International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 01 | Jan 2023 www.irjet.net p-ISSN: 2395-0072

# Online Voting System Using Blockchain Technology

**Sandeepkumar Chauhan[1], Sanjeev Kumar Mandal[2], Santanu Jha[3], Prashant Koparde[4]**

*[1,2,3]Students, Department of Computer Science & Engineering, T John Institute of Technology, Bengaluru, India*
*[4]Asst. Professor, Department of Computer Science & Engineering, T John Institute of Technology, Bengaluru, India*

---

**Abstract -** *Voting is a complex process with a lot depending on it. Building an e-voting system that can guarantee anonymity, verifiability, and transparency together is a challenging task. Continuous efforts are being made to improve the voting system to achieve these properties. Recently, blockchain has hit the technology space with many promises, especially to make verifiable and transparent decentralized systems. However, a major challenge faced with blockchain-based e-voting systems is to achieve the users' anonymity while ensuring only authorized voters should be able to vote, and that too only once. To address these issues, this paper proposes a blockchain-based e-voting system with secret contracts. We have used Enigma (a secure multiparty computation platform) to design secret contracts. The proposed system meets most of the features required to conduct free and fair voting electronically.*

***Key Words***: **Online Voting System Using Blockchain technology, Decentralized Voting System, Blockchain Technology Based Project, A Secured Balloting Voting System Using Blockchain Technology.**

## 1. INTRODUCTION

India is a democratic country and has a democratic country. As now all Indian citizen become a part of the growing digital India with a digital ID that is Aadhaar card. Voting schemes have evolved from counting hands in early days to systems that include paper, punch card and electronic voting machine.

The idea behind an e-voting system based on blockchain is similar to digital wallets. The system or authority can issue a digital wallet to each participant after verifying its identity. The wallet issued must contain the user credentials and a single coin representing a single chance to vote When a user selects a candidate and casts its vote, the coin in the user's wallet is transferred to candidate's account or wallet. At the end the number of coins in each candidate's wallet represents the number of votes cast to him. Electronic and online voting system can provide more security and integrity than EVM. It also provides privacy to the users; as the eligible voters can cast their votes using their computers or even smartphones, thus maintaining anonymity. It also builds trust on the system as it is online and completely an open system. It can also increase the number of participants' involvement.

### 1.1 MOTIVATION

Our motivation here is to provide a platform for people where they are able to completely trust the system, carry out the transactions and not be worried about miscounting of their votes, their choice and their decision remains unaltered and thus cannot be manipulated in any way. The platform that we are going to provide can be accessed from any device running the blockchain node such as computers, servers, embedded systems or even mobile devices in the near future.

### 1.2 AIM

Our aim is to design and develop a software platforms online voting system based on blockchain technology which helps to securely conduct votes and elections. As a digital platform, they eliminate the need to cast the votes using paper or having to gather in person. They also protect the integrity of your vote by preventing voters from being able to vote multiple times. E-voting has fundamental benefits over paper based systems such as increased efficiency and reduced errors. The electronic voting system tends to maximize user participation, by allowing them to vote from anywhere and from any device that has an internet connection. The blockchain is an emerging, decentralized, and distributed technology with strong cryptographic foundations that promises to improve different aspects of many industries. Expanding e-voting into blockchain technology could be the solution to alleviate the present concerns in e-voting. Here we propose a blockchain-based voting system that will limit the voting fraud and make the voting process simple, secure and efficient.

## 2. LITERATURE SURVEY

Currently increasing digital technology helped many people lives. In contrast to the electoral system, there are many conventional uses of paper in its implementation. The aspect of security and transparency is a threat from still widespread election with the conventional system .Block chain technology is one of solutions, because it embraces a decentralized system and the entire database are owned by many users. There is no doubt that the revolutionary concept of the blockchain, which is the underlying technology behind the famous crypto currency Bit coin and its successors, is triggering the start of a new era in the Internet and the online services. In this work, we have implemented and tested a sample e-voting application as a smart contract for

