TRIBHUWAN UNIVERSITY

INSTITUTE OF ENGINEERING

HIMALAYA COLLEGE OF ENGINEERING

A THIRD YEAR MINOR PROJECT PROPOSAL ON

BLOCKCHAIN BASED E-VOTING SYSTEM

PROJECT MEMBERS

Anil Shrestha (HCE076BEI002)

Sandesh Bhusal (HCE076BEI014)

Yubraj Bashyal (HCE076BEI015)

DEPAREMENT OF ELECTRONICS AND COMPUTER ENGINEERING

HIMALAYA COLLEGE OF ENGINEERNNG

Chyasal, Lalitpur

January, 2023

BLOCKCHAIN BASED E-VOTING SYSTEM

A THIRD YEAR MINOR PROJECT PROPOSAL

“A THIRD MINOR PROJECT

REPORT SUBMITTED FOR PARTIAL FULILLMENT OF THE

DEGREE OF BACHELOR’S IN ELECTRONICS,

COMMUNIATION AND INFORMATION ENGINEERING”

SUBMITTEED TO

HIMALAYA COLLEGE OF ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER

ENGINEERING

Chyasal, Lalitpur

SUBMITTED BY

Anil Shrestha (HCE076BEI002)

Sandesh Bhusal (HCE076BEI014)

Yubraj Bashyal (HCE076BEI015)

January, 2023

ABSTRACT

An electronic voting system based on blockchain technology offers a secure and transparent means of conducting elections. The use of a decentralized, distributed ledger allows for each vote to be recorded and verified in a tamper-evident manner, ensuring the integrity of the electoral process. One potential implementation of a blockchain-based e-voting system involves the use of smart contracts to automate the voting process. Voters can cast their ballots electronically, with their votes being recorded and stored on the blockchain. The blockchain structure can then be used to tally the votes and determine the outcome of the election. In addition to providing security and transparency, a blockchain-based e-voting system also has the potential to increase accessibility and convenience for voters. It allows individuals to cast their votes remotely, eliminating the need to physically go to a polling station. This can be particularly beneficial for individuals with disabilities or those who live in remote areas. Overall, a blockchain-based e-voting system has the potential to revolutionize the way elections are conducted, providing a secure and transparent means of conducting elections that is accessible to all voters.

Keywords: blockchain, hash, node, proof of work

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 BACKGROUND

Elections are a cornerstone of democratic societies, providing citizens with the opportunity to select their leaders and make decisions on important issues. However, the traditional voting process can be prone to issues such as fraud, intimidation, and the logistical challenges of physically going to a polling station. In recent years, there has been increasing interest in using electronic voting systems as a way to improve the voting process. These systems allow individuals to cast their votes electronically, typically through the use of computers or other electronic devices. However, traditional electronic voting systems can still be vulnerable to hacking and other forms of tampering, which can undermine the integrity of the electoral process. Blockchain technology offers a potential solution to these issues by providing a secure and transparent means of conducting elections. A blockchain is a decentralized, distributed ledger that allows for the recording and verification of transactions in a tamper-evident manner. This makes it ideal for use in an e-voting system, as it allows each vote to be recorded and verified in a secure and transparent manner.

In this project, we will explore the potential benefits and challenges of using a blockchain-based e-voting system, as well as potential implementation approaches. We will also consider the potential impact of such a system on the electoral process and on democracy more broadly.

1.2 OBJECTIVE

The objective of the project:

- To develop the blockchain based electronic voting system.

1.3 SCOPE

- Research and analysis: This could involve conducting a thorough review of existing e-voting systems and exploring the potential benefits and challenges of using blockchain technology to improve the voting process.

- System design and development: This could involve designing and building the technical infrastructure for the e-voting system.

- User interface design: This could involve designing a user-friendly interface that makes it easy for voters to cast their ballots electronically.

1.4 APPLICATION

The voting system could be used to enable individuals to cast their ballots remotely, eliminating the need to physically go to a polling station. This could be particularly beneficial for individuals with disabilities or those who live in remote areas.

1.5 PROBLEM STATEMENT

The blockchain-based voting system aim to address is the need for a secure, transparent, and auditable voting process. In many traditional voting systems, there is a risk of fraud, tampering, or other types of interference that can undermine the integrity of the vote. A blockchain-based voting system could help to mitigate these risks by providing a tamper-evident record of the votes that is transparent, secure, and auditable.

2. LITERATURE REVIEW

The first things that come to mind about the blockchain are cryptocurrencies and smart contracts because of the well-known initiatives in Bitcoin and Ethereum. Bitcoin was the first crypto-currency solution that used a blockchain data structure. Ethereum introduced smart contracts that leverage the power of blockchain immutability and distributed consensus while offering a crypto-currency solution comparable to Bitcoin. The concept of smart contracts was introduced much earlier by Nick Szabo in the 1990s and is described as "a set of promises, specified in digital form, including protocols within which the parties perform on these promises"' [1]. In Ethereum, a smart contract is a piece of code deployed to the network so that everyone has access to it. The result of executing this code is verified by a consensus mechanism and by every member of the network as a whole.

Today, we call a blockchain a set of technologies combining the blockchain data structure itself, distributed consensus algorithm, public key cryptography, and smart contracts [2]. Below we describe these technologies in more detail.

Blockchain creates a series of blocks replicated on a peer-to-peer network. Any block in blockchain has a cryptographic hash and timestamp added to the previous block, as shown in Figure 1. A block contains the Merkle tree block header and several transactions [3]. It is a secure networking method that combines computer science and mathematics to hide data and information from others that is called cryptography. It allows the data to be transmitted securely across the insecure network, in encrypted and decrypted forms [4].

As was already mentioned, the blockchain itself is the name for the data structure. All the written data are divided into blocks, and each block contains a hash of all the data from the previous block as part of its data [5]. The aim of using such a data structure is to achieve provable immutability. If a piece of data is changed, the block's hash containing this piece needs to be recalculated, and the hashes of all subsequent blocks also need to be recalculated [6]. It means only the hash of the latest block has to be used to guarantee that all the data remains unchanged. In blockchain solutions, data stored in blocks are formed from all the validated transactions during their creation, which means no one can insert, delete or alter transactions in an already validated block without it being noticed [7]. The initial zero-block, called the "genesis block," usually contains some network settings, for example, the initial set of validators (those who issue blocks).

Blockchain solutions are developed to be used in a distributed environment. It is assumed that nodes contain identical data and form a peer-to-peer network without a central authority. A consensus algorithm is used to reach an agreement on blockchain data that is fault-tolerant in the presence of malicious actors. Those that are intended to be used in fully decentralized self-organizing networks, such as cryptocurrency platforms, use algorithms such as proof-of-work or proof-of-stake, where validators are chosen by an algorithm so that it is economically profitable for them to act honestly [8]. When the network does not need to be self-organized, validators can be chosen at the network setup stage [9]. The point is that all validators execute all incoming transactions and agree on achieving results so that more than two-thirds of honest validators need to decide on the outcome.

As is the case with any other technology, blockchain technology has its drawbacks. Unlike other distributed solutions, a blockchain is hard to scale: An increasing number of nodes do not improve network performance because, by definition, every node needs to execute all transactions, and this process is not shared among the node [10]. Moreover, increasing the number of validators impacts performance because it implies a more intensive exchange of messages during consensus. For the same reason, blockchain solutions are vulnerable to various denial-of-service attacks. A network can be attacked by merely sending a considerable number of transactions: At some point, the system will refuse to receive anything else. In cryptocurrency solutions, all transactions have an execution cost: the more resources a transaction utilizes, the more expensive it will be, and there is a cost threshold, with transactions exceeding the threshold being discarded. In private blockchain networks [11] [12], this problem is solved depending on how the network is implemented via the exact mechanism of transaction cost, access control, or something more suited to the specific context.

3. REQUIREMENTS

Functional Requirement

1. It should manage voting category, candidate and voters.
2. Cast vote.
3. Talley Votes.
4. Display voting result.

3.1 Non-Functional Requirement

1. It will be distributed, decentralized and secured system.
2. It will provide immutable and transparent result.
3. It will verify result integrity.
4. User friendly UI.

