**ABSTRACT**

Decentralized voting Using Ethereum Blockchain is a secure, transparent and tamperproof way of conducting online voting. It is a decentralized application built on the Ethereum blockchain network, which allows participants to cast their votes and view the voting results without the need for intermediaries. In this system, voters are recorded on the blockchain, making it impossible for anyone to manipulate or alter the results. The use of smart contracts ensures that the voting process is automated, transparent, and secure. The use of the blockchain technology and the implementation of a decentralized system provide a reliable and cost-effective solution for conducting trustworthy and fair elections.

**Keywords:** Blockchain, Ethereum, Decentralized Voting, Smart Contracts, