifier and the outcome of all required authentication stages. The voting request is then compared against the existing voting state. If the voter has already been marked as having voted, the application rejects the request. If not, the smart-contract operation proceeds and updates the relevant state according to the contract rules.

The final duplicate-voting check should remain in the smart contract because a check performed only by the backend may be insufficient when requests are repeated or sent through alternative application paths. To avoid multiple accepted votes from concurrent requests, the smart contract should apply the one-vote-per-eligible-voter rule atomically. The success of this protection depends on smart-contract correctness and adequate testing [5], [6].

I. Vote Casting and Transaction Recording

After registration verification,facial matching, and eligibility checking have been completed successfully, the voter may access the candidate-selection interface. The voter chooses a candidate and submits the vote through the application. The frontend sends this request to the Java backend, which validates it and coordinates the required blockchain communication.

Through Web3j, the backend calls the voting smart contract. The contract verifies whether the voting action is allowed and updates the relevant voting state. The transaction is submitted to the configured Ethereum network for processing and recording. Ganache is used to test this transaction process during development [5], [6].

Before blockchain submission, the proposed workflow applies SHA-256 hashing to vote-related data. The implementation must clearly define the information included in the hash and the relationship between the hash and the stored vote. To maintain ballot secrecy, personally identifiable voter information should not appear in public blockchain transaction data, and the design should avoid directly associating a voter's identity with a selected candidate [4].

After the transaction is processed successfully, the application may present a confirmation to the voter. This confirmation should be displayed only after the backend has checked the transaction result, rather than immediately after the voter selects the submit button.

J. Complete System Workflow

The methodology follows a sequence involving registration, authentication, eligibility validation, vote submission, and transaction confirmation.

Voter registration: The voter submits the required details, voter identifier, and facial image.

Face detection: OpenCV and RetinaFace process the image and identify the facial region [2], [7].

Feature extraction: FaceNet512 produces a 512-dimensional facial embedding [1].

Reference comparison: Euclidean distance and FAISS support similarity comparison with registered facial representations [3].

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Eligibility validation: The backend checks registration status and applicable voting conditions.
Duplicate-vote check: The system reviews the voter's voting state and rejects a record that has already been used.
Candidate selection: An authenticated and eligible voter chooses a candidate and submits the vote.
Hash generation: SHA-256 is applied to vote-related data according to the implementation design [4].
Smart-contract execution: The Java backend communicates with the Solidity smart contract through Web3j [5], [6].
Blockchain recording: The transaction is processed on the Ethereum network, with Ganache used for local development and testing.

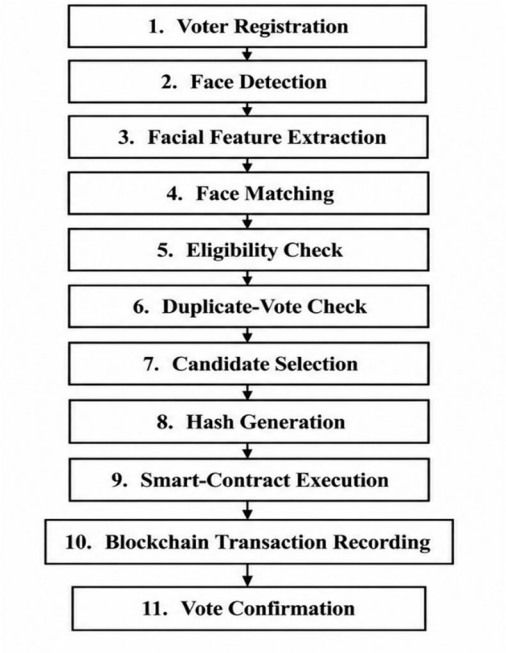


Fig. 3. Methodology Flow of the Proposed Voting System.

K. Security and Reliability Considerations

The architecture combines several mechanisms to improve the security and reliability of voting operations. Biometric verification helps determine whether the person accessing the application resembles the registered voter .verification provides an additional authentication stage, while backend access controls limit access to protected functions. Smart-contract rules offer another mechanism for enforcing voting requirements, including rules that prevent repeated voting.

The local server and blockchain have separate functions. Application-level information is managed by the local server, while the blockchain records transactions and voting-state changes defined by the deployed smart contract. This division enables the application to manage operational information without using the blockchain as a general-purpose repository for personal data.

Facial data, voter identifiers, and vote-related records must all be protected. Access to sensitive information should be limited, and unnecessary personal data should not be written to a public blockchain. The implementation should be evaluated against failed authentication attempts, duplicate submissions, invalid transactions, network interruptions, and unsuccessful smart-contract calls.

The proposed architecture establishes a foundation for an AI-assisted and blockchain-supported voting application. However, architecture alone cannot demonstrate that the system is suitable for real elections. Further assessment is necessary to evaluate biometric accuracy, performance under load, smart-contract correctness, privacy, accessibility, ballot secrecy, and resistance to coercion and other election-specific threats.