4. SYSTEM DESIGN

4.1 Data flow diagram (DFD):

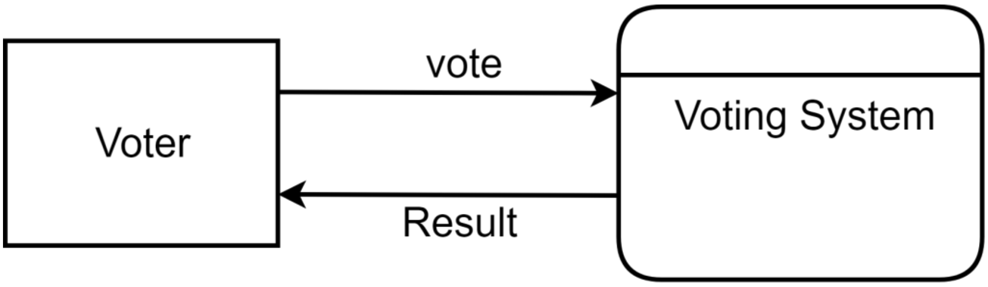


Figure 1: level 0 DFD

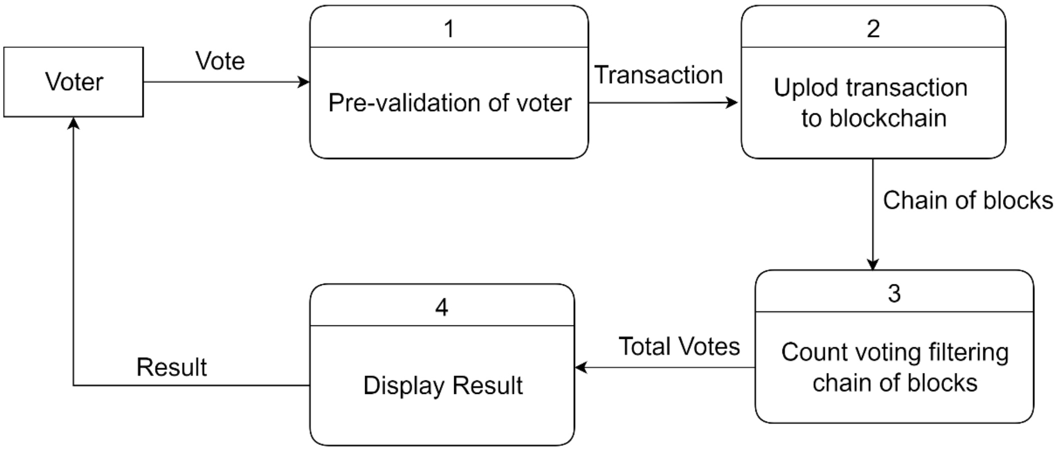


Figure 2: level 1 DFD

5. METHODOLOGY

5.1 The block diagram of the project:

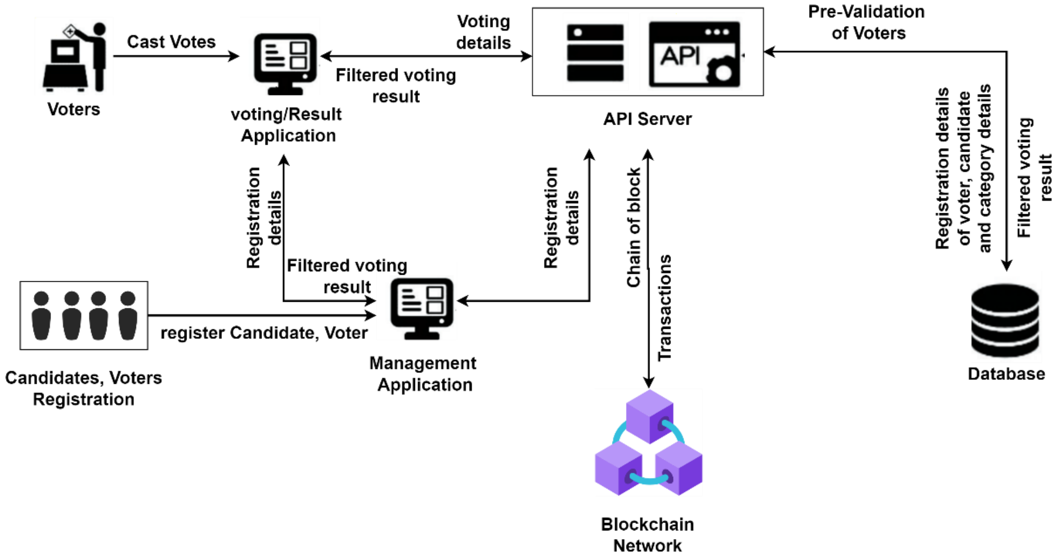


Figure 3: Block diagram of project

5.2 Management Application

The management application implies the features like register candidate, register voter, adding category for the voting and calling endpoints for storing candidate and voter details in the database. The management application comprises the following methods:

registerCandidate: for registration of candidate

registerVoter: for registration of Voter

addCategory: for adding category of voting

fetchRegistrationDetails: for fetching the registration details from database

5.3 Voting/Result Application

The voting/result application implies the features like casting vote and displaying voting result. It comprises the methods:

vote: for voting the candidate

dispVotes: for displaying the total votes of each candidate:

5.4 Blockchain Network

It is the underlying blockchain platform and infrastructure that powers the system. It is responsible for recording and storing votes on the blockchain, as well as maintaining the integrity and security of the system. It has a class Blockchain with attributes and methods:

5.4.1 Attributes:

chain []: for keeping track of the blocks in blockchain.

pendingTransactions []: for keeping track of the pending transactions.

currentNodeUrl: for keeping the track of currently running node.

nodeNetworks []: for keeping the track of number of nodes existing in the network.

5.4.2 Methods:

createNewBlock: this method creates the new block in the blockchain taking an argument nonce, previousBlockHash, hash.

createNewTransaction: this method creates the transaction taking an argument candidate name, candidate unique address, category, voter unique address.

hashBlock: this method creates the hash of the block data taking an argument previousBlockHash, blockHash, nonce.

proofOfWork: It is mechanism used by blockchain networks to validate transactions and create new blocks. The proofOfWork method returns the value of nonce for the hash generated that contains first four zeros.

addTransactionToPendingTransaction: this method pushes the current transaction to the pending transactions so that it can be included in the succeeding block.

getLastBlock: the method returns the last block of the chain.

consensus: the method implements the consensus algorithm which checks whether the block is valid or not, if not replace by the longest valid chain in the network (implements longest chain rule).

5.5 API Server

The API Server of a blockchain-based voting system is responsible for managing the underlying infrastructure and processes required to run the system. It handles tasks such as storing candidate and voter's data in the database, validating voters, updating the blockchain ledger and filter the votes stored in chain and updating in the database. The Server comprises the following methods:

initializeBlockchain: handles the event for initializing and setting up every node in the network.

createCandidate: handles the event for registering the candidate.

createVoter: handles the event for registering the voter.

vote: handles the event for registering vote, voted by the voter to the candidate.

voteCount: handles the event for calculating the votes obtained by each candidate.

voteStore: handle the event for storing the votes voted by the voter in the database.

Also, the API Server has the endpoints for triggering createCandidate, createVoter and vote methods and to make interactions with the blockchain attributes and methods:

/create-candidate: triggers createCandidate method

/create-voter: triggers createVoter method

/create-vote: triggers vote method

/blockchain: it returns the entire blocks available in the network.

/register-and-broadcast-node: current node register and broadcast new node to the other nodes in the network.

/register-node: every node in the network register the new node.

/register-node-bulk: the new node registers every other node available in the network.

/transaction: it is used by the nodes to push the transaction in the pendingTransactions.

/transaction-broadcast: the current node pushes the transaction to the pendingTransactions and broadcast the transactions to other nodes in the network.

/mine: current node creates a newBlock and broadcast the block to other nodes in the network.

/receive-new-block: the other nodes receive the newBlock and push into the chain with verification.

