

## A Secure and Transparent E-Voting Framework Using Ethereum Blockchain

Gayitri H M<sup>1</sup>, Kruthi Krishnagiri<sup>2</sup>, Abhishek Gaviyappa<sup>3</sup>

<sup>1,2,3</sup> Department of Electronics and Communication Engineering, Sri Jayachamarajendra College of Engineering, JSS Science & Technology University, Mysore – 570006, India

Corresponding Author: E-mail: [gayitrikumar@sjce.ac.in](mailto:gayitrikumar@sjce.ac.in)

---

### ARTICLE INFO

### ABSTRACT

Received: 15 July 2025

Revised: 27 Aug 2025

Accepted: 07 Sept 2025

Ethereum blockchain technology, this study suggests a decentralized electronic voting system to improve digital elections' security, transparency, and credibility. In order to automate crucial procedures like voter registration, candidate administration, vote casting, and result computation, the system makes use of smart contracts, guaranteeing that every operation is verifiable and impenetrable. While MetaMask integration allows for secure user authentication and transaction signature, a frontend built on React.js offers an intuitive user experience. IPFS is used in the implementation of off-chain storage to effectively handle sensitive data and lessen on-chain burden. To verify the efficacy of the system, functional testing was carried out on the Ethereum test net and Ganache. Offering a scalable and reliable foundation for contemporary, easily accessible, and secure voting procedures, the modular design shows great promise for practical implementation.

**Keywords:** JavaScript, Solidity, React, Node.js, Ganache, MetaMask

---

### Introduction

Blockchain is a decentralized digital ledger that is perfect for electronic voting since it securely records transactions without the need for middlemen. Election data is transparent and impenetrable since each block has an unchangeable record. Data is shared throughout a peer-to-peer network of nodes, removing single points of failure, in contrast to centralized systems. Every vote is forever recorded and verifiable, and voters can cast their ballots in confidence. In addition to preventing electoral fraud, this improves trust and permits real-time counting. As a result, blockchain provides a safe, open, and effective replacement for contemporary voting systems.[1]

Voting systems have mainly not changed despite scientific and technological breakthroughs; they have only evolved in appearance rather than functionality. Due to its continued reliance on centralized infrastructures, traditional electronic voting is susceptible to problems with security and transparency. Although cryptographic methods made considerable improvements, they were unable to provide completely safe, scalable, and mobile-friendly solutions. Given the increased mobility of people in today's society, this restriction becomes even more crucial. By using smart contracts to enable transparent, tamper-proof, and remote voting, blockchain-based decentralized voting on platforms such as Ethereum presents a possible substitute. This strategy promotes greater voter engagement, removes middlemen, and builds trust.[2]

Electronic voting, which allows voters to cast their ballots remotely using smart devices like smartphones and tablets, has gained significant attention in recent years. Real-time counting, immediate results, transparency, anonymity, and a smaller environmental impact are some advantages of switching from paper-based to electronic voting. Additionally, it promotes decentralization and reduces errors. However, there are a number of security issues, weaknesses, and possible attacks

associated with the growing use of digital voting systems. It is still crucial to provide correctness, consistency, verifiability, anonymity, and authenticity. Building a reliable and secure electronic voting system requires addressing these issues.[3]

### **1.1 Motivation**

The Traditional electronic voting systems continually have security vulnerabilities, lack of transparency, and limited auditability, which undermine public faith in democratic processes. Human manual monitoring and centralized administration also expose elections to manipulation and incompetence. Blockchain technology, in the case of Ethereum, offers a decentralized, tamper-evident environment with end-to-end transparency and secure storage of votes. Smart contracts can be employed to self-executing record, verify, and count votes without the need for manual intervention. Voter confidentiality is preserved while making it possible to have real-time observation and traceability. The potential to reduce costs and make it more accessible by enabling individuals to vote remotely improves its scalability. The system aims to eliminate fraud and in still trust into electoral processes. The motivation is to create a secure, transparent, and inclusive system for future democratic polls.

### **1.2 Problem Definition**

Traditional electronic voting systems sometimes depend on centralized infrastructures that have a number of serious problems, such as single points of failure, data tampering, and cyberattack susceptibility. Additionally, these systems are commonly criticized for their poor auditability, lack of transparency, and inadequate protection of voter confidentiality. These solutions are ineffective at guaranteeing the validity of the voting process in situations involving large-scale elections. The possibility of manipulation or administrative errors is further increased by manual intervention during registration, vote counting, and result confirmation. Furthermore, traditional voting platforms are not sufficiently scalable or capable of supporting remote participation, which restricts accessibility for citizens in emergency situations, disabled people, and remote voters.

### **1.3 Research Contribution**

**The main contributions of this study are summarized as follows:**

- Using Ethereum smart contracts, a scalable, safe, and flexible architecture is created to establish a decentralized electronic voting system. This approach guarantees tamper-resistant data storage, does away with centralized control, and permits trust less vote execution. The voting process's main logic is comprised of smart contracts that are deployed using Truffle and tested using Ganache, guaranteeing vote counting's transparency and verifiability
- MetaMask wallet integration is used to provide a decentralized authentication and verification scheme. By securely authenticating voters using Ethereum account addresses, the technology facilitates distant voting. By confirming eligibility prior to voting, admin-level cross-verification improves integrity without jeopardizing anonymity
- To handle sensitive data, SHA-256 hashing and IPFS are used to enforce cryptographic security. Voter information is hashed and kept off-chain to protect privacy, even if votes are permanently recorded on-chain. Secure and verified participation is ensured by private key-based access and multi-factor verification
- React.js is used in the front-end interface to provide a responsive and easy-to-use voting environment for administrators and voters alike. Real-time election updates, candidate registration, and transparent vote tracking are made possible by the user interface , which enhances accessibility and usability for all parties involved
- Private Ethereum blockchain configurations and simulation tools are used to introduce an energy-efficient and economical voting framework. In contrast to conventional systems, the

design preserves the election process's real-time integrity and auditability while drastically lowering infrastructure costs and enabling widespread deployment.

