International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

SecureVote: Design and Development of an Online Voting Portal Incorporating Blockchain-Based Verification

Ajay Kumar Uthaya Kumar[1], Dr. Manoj Singh[2]

[1](Computer Science, SIES College of Arts, Science and Commerce, Sion (West)

Email: u.ajaykumar7616@gmail.com)

[2](Head of Computer Science Department, SIES College of Arts, Science and Commerce, Sion (West)

Email: manojks@sies.edu.in)

----------**********----------

Abstract:

This research presents the design and development of an online voting system named, 'SecureVote' incorporating blockchain-based verification. The proposed system ensures a robust security framework through features such as location-based voter classification, secure blockchain transaction storage and verification, constituency mapping, virtual voting, real-time monitoring, nine layers of security and role-based access control for system authorities. The system aims to be a reliable alternative to the traditional voting methods, particularly in places where physical voting is not possible.

Keywords — **Online Voting System, Blockchain, Web Development, i-voting, e-voting, internet voting**

----------**********----------

I. INTRODUCTION

In today's modern times, democracy and people's rights have become the core pillars of nations. Every nation's people decide which party and leader will serve them as the country's head. If a government fails or misleads the country, people have the right to remove the government. Elections are the mechanism that gives people their rights.

Each country has its own voting system: a ballot system, a paper voting system and an internet voting system. The country which adopted an internet voting system is Estonia in 2009. Estonia uses Keyless Signature Infrastructure(KSI) blockchain, a private blockchain [9]. Other countries, such as Sierra Leone, tested blockchain voting in 2018, the United States in 2018 accepted legally binding blockchain voting for overseas and military personnel in a federal midterm election, and the Thai Democratic Party in 2018 adapted using blockchain for a large-scale primary election, with over 120,000 votes cast, and many more [3], [8].

With a growing technological era and different nations staying aboard due to work and other reasons, countries have started adopting online voting systems. In Estonia's 2005 elections, only 2% of voters used i-voting, but after years of adoption, this grew to over 46% by 2019 [9]. During government instability in countries such as Nepal, Sri Lanka, and Bangladesh, the online voting system will provide strong support to run elections. In the future, the online voting system will become a major voting system and blockchain will pioneer its role in it due to its hash transactions [1], [2].

The main purpose of this research was to design and develop an online voting system with features such as secure transaction storage, voter authentication, constituency mapping, real-time monitoring, and multi-layered security procedures. The system helps to improve election security, voting transparency and accessibility. The project

offers a dependable and scalable alternative to the conventional voting system [4], [7].

By incorporating blockchain technology with modern web-based systems, it is particularly useful where physical voting is not possible. By addressing important issues with current electronic voting and providing a framework, this system provides trust, security and usability [5], [6], [8].

II. Current Voting system and Online Voting Portal Issues

A. Preplanning before election

The election needs to be pre-planned from start to end. Like ward, constituency and division of ward. Also, a voter list needs to be taken, and upon this, all slight state issues like riots are raised. Then the cancellation of an election is also a high risk. So, the cost is the total loss of the election commission [3], [8].

B. Dedicated Portal

A key issue with online voting portals is the lack of a dedicated portal which supports end-to-end creation of elections, adding candidates, adding voter lists according to address. People need to be able to vote and view the results as well. With these adequate issues, most countries must face alternative ways to conduct elections [5], [6].

C. Vulnerability to hacking

The main and prominent issue of digital voting is the vulnerability and hacking of voting. One flaw in the system could expose and risk failure of the total voting system [1], [2].

D. Data tampering

Unlike paper voting and ballot voting, online voting has a risk of data tampering. Small access to databases could lead to risk of vote data tampering [4], [7].

E. Tracking & surveillance risks

With online voting, the risk of tracking and surveillance of voters is quite high. Candidates may influence or pressure voters both offline and online before voting. Only during voting, it is up to the voter to decide. Physically, it happens before election authorities, while online is a questionable thing [2], [8].

F. Insider threats

The threats outside the election commission are equivalent to those inside the election commission. Corruption and bribing are quite rooted in society. So, monitoring inside is also quite inefficient [3], [7].

G. Inadequate testing

The risk of failing in real conditions is quite evident. Estonia, which adapted to online voting, had too many risks and the acceptance of voters has been quite low in 2009 [9]. But with time and technology it has also changed.