5.6 The working model of API Server on Voting

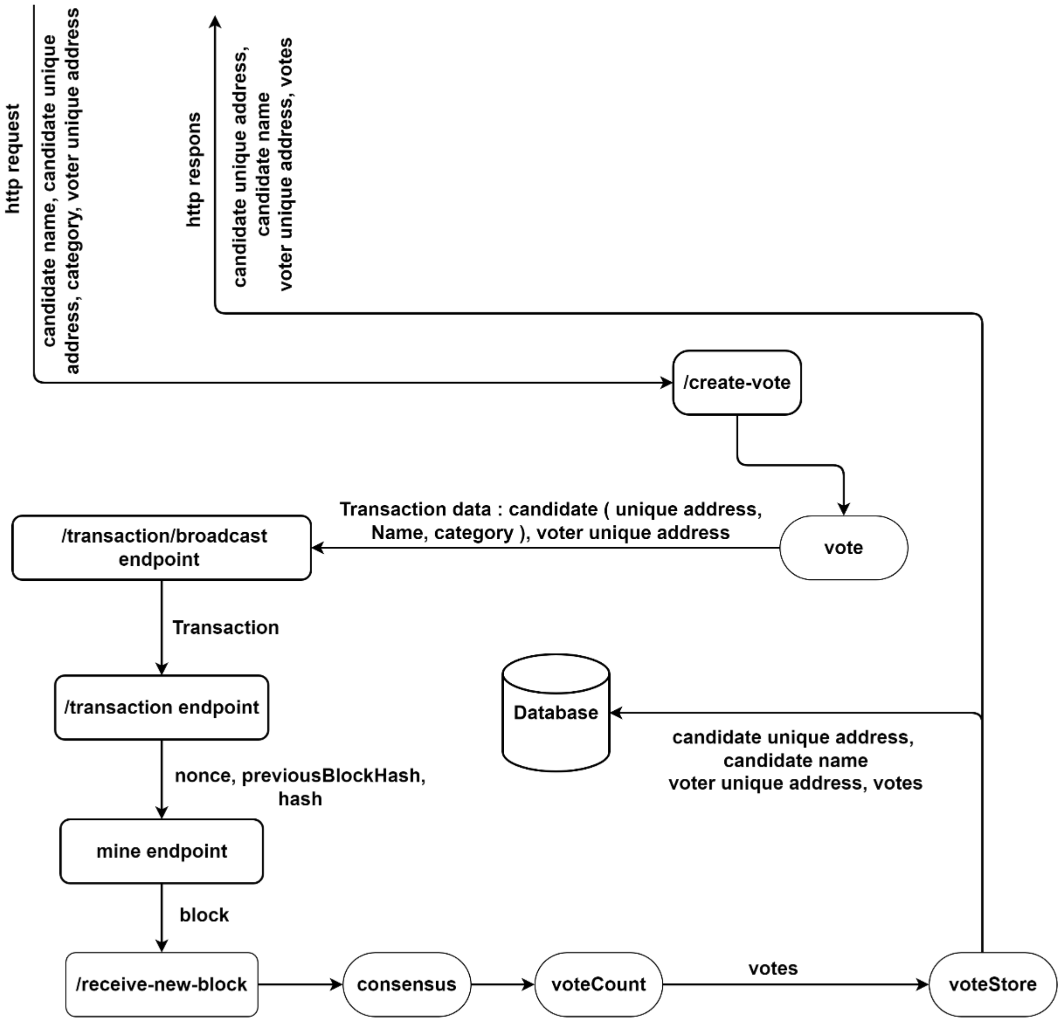

http request
candidate name, candidate unique address, category, voter unique address
http respons
candidate unique address, candidate name voter unique address, votes
/create-vote
vote
Transaction data : candidate (unique address, Name, category), voter unique address
/transaction/broadcast endpoint
Transaction
/transaction endpoint
nonce, previousBlockHash, hash
mine endpoint
block
/receive-new-block
consensus
voteCount
votes
voteStore
Database
candidate unique address, candidate name voter unique address, votes

Figure 4: Working model of API server

The algorithm for the working model of the backend:

- The http request is received with end point /create-vote triggers the Vote () function containing unique candidate address, candidate name, category and voter unique address.
- The /transaction/broadcast endpoint is triggered with the transaction data containing candidate unique address, candidate name, category and voter unique address and creates transactions for each category and then broadcasted to all the nodes in the network.

- The /transaction endpoint is triggered in each node then pushed into their pendingTransactions array.
- Then the /mine endpoint is triggered which create a block, push into chain array and broadcast to other nodes.
- The /receive-new -block endpoint is triggered on each node in which a new block is pushed into their chain array.
- The consensus () method is invoked which validates the chain of blocks across the network.
- The voteCount () is invoked which counts the votes obtained by the candidates.
- The voteStore () method is then invoked which stores the data (candidate unique address, candidate name, category, voter address) into the database and also return the same copy of data as http response.