## **2. Related work**

“Blockchain-Based E-Voting System Using Smart Contracts e-voting system “ utilises smart contracts on the Polygon blockchain for transparency, integrity, and privacy in the election. The system utilizes OTP-based voter authentication and Paillier homomorphic encryption to enable anonymous, secure voting submission and encrypted on-chain tallying. Smart contracts provide voting rules, double prevention, and coercion resistance while allowing verifiability through unalterable records. With a promising design, usability improvements and prototype testing in large-scale electoral environments are future prospects[1]

“Impact of Decentralization on Electronic Voting Systems: A Systematic Literature Survey” a structured literature review of 133 studies compressed to 33 practical decentralized e-voting systems, the majority of which are based on blockchain. The systems are contrasted in terms of cryptography techniques, architecture design, and decentralization impact. Filtering, classification, and comparative evaluation are part of the process coupled with the most significant security and performance metrics. Findings highlight enhanced voter convenience and anonymity through decentralization but report increased system complexity and the need for standardized measures[2]

”Development Principles for Electronic Voting System Using Distributed Ledger Technology” This book presents a structured literature review of 133 studies compressed to 33 practical decentralized e-voting systems, the majority of which are based on blockchain. The systems are contrasted in terms of cryptography techniques, architecture design, and decentralization impact. Filtering, classification, and comparative evaluation are part of the process coupled with the most significant security and performance metrics. Findings highlight enhanced voter convenience and anonymity through decentralization but report increased system complexity and the need for standardized measures[3]

“Blockchain Based Electronic Voting System ” Legacy e-voting systems are likely to be insecure against violations, are not transparent, and are highly limited in their auditability, which can compromise the integrity of elections and public trust. Manual management of votes and centralized management increase the stakes of tampering, fraud, and inefficiencies. The systems also tend to restrict voter accessibility, especially for distant or disabled voters. There is a critical need for a secure, open, tamper-evident, and scalable voting system that can ensure voter anonymity, facilitate real-time smooth verification of votes, and maximize participation in the democratic process[4]

“ Impact of Decentralization on Electronic Voting Systems: A Systematic Literature Survey ” conducts a systematic literature review of 133 e-voting studies, whittling down to 33 deployable decentralized systems for careful analysis. The process applies structured filtering and cross-comparative analysis based on cryptographic techniques, architectural styles, and degrees of decentralization. It concludes that blockchain enhances voter mobility and anonymity but increases complexity, and Ethereum and smart contracts are dominant. It suggests the use of standardized benchmarks and application studies in the field for these systems to ensure their validity[5]

“ Blockchain-Based E-Voting System ” proposes a cost-effective, open, and secure e-voting system on a private Ethereum blockchain with PoA consensus. Implementation is based on a method of holding elections as smart contracts on district nodes and validating transactions through institutional authorities. Votes are appended upon consensus to preserve integrity, and verifiability is facilitated through transaction IDs. Scalable at the regional level, the research identifies scalability concerns and suggests optimization for nationwide deployment[6]

“ Decentralized Online Voting System using Blockchain ” presents a decentralized e-voting system on the Ethereum blockchain with Aadhaar-based biometric login to enable verifiable and secure elections. The solution combines OTP-based login and facial and fingerprint recognition using Haar Cascades and facial embeddings in a web interface. Encrypted votes are stored on chain, with one-person-one-vote and transparency provided using smart contracts. Despite encouraging participation and security, the system is faced with issues of accessibility, internet dependency, and finiteness of biometric information[7]

“ A Smart Contract System for Decentralized Borda Count Voting ” presents a decentralized Borda count voting scheme on the Ethereum blockchain to facilitate self-tallying without a central authority figure. The process is a two-round scheme in which voters first publish public keys, and then cast randomized encrypted ballots with non-interactive zero-knowledge proofs for privacy and correctness. Ethereum is used as a public bulletin board, providing transparency and verifiability. Although presently appropriate for small-scale elections, future development will work towards improving scalability and efficiency[8]

“ Blockchain-Based Voting Considered Harmful ” proposes a secure, transparent, and low-cost e-voting scheme using a private Ethereum blockchain with Proof-of-Authority consensus. The method entails running elections as smart contracts on district nodes and verifying voters using institutional authorities. Votes are appended after consensus for integrity, with verifiability enabled through transaction IDs. While potent at a regional level, the work illustrates scalability limitations and advises optimization for national deployment[9]

“ A Distributed Self-Tallying Electronic Voting System Using the Smart Contract ” proposes a distributed self-tallying e-voting system resolving abortive and adaptive issues in the absence of trusted authority, utilizing smart contracts on Ethereum. Methodology integrates an optimized Group Encryption Scheme and Exponential ElGamal cryptosystem with zero-knowledge proofs and homomorphic encryption to ensure privacy and verifiability. Votes are kept securely on-chain and counted securely without loss of accuracy and resilience. While more efficient than previous models, computational complexity and proof size optimization present challenges for use at large scale[10].

## **2.1 Literature Summary**

The e-voting system based on blockchain utilizes smart contracts on the Polygon blockchain to guarantee transparency, integrity, and privacy. It employs OTP-based verification and Paillier homomorphic encryption for secure and anonymous voting and encrypted on-chain counting. Smart contracts enforce the rules of voting and make it possible for tamper-proof proofs of verifiability[1]. The survey of the literature examines 133 studies, which were narrowed down to 33 operational decentralized e-voting systems, which are primarily blockchain-based. It compares them against each other using cryptographic techniques, architecture, and decentralization impacts, assessing security and performance measures. The study discovers enhanced voter anonymity and ease of use but mentions increased system complexity and non-standardization.[2]. This book examines 133 studies and finds 33 practical decentralized e-voting systems, which are predominantly blockchain-based. It examines them in terms of cryptographic approaches, architecture, and decentralization effect with major security and performance metrics. It concludes that decentralization enhances voter anonymity and convenience but increases complexity and underscores the need for standardization[3]. Legacy e-voting systems have problems such as weak security, transparency, and limited auditability that threaten election integrity. Centralized and manual vote handling enhances the possibilities of fraud and limits accessibility. There is a need for a blockchain system to provide security, voter anonymity, real-time verification, and increased participation[4]. A systematic review of 133 e-voting studies, shortlisted to 33 decentralized systems for close examination. It compares them in terms of cryptography, architecture, and decentralization, with the standout technologies being Ethereum and smart contracts. The research concludes that decentralization enhances voter mobility and anonymity