III. FEATURES

A. Ward Mapping

SecureVote incorporated a map feature for mapping the ward. The mapping will be useful in segregation of voters. This mapping will work for bigger areas, due to constraints of current technology. It will not work for smaller areas, like buildings, areas in a constituency, etc. Also, it will be used for bigger areas.

B. Voter Segregation

If you have an Excel sheet of voters which has a mix of different locations. Then, in ward making, they opt for no separate voter list. When the Excel file is uploaded, it is automatically segregated by voters who fall under the location. Also, a separate login ID will be sent to respected voters via email.

C. Hierarchy-wise distribution of access

The system has three stakeholders: Super admin, Admin and Voter. The voter will have a special login address which has been meant for one-time voting and strict access to voting. The administration has authority to manage elections, wards and candidates. While strictly restricted from features such as deleting elections and admin creation, Super admin has access to every part of the system, from admin creation to seeing election results.

D. Resolution of vote tampering

The main issue with voting is voting tampering. To resolve it, Blockchain came into as the main player. The voting is stored on both the database side and the blockchain side. In case any voting tampering issue is raised, there is a place to verify it. It shows you whether voting is tampered with or not. In this way, risk tampering is resolved.

E. Monitoring and surveillance of voters

During voter voting, a video movement and audio interference detection are added. If video and audio detection happens, a warning sign will pop up. two chances are given, even though detection still happens. The vote is automatically recorded as NOTA, so reducing the risk of voting manipulation during live voting.

F. Heavy layer security

The system has 9 distinct security layers across 4 zones:

- Zone 1 encompasses network security such as HTTPS / TLS, CORS policy, Rate limiting and CSP headers.
- Zone 2 is an encryption layer. Both requests and responses are encrypted and will be decrypted according to the required request structure and keys.
- Zone 3 is Authentication, where OTP login, role-based segregation and token encryption happen with a rate limit.
- The last zone is voting integrity, environmental background monitoring, blockchain records, and vote fraud detection. This zone is made especially for voters.

Also, if admins do something, it is also recorded in a database, which can be seen by database authorities, super admin and co-admin.

IV. METHODOLOGY

A. System Overview

SecureVote is a secure, role-based web application designed to implement digital elections for small-level scale like office, school, college and municipal level elections. It has three types of users: Super Admin, Admin and Users. Each user is defined with clear access levels and responsibilities. The system is built upon a React JavaScript framework for frontend interface, Node.js for backend support, PostgreSQL database extended with PostGIS for geospatial ward boundary management, and implemented with the Ethereum blockchain network to verify all cast votes are tamper-proof and independently verifiable. OTP-based authentication is implemented for each login to prevent unauthorized access.

B. System Initialization and Super Admin Setup

The system is initialized with a single Super Admin account created directly in the database at the time of deployment. This account holds the highest authority level and cannot be created through the portal itself. The Super Admin is responsible for the entire lifecycle of the election system. From creating admins, elections, and wards to monitoring results and verifying the votes. There is only one Super Admin in the system at any time, and this account cannot be deleted through the portal.

C. Authentication

The system follows a two-factor process for all user types. When a Super Admin or Admin sign in to their account, they enter their email and password, which are passed to the system. The system verifies the user data in the users table. If valid, the system creates an encrypted OTP with AES-256-GCM algorithm. The system inserts the encrypted one-time password into the otp_verifications table together with the expiration time while it dispatches the one-time password to the user's registered email address. The user needs to input the received one-time password which will then be used to authenticate their identity. The system marks the one-time password as validated after successful verification and produces an AES-256-GCM token which grants the user entry to their designated dashboard.

To reset the password, there is a separate flow. The user requests a password reset by submitting their new password and confirming the password on the change password page. The system generates a new encrypted AES-256-GCM one-time password and date is passed to password_otp_verifications table together with the expiration time while it dispatches the one-time password to the user's registered email address. The user needs to input the received one-time password. The system will verify the one-time password marks as verified, hash the password using Bcrypt hashing and update the user's record. This prevents cross use.

The voter authentication is another separate flow, the voters don't follow a traditional login. They get a unique login id when they are added to the voters' table. To sign in, the voters input the login id they got at their respective email address. The system goes through the voter's table, generates an encrypted OTP with the AES-256-GCM algorithm. The system inserts the encrypted one-time password into the voter_login_otps table together with the expiration time while it dispatches the one-time password to the voter's email address. The voters need to input the received one-time password, which will then be used to authenticate their identity. The system marks the one-time password as validated after successful verification and produces an AES-256-GCM token which grants the user entry to the voters' dashboard. It restricts access to only voting-related functionality.