5.7 The working model of User Interface

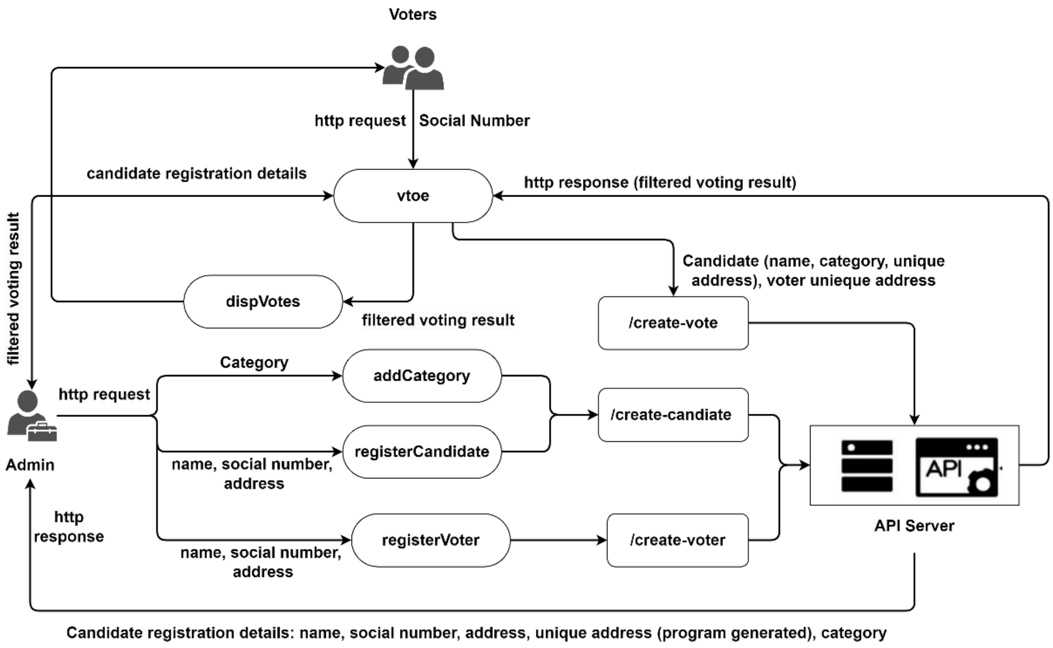

Voters
http request
Social Number
candidate registration details
vtoe
http response (filtered voting result)
filtered voting result
Candidate (name, category, unique address), voter unieque address
dispVotes
filtered voting result
/create-vote
Category
addCategory
http request
/create-candiate
Admin
registerCandidate
name, social number, address
API
API Server
http response
registerVoter
/create-voter
name, social number, address
Candidate registration details: name, social number, address, unique address (program generated), category

Figure 5: Working model of User Interface

The algorithm for the working model of the user interface:

- The admin sends http request to the API server calling endpoint /create-candidate and with parameters category name, social number and address to register candidate and in http response the server returns the registration details of the candidate with program generated unique candidate address.
- The admin sends http request to the API server calling endpoint /create-voter with parameters name, social number and address to register voter.
- The voters send http request to the server invoking the method vote () which call an endpoint /create-vote with parameters registration details of candidate and voter unique address and in http response the server sends the filtered voting result which is further passed to method dispVotes () (which displays result to the voters) and admin.

5.8 The flow chart for broadcasting transactions

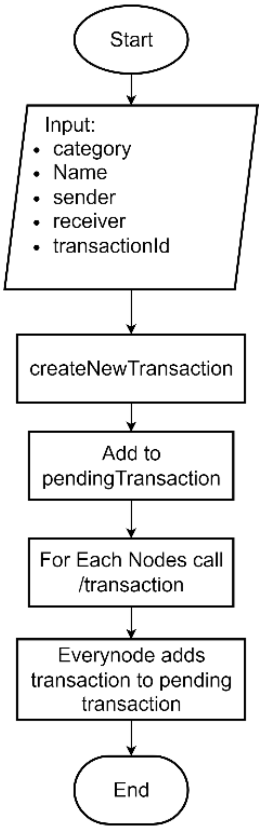

Start
Input:
• category
• Name
• sender
• receiver
• transactionId
createNewTransaction
Add to pendingTransaction
For Each Nodes call /transaction
Everynode adds transaction to pending transaction
End

Figure 6: Flowchart of /transaction/broadcast

The algorithm for broadcasting transactions:

1. The category, name, sender, receiver parameters are fed into the function createNewTransaction which return a transaction and pushed into int pendingTransactions array.
2. For each node /transaction is called with parameter a transaction created in 1 and every node pushes the transaction into their pendingTransactions array.