but creates complexity, suggesting the use of standardized benchmarks for future work[5] The system provides a decentralized e-voting platform on Ethereum based on Aadhaar-based biometric login with OTP, face, and fingerprint recognition. The votes are encrypted and kept on-chain for transparency and one-person-one-vote via smart contracts. It increases security and participation but is challenged by accessibility, internet dependency, and limited availability of biometric information[6] The paper suggests a decentralized Borda count election system on Ethereum, which allows self-tallying without central control. It employs a two-round mechanism with ciphertext ballots and NIZK proofs to maintain privacy and correctness. Although suitable for small-scale elections, the system hopes to scale up and become more efficient in further work[7] The paper introduces an e-voting system on blockchain with a private Ethereum network based on PoA consensus and smart contracts on district nodes. Institutional authorities manage the verification of voters, with votes made traceable through transaction IDs [8] The paper identifies scalability issues and suggests enhancements for wider, national-level use. It presents a transparent and safe electronic voting system utilizing a private Ethereum blockchain with PoA consensus and smart contracts[9] The presents a decentralized electronic voting system that ensures privacy and verifiability without the need for a trusted authority. Large proof sizes and computational complexity limit scalability despite its increased efficiency[10]

**Table1: Simplified Comparison Table of E-Voting Systems**

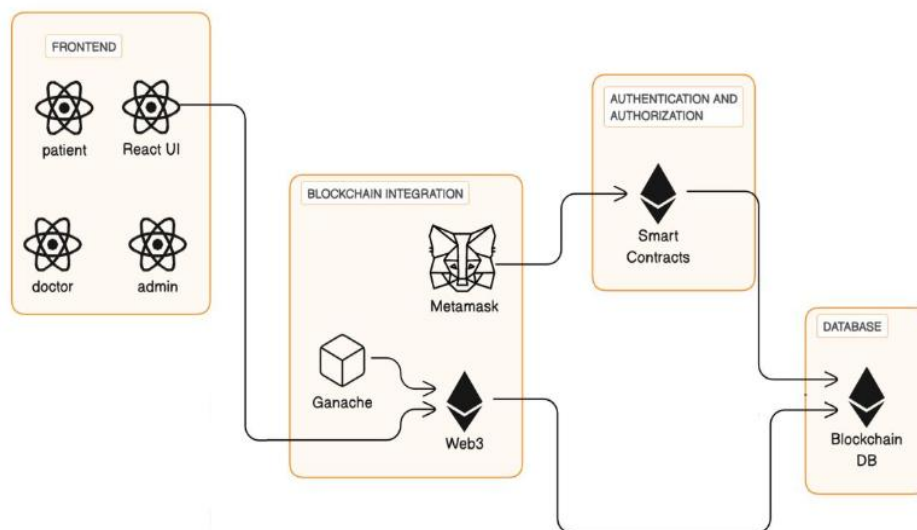
Ref	Title/System	Blockchain	Authentication	Privacy & Security	Smart Contracts	Key Feature
[1]	Smart Contracts E-Voting	Polygon	OTP	Homomorphic Encryption	Included	Encrypted on-chain tallying
[2][5]	Decentralized Voting Survey	Multiple	Varies	Varies	Varies	Broad comparative review
[3]	Dev. Principles for E-Voting	Multiple	Varies	Varies	Varies	Design principles, no system
[4]	Blockchain-Based E-Voting	Not specified	None	Basic	Included	Calls for transparency
[6][9]	PoA-Based Voting	Ethereum	Institutional	Moderate	Included	Regional voting, low cost
[7]	Aadhaar-Based Voting	Ethereum	Boimetrics + OTP	Encrypted votes	Included	Biometric login integration
[8]	Borda Count Voting	Ethereum	Public Keys	ZKP + Encryption	Included	Self-tallying with Borda method
[10]	Self-Tallying Voting	Ethereum	Public Keys	ZKP ElGamal +	Included	No central authority
This Work	Proposed System	Ethereum	MetaMask + Admin	SHA-256 + IPFS	Included	Full-stack + hybrid storage



### 3. Proposed blockchain-based smart contract e-voting system

The Solidity programming language can be used to create and implement smart contracts and decentralized apps on Ethereum, an open-source, decentralized blockchain platform. With the help of a network of nodes that carry out and verify transactions independently of a centralized authority, it operates as a worldwide virtual computer. Ethereum uses decentralized governance, consensus processes, and cryptographic protocols to provide transparent, tamper-proof, and trustless operations. The main programming language for Ethereum, Solidity, is contract-oriented and made to build safe, effective smart contracts that operate on the Ethereum Virtual Machine. Ethereum improves security and efficiency in all digital interactions by eliminating middlemen and facilitating automation. By rethinking the practical applications of trust, transparency, and decentralized control, this unique platform has revolutionized sectors like healthcare, banking, and supply chain management.[11]

#### 3.1 Blockchain Technology



**Figure1:** System architecture of decentralized System

To guarantee safe, transparent, and impenetrable vote recording, the created electronic voting system was implemented on the Ethereum blockchain. Ganache was used as a local Ethereum blockchain emulator to make development and testing easier. Before deploying to a live network, developers were able to replicate and validate all blockchain interactions thanks to Ganache's controlled environment, which included pre-funded accounts, quick block mining, and insight into transaction behavior. The Truffle Suite was used for writing, compiling, and migrating smart contracts. Reliability was ensured and deployment mistakes were reduced thanks to Truffle's robust framework, which enabled structured development with migration scripts and automated testing capabilities utilizing the Mocha framework.

MetaMask was incorporated as a browser-based Ethereum wallet to help with communication between the user interface and the blockchain. Users (administrators and voters) could sign transactions without disclosing their private keys thanks to MetaMask, which also managed secure user authentication. This made sure that every blockchain transaction, including voting and candidate registration, was both cryptographically secure and verifiable. Only verified MetaMask accounts that were registered by the administrator were permitted to take part in the voting process, thanks to the system.

The user interface on the front end was created especially to handle administrative features including candidate administration, voter registration, and real-time voting process tracking. The admin

dashboard offered dynamic feedback on contract state and blockchain events, facilitating smooth interaction with deployed smart contracts using Web3.js. The voting system provided dashboard insights to show real-time changes in election data, validated Ethereum address linking, and real-time transaction updates.