the Ethereum network using the Ethereum wallets and the Solidity language. Block chain was first introduced by Satoshi Nakamoto (a pseudonym), who proposed a peer to-peer payment system that allows cash transactions through the Internet without relying on trust or the need for a financial institution. Block chain is secure by design, and an example of a system with a high byzantine failure tolerance. E-voting is a potential solution to the lack of interest in voting amongst the young tech savvy population. For e-voting to become more open, transparent, and independently auditable, a potential solution would be base it on block chain technology. Block chain technology has a lot of promise; however, in its current state it might not reach its full potential. Electronic voting has been used in varying forms since 1970s with fundamental benefits. over paper based systems such as increased efficiency and reduced errors. With the extraordinary growth in the use of block chain technologies, a number of initiatives have been made to explore the feasibility of using block chain to aid an effective solution to e-voting. It presented one such effort which leverages benefits of block chain such as cryptographic foundations and transparency to achieve an effective solution to e-voting. The proposed approach has been implemented with Multichain and in-depth evaluation of approach highlights its effectiveness with respect to achieving fundamental requirements for an e-voting scheme. Recent major technical challenges relating to e-voting systems embrace, however not restricted to secure digital identity management.

## 3. BACKGROUND

### A. Ethereum Blockchain

Ethereum [3][6] is an open-source, blockchain-based platform. that helps build decentralized applications (Dapps). It features scripting functionalities. These scripting features - called smart contracts [3] - are logic that operates on the network of computers in the blockchain and is responsible for the interaction of Dapp with blockchain in the form of transactions. By definition, Blockchain is a public ledger, so achieving data privacy using blockchain in its current form is not possible. Data privacy, however, is crucial for any application looking to gain its user's trust, and hence the full potential of blockchain and, in extension, decentralization could not be harnessed. That was the case until Guy Zyskind (now CEO of ENIGMA) introduced the idea of the off-chain storage in [7] and then extended it to off-chain computation in Enigma protocol [8] and quickly became the foundation stone for all further talk on the topic of decentralization and privacy. Enigma is a decentralized computation platform with guaranteed privacy. It aims to enable end-to-end decentralized applications to be built without any need for a trusted third party. In [7], tasks functions are divided into two types, off chain, handling private data and computations, and on-chain for consensus (validation).

### B. Enigma

Enigma is designed to work over an existing blockchain by adding an off-chain network of multiple nodes called secret nodes where private computations/storage on/of data is offloaded. Each node must have in-built support for TEE (Trusted Execution environments); otherwise, nodes won't be allowed to become a part of the network. TEE technology (enigma relies on intel's SGX) is secure hardware that decrypts sensitive data inside a defined private region of memory called enclaves and whose contents are protected and can not be read or accessed by any program or entity outside the enclave, including processes with higher privilege levels and even the owner of the hardware. These TEE-compatible nodes, along with multi-party computation (MPC), help in privacy preserving computations. Enigma uses MPC as its computation model. Threshold cryptosystems are normally used here, with the general one being Shamir's secret sharing. Secret sharing is where a secret is shared among n users, and t+1 is required to reconstruct the secret. Any subset of t and nothing can be learned about the secret. The Enigma network cryptographically guarantees privacy by encrypting data before it leaves the user's premises. When any data input into the network, it is encrypted client-side. The data is decrypted inside the enclave, which does not reveal anything outside, preserving secrecy. Finally, results are encrypted too. Enigma also extends the functionality of smart contracts by introducing secret contracts. The biggest benefit of this was the secret state, that is, inputs and outputs, but now the state can also be secretly maintained, and thus it is not accessible to network participants. This helps provide at least the same functionality as Ethereum smart contracts, probably much more. Secret contracts in enigma are written in Rust and compiled into Web Assembly (WASM) bytecode. Rust is an excellent programming language that simultaneously optimizes for both safety and performance. Also, rust makes it easy to use machine learning components and perform other complex calculations in contracts. Secret contracts are interoperable with Ethereum, which means Ethereum developers can continue to build on Ethereum and use the Enigma network for secret computations. A secret contract is broken down by an interpreter, where the parts dealing with private data are sent to the enigma network, and the parts dealing with public data are sent to the blockchain network. Enigma working can be explained as below: When a user enters a task to be computed by the enigma network, a randomly-selected node picks up the inputs and propagates the encrypted inputs to a group of secret nodes working on that particular secret contract. The selected secret nodes can decrypt the inputs inside their enclaves, compute on the data, and reach a simple consensus on the result. Once this consensus is reached, the secret contract's state is updated, and Ethereum smart contracts may be called if required. The key management module ensures that a random secret node can compute an encrypted input sent by the user. This is enabled by using

the Diffie Hellman scheme, where keys are exchanged between the user client and workers in the network. Along with the above, enigma introduces scalability. In a traditional blockchain, every node executes the same code and maintains the same public state, causing quite a lot of redundancy and also delay when accepting a new block to the chain. However, in an enigma, the computation is efficiently. distributed across the subset of secret nodes that computes and gives back results. As only a subset of nodes takes part in a particular computation, the network is more scalable.