IV. IMPLEMENTATION AND RESULTS

A. Implementation Environment

The proposed voting system is delivered through a web-based platform called VoteSecure AI – Digital Voting Portal. The interface includes navigation for voter entry, dashboard details, candidate information, and election-result viewing. The project architecture specifies HTML5, CSS3, and JavaScript for the frontend; Java with Spring Boot for backend operations; and a local server to maintain voter and application data. Planned biometric technologies include OpenCV, RetinaFace, FaceNet512, and FAISS, whereas Ethereum, Solidity, Ganache, and Web3j are identified for blockchain-related functions.

©IJRASET: All Rights are Reserved

To support a clear voter journey, the implementation separates the application into distinct functional screens. The home page introduces the portal, the login page collects user credentials, the dashboard shows voter-related information, the candidates page displays candidate profiles, and the results page presents election figures and candidate rankings.

The screenshots included in this paper illustrate the application interface and the visible system states. However, they do not independently confirm that all backend services, biometric methods or blockchain transactions have been integrated and tested successfully. Therefore, the results discussed below are limited to the observable features of the prototype.

B. Home Page and Voter Access Interface

The VoteSecure AI home page acts as the starting point for users entering the system. It describes the application as a secure digital voting portal and offers navigation options including Home, How It Works, Security, Candidates, Results, and Help. Voter login and start-voting controls are also available on the page.

An election-information area displays the title General Election 2026 and presents the election as active. Users are also provided with an option to view candidate information before beginning the voting process. The page identifies identity verification, one-person-one-vote enforcement, and tamper-resistant records as intended system features.

By placing the main options within a single screen, the home page is designed to provide voters with a straightforward entry into the application. Its layout offers an initial path toward authentication and other voting-related activities.

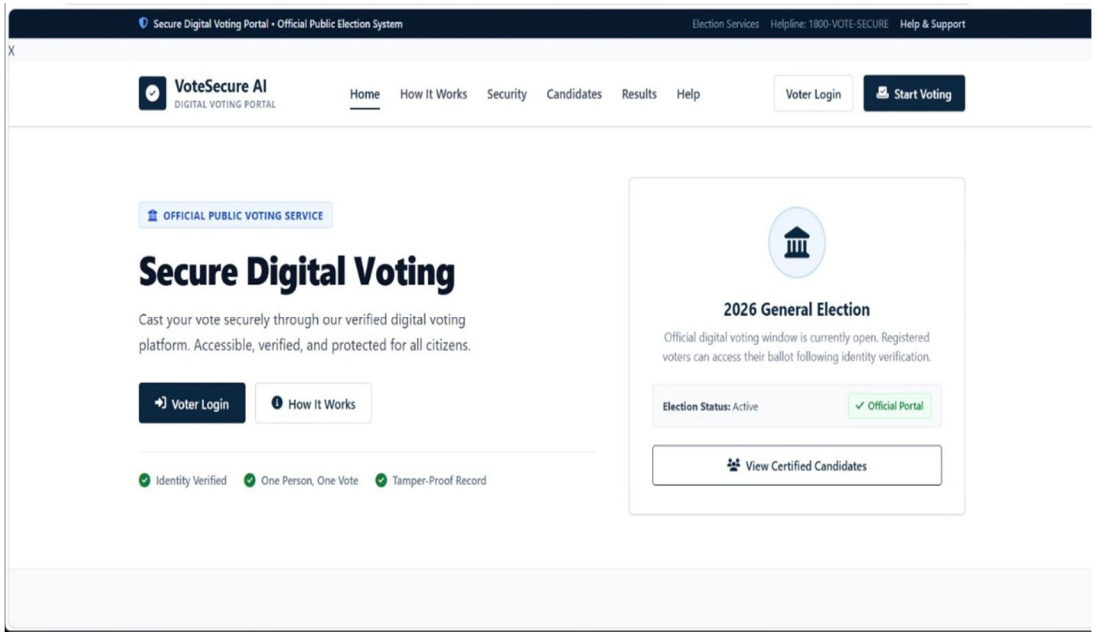


Fig. 4. Home Page of the Vote Secure AI Digital Voting Portal

C. Voter Login Interface

The voter login page functions as the first authentication screen. It provides input fields for a voter ID or registered email address and password. The page also includes a session option labelled “Remember this device for this session,” along with a Login button.

New users can access registration through a navigation option, while a demo-account autofill control is also provided. Together, these components show the basic structure of the login interface and its intended purpose of collecting credentials before users are allowed to enter the voting portal.