5.9 The flow chart for consensus algorithm

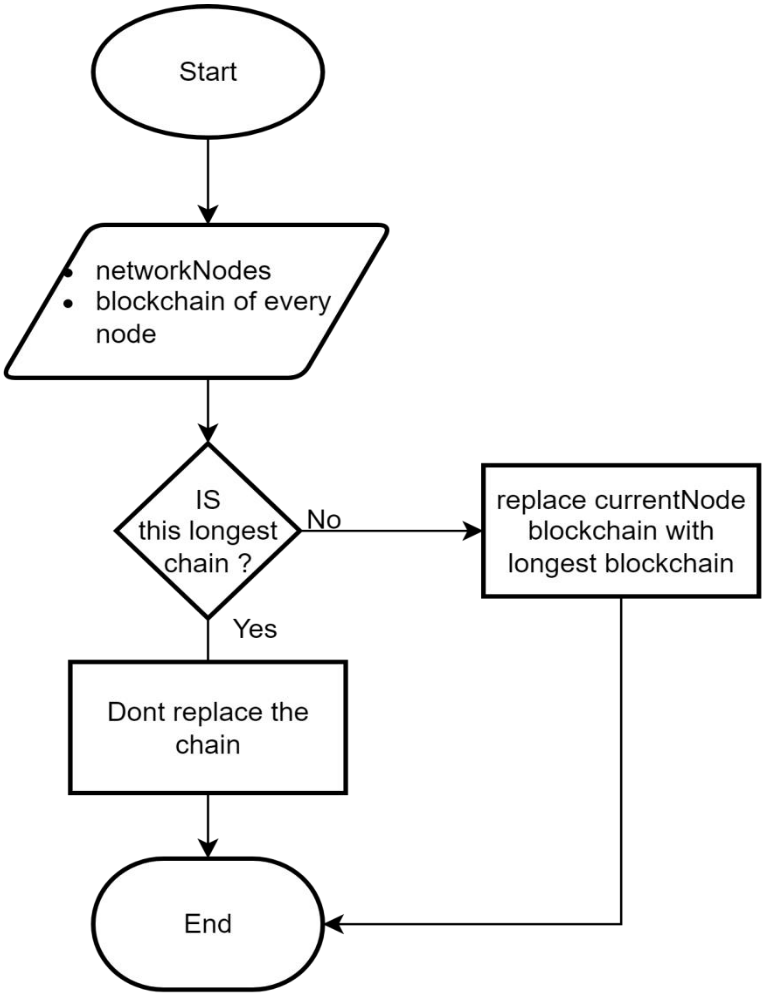
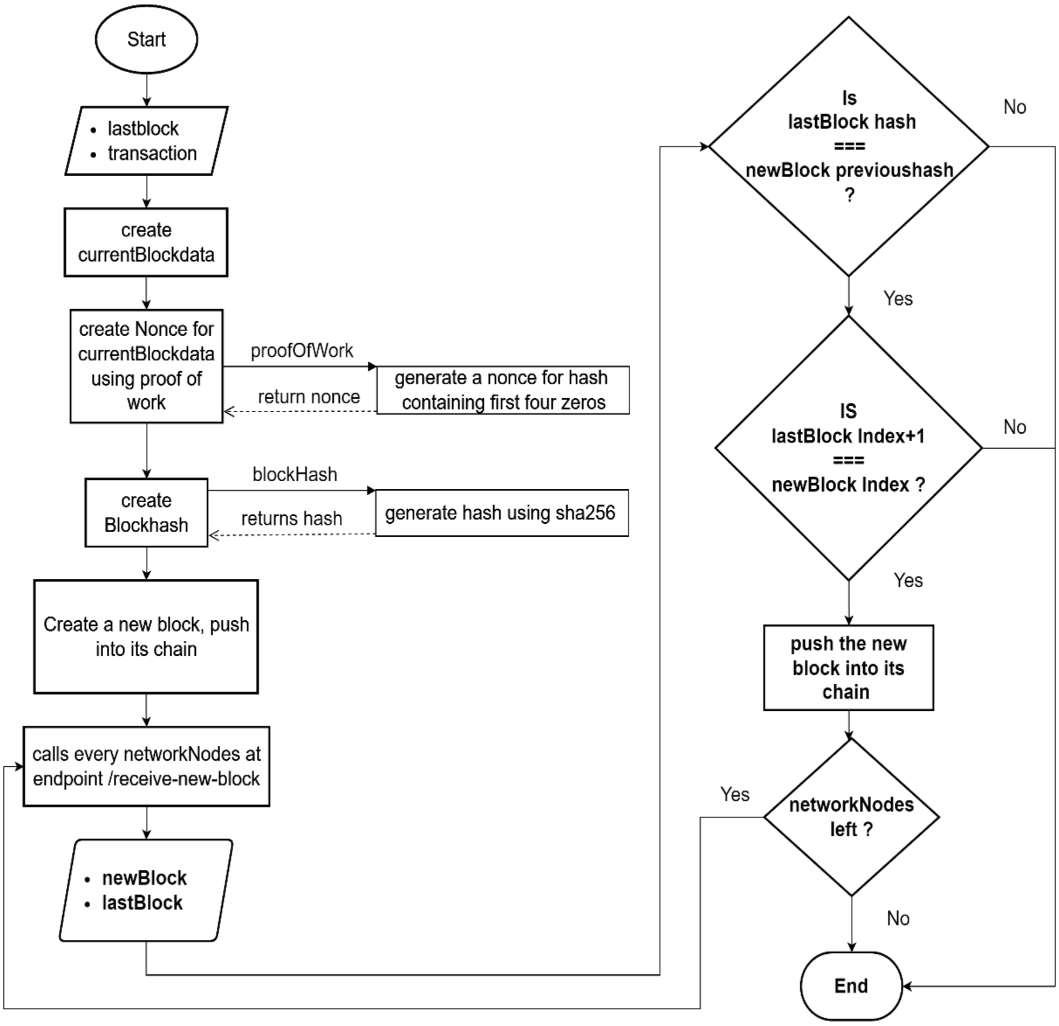


Figure 7: Flow chart for consensus algorithm

5.10 The flow chart for mining block



Start
• lastblock
• transaction
create currentBlockdata
create Nonce for currentBlockdata using proof of work
proofOfWork
return nonce
generate a nonce for hash containing first four zeros
blockHash
returns hash
create Blockhash
generate hash using sha256
Create a new block, push into its chain
calls every networkNodes at endpoint /receive-new-block
• newBlock
• lastBlock
Is lastBlock hash === newBlock previoushash ?
No
Yes
IS lastBlock Index+1 === newBlock Index ?
No
Yes
push the new block into its chain
networkNodes left ?
Yes
No
End

Figure 8: Flow chart for mining block

The algorithm for mining block:

1. The parameter currentBlockdata is created using the data lastblock and transaction and nonce generated passing into the method proofOfWork which is fed into the method BlockHash which return the hash of the newBlock.
2. New Block is created by passing the parameters nonce, previousBlockHash, hash into the method createNewBlock which is pushed into the current Node and broadcasted to every node in the network.