## 4. RELATED WORKS

In [9], the Author proposed an improvisation on FOO voting scheme [10] using smart contracts and hyper ledger fabric-based blockchain network by addressing the limitation or weakness in existing FOO voting system like third party involvement being replaced with smart contracts and others. The authors of [11] proposed a blockchain-based framework where verification is done through a central party. To guarantee anonymity to the voters, the system uses NIZKP for wallet generation, but there is no clear specification of how exactly NIZKP will be integrated. They also ask that the voters trust the third party for authentication/verification and wallet generation, which is a deal-breaker. In [12], the Author proposed a secure and reliable e voting system based on smart contracts and an Ethereum-based private blockchain network. This system enables all users to view the voting data, and there is no involvement of any third party. The proposed system maintains the required properties like voter's anonymity, prevents re-voting, etc. In [13], the Authors proposed a secure end-to-end verifiable Internet-voting system based on an identity-based blind signature scheme using certain features bio-metric features and voter's unique identification. However, the authors used elliptic curve discrete logarithm to achieve privacy of the proposed system. The system enables users to verify the voter's cast vote, and even each voter can verify the counting process of recorded ballots correctly. The author showed that the proposed system takes the least machine cycles while comparing with some other works. In [4], the authors proposed an e-voting system based on blockchain as it is considered a robust consensus mechanism and provides immutability, transparency. In addition to that, the Authors incorporated homomorphic encryption schemes to protect the privacy of the voters. Casted Votes will be stored onto the blockchain in the encrypted form by performing homomorphic encryption so that there is no need to perform decryption to count the votes. In addition to the properties provided by the blockchain, the Authors designed a self-tallying e-voting system in paper [14] where anyone can verify the ballot's validity and tally the ballots without any central authorities and any third parties. It is achieved by generating and distributing the random numbers for the ballot security. These random numbers allow counting the ballots as these numbers will be cancelled out when multiplying all ballots. This scheme's limitation is that all voters have to cast their votes; otherwise, the system would not work. In [15], the authors suggest using a private blockchain instead of a public blockchain. The private blockchain would then have to maintain a sizable number of peers in the system to achieve transparency. The system also uses a Central authority for identity verification. In this paper, a voter would send a digital commitment of his vote to the central authority along with his public key after applying a blinding function. They propose that the blinding function will hide a person's. vote from parties involved. Still, the issue persists as a central authority can link a person's identity to his/her public key.

## 5. PROPOSED SYSTEM

The protocol is designed while keeping decentralization in mind as the primary goal. Besides that, we give high consideration to maintaining the voters' anonymity, votes even during the tallying phase, and verifiability of the tally. Earlier works use a central authority as a trusted third party to maintain the secrecy of voters' identities whilst allowing only eligible voters to participate. On the other hand, our proposal transfers this role of the central authority to a secret contract so that the system is more under the control of peers rather than one single organization. Before providing a high-level description of our proposed solution, we first describe some protocol terms.

**A. Roles and Entities Wallet**: Represents a digital wallet and is referenced by a unique address on the blockchain.

**Vote**: A vote is represented by a token in the system. A vote casting to a candidate transfers the token from the voter's wallet to the candidate's wallet.

**Voter**: Voter is a user who wants to cast a vote. Until and unless mentioned otherwise, a voter refers to an eligible voter only. Voters can be identified by the system in two ways. Using secret contracts, voters interact using encryption of his government-issued ID, whereas, with smart contracts, they can only interact using their blockchain addresses.

**Election Authority**: An authority which holds the responsibility of deploying contracts, registering candidates, starting elections, and ending it. Every action of this authority is now under public scrutiny. Candidate: the contestant of election registered by election authority. They have already been vetted offline by due process.