Under the intended system design, submitted credentials are checked by the backend before protected sections can be accessed. Spring Security and JWT are specified as the authentication and authorization technologies. Nevertheless, the supplied login screenshot only presents the credential-entry interface; it does not show completed successful facial matching.

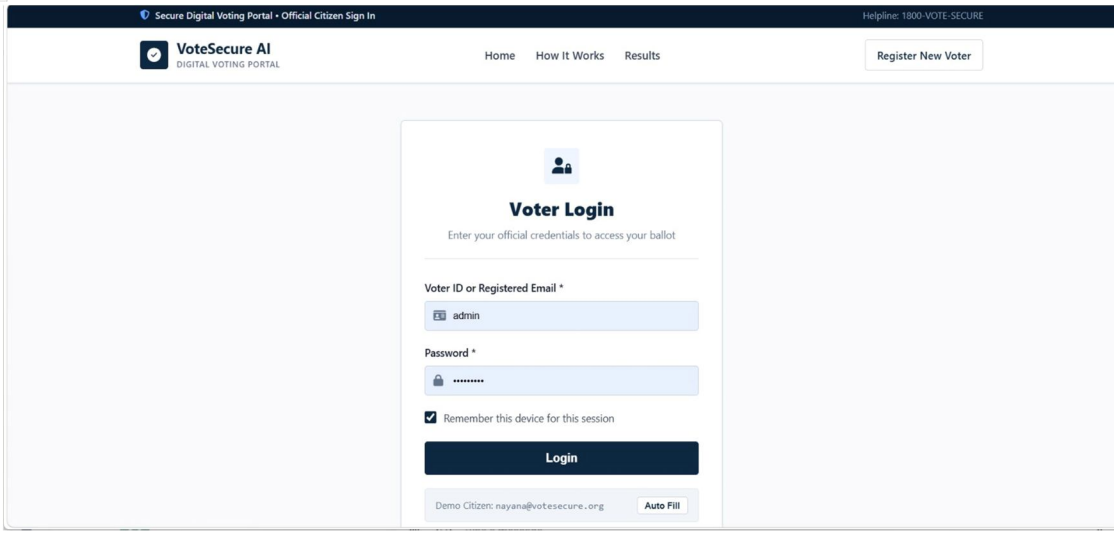


Fig. 5. Voter Login Interface

D. Voter Dashboard and Electoral Status

The voter dashboard provides a personalized interface that displays the voter name and government voter identifier. It contains a "View Candidates" control and a section named "Voter Status & Electoral Clearance."

Four status indicators appear in the panel: Registered, Identity Verified, Eligible to Vote, and Voting Status. These indicators are intended to summarize the voter's registration status, identity-verification progress, eligibility, and voting activity.

The dashboard further displays the statement "Your Vote Has Been Cast" and includes a "View Blockchain Receipt" link. This design is intended to inform the voter about vote completion and provide access to a transaction-related receipt.

The screenshot also presents an "Identity Verified" indicator marked "Pending Verification," although the remaining indicators show positive states. As a result, the dashboard should be regarded as a visible application state rather than proof that all verification procedures were completed. Additional testing is needed to confirm that these status indicators match the actual authentication and transaction records.

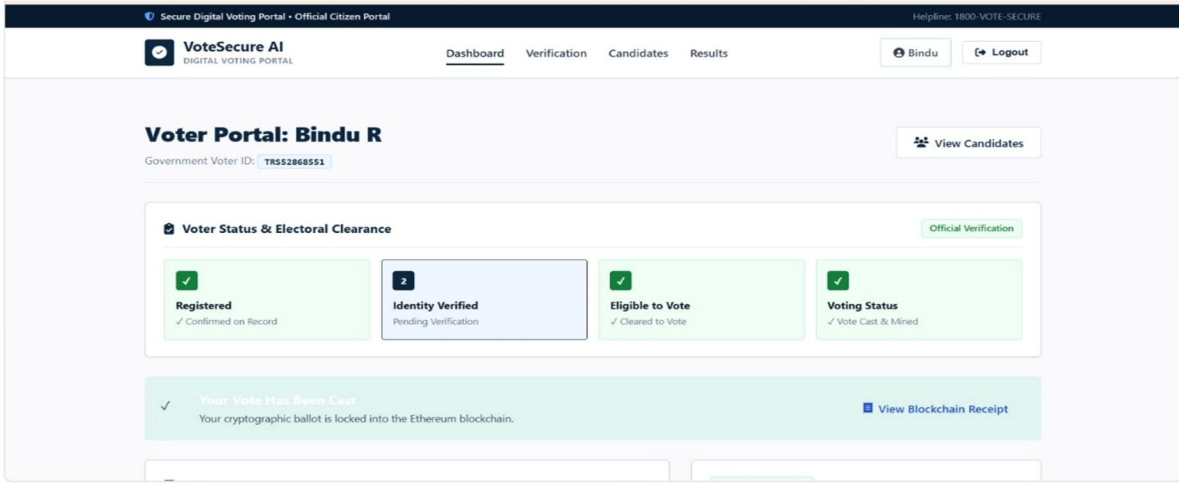


Fig. 6. Voter Dashboard and Electoral Clearance Status

E. Certified Candidate Listing

The candidate page displays profiles under the heading "Certified Candidates." Each candidate card contains a name, candidate identifier, symbol, group or party label, brief description, and associated activity areas. The examples visible on the page include M S Nishamya, Thanushree H R, and Bindu R.