The smart contracts, frontend, wallet authentication, and blockchain all functioned separately yet in unison under this deployment model's modular and completely decentralized architecture. The system was designed to be compatible with Ethereum test nets, which ensured that it could grow toward real-world, production-ready environments, even though Ganache was utilized for local testing. This implementation demonstrates that blockchain technology may be used to create safe, scalable, and auditable electronic voting systems.

#### 4. Implementation

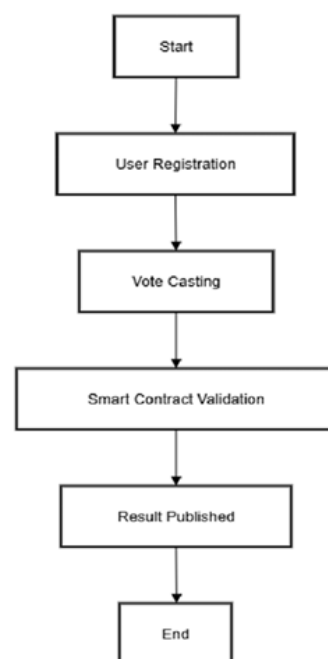
**Candidate Dataset:** The administrator registers the candidate's name and unique identification number. To preserve transparency and immutability, these records are kept on the blockchain via smart contracts.

**Blockchain Transaction Data:** On the Ethereum blockchain, each vote results in a transaction. These consist of wallet addresses, timestamps, block numbers, and transaction hashes. The election process's unchangeable audit trail is made up of this dataset.

**Off-Chain Storage Data :** The Inter Planetary File System is a secure location for sensitive data, including voter KYC information and biometric IDs . Only the encrypted hash references are kept on the blockchain, guaranteeing anonymity and lowering on-chain costs.

**Testing and Validation Data:** During development, a local blockchain environment with pre-funded Ethereum accounts was simulated using Ganache. The dataset used to verify the voting procedure prior to deployment on the Ethereum test net consists of these accounts and the transactions connected to them.

In conclusion, rather than coming from static sources, the system's dataset is transactional and real-time in nature, created when the voting platform is really in use. This method improves openness, security, and authenticity while adhering to the tenets of blockchain-based electronic voting.



**Figure 5.2: Flow Diagram of proposed work**

This flowchart illustrates how an Ethereum-based blockchain electronic voting system operates. Here is a detailed breakdown of every block:

**1. Start:**

- When the system is initialized, the procedure starts.
- This is the point of entry for users to take part in the voting process.

**2. User Registration**

- Voters who are eligible create accounts using the interface that is given (for example, MetaMask).
- In order to register, you must provide your wallet address and identifying data.
- By comparing the voter's eligibility to a whitelist or other authentication methods, the system confirms their eligibility.

**3. Vote Casting**

- Voters can use the interface to cast their ballots after registering.
- On the blockchain, the vote is transmitted as a transaction.
- The method guards against tampering during submission and guarantees that each voter may only cast one vote.

**4. Validation of Smart Contracts**

- A Solidity-written smart contract handles the voting transaction.
- The contract confirms that the vote was legitimate:
- confirms the voter's registration.
- Verifies that the voter hasn't cast their ballot yet.
- The vote is permanently recorded in the blockchain ledger.
- At this point, every invalid vote is disregarded.

**5. Result Published**

- Following validation, votes stored on the blockchain are aggregated to determine the outcomes.
- The outcomes are then made available to the public and reported in real time.
- The data is unchangeable, preventing any tampering, and transparency is guaranteed.

**6. End**

- After the final results are released, the procedure is over.
- All transactions are permanently documented on the blockchain, and the voting event has ended.

**Key Takeaways**

- The process flow shown in the diagram is secure but linear.
- Smart contracts facilitate the smooth process of user authentication, vote casting, and certification.



**TABLE 2: Dataset Description for Ethereum-Based E-Voting System**

Dataset Type	Description	Source / Storage	Purpose
Voter Registration Data	Includes voter name, phone number, and Ethereum wallet address.	Frontend (React.js) + Admin validation	To authenticate and register eligible voters.
Candidate Dataset	Candidate details such as name and unique identifier.	Smart Contracts on Ethereum Blockchain	To store candidate information immutably and transparently.
Blockchain Transactions	Vote transactions including transaction hash, block number, timestamp, wallet ID.	Ethereum Blockchain (Ganache)	To record votes securely and provide auditability.
Off-Chain Storage Data	Sensitive data such as voter phone Number, stored as encrypted hashes.	Inter Planetary File System	To ensure privacy while reducing blockchain storage load.
Testing Data	Sensitive data such as voter Phone Number, stored as encrypted hashes.	Ganache Local Blockchain Emulator	To simulate the voting process before real deployment.

To ensure seamless operation during development, testing, and deployment, the suggested blockchain-based electronic voting system implementation calls for a reasonably equipped machine with a processor running at 2 GHz or faster, at least 4 GB of RAM, and 100 GB or more of free disk space. Performance bottleneck-free frontend rendering, database operations, and local blockchain environments are all supported by these specs.

The system uses Web3.js for blockchain interaction and Node.js as the runtime on the software side. Solidity is used to create smart contracts, which are then implemented on Ganache using Truffle While MySQL and Fast API take care of backend services and data storage, MetaMask handles wallet-based authentication.