3. The /receive-new-block endpoint is then called in every node in which every node pushes the new block into their chain by checking the condition:
    a. If lastBlock hash == newBlock previous hash
    b. lastBlock index + 1 == newBlock index

5.11 The flow chart for connecting and synchronizing the nodes

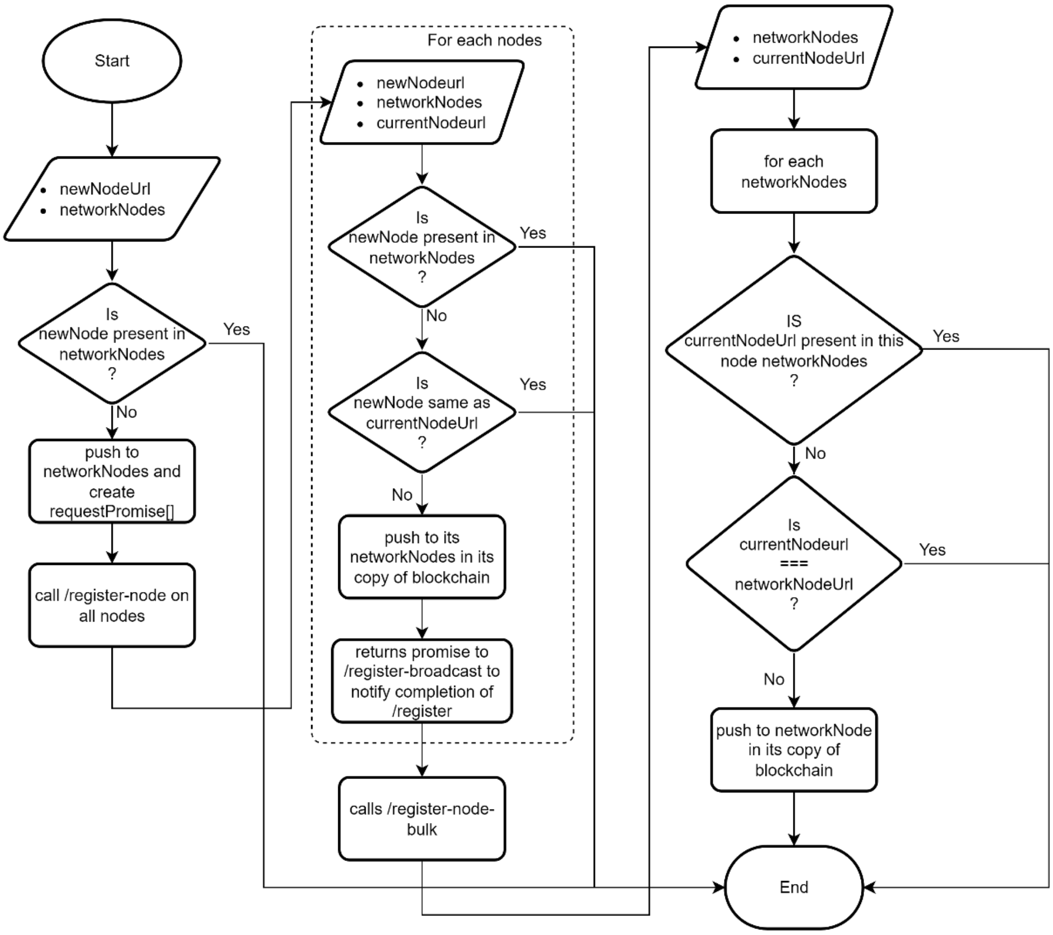

Start
• newNodeUrl
• networkNodes
Is newNode present in networkNodes ?
Yes
No
push to networkNodes and create requestPromise[]
call /register-node on all nodes
For each nodes
• newNodeurl
• networkNodes
• currentNodeurl
Is newNode present in networkNodes ?
Yes
No
Is newNode same as currentNodeUrl ?
Yes
No
push to its networkNodes in its copy of blockchain
returns promise to /register-broadcast to notify completion of /register
calls /register-node-bulk
• networkNodes
• currentNodeUrl
for each networkNodes
IS currentNodeUrl present in this node networkNodes ?
Yes
No
Is currentNodeurl === networkNodeUrl ?
Yes
No
push to networkNode in its copy of blockchain
End

Figure 9: Flow chart for connecting nodes and synchronizing the nodes

The algorithm for connecting and synchronizing nodes:

1. The networkNodes lists every node connected with the current node. newNodeUrl is checked whether it is present in the networkNodes or not. If not, push into the networkNodes and broadcast the newNodeUrl to other Nodes in the network.

2. Every node in the network receive the newNodeUrl and pushes into the their networkNodes array checking the condition:
    a. newNodeUrl not present in networkNodes.
    b. newNodeUrl is not currentNodeUrl.

    And send their nodeUrl as response to current node.
3. The current node captures the response sent by every node and send to the new node.
4. The new node pushes the every networkNodeUrl into its networkNode array.

6. DISCUSSION

A blockchain-based voting system has the potential to significantly improve the security, transparency, and accessibility of elections. Some potential benefits of a blockchain-based voting system include:

Security: One of the main benefits of using a blockchain for voting is the increased security it provides. A blockchain is an immutable distributed ledger, which means that once a transaction (in this case, a vote) is added to the chain, it cannot be altered or deleted. This makes it much more difficult for votes to be tampered with or fraudulently altered.