A search field and party-selection dropdown are included to assist users in finding candidates and filtering the available profiles. The card-based format organizes candidate information so that voters can examine profiles before continuing to the voting stage.

The screenshot shows the phrase "Vote Already Cast" on the candidate cards. This suggests that the displayed account is presented as having completed voting. However, the image alone does not demonstrate whether this status is based on a verified blockchain transaction or on application-level data.

Before final evaluation, candidate identifiers and group labels should be verified for consistency across the candidate listing, voting records, and results page.

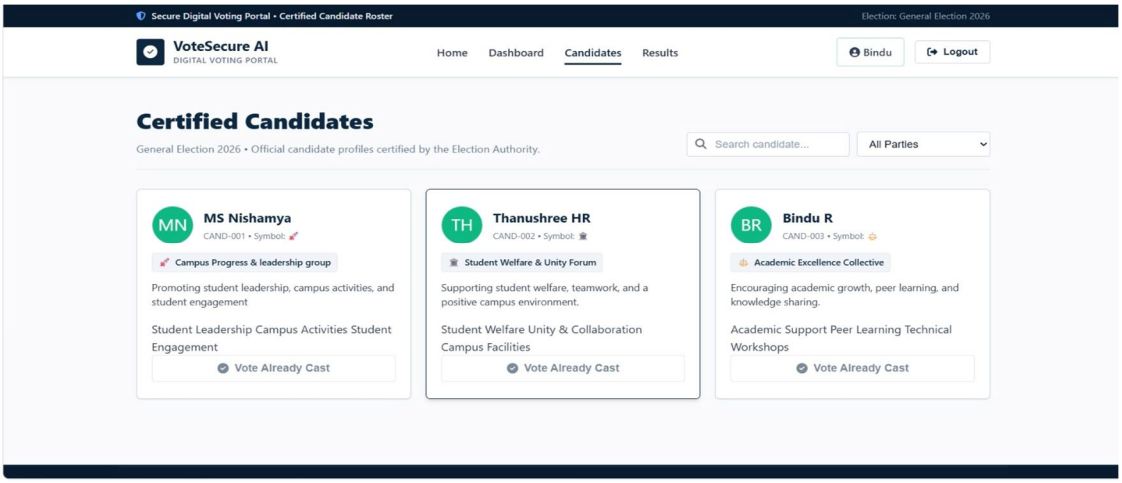


Fig. 7. Certified Candidate Listing Interface

F. Election Results Dashboard

The election results page summarizes the visible election tally. It identifies the election as "General Election 2026" and describes the tally as an official certified tally. An automatic-update option and a Refresh Data control are available on the page.

Four summary values are displayed: 10,852 total votes cast, 72.3% voter turnout, 100% verified ballots, and a Certified electoral status. These figures are reported as values shown in the supplied screenshot and have not been independently checked against voter records or blockchain transactions. The Candidate Vote Standings section presents candidate positions, vote totals, and vote percentages. In the visible part of the screenshot, Thanushree H R appears in first position with 4,211 votes and 38.8%, while Bindu R appears in second position with 3,891 votes and 35.9%. Progress bars visually compare the displayed candidate standings. The refresh and automatic-update features are intended to present updated election information. However, the correctness of the displayed tally and its connection to the authoritative vote record can only be confirmed by comparing it with the underlying application data and, where applicable, the verified smart-contract state.