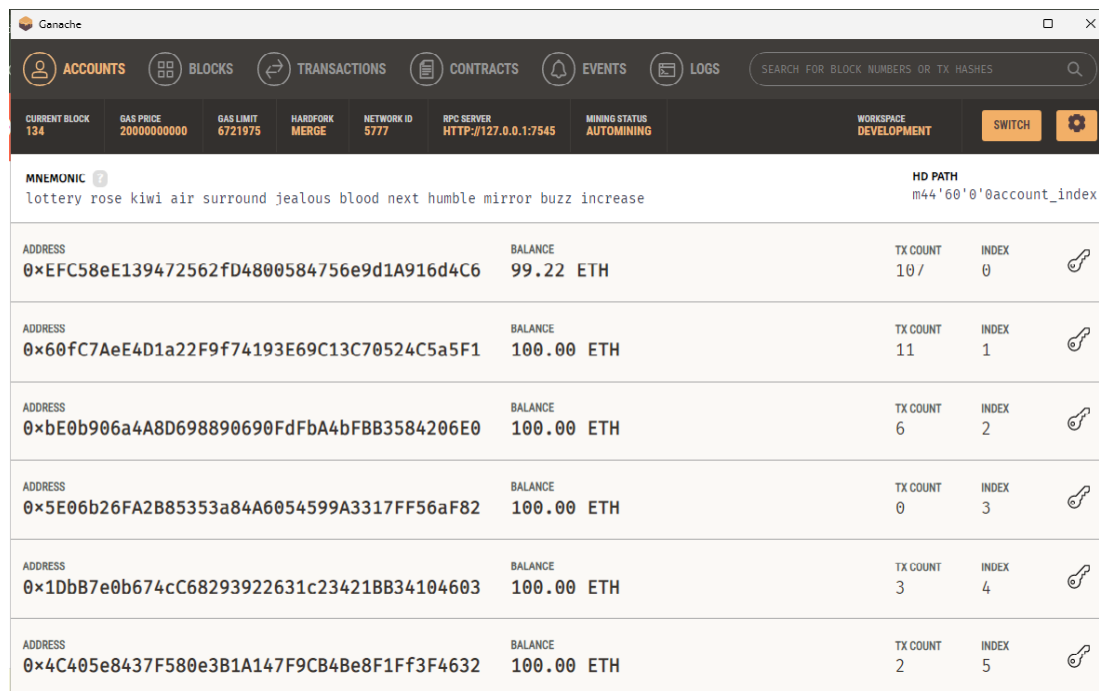
Web3.js was used as the middleware to enable communication between the Ethereum blockchain and the frontend application. The frontend can interact with blockchain-deployed smart contracts thanks to this JavaScript framework. MetaMask was incorporated into the system to securely sign blockchain transactions and manage Ethereum accounts. With the help of the popular browser extension MetaMask, users may communicate with decentralized apps straight from their browser. In order to verify functionality and behaviour prior to live deployment, MetaMask was set up to connect to a local Ethereum blockchain simulation using Ganache during development and testing.

React.js, a powerful JavaScript toolkit made for creating dynamic, responsive web applications, was used to create the user interface. Voter registration, candidate selection, and safe vote casting were among the crucial functions offered by the frontend. Accessibility was improved with a voter-friendly layout that guaranteed usability for people with little technical expertise. Smart contracts handled user activities like voting as they engaged with the interface, and the resulting blockchain state was retrieved in real-time to show the most recent election data. This strategy guaranteed responsiveness and

transparency, which are essential for any voting system.

The system used IPFS for off-chain storage in order to solve concerns about data privacy and storage, especially for sensitive information like biometric data and Know Your Customer details. The SHA-256 cryptographic hash function, which creates a fixed-size hash result from any data input, was used to encrypt the data before it was uploaded to IPFS. By acting as a distinct fingerprint of the original material, this hash value guaranteed confidentiality and integrity. Only the encrypted hash references were stored on-chain, as opposed to raw data, which greatly reduced on-chain storage overhead and allowed verification without disclosing personal information.

The system was thoroughly tested in the Ganache environment to verify every step of the voting process, from registration to the calculation of the results. The testing stage made sure that every Ethereum address that was registered could only vote once and that every transaction was accurately recorded on the blockchain, free from any chance of fraud or duplication. The solution was put to the Ethereum test net after verification, allowing for more extensive engagement with actual blockchain behaviour. End users used MetaMask for authentication, the React-based frontend to access the Dapp, and smart contracts to safely cast their votes. Data integrity, voter anonymity, and complete transparency all essential e-voting features were upheld throughout this process, proving the system's feasibility as a safe and decentralized electronic voting solution.



The screenshot shows the Ganache application window. At the top, there's a navigation bar with icons for Accounts, Blocks, Transactions, Contracts, Events, and Logs. Below this is a status bar displaying various metrics like Current Block (134), Gas Price (2000000000), Gas Limit (671975), Hardfork (MERGE), Network ID (5777), RPC Server (HTTP://127.0.0.1:7545), Mining Status (AUTOMINING), and Workspace (DEVELOPMENT). The main area shows the 'ACCOUNTS' tab with a mnemonic phrase: 'lottery rose kiwi air surround jealous blood next humble mirror buzz increase'. Below the mnemonic is a table listing several pre-funded accounts, each with an address, balance (100.00 ETH), transaction count, and index.

ADDRESS	BALANCE	TX COUNT	INDEX
0xEFC58eE139472562fD480584756e9d1A916d4C6	99.22 ETH	10 /	0
0x60fC7Ae4D1a22F9f74193E69C13C70524C5a5F1	100.00 ETH	11	1
0xbE0b906a4A8D698890690FdFbA4bFBB3584206E0	100.00 ETH	6	2
0x5E06b26FA2B85353a84A6054599A3317FF56aF82	100.00 ETH	0	3
0x1DbB7e0b674cC68293922631c23421BB34104603	100.00 ETH	3	4
0x4C405e8437F580e3B1A147F9CB4Be8F1f3F4632	100.00 ETH	2	5

**Figure 2:** Ganache

To facilitate the creation and testing of smart contracts, Ganache is used as a local Ethereum blockchain emulator Fig. 2. It offers a controlled environment with real-time block confirmations and thorough transaction monitoring by simulating an actual blockchain network. By providing several pre-funded Ethereum accounts that are all started with 100 ETH by default, Ganache makes it possible for developers to deploy and communicate with smart contracts without having to pay real gas fees. This configuration makes it possible to test features like admin control, vote casting, and voter registration quickly. Its transaction visibility and integrated GUI aid in the early detection of faults during the development cycle. The e-voting system uses Ganache to provide secure operation and strong validation before going live on public Ethereum testnets or the main network.

The screenshot shows the 'ADMIN' dashboard with a navigation bar containing 'Verification', 'Add Candidate', 'Registration', 'Voting', and 'Results'. A light blue box at the top indicates 'Total registered voters: 0'. The 'Registration' section is active, showing a form to 'Register to vote'. The form includes three input fields: 'Account Address' (containing a long hexadecimal string), 'Name' (containing 'Kruthi Krishna'), and 'Phone number \*' (containing '9886027649'). A 'Note' section below the form states: 'Make sure your account address and Phone number are correct. Admin might not approve your account if the provided Phone number does not match the account address registered in admin's catalogue.' A 'Register' button is at the bottom right of the form.