D. Admin Management

Once the super admin logs in, the super admin can create admin accounts by providing appropriate details and it will be added to the users' table. The super admin has rights to view the list of all admins in the system, update the admin details and delete admin. Admin accounts are designated to access elections which they either access or create. But the ultimate control is enforced by the super admin. Also, the admin has been restricted from some tasks and will be safeguarded through the backend service layer on every request.

E. Election Management

The Super admin or Admin who has been given authority can create an election by providing the form details. The details will be passed down to the elections table with the initial status scheduled. The election table has a trigger called trigger_update_election_status. This trigger is really important because it automatically updates the election status. The election status changes based on the date and time. The election status can be in one of three states. These states are: Active, Completed, and Scheduled. The trigger_update_election_status trigger is what makes sure the election status is always correct. The status will transition according to the start time.

F. Super Admin Privileges and Control

The Super Admin holds the highest level of authority in the system. This role includes the ability to create and manage admin accounts, monitor elections, access results, and verify votes.

The Super Admin has exclusive rights to delete elections and associated wards, candidates, and voters. Additionally, the Super Admin can view detailed election statistics and ensure the integrity of the system through administrative oversight.

All actions performed by the Super Admin are validated through backend controls to maintain system security and prevent unauthorized operations.

G. Ward Management with PostGIS

The ward represents the geographical division of the area where the election is going to happen. While creating the ward, the admin or super admin provides the ward details along with the ward geographical boundary as a GeoJSON polygon. The system will process this GeoJSON polygon through PostGIS's ST_GeomFromGeoJSON function to convert it into a native geometry(Polygon 4326) type, which is added to the ward table's boundary polygon table. A GIST spatial index is added to this column to make fast retrieval. A trigger named trigger_update_ward_count is created to automatically increment ward counts in the parent election.

Ward details can be updated and deleted along with voter's records. The ward boundary is optional. This ward boundary can be used for larger scale parts rather than smaller areas due to technical limits. A history button will be available to see who uploaded the voter's excel file and an audit of success/ failure, with a timestamp to monitor. Also, usage is explained in detail in the next section

H. Voter List Upload via Excel and Geo-Matching

The Voter's records sometimes have different addresses or same addresses. To address these issues, the system introduced the administrator to upload voters through Excel file upload and to resolve address discrepancies. The boundary polygons on the ward tables are there.

The administrator can select a ward and upload an Excel file. The Excel file contains columns such as voter ID, name, email address, mobile number and address. The system parses the Excel file and checks if geo-matching is enabled. If it is not enabled, then the system adds every voter in the Excel file to the voter's table along with appropriate election_id and ward_id.

I. Candidates Management

The candidates are registered for specific wards and elected by the administrator. The administrator will give the candidate's name, party name, symbol, ward and election. The system validates the ward id and election ID and inserts the records into the candidate table. Also, respective administrator who has access to the election can update, delete and view the full list of candidates. A search bar is also provided on the candidates list for filtering. Also, with the admin adding a candidate, a candidate named NOTA is also added along with it.

The voter's dashboard will display the instructions for voters to follow and a button to navigate to the voting page. On the voting page, there will be a check for whether voters' device, camera and mic settings are on or not. If not allowed, they wouldn't be allowed to vote. Once the check has been done, the sound and movement detection will be active. The voter will be displayed with a list of candidates, the voter will decide one and cast their votes. If sound and movement are detected, an alert message pops up. Two chances will be given, the voters' vote will be cast for NOTA.

If all checks pass, the system will pass the vote data for further processing. It first identifies the voter, and then a SHA-256 hash is created which contains login ID, candidate ID, the election ID, timestamp and a random number. The hashing is then broadcast onto the Ethereum network using Keccak-256. Only after the blockchain confirms the transaction and returns a transaction_hash and blockchain_hash is the vote added to the vote table. The system sets has_voted = true on the voter record, increment the count for the respective candidate, and increment vote_cast_count for the respective ward. The voter will be displayed with a success message and for whom they voted.

If the blockchain transaction fails at any process, the system will perform a rollback and will not record the voting data. This ensures that the database and blockchain network are in full sync.

J. Result & Statistics

Election results can be seen by Super Admins and Admins after the election time has crossed. The results page shows how many votes each candidate got, grouped by the area they are running in. It also shows a list of candidates in each area ranked by how many votes they got, and it tells you what party they are from. For each area you can see how many people are registered to vote, how many people actually voted and what percentage of people voted. These numbers come from the list of registered voters and the number of votes that have been cast which are always up to date because the database updates them automatically when new voters are added or when people vote. The database does this with triggers that keep everything current.