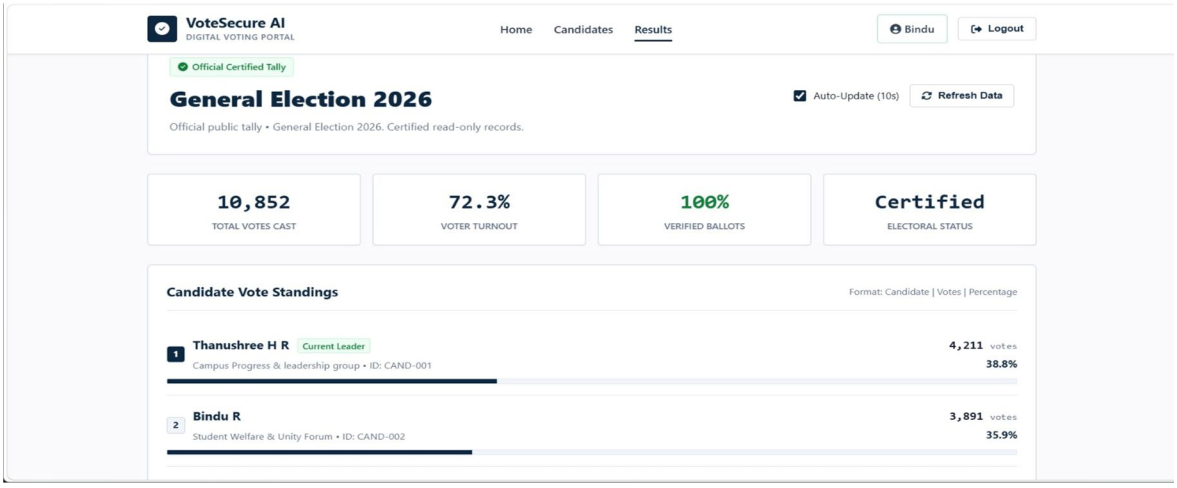


Fig. 8. Election Results Dashboard and Candidate Vote Standings

G. Observed Results and Functional Interface Summary

The supplied screenshots show the main user-facing screens of the Vote Secure AI prototype. The home page provides entry to portal functions, the login page collects voter credentials, the dashboard presents voter-related status information, the candidate page displays profiles, and the results page presents election statistics and candidate standings.
The table below summarizes the visible functions documented through the screenshots.

Table I. Summary of Observed Interface Outputs

Module	Observed Output	Evidence
Home Page	Voting portal introduction, navigation links, election information, and access controls	Fig. 4
Voter Login	Voter ID/email and password fields, session option, and login control	Fig. 5
Voter Dashboard	Voter identifier, electoral status indicators, and vote-status message	Fig. 6
Candidate Listing	Candidate profiles, identifiers, descriptions, search, and party filter	Fig. 7
Election Results	Displayed vote totals, turnout, verification indicator, and candidate standings	Fig. 8

H. Implementation Components

The proposed Vote Secure AI system combines web-based technologies, backend services, biometric recognition methods, and blockchain elements. Each element has a defined function in voter registration, authentication, candidate selection, and voting operations. Through its modular structure, the system separates the user interface from application processing, biometric validation, and blockchain communication. HTML5, CSS3, and JavaScript are used to create the frontend interface through which users access the voting portal. Application logic and communication between the frontend and other system modules are managed by the Java and Spring Boot backend. The project design incorporates Spring Security and JSON Web Tokens (JWT) to provide authentication and access control. Voter and election-related information is maintained through a local server.
For biometric verification, the proposed module uses Python, OpenCV, and Retina Face to process and detect facial images. FaceNet512 produces facial embeddings, whereas Euclidean distance and FAISS are used to compare and search facial similarities. Together, these elements are intended to determine whether a voter corresponds with the registered facial information.
Ethereum and Solidity smart contracts form the blockchain module, which manages vote-related transactions and applies the configured voting rules. Ganache offers a local blockchain setting for development and testing, and Web3j enables communication between the Java backend and the Ethereum network. The proposed workflow also includes SHA-256 hashing to support the integrity of vote-related data. These implementation components collectively support an organized voting process. The frontend receives user input, the backend manages authentication and application processing, the biometric module assists with facial verification, and the blockchain layer performs voting operations specified through the smart contract. The overall system's performance depends on the proper integration and testing of all these components.

Table II. Implementation Components of Vote Secure AI

Component	**Technology**	**Role in the System**
Frontend interface	HTML5, CSS3, JavaScript	Offers the user interface for voter access, candidate review, and election-result viewing
Backend server	Java, Spring Boot	Manages application logic and coordinates communication among system modules
Data management	Local Server	Maintains and organizes voter and election-related application information
Face detection	OpenCV, Retina Face	Enables facial image processing and face detection
Facial feature extraction	FaceNet512	Produces 512-dimensional facial embeddings for comparison
Facial matching	Euclidean Distance, FAISS	Enables facial similarity comparison and searches across

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

		facial embeddings
Face recognition service	Python	Processes requests related to facial verification
Blockchain module	Ethereum, Solidity	Manages smart-contract-based voting rules and transaction recording
Local blockchain network	Ganache	Offers an Ethereum-compatible setting for development and testing
Blockchain integration	Web3j	Links the Java backend to the Ethereum blockchain
Vote-data hashing	SHA-256	Creates a cryptographic hash for vote-related data

I. Result Interpretation and Verification Requirements

The available visual material supports a qualitative discussion of the prototype interface. Candidate cards, voter-status indicators, and result summaries demonstrate how the system presents information to users. However, evaluating the full voting system requires evidence beyond the appearance of these screens.