### B. System Design

We divide the process into four phases, each of which is detailed below

**1) Initialization phase**: In this phase, election authorities deploy both smart contract and secret contract, separately announce an election start date and end date, and host an updated android application. A voter can interact with the system only through this updated application as this contains he new contract addresses. Secret contracts are an enhanced version of smart contracts and are designed specifically for privacy because they do not reveal the input data they interact with. In this phase, the election authority also uploads a list of valid candidates' government-issued ID numbers along with appropriate verification credentials. We assume that the credentials have been shared with the voters before. This list is a part of the secret contract state and is only accessible by secret contract nodes within their enclaves. The secret

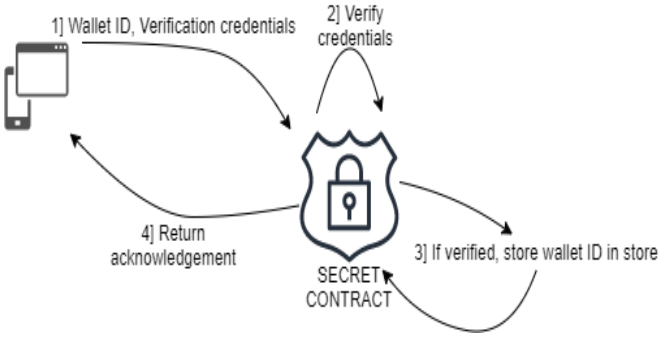


Fig. 1: Verification Phase

contracts use this list to verify the identity of voters during the verification phase. Storing IDs in a secret state makes the secret contract self-reliant as it does not need to contact any other third party DB after this step. For scalability, we suggest a separate set of contracts for each district, but we intend to discuss and outline our system with respect to one district, and the same process. can be followed for any other. We also assume that each voter has an Ethereum account needed to interact with the system. Ethereum accounts are required only because we are basing this project on the Ethereum blockchain. If any other blockchain was used, it accordingly could be integrated.

**2) Verification Phase**: This phase begins as soon as the initialization phase is over. In this phase, we expect eligible voters to download the app and register themselves. Internally the application flow is as follows: voters put a registration request through an app on their device using their ID numbers and verification credentials (maybe fingerprint or some other secret question as necessitated by the system). As shown in figure 1, a request is passed to the secret contract with the above information and the voter's blockchain wallet address, which is to be created by the voter beforehand. The ID and other personal information help secret contracts validate the voter who will verify the. voter. For the time being, this verification is a simple matching of ID numbers in the secret contract enclaves, but as more power is added to secret contracts, a more robust verification protocol can be introduced.

Secret contract stores only the wallet addresses of voters, which have been verified to be used in the next phase. These wallet addresses should be stored in a list separate from the government ID's list. Thus there is no way to establish a link between the two. Also, from time to time, we can randomize the list to maintain secrecy.

This phase has one additional task. The election authority registers valid candidates through a smart contract through an app we made separately for election authority. Internally this means creating a wallet address corresponding to the candidate and storing it corresponding to the candidate ID in a smart contract.

**3) Wallet Initialization Phase:** This phase begins after the previous phase is over according to the timeline set by the election authorities. When the election authority indicates the secret contract that the verification phase is over, as shown in figure 2, it triggers a call to the smart contract to pass the list of stored wallet IDs. This function contacts the smart contract's "register Voter" function with verified

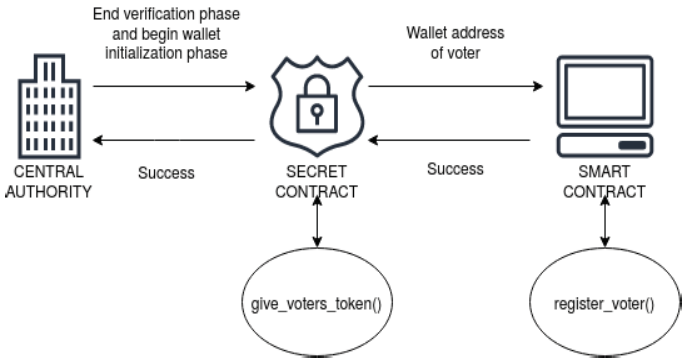


Fig. 2: Wallet Initialization

voters' addresses, which were stored in the previous phase. In this function of the smart contract, the wallet represented by each address will be credited with one token. Each token will represent one vote in our system. We have separated this phase from the Verification phase to ensure a timeline cannot be built between voter registration and transaction with the smart contract.