**Figure 3:** The Voter Registration Process

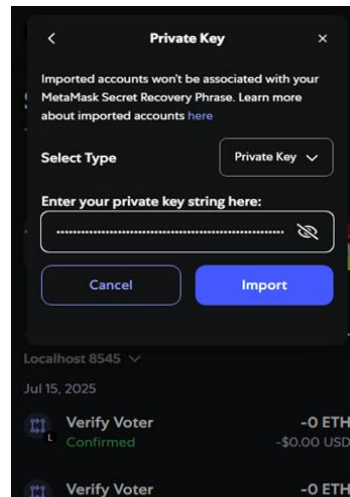
The e-voting system's voter registration screen, where users enter their name, phone number, and Ethereum wallet address, is shown in Figure 3. React.js was used in the development of this front-end interface, which is made to be user-friendly even for those with no technical expertise. The system verifies the Ethereum address entered by the user against the administrator's approved voter catalogue once the user submits the form. This guarantees that only users who are eligible and allowed can cast ballots. Registration is denied if the address is not on the administrator's whitelist. By detecting fraudulent or duplicate entries and guaranteeing that only confirmed users can move on to the secure voting phase, this validation procedure improves the voting system's integrity.

The screenshot shows the 'ADMIN' dashboard with a navigation bar containing 'Verification', 'Add Candidate', and 'Registration'. The 'Add a new candidate' section is active, showing a form to 'Add a new candidate'. The form includes two input fields: 'Header' (containing 'kruthi') and 'Slogan' (containing 'The spirit of adventure'). An 'Add' button is at the bottom right of the form. On the right side of the screen, a 'Transaction request' modal is open, showing details for 'Account 2' (Localhost 8545). The modal includes a 'Request from' field (HTTP localhost:3000), an 'Interacting with' field (0xF1915...8C542), a 'Network fee' field (0.0025 ETH), and a 'Speed' field. At the bottom of the modal are 'Cancel' and 'Confirm' buttons.

**Figure 4:** Adding New Candidate

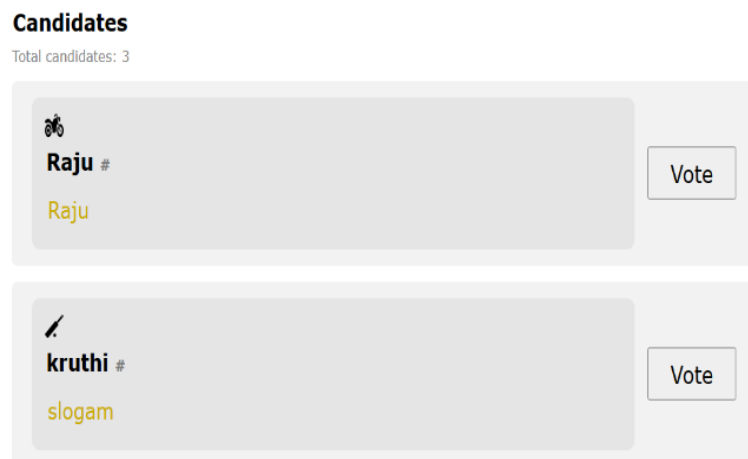
The admin interface for introducing new candidates to the e-voting system is shown in Figure 4.

Through a structured React.js form, this interface enables the administrator to enter the candidate's name and pertinent information. The system initiates a blockchain transaction upon submission, updating the smart contract with the new candidate's details. A MetaMask pop-up prompts the administrator to examine and authorize the transaction using their Ethereum account in order to guarantee the legitimacy and security of this action. The candidate is not formally registered on the blockchain until this confirmation has been received. By avoiding illegal alterations and guaranteeing that all changes to the election setup are permanently documented, this procedure strengthens the election's openness and reliability.



**Figure 5:** Private Key to Import

The Private Key Import feature in the MetaMask UI allows users to import an existing Ethereum account using a private key. By entering the corresponding private key, users can access or restore a wallet that has already been generated. The user goes to the "Import Account" section of the MetaMask browser extension, chooses "Private Key" as the import method, and then enters the private key in the input field. After submission, MetaMask safely adds the account to the wallet, allowing users to utilize it to communicate with decentralized appss. For developers or voters who need to access test accounts or pre-funded wallets for blockchain testing or voting system simulations on platforms such as Ganache, this feature is especially helpful.



**Figure 6:** Candidate Voting Interface

The interface that shows a registered voter's information and provides the administrator with options to approve or confirm the voter's eligibility is shown in Figure 7. The interface displays important data that was provided throughout the registration process, including the voter's name, phone number, and Ethereum wallet address. Based on a predetermined eligibility catalogue, the administrator examines this data and applies the verification controls to approve or reject the voter. After being accepted, the voter's address is added to the whitelist, allowing them to cast a ballot. This verification process ensures that only valid users can proceed to cast their ballots, safeguarding the system's integrity and avoiding illegal access or duplicate registrations within the decentralized voting application.

The screenshot shows the ADMIN interface with a navigation bar containing 'Verification', 'Add Candidate', 'Registration', 'Voting', and 'Results'. The 'Verification' section is active, displaying 'Total Voters: 8' and a 'List of registered voters' button. Below this is a table with the following data:

Account address	0x60fC7AeE4D1a22F9f74193E69C13C70524C5a5F1
Name	yashu
Phone	8765894367
Voted	False
Verified	False
Registered	True

At the bottom of the table is an 'Approve' button.

Figure 7: Voter Verification Panel

The election candidates are shown on the user interface of a decentralized voting application, usually in an organized and aesthetically pleasing arrangement. Every candidate is listed with pertinent information, including their name, party affiliation (if any), and a **"Vote"** button next to or beneath their details. With this configuration, registered voters can quickly go over their options and cast their ballot by just pressing the appropriate button. To provide immutability and transparency, a smart contract function is triggered when a vote is submitted, recording the vote on the blockchain. Web3.js is used to refresh the UI in real-time, reflecting the vote progress and transaction status. The voting process on the decentralized platform is guaranteed to be smooth, safe, and easy to use thanks to its design.