Authentication testing should first confirm that valid credentials are accepted and invalid credentials are rejected. Where facial verification are implemented, separate tests should document both successful and unsuccessful verification paths.

Duplicate-voting testing should then verify that a voter cannot submit another accepted vote after voting once. Confirmation should come from the authoritative voting state rather than only from a disabled control or a “Vote Already Cast” message.

Blockchain integration should be evaluated by recording the transaction identifier, transaction status, and corresponding smart-contract state for a test vote. The totals displayed by the application should subsequently be compared with the authoritative tally source.

Candidate identifiers, names, and voting statuses should also remain consistent across the candidate listing, dashboard, and results page. Differences appear in the candidate identifiers and group labels across the supplied candidate and result screens; therefore, these records must be reconciled before the tally can be considered validated.

Accordingly, the findings presented in this section are restricted to observable interface outputs and displayed values. Claims concerning security, biometrics, and system performance should be included only after the required tests have been performed and documented.

V. SECURITY ANALYSIS AND PRIVACY CONSIDERATIONS

Vote Secure AI is proposed as an electronic voting system that combines biometric verification, authentication measures, voter-eligibility validation, and blockchain-based transaction recording to strengthen election security. Because the system processes voter details and voting-related records, security and privacy protections must be addressed during registration, authentication, vote casting, and result display .Several components are incorporated to address different security needs. Facial recognition is intended to confirm voter identity ,backend access controls limit sensitive operations, and smart-contract logic is designed to stop repeat voting. The proposed workflow also includes SHA-256 hashing for checking the integrity of vote-related data. Nevertheless, the protection offered by these mechanisms depends on proper implementation and testing, as no individual component can ensure complete election security.

A. Voter Authentication and Access Control

Authentication is a key security requirement because voting functions should be available only to eligible and authorized voters. The proposed system combines credential-based access with facial recognition. Through the login interface, voters provide their voter ID or registered email address together with a password, while the biometric module compares the captured facial representation against the registered reference.

Spring Security and JWT are included to manage authentication and access control. When configured properly, these technologies can limit protected application functions to authorized users Facial matching is intended to lower the risk of unauthorized access by individuals using another person’s credentials.

Failed facial matches should result in denied access. Voting access should be provided only when all required verification conditions have been met. Authentication logs and session-related information must also be safeguarded from unauthorized access.

B. Data Privacy and Protection

During registration and authentication, the system handles voter identifiers, account details, and facial information. These data require careful management to minimize the possibility of unauthorized disclosure. Voter records should be accessible only through authorized application functions, while sensitive information should not be unnecessarily revealed through interfaces, logs, or blockchain transactions.

The local server is intended to manage voter and election-related application information, whereas the blockchain layer processes transactions and voting states established by the smart contract. This separation can reduce the need to store all personal information on the blockchain. Facial data should also be secured through suitable storage practices and access restrictions.

SHA-256 hashing does not encrypt information or automatically guarantee ballot secrecy. For this reason, voter identities and directly identifiable vote choices should not be placed on a public blockchain. The connection among voter records, transaction details, and candidate selections must be carefully structured to preserve voting privacy.

C. Duplicate-Voting Prevention and Data Integrity

Preventing duplicate voting is necessary to preserve the reliability of the voting process. Before a new vote is accepted, the proposed system checks the voter's existing voting status. Smart-contract logic is intended to maintain the required voting state and reject attempts to vote more than once.

The final validation against duplicate voting should be carried out by a trusted backend or the smart contract, rather than depending solely on the displayed state of a page or button. The system must also correctly manage repeated and simultaneous requests so that concurrent submissions cannot result in multiple accepted votes from one voter.

The proposed workflow uses SHA-256 hashing to produce a cryptographic hash of vote-related information. When comparisons are performed correctly, this hash can help reveal changes made to the represented data. However, hashing by itself cannot prevent all forms of tampering, verify voter eligibility, or demonstrate that a vote was counted accurately. These outcomes depend on suitable application logic, secure transaction processing, and validation of the recorded voting state.

Table III. Security and Privacy Considerations of VoteSecure AI

Security Aspect	Mechanism in the Proposed System	Purpose and Consideration
Voter authentication	Credentials, and facial verification	Helps confirm voter identity before system access is provided
Facial verification	Retina Face, FaceNet512, Euclidean distance, and FAISS	Supports face detection and comparison with registered facial representations
Voter data protection	Local server and controlled access	Manages voter information while reducing unnecessary exposure
Duplicate-vote prevention	Voting-state checks and smart-contract logic	Designed to reject attempts at repeated voting
Data integrity	SHA-256 hashing	Creates a hash of vote-related information for integrity verification
Transaction recording	Ethereum and Solidity smart contracts	Stores transactions and applies the voting rules defined in the contract
Ballot privacy	Careful handling of voter and vote-related information	Prevents unnecessary exposure of personal identifiers and candidate selections

D. Security Limitations

Although the proposed architecture includes several security mechanisms, it also has limitations that must be considered. Facial recognition performance can be influenced by lighting conditions, image quality, camera angle, and the threshold selected for determining a match. Blockchain records may offer a tamper-evident history of transactions, but blockchain technology alone cannot ensure voter eligibility, ballot secrecy, protection from coercion, or the overall correctness of an election. Errors in smart contracts, insecure backend processes, compromised user devices, and weaknesses in voter registration can still reduce system reliability.