**4) Voting Phase:** This is the final phase of the system and is performed on the Ethereum blockchain. The start of this phase is indicated by the election authority calling a function "set start election" of a smart contract through their app.

Once the election has started, registered voters can vote by transacting their tokens to their chosen candidate's wallet. As per Ethereum flow, the transaction will generate a transaction ID that the voters can use to ensure that their votes have been correctly cast. Thus providing verifiability.

At the end of the voting cycle, each candidate's wallet balance can be checked, and the candidate with the highest tally wins. Results can be verified easily as all transaction details are stored in the public blockchain. Application flow is as follows: Each voter using his wallet will transact a token to a candidate's wallet (the interface will be handled by app automatically). The transaction of a token represents casting of a vote. A voter will get a transaction ID which can be used to verify that his vote has been correctly casted. After the voting phase, the candidate's wallet balance will show the final tally. Tally of all candidates can now be compared to decide and set the winner.

### C. Protocol Analysis

In the following, we explain how each required property is achieved by the proposed protocol.

**1) Eligibility**: While deploying a secret contract, the election authority initializes it with a list of encrypted ID's of only eligible voters. So while executing a secret contract will not verify any person whose ID could not be verified with a list.

**2) No multiple voting fraud**: Secret contract maintains a state variable which stores if an ID was previously verified. If so, it means some address has already been stored corresponding to it. Thus only one address at max can be there for one ID. Also, When the wallet initialization phase begins smart contract ensures that only one token is granted to an address. Thus a voter can have only one token resulting in him only being able to vote once. Also, blockchain itself makes sure. that double spending has not been done.

**3) Anonymous voting**: As states of the secret contract are private and also incoming and outgoing data is encrypted. This perfectly helps us achieve no identifiable link between a person and his wallet and thus with his vote.

4) Verification: Smart contracts store a state of which address voted to which candidate using that a voter can easily verify his vote. The voter has a transaction number stored by an event on the blockchain, which can be queried to get data from the blockchain and verify. Also, following the trail of transactions from the time since the voting phase began, one can easily verify the tally independently on his own. Also, the design of the blockchain itself makes the system tamper-proof.

**5) Decentralization**: Blockchain being the center of the whole application clearly achieves decentralization to the system.

**6) Consensus**: The final result is the consensus of all the voters.

## 6. IMPLEMENTATION

The system activity diagram, which shows the interaction among phases mentioned in the previous section, can be seen in figure 3. We are using a Ropston Ethereum, a testnet version of Ethereum-based blockchain. It provides the same environment as the mainnet of blockchain; only it does not cost money. It is a public blockchain, and it uses the Proof of Work (PoW) Algorithm. Access to this testnet can be achieved through Remix online IDE, which is used to create, compile, deploy, and interact with smart contracts. It needs a wallet like meta mask to support transactions with smart contracts. We have tested our smart contract using this IDE with the Ropsten Testnet and test ether. The secret contract idea is to set up a local enigma network which comes with its own local blockchain. A secret contract written in rust and a smart contract are deployed on this network, and android applications are used to interact with these contracts. The secret contract is intended to perform the following functions

- Register a Voter - Verify uniqueness and identity of a voter.
- Start wallet initialization when the end of this phase is marked.

Algorithm 1: verification-voter verification

```
Procedure verify id using list(encrypted ID proofs)
Result: True if valid else false
return list[encrypted ID proofs]
Procedure register a voter( wallet address,
encrypted ID proofs)
require(verification phase end == false)
res=verify id using list(); //this can be replaced by a
more sophisticated later
if res == true then
add address to eligible voters list;
else
;
end
```

Algorithm 2: Wallet Initalization

```
Procedure give voters token ()
    require (verificationphaseend == true)
    for address in eligible voter list
    register voter(address,1); // register voter is a
smart
  contract function
   Procedure end verification phase()
  require( msg.sender == owner)
require (verification phase end == false)
verification phase end == true
give voters token();
```