K. Vote Verification & Audit

On the election polls page, super admin and admin can see the list of voters who voted for which candidates. Also, to double verify there is no

tampering done. The blockchain comes into play. Along with voting, a blockchain_hash will be created. The admin or super admin can verify any vote by sending the blockchain_hash. The system will go through the records and retrieve the transaction_hash associated with the voters. Then the system queries the Ethereum network and verifies the return data such as the transaction's existence, block number, timestamp, and confirmation count. This 2-step verification ensures that every vote can be independently audited without relying solely on the portal's own database.

L. Feedback and Issue Management

The system includes feedback features and a reporting module which can be reported to anyone and free to access. The public can submit their name, email, mobile number, issue subject and detailed report of the issue. Super Admin and Admin can view, mark as read, resolve the issue and update the issue to state whether it is open or closed. This module gives us a lightweight support channel instead of a separate helpdesk system.

M. Security Design Principle

The system makes sure that everything is secure in a way. All the places where people connect to the system are protected by an encrypted token created by AES-256-GCM and API encryption. Two-factor one-time password verification has been made mandatory for every login. The one-time password are stored in encrypted format with expiry timestamps and marked as verified after use. Voting security and integrity is confirmed by the has_voted flag. This unique Boolean constraint will reject voter from further voting if already voted. Along with voting data, the system records IP address and user agent for cybersecurity purpose. In case a legal issue arises regarding the voting, the database administrator can handover the voter record for further investigation. Admin access is limited to the data they own or have been specifically given access to. This is checked at every request at the service layer. All ward boundaries are stored in a format called PostGIS geometries. This is better than storing them as simple coordinate strings. It helps to make sure that any queries that use location data are accurate. It also prevents invalid boundary data from being inserted into the system.

V. SYSTEM ARCHITECTURE

A. Architectural Overview

SecureVote was designed with a clear, distinct, layered and role-based web application following proper client-server architecture. The responsibilities of architecture are divided into frontend, backend, database and external service layers. The system's main aim is to build small to mid-scale elections such as office, school, college and municipal level elections. The user roles are divided into three types: Super Admin, Admin and Voter. With hierarchy in the system, each interacts with the system with restricted API endpoints and dedicated interfaces. The five core pillars of the architecture were the React-based frontend, Node.js backend API, PostgreSQL database, extended POSTGIS, an Ethereum blockchain network for voting integrity and multizone security that spans all layers of the system.

B. Frontend Layer

The frontend system was developed using React JavaScript Library, which provides a dynamic and user-friendly response. Super Admin and Admin have been given access to managing elections, wards, candidates, voters, and results, while Super Admin has added on feature of managing admins designed for the role. The voter portal is accessible only through a unique login_id, leading to the voter dashboard and voting pages. The communication between frontend and backend is completely encrypted through AES-256-GCM at the application layer and exclusively through encrypted HTTPS requests.

C. Backend Layer

The backend of SecureVote is developed using Node.js and exposed using RESTful API using JSON format. The backend is organized around different service modules corresponding to each

functional service, such as: Auth, Election, Voter, Vote, Candidate, Election, Ward and Feedback. Also, role-based access is implemented in each API call with token verification. All requests and responses pass through AES-256-GCM encryption and decryption middleware, ensuring payload confidentiality beyond the network layer.

D. Database layer

The system uses a PostgreSQL relational database with the extension of PostGIS for geospatial processing. The architecture of the database consists of a structured schema with thirteen core tables: users, elections, wards, candidates, voters, votes, election_admins, voter_list_uploads, voter_login_otps, otp_verifications, password_otp_verifications, feedback_and_issues, and spatial_ref_sys (a PostGIS system table).

E. Blockchain layer

The Ethereum blockchain layer serves as a verification audit for voters. The network captures the voting records along with the PostgreSQL database, which serves as external proof for the voter, two-step verification for votes and an independent audit path separately from the database.

F. External Services

The online system has two main external services. The first one is email notification service, which is used to send the OTP codes to Super Admin, Admin and Voters for authentication. Also, the unique login_id credentials for the voters after the records are added. The second is the Ethereum blockchain network. The backend communicates with the blockchain network through the standard Web3 interface, broadcasting vote transactions and querying transactions. These services are dependencies that are isolated within the respective service modules, ensuring that failure doesn't affect other systems.