The provided interface screenshots show the visible portal screens and displayed status details, but they do not independently confirm biometric accuracy, smart-contract correctness, or end-to-end security. Before the system can be regarded as appropriate for real election use, separate functional, security, and privacy testing is required.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VI. CONCLUSION AND FUTURE SCOPE

A. Conclusion

VoteSecure AI – AI-Driven Voter Access and Authentication in a Blockchain-Based Secure Voting System is proposed to strengthen the security, accessibility, and transparency of electronic voting through the combined use of artificial intelligence, biometric authentication, and blockchain technology. The system is intended to address key concerns in electronic voting, such as unauthorized access, voter identity validation, duplicate voting, and the preservation of voting-record integrity.

The proposed design connects a web frontend built with HTML5, CSS3, and JavaScript to a Java Spring Boot backend. Its biometric verification process uses RetinaFace for face detection and FaceNet512 to create 512-dimensional facial embeddings. Euclidean distance and FAISS are used for facial comparison and similarity searches.

Ethereum and Solidity smart contracts form the blockchain portion of the system, supporting the enforcement of voting rules and the recording of transactions. Ganache is used as a local blockchain setting for development and testing, whereas Web3j enables interaction between the Java backend and the blockchain. The proposed workflow also includes SHA-256 hashing to check the integrity of vote-related information.

The provided interface screenshots show the structure of the voting portal, including the home page, voter login, voter dashboard, candidate list, and election-results dashboard. These screens indicate how voter details and voting-related outcomes are displayed within the application.

Overall, the project presents a modular method for combining AI-supported voter verification with blockchain-based voting processes. To determine whether the system is reliable, privacy-preserving, and appropriate for real-world elections, further implementation validation, security evaluation, and controlled testing are needed.

B. Future Scope

The proposed system can be improved in several areas related to functionality, security, scalability, and usability.

1) Improved Biometric Verification: Future work may assess facial recognition performance across varying lighting conditions, camera positions, and image-quality levels. Testing with representative datasets and calibrating the matching threshold can support measurement of false acceptance and false rejection rates. Presentation-attack detection could also be added to reduce the possibility of identity spoofing.
2) Stronger Authentication Security: The authentication process could be strengthened through limits on verification attempts, secure session handling, and safeguards against repeated authentication requests. Monitoring features may also help detect suspicious login activity while protecting voter credentials.
3) Enhanced Blockchain Verification: Later development can examine the smart contract for correctness, prevention of duplicate votes, transaction failures, and simultaneous submissions. The application may also generate verifiable transaction receipts and compare the displayed election outcomes with the authoritative voting state.
4) Improved Ballot Privacy: Future development should consider approaches that separate voter identity from candidate selection. Sensitive voter details should not be stored unnecessarily on a public blockchain. Privacy-preserving voting methods and independent security assessments can improve confidentiality and reduce the risk of connecting voters with their selected candidates.
5) Scalability and Performance Evaluation: The system may be evaluated with progressively larger numbers of registered voters and voting requests. Tracking response time, authentication latency, transaction-confirmation time, and resource use can reveal performance constraints and support optimization.
6) Multilingual and Accessible Interface: The frontend could be expanded to include multiple Indian languages and accessibility features. Usability may improve through clear instructions, responsive page layouts, keyboard navigation, and support for users with different levels of digital literacy.
7) Comprehensive Testing and Deployment: Future work should involve structured functional testing, biometric assessment, smart-contract auditing, privacy evaluation, and usability testing. Moving beyond the local Ganache development environment would require careful assessment of the selected blockchain network, transaction costs, operational controls, and election-specific requirements.