The screenshot shows the ADMIN interface with a navigation bar containing 'Verification', 'Add Candidate', 'Registration', 'Voting', and 'Results'. The 'Voting' section is active, displaying 'Your Account: 0xEFC58eE139472562fD4800584756e9d1A916d4C6'. Below this is the 'Election Title' section, which includes 'Organization Title', 'Admin' (Admin Name (Admin Title)), and 'Contact' (admin@example.com). At the bottom, there is a button labeled 'End'.

Figure 8: Election Status

The election summary interface, shown in Figure 8, shows key election information such as the election title, the total number of registered candidates and voters, and the election's current state. Administrators can access this portal, which offers a centralized overview for real-time voting event monitoring. It also provides the administrator's contact details so that voters or interested parties can get in touch if they need help or clarification. To show the present stage, the election status such as "Ongoing," "Upcoming," or "Closed" is conspicuously shown. In addition to assisting the administrator in effectively managing and supervising the election workflow within the decentralized voting platform, this feature guarantees openness and unambiguous communication throughout the election cycle.

#### **4.1 Blockchain Interaction**

An essential part of the electronic voting system is the blockchain connectivity layer, which facilitates communication between the decentralized Ethereum blockchain and the user-facing frontend. Web3.js, a robust JavaScript package that offers an extensive range of features for communicating with Ethereum nodes, was utilized to do this. This enabled the application to track blockchain events, send transactions, and read data from smart contracts. Every step of the voting process was made seamless and dependable by Web3.js, which acted as a bridge between the Ethereum blockchain and the client-side application.

MetaMask was incorporated into the system to enable safe user engagement with the blockchain. Users can authenticate and sign blockchain transactions using MetaMask, a browser-based cryptocurrency wallet and Ethereum account management, without disclosing their private keys to the program. This improved security and usability by removing the need for users to manually enter credentials or manage private key files. MetaMask was set up to connect to Ganache, a local blockchain simulator that replicates the Ethereum network's behaviour, for development and functional testing. This gave developers complete control over the test environment and sped up transaction confirmations.

#### **4.2 Front-end User Interface**

React.js, a well-liked open-source JavaScript toolkit renowned for its component-based architecture and excellent responsiveness, was used to develop the e-voting platform's frontend. Given the significance of accessibility in public voting systems, the user interface was essential to guaranteeing usability and a seamless user experience. Voter registration, candidate listings, voting interfaces, and real-time result updates were among its essential features. Because these elements were designed to be less complicated, persons with less technical expertise may comfortably take part in the voting process.

Every vote a user cast using the React interface caused the deployed smart contract to perform a matching function. On the blockchain, these functions were in charge of registering the vote. Updates were reflected instantly since Web3.js was used to connect the frontend to the blockchain directly. The updated vote total, for instance, might be obtained and shown to the user practically immediately after a vote was made. One of the fundamental needs of a reliable and efficient voting platform is transparency, which was greatly enhanced by the system's high degree of responsiveness and interaction.

#### **4.3 Storage of Off-Chain Data**

It was not practical to store personal data directly on the blockchain because of its bulk and sensitive nature, such as biometrics and Know Your Customer documentation. Rather, the system used a mixed on-chain/off-chain architecture, with Interplanetary File System being used to store sensitive data off-chain. IPFS is a distributed, peer-to-peer file storage protocol that ensures data is stored in a decentralized and tamper-proof manner.



Any sensitive data was encrypted using the SHA-256 hashing algorithm, which created a distinct and irreversible hash of the data, prior to being stored on IPFS. The only data kept on the blockchain was this hash, which acted as a safe link to the original data. By doing this, the system avoided the high cost and storage limitations that come with directly on-chaining huge files while still guaranteeing data privacy, immutability, and traceability. This technique helped maintain compliance with data protection regulations and provided a layer of privacy preservation by limiting direct access to personal data via the blockchain.

#### **4.4 Testing and Validation**

Ganache, which offered a regulated Ethereum-like environment for mimicking real-world interactions, was used for extensive testing to confirm the accuracy and dependability of the system. In this setting, the complete electronic voting process from voter registration to voting and final result counting was tested. Ganache made it simple to deploy smart contracts, provide quick feedback, and inspect transactions all of which were essential for finding and fixing errors early in the development cycle.

A number of crucial characteristics were validated throughout the testing phase: each registered Ethereum address could only cast one vote, avoiding fraudulent manipulation or double voting. On the blockchain, every transaction including casting a vote was permanently documented. In order to make sure the system remained stable and resilient under pressure, it was also tested in boundary scenarios such simultaneous voting, unauthorized account access, and failure recovery. Before the program was put into use in the actual world, these stringent testing procedures improved its dependability and credibility.

#### **4.5 Ethereum Test net Deployment**

The application was uploaded to the Ethereum testnet for real-world testing under more realistic network conditions after a successful local validation. Users could interact with the program using actual Ethereum wallets and smart contracts maintained on a public blockchain in this deployment, which mimicked an actual election scenario. Without centralized credentials, users were able to sign transactions and verify their identities thanks to MetaMask, which acted as the safe authentication method.

Through a React-based web interface, users were able to access the platform, log in with their MetaMask wallets, and engage directly with the deployed smart contracts to cast their votes. In addition to demonstrating the application's preparedness for mainnet deployment, this real-world testnet deployment assisted in identifying edge cases that were not visible in the local testing environment. It also demonstrated how the platform may operate without the need for centralized servers or middlemen in decentralized, trustless ecosystems.

#### **4.6 Data Integrity, Security, and Privacy**

The design of the system was based on fundamental security and privacy principles. Once hashed, sensitive data could not be altered or reverse-engineered thanks to the usage of SHA-256 encryption. Furthermore, data integrity was automatically maintained due to the decentralized nature of both IPFS and the Ethereum blockchain; any effort to modify stored data would cause a discrepancy with its hash, instantly revealing the tampering.

Furthermore, by making sure that individual votes could not be connected to voter identities, the system preserved voter anonymity. The blockchain only displayed hashed references to vote transactions and user data. This approach preserved voter anonymity without sacrificing the election's auditability. The blockchain made every vote and transaction publicly verifiable, increasing system transparency and enabling independent third-party audits to confirm election results.

#### **4.7 Trust and User Experience**

To gain users' trust, the program placed a strong emphasis on usability and transparency. Voters may interact with the platform with confidence thanks to the combination of safe backend features and an easy-to-use UI. The ecosystem created by the open-source smart contracts, tamper-proof voting records, and real-time feedback gave users confidence that their votes were cast fairly and properly.