Transparency: A blockchain-based voting system would allow anyone to verify the accuracy of the vote count by simply reviewing the transactions on the chain. This can help to build trust in the electoral process and ensure that the results are fair and accurate.

Accessibility: A blockchain-based voting system could potentially make it easier for people to vote, especially for those who may have difficulty physically accessing polling stations. For example, individuals could potentially vote using their smart phones or other devices, which could make voting more convenient and accessible for everyone.

Efficiency: A blockchain-based voting system could potentially reduce the amount of time and resources needed to conduct an election. For example, it could eliminate the need for manual vote counting, which can be time-consuming and error-prone.

There are also some challenges to consider when implementing a blockchain-based voting system, such as ensuring the privacy of individual votes and ensuring the security of the system against attacks. However, with careful planning and the use of proven security measures, these challenges can be overcome to create a secure, transparent, and efficient voting system.

7. EXPECTED OUTCOME

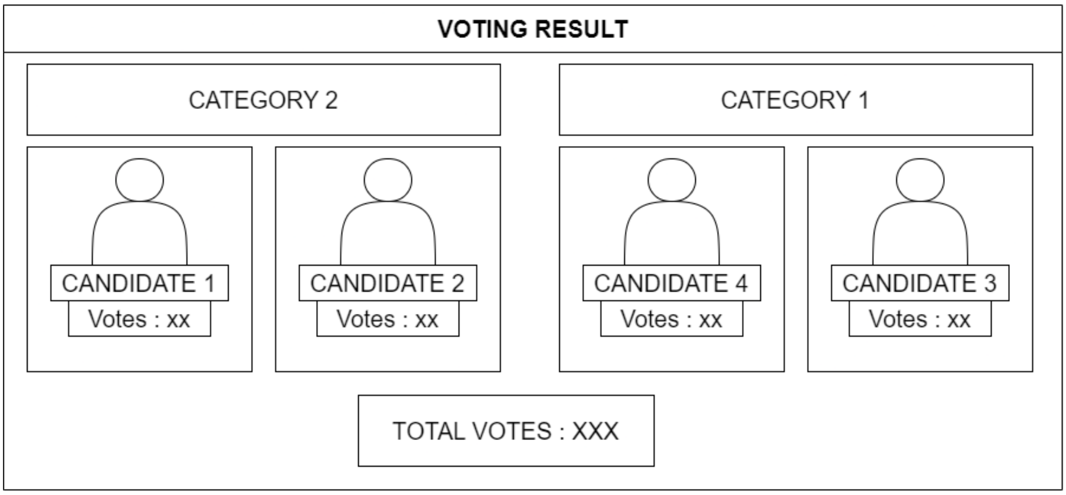


Figure 10: Expected Output

The voting system displays the total votes and the vote obtained by each candidate according to the category as shown in the fig. 13. The candidate obtaining the highest votes according to their respective category

REFERENCES

[1] N. Szabo, "Formalizing and securing relationships on public networks," 1997.

[2] W. Tan, H. Zhu, J. Tan, Y. Zhao, L. D. Xu and K. Guo, "A novel service level agreement model using blockchain and smart contract for cloud manufacturing in industry 4.0," Taylor & Francis, 2022.

[3] S. Nakamoto, "bitcoin.org," 2008. [Online]. Available: bitcoin.org/bitcoin.pdf.

[4] K. Garg, P. Saraswat, S. Bisht, S. K. Aggarwal, S. K. Kothuri and S. Gupta, "A comparitive analysis on e-voting system using blockchain," IEEE, 2019.

[5] S. Abed, R. Jaffal, B. J. Mohd and M. Al-Shayeji, "An analysis and evaluation of lightweight hash functions for blockchain-based IoT devices," Springer, 2021.

[6] M. Nofer, P. Gomber, O. Hinz and D. Schiereck, "Blockchain," Springer, 2017.

[7] L. Zhang, M. Peng, W. Wang, Z. Jin, Y. Su and H. Chen, "Secure and efficient data storage and sharing scheme for blockchain-based mobile-edge computing," Wiley Online Library, 2021.

[8] B. L. a. R. Clayton, " Proof-of-Work Proves Not to Work.".

[9] D. Prashar, N. Jha, S. Jha, G. P. Joshi and C. Seo, "Integrating IOT and blockchain for ensuring road safety: An unconventional approach," Multidisciplinary Digital Publishing Institute, 2020.

[10] Z. Zheng, S. Xie, H.-N. Dai, W. Chen, X. Chen, J. Weng and M. Imran, "An overview on smart contracts: Challenges, advances and platforms," Elsevier, 2020.

[11] M. T. Oliveira, G. R. Carrara, N. C. Fernandes, C. V. N. Albuquerque, R. C. Carrano, D. S. V. Medeiros and D. M. F. Mattos, "Towards a performance

evaluation of private blockchain frameworks using a realistic workload," IEEE, 2019.

[12] H. A. Hussain, Z. Mansor and Z. Shukur, "Comprehensive Survey And Research Directions On Blockchain Iot Access Control," Science and Information (SAI) Organization Limited, 2021.

[13] R. S. S. S. S. S. N. B. A. Raj Shrestha, "Blockchain Interfaced Sacure E-Voting System," Journal of the Institute of Engineering, 2019.