These developments can help transform the prototype into a more thoroughly assessed voting application. Before any real-election deployment, the system would need independent technical, legal, privacy, accessibility, and election-security evaluations.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

REFERENCES

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823, 2015. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298682

[2] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5203–5212, 2020. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00525

[3] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The Faiss Library," arXiv preprint arXiv:2401.08281, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2401.08281

[4] National Institute of Standards and Technology, "Secure Hash Standard (SHS)," Federal Information Processing Standards Publication 180-4, 2015. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.180-4

[5] V. Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform," Ethereum Whitepaper, 2014. [Online]. Available: https://ethereum.org/whitepaper/

[6] Ethereum Foundation, "Solidity Documentation." [Online]. Available: https://docs.soliditylang.org/

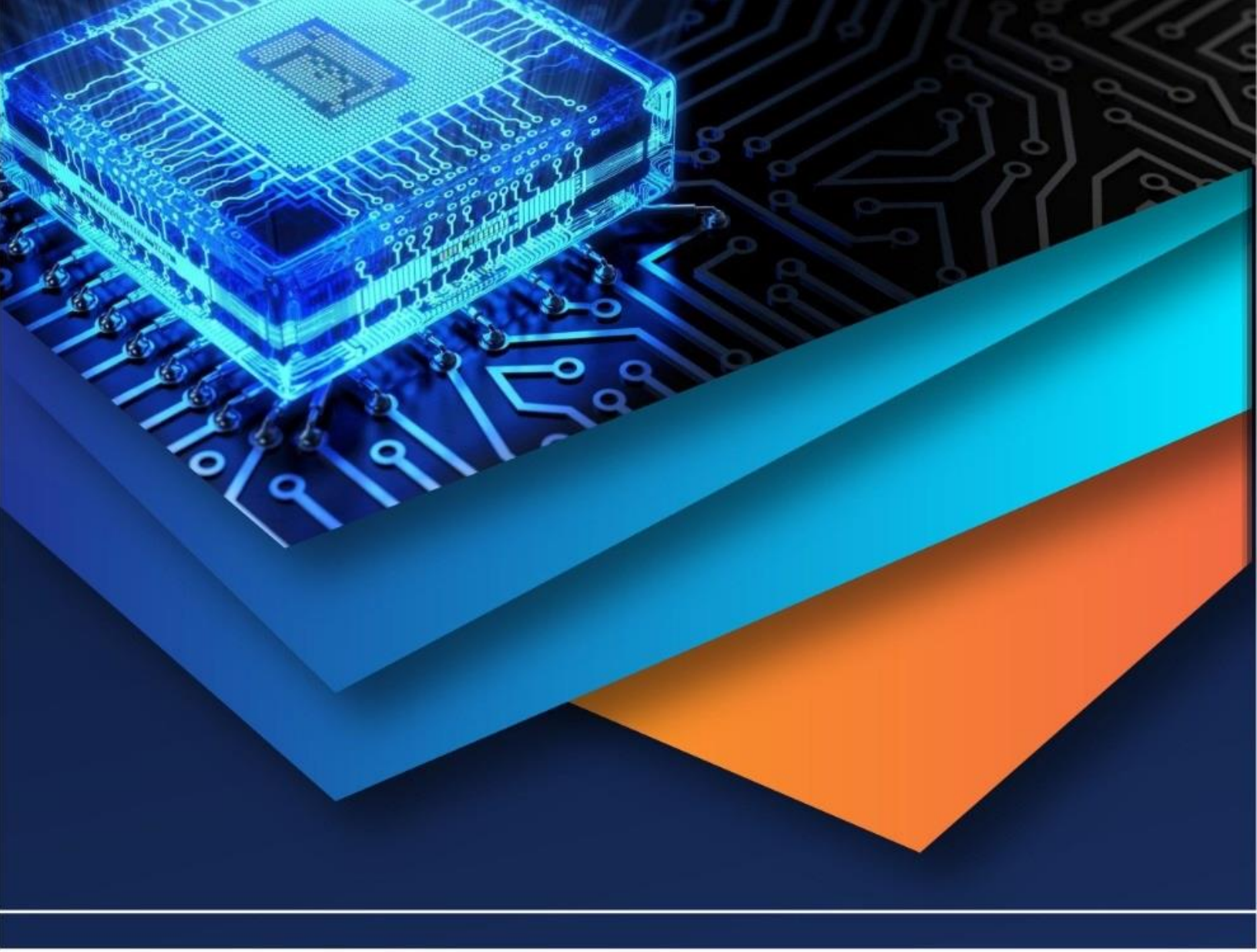
[7] OpenCV, "OpenCV Documentation." [Online]. Available: https://docs.opencv.org/

[8] Spring, "Spring Security Reference Documentation." [Online]. Available: https://docs.spring.io/spring-security/reference/

[9] National Institute of Standards and Technology, "Digital Identity Guidelines: Authentication and Lifecycle Management," NIST Special Publication 800-63B. [Online]. Available: https://pages.nist.gov/800-63-3/sp800-63b.html

[10] H. Baniata and G. Caluna, "BP-Vot: Blockchain-Based e-Voting Using Smart Contracts, Differential Privacy, and Self-Sovereign Identities," IEEE Access, vol. 13, pp. 46106–46123, 2025. [Online]. Available: https://doi.org/10.1109/ACCESS.2025.3548404

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)