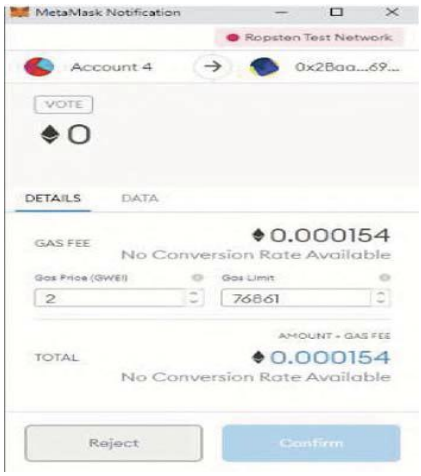


Fig.3: Metamask transaction confirmation

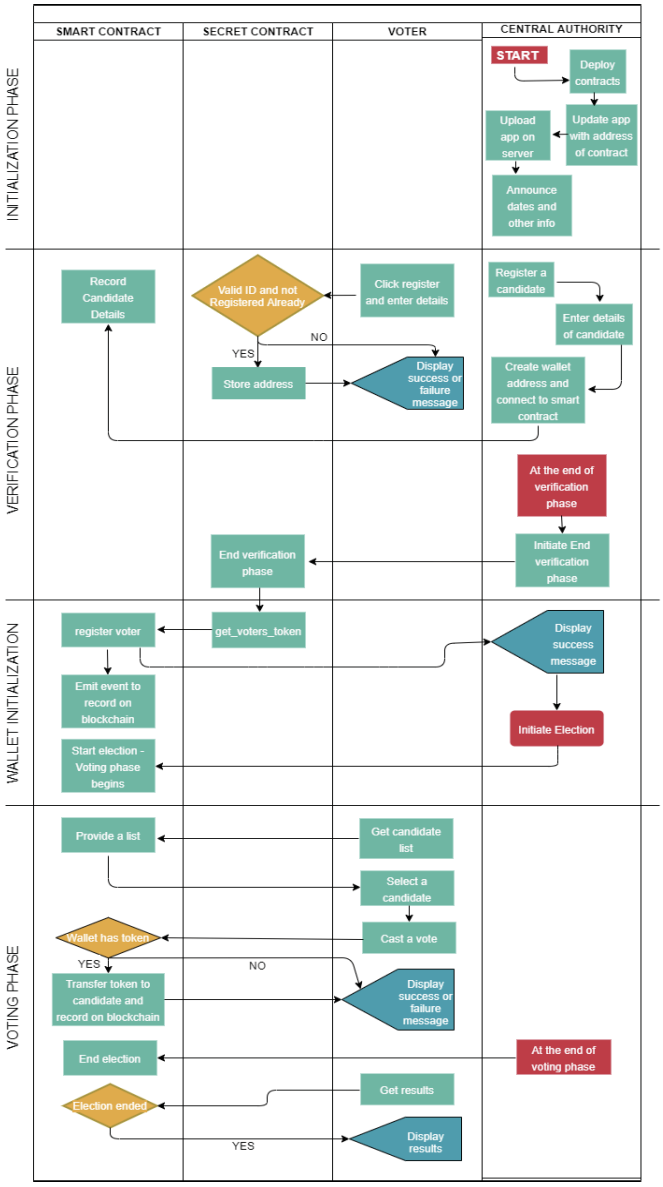


Fig. 4: System Activity Diagram

Also the Smart contract provides the following functionality to support different phases of the election process.

• Register a Candidate (The call to this can only go through owner of contract)

• Transfer of token to voter (The call to this can only go through secret contract)

• Start election (only owner of contract - election commission in this case - can call this)

• Vote Casting (As a transfer of token to a registered candidate only)

• End election (only owner of contract – election commission in this case - can call this)

• Show results and verify Vote

Algorithm 3, 4, and 5 provide pseudo-code for Candidate registration, token transfer to voter and vote casting respectively.

Algorithm 3: verification - Candidate registration

```
Procedure Register candidate(address , name );
  require( msg.sender == owner)
   for address in eligible voter list
  require(electionstart = false and electionend = false)
 add candidate details to state variable and also set
  corresponding balance = 0 //balance dictates
  votecount
```

Algorithm 4: Wallet initialization

```
Procedure register voter( address ,tokens );
      require( msg.sender == secret contract)
       require(electionstart = false and electionend = false)
       require(tokens==1)
       balance[owner]=balance[owner] - 1
       balance[address]=balance[address] + 1
       emit transfer event //to record data on blockchain
```

Algorithm 5: Voting Phase

```
Procedure cast a vote( name, address of candidate,
   tokens );
    require(electionstart=true and electionend=false)
    require(tokens==1)
    balance[msg.sender]=balance[msg.sender] – 1
   balance[addressofcandidate] = balance[address] + 1
   emit transfer event //to record data on blockchain
```

Two applications have been created. One provides a user interface for election commission and another one for Voters.