G. Data Flow Diagram

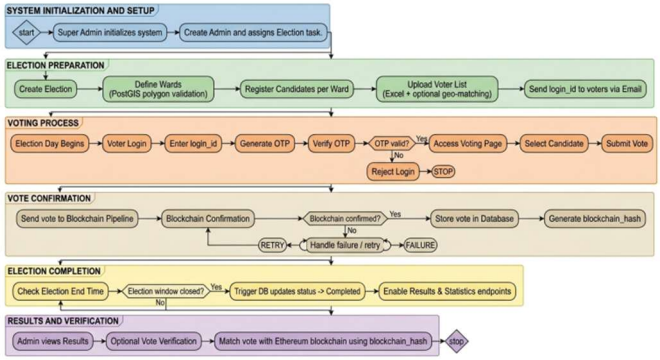

Fig.1 Data Flow Diagram

Fig 1 shows how data is transferred from admin page to election results.

VI. SECURITY ANALYSIS

The security architecture of the system is organized and categorized into 4 distinct zones. Each addresses a specific layer:

A. Zone 1 (Network Security)

The network layer is the defensive layer. All the connections, between the client and the server take place in the network layer. The network layer is protected by HTTPS and TLS encryption. The network layer has a Cross-Origin Resource Sharing policy that only lets authorized origins access the network layer. The network layer also has rate limiting to stop force and denial-of-service attacks on the network layer. To prevent cross-site scripting and content injection attacks Content Security Policy headers are set up in the network layer.

B. Zone 2 (Encryption Layer)

The encryption layer operates at the application level. All requests from the SecureVote are encrypted by the client using the AES-256-GCM algorithm before requesting, and vice versa. The responses are also similarly encrypted at the backend before transmitting. The decryption happens at endpoints by using the corresponding correct keys and initialization vectors. These layers make sure that even if a network packet is bypassed, the request response content remains protected. Before the database storage, the data is encrypted using AES-256-GCM and password encrypted using the bcrypt algorithm.

C. Zone 3 (Authentication and Authorization)

This zone covers the identity verification and access control. For every user login, two-factor OTP authentication is enforced. After OTP verification, a token is issued which is encrypted using AES-256-GCM and carries role-based authentication for every incoming request which is validated by the backend.

D. Zone 4 (Voting Integrity)

This zone is specially dedicated to the voter-facing portion of the system. It enforces the environmental background monitoring system(camera and microphone detection), the blockchain layer for voter recording and detection of voting duplication using the has_voted boolean flag. This zone is protected from both external coercion and internal data manipulation. Also, all the administrative actions performed on the election-related data are properly recorded in the database, which is visible to the database administrator and Super Admin.

VII. Project Execution Using Multiple Terminals

To run the system successfully, four terminals are required. Each terminal will manage a different part of the website.

A. Terminal 1: Frontend

- cd frontend
- npm start

B. Terminal 2: Blockchain

- cd contracts
- npx hardhat node

C. Terminal 3: To Get Contract Address

- cd contracts
- npx hardhat run scripts/deploy.js --network localhost

D. Terminal 4: Backend

You will get a list of private key and contract address from terminal 2(only after running terminal 2 and terminal 3. Then only the key will be displayed). Paste those keys in backend dotenv file, then run the commands.

- cd backend
- npm start

Now SecureVote is ready to run a democratic election

VIII. Implementation Details

The system was successfully implemented based upon the proposed architecture. The following user interface will demonstrate the working module of different roles.