Additionally, the system's decentralized structure decreased dependence on any one central authority, making it impervious to internal manipulation, censorship, and single points of failure. Interoperability and future scalability were further guaranteed by the use of open standards and widely used technologies such as Web3.js, Ethereum, IPFS, and MetaMask. When combined, these characteristics showed that the system might operate as a blockchain-based electronic voting platform that is transparent, safe, and scalable.

#### **5. Conclusion**

This article suggests utilizing Ethereum blockchain technology to create a scalable, transparent, and safe electronic voting system. The solution guarantees real-time vote tracking, tamper-proof records, and privacy-preserving voter registration by combining smart contracts, React.js, MetaMask-based authentication, and IPFS for off-chain storage. Decentralization eliminates single points of failure and increases voter trust, while SHA-256 encryption improves data integrity. The system's dependability and preparedness are confirmed by functional testing on Ganache and deployment on the Ethereum test net. The popular programming tools and modular architecture provide a useful basis for practical application. Although there is still room for improvement in terms of scalability, anonymity, and interaction with national ID systems, the system shows how blockchain has the potential to completely transform electoral processes by enabling safe, verifiable, and decentralized voting.

##### **5.1 Summary of the Proposed Mechanism**

The Ethereum blockchain is used by the suggested electronic voting system to guarantee a safe, open, and impenetrable election process. Solidity-written smart contracts ensure immutability and one-person-one-vote enforcement by managing essential operations including voter registration, candidate addition, and vote casting. Users may safely authenticate and sign transactions using the frontend, which was created with React.js and communicates with the blockchain via Web3.js and MetaMask. Only the hash is kept on-chain; sensitive data is hashed using SHA-256 and stored off-chain using IPFS to preserve voter privacy. To verify functionality in real-time, the system was deployed on the Ethereum testnet and evaluated on Ganache. This architecture is appropriate for safe online elections since it promotes decentralization, voter anonymity, and verifiability.

##### **5.2 Limitations**

Despite its functionality and secure design, the suggested electronic voting method has a number of drawbacks. First of all, because public blockchain networks may have high gas prices and transaction delays, it is not scalable for large-scale elections. Second, privacy protection during vote tallying is limited due to the lack of use of sophisticated cryptographic techniques such homomorphic encryption and zero-knowledge proofs. Thirdly, voters who are not technical or live in remote areas may find the system less accessible due to its reliance on MetaMask and internet connectivity. Fourthly, possible vulnerabilities have not been addressed since smart contracts have not been the subject of rigorous security audits. Finally, voter identity verification is restricted by the lack of biometric or official ID integration. These limitations imply that, despite its potential, the system requires additional work

before it can be used effectively in national elections.

### **5.3 Future Research Directions**

The suggested e-voting system will be improved in the future to increase scalability, privacy, and practicality. For large-scale elections, integrating Layer-2 blockchain solutions like Polygon or zk-Rollups can greatly save gas costs and enhance performance. Advanced cryptographic methods like homomorphic encryption and zero-knowledge proofs can be used to increase privacy. Secure voter verification can be achieved through integration with national identity platforms such as Digi Locker or Aadhaar. Multi-factor login and biometric authentication can enhance identity validation even more. To confirm the security of smart contracts, formal auditing techniques such as Myth X or Slither should be utilized. Furthermore, the system will become more transparent, accessible, and appropriate for both domestic and international deployment by incorporating energy-efficient consensus techniques like Proof-of-Stake, enabling offline and mobile voting, and incorporating real-time analytics dashboards.

### **References**

- [1] A. H. M. Alam, R. Arshad, and R. Akhtar, "Decentralized Voting Platform Using Blockchain and Smart Contracts," *International Journal of Computer Applications*, vol. 182, no. 38, 2019.
- [2] Ricardo Lopes Almeida, Fabrizio Baiardi, "Impact of Decentralization on Electronic Voting Systems: A Systematic Literature Survey", *IEEE Access*, 2023.
- [3] Kateryna Isirova, Oleksandr Potii, "Development Principles for Electronic Voting System Using Distributed Ledger Technology", *IEEE*, 2020.
- [4] Albin Benny, Aparna Ashok Kumar, "Blockchain Based Electronic Voting System", *IEEE Access*, 2023
- [5] M. N. Koenig and D. Song, "Indisputable Voting: Privacy-Preserving Verifiable Elections on the Blockchain," *IEEE Security & Privacy*, vol. 17, no. 4, pp. 42–51, 2019.
- [6] C. B. Gritzalis, "Secure Electronic Voting," *Advances in Information Security*, vol. 7, Springer, 2003, pp. 165–180.
- [7] M. T. Alam, I. Khalil, and X. Yu, "A Secure Data Sharing Scheme Using Blockchain and Cloud Computing," *Future Generation Computer Systems*, vol. 105, pp. 255–269, 2020.
- [8] A. Sharma, M. Dave, and S. Saxena, "Blockchain Technology for Secure E-Voting System," *Procedia Computer Science*, vol. 132, pp. 851–856, 2018.
- [9] J. Ayed, "A Conceptual Secure Blockchain-Based Electronic Voting System," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 9, no. 3, pp. 01–09, 2017.
- [10] Truffle: <https://truffleframework.com>
- [11] Ethereum project: <https://ethereum.org>
- [12] Ganache: <https://truffleframework.com/ganache>
- [13] <https://www.trufflesuite.com/docs/ganache/overview>
- [14] M. Gupta, A. Negi, and R. Kumar, "Securing IoT Networks Using Blockchain: A Survey," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, 2020, pp. 512–517, doi: 10.1109/ICESC48915.2020.9155918.
- [15] Y. Zhang, Y. Li, L. Fang, P. Chen, and X. Dong, "Privacy-protected Electronic Voting System Based on Blockchain and Trusted Execution Environment," 2020, doi: 10.1109/iccc47050.2019.9064387.
- [16] M. Glaser and D. Bez, "Online Voting with Ethereum Blockchain in Practice," *Journal of Information Security Research*, vol. 1, no. 1, 2019.
- [17] A Dasgupta, S. Bose, and S. Roy, "Security Vulnerabilities in IoT Devices and Proposed Solutions Using Blockchain," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9282–9293, 2019.