• Election commission can register a candidate, start or end an election, and start the wallet initialization phase.

• Voter gets to perform following function:
– Register with the secret contract to get a token.
– Check registered candidates.
– Cast their vote.
– Check and verify results.

To integrate smart contract with android web3j library is used to first create wrapper java classes for smart contract and then functions of this class are called to interact with blockchain. Also infura is used to act as ethereum client. It is a cloud based service that provides access to blockchain services for devices which cannot run their own nodes.

## 7. CONCLUSION

This paper introduces a blockchain-based framework and protocol to perform e-Voting in a distributed manner. Along with the blockchain, the proposed framework uses secret contracts to address the problems of maintaining voter anonymity and the confidentiality of the vote cast by him. The proposed framework uses a secure multiparty computation platform Engima to design secret contracts. The protocol also focuses on decentralizing verification in a secure manner, which means that the central authority can not trace a vote back to the voter even if wanted to. Protocol analysis shows that the proposed framework and the protocol achieve most of the requirements to conduct free and fair elections.

## 7. ACKNOWLEDGEMENT

We thank, Dr. Thomas P John (Chairman), Dr. Suresh Venugopal P (Principal), Dr Srinivasa H P (Vice-principal), Ms. Suma R (HOD – CSE Department), Dr. John T Mesia Dhas (Associate Professor & Project Coordinator), Mr. Prashant Koparde. (Assistant Professor & Project Guide), Teaching & Non-Teaching Staffs of T. John Institute of Technology, Bengaluru – 560083.

## REFERENCES

[1] Y. Abuidris, R. Kumar, and W. Wenyong, "A survey of blockchain based on e-voting systems," in Proceedings of the 2019 2nd International Conference on Blockchain Technology and Applications, pp. 99–104, 2019.

[2] K. M. Khan, J. Arshad, and M. M. Khan, "Investigating performance constraints for blockchain based secure e-voting system," Future Generation Computer Systems, vol. 105, pp. 13–26, 2020.

[3] B. Singhal, G. Dhameja, and P. S. Panda, Beginning Blockchain: A Beginner's Guide to Building Blockchain Solutions. Springer, 2018.

[4] G. M. C. Sravani, "Secure electronic voting using blockchain and homomorphic encryption," International Journal of Recent Technology and Engineering (IJRTE), vol. 8, 2019.

[5] "Enigma Protocol Overview." https://enigma.co/protocol. Accessed:2020-6-5.

[6] V. Buterin et al., "A next-generation smart contract and decentralized application platform," white paper, vol. 3, no. 37, 2014.

[7] G. Zyskind, O. Nathan, et al., "Decentralizing privacy: Using blockchain to protect personal data," in 2015 IEEE Security and Privacy Workshops, pp. 180–184, IEEE, 2015.

[8] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," arXiv preprint arXiv:1506.03471, 2015.

[9] Y. Zhou, Y. Liu, C. Jiang, and S. Wang, "An improved foo voting scheme using blockchain," International Journal of Information Security, vol. 19, no. 3, pp. 303–310, 2020.

[10] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in International Workshop on the Theory and Application of Cryptographic Techniques, pp. 244–251, Springer, 1992.

[11] F. . Hj´almarsson, G. K. Hreiarsson, M. Hamdaqa, and G. Hj´almt`ysson, "Blockchain-based e-voting system," in 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), pp. 983–986, IEEE, 2018.

[12] H.-D. Park, "A decentralized e-voting system based on blockchain network," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, 2019.
[13] M. Kumar, S. Chand, and C. P. Katti, "A secure end-to-end verifiable internet-voting system using identity-based blind signature," IEEE Systems Journal, vol. 14, no. 2, pp. 2032–2041, 2020.

[14] G. Zeng, M. He, and S. M. Yiu, "A secure and self-tallying e-voting system based on blockchain," in International Workshop on Information Security Applications, pp. 67–76, Springer, 2019.

[15] F. S. Hardwick, A. Gioulis, R. N. Akram, and K. Markantonakis, "Evoting with blockchain: An e-voting protocol with decentralisation and voter privacy," in 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 1561–1567, IEEE, 2018.