A. Admin Login Page

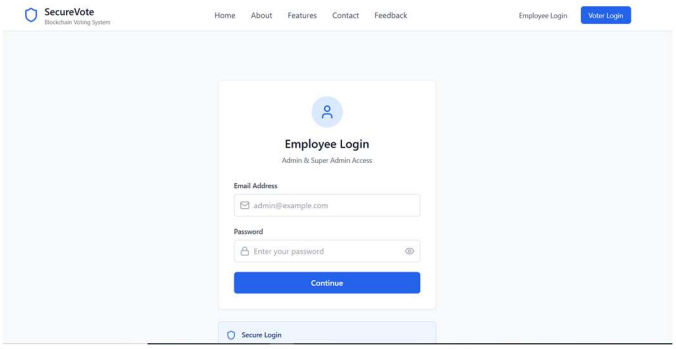

Fig.2 Admin Login Page

B. Admin Dashboard

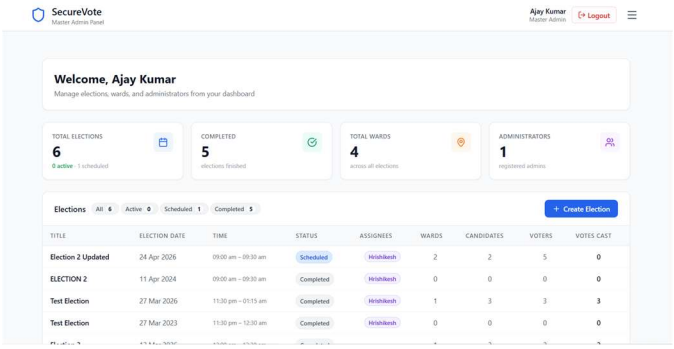

Fig.3 Admin Dashboard

C. Election Creation

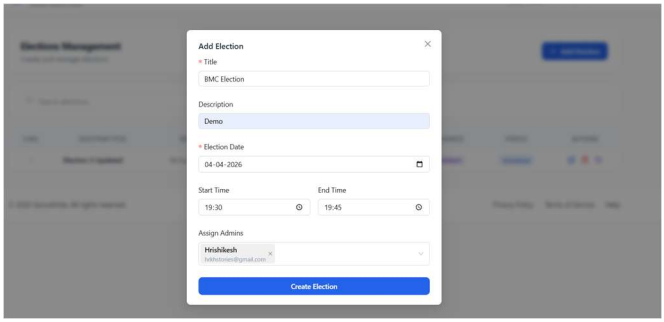

Fig.4 Election Creation

D. Ward Mapping

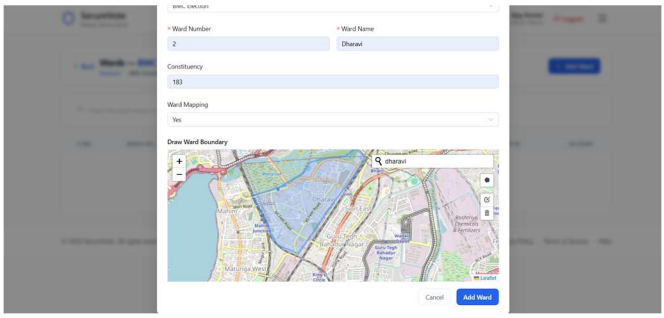

Fig.5 Ward Mapping

E. Voter Login

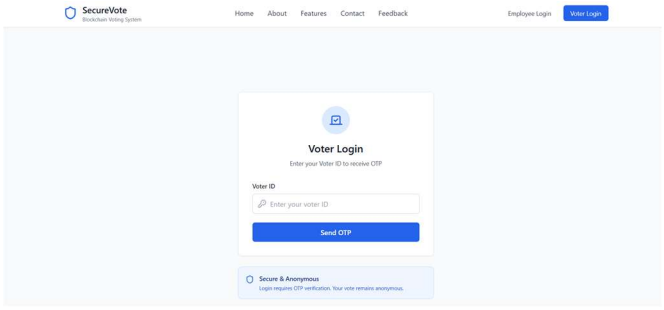

Fig.6 Voter Login

F. Voter Dashboard

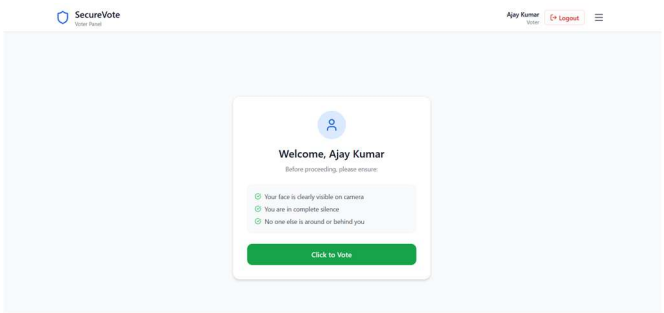

Fig.7 Voter Dashboard

G. Voting Screen

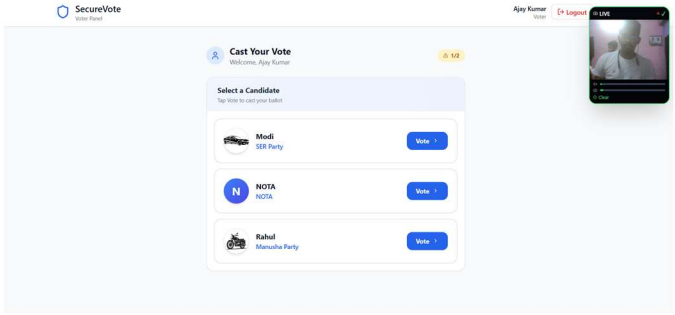

Fig.8 Voting Screen

H. Voting Result Screen

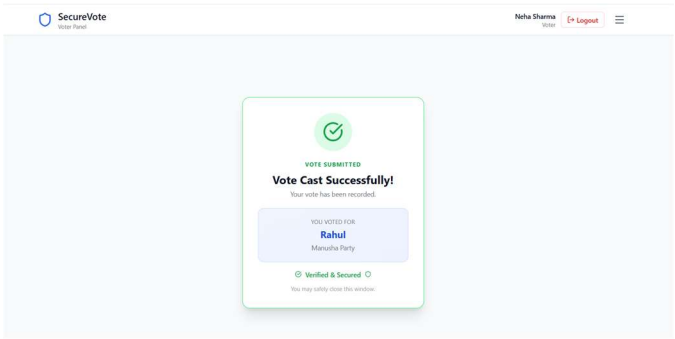

Fig.9 Voting Result Screen

I. Results Page Dashboard

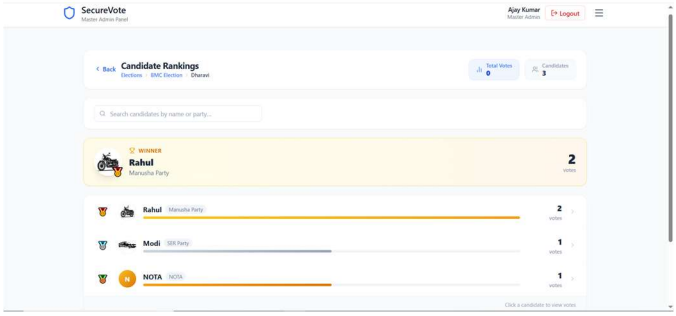

Fig.10 Result Page Dashboard

J. Voter Verification

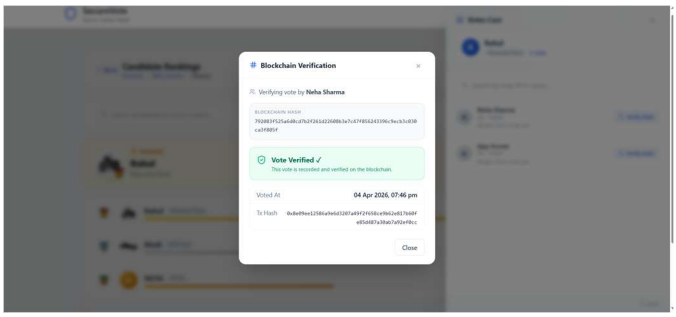

Fig.11 Voter Verification

IX. GitHub Repository

The complete source code of the project is available on GitHub:

https://github.com/Ajay7616/SecureVote

X. RESULT & DISCUSSION

The system was tested across multiple modules including authentication, election management, voting, and result computation. OTP-based login was successfully validated for all user roles. Each vote was recorded securely and linked with a blockchain transaction hash, ensuring tamper-proof storage. The has_voted constraint effectively prevented duplicate voting. Ward-based voter segregation using geospatial mapping produced accurate results for valid address inputs.

The use of blockchain significantly improves trustworthiness and transparency compared to the traditional ballot voting system. Unlike the ballot system or conventional systems, where trust is placed in authorities in this system, the trust is decentralized [1], [2]. However, the system has certain limitations such as dependency on internet connectivity, hardware requirements (camera and microphone access), and challenges in precise geospatial mapping for smaller regions.

XI. Comparison with Existing Systems

Feature	Traditional Voting System	Conventional E-Voting Systems	Proposed Blockchain-Based System
Security	Prone to fake voting and manual tampering	Moderate, depends on central server security	High, due to multi-layer security, OTP authentication, and encryption
Transparency	Limited; process is not fully visible to voters	Limited; controlled by system administrators	High; votes are recorded and verifiable via blockchain and database logs
Trust Model	Trust in election authorities	Trust in centralized system	Trustless; decentralized verification using blockchain
Remote Access	Not available; requires physical presence	Available	Available with secure authentication
Tamper Resistance	High chances of manipulation	Moderate; depends on system design	Very high; ensured through blockchain immutability and database checks
Auditability	Manual and time-consuming	Limited logging mechanisms	Real-time audit using blockchain and database verification
Single Point of Failure	Low; distributed physically	High; centralized server dependency	Low; decentralized and distributed architecture

XII. LIMITATIONS

Although the proposed system has several pros, the system carries several limitations that need to be acknowledged. The PostGIS-based ward boundary mapping and voter geo-segregation is quite limited, and it wouldn't be suitable for smaller areas such as individual buildings, narrow lanes, or densely packed urban constituencies. The environmental monitoring system is quite effective, but it captures only audio and movement only and cannot identify anyone who may be in the room or verify the voter's identity beyond OTP login. The system is designed for small level elections such as office, school, college, and municipal levels, and is not valid for infrastructure demands, concurrent user load, or regulatory requirements of large-scale state or national elections. Also, voter participation depends on digital literacy, access to a device, internet connection, and registered email address, which may exclude a large section of voters. Also, the reliance on blockchain depends on confirmation of network availability and blockchain transactions, and fees may become a cost consideration as the voter number scales up.

XIII. FUTURE SCOPE

Now, the election level is restricted to a smaller level. In the future, it will expand to conduct elections on a larger scale like state and national-level elections and infrastructure with enhanced scalability, load handling, regulatory compliance, and multi-jurisdiction election management capabilities. The environmental monitoring feature captures only video and audio during the election. In the future, it will be expanded to real-time facial recognition, surrounding environment capturing, and identity verification for voter fraud prevention. The scale of ward mapping and voter geo-segregation features are quite limited in small areas, which will be enhanced to capture smaller areas in the future. Beyond these expansions, the system will also explore enhancing the issue of feedback system, admin and super admin voter dashboard, multilingual voter interface and integration with national voter identity to further strengthen voter authentication [3], [8].

XIV. CONCLUSION

SecureVote is a significant push towards digitalization and addresses the longstanding issue with traditional elections. Also, the proposed system adopts React and Node.js for web applications with PostgreSQL database, PostGIS for geospatial processing, the Ethereum Blockchain network for tamper-proof vote storage and end-to-end encryption and four-zone security. This makes SecureVote secure, transparent, and accessible for small level elections.

While the system carries some limitations, it has established a strong foundation in both the technical and the practical sides which can be extended and adapted for proper security requirements. Overall, SecureVote is a secure, viable alternative to the traditional voting system, and contributes meaningfully to the ongoing effort to make democratic practice more accessible, trustworthy and resilient in the internet age.

ACKNOWLEDGMENT

I take this opportunity to express my profound gratitude to my Computer Science Department, for giving me the opportunity to accomplish this research work. I am also deeply thankful to Dr. Radhika Birmole, Principal in charge, for their continuous support and encouragement.

REFERENCES

[1] 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, 2018.

[2] F. S. Hardwick, A. Gioulis, R. N. Akram, and K. Markantonakis, "E-Voting with Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy," in *Proc. IEEE International Conference on Internet of Things (iThings)*, 2018.

[3] Md. Atik, S. S. Tarin, M. Rahman, A. A. Alvi, and Md. F. Sohan, "A Comprehensive Analysis of Blockchain-Based Voting Systems: Enhancing Transparency and Security," Journal of Computer Science and Technology Studies, 2023.

[4] F. P. Hjálmarsson, G. K. Hreioarsson, M. Hamdaqa, and G. Hjálmtýsson, "Blockchain-Based E-Voting System," in Proc. IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2018, pp. 983–986.

[5] A. S. Yadav, A. U. Thombare, Y. V. Urade, and A. A. Patil, "E-Voting using Blockchain Technology," International Journal of Engineering Research & Technology (IJERT), vol. 9, no. 5, 2020.

[6] A. A. Lahane, J. Patel, T. Pathan, and P. Potdar, "Blockchain technology based e-voting system," International Journal of Research in Engineering, Science and Management, 2021.

[7] G. Gopalakrishnan, A. Jadhav, K. Parakhaya, R. Singh, and K. S. Suresh Babu, "Online Voting System using Blockchain," International Journal of Advanced Research in Computer and Communication Engineering, vol. 9, no. 3, 2020.

[8] U. Jafar, M. J. Ab Aziz, and Z. Shukur, "Blockchain for Electronic Voting System—Review and Open Research Challenges," Sensors, vol. 21, no. 17, p. 5874, 2021.

[9] P. Ehin, M. Solvak, J. Willemson, and P. Vinkel, "Internet voting in Estonia 2005–2019: Evidence from eleven elections," Government Information Quarterly, vol. 39, no. 4, p. 101718, 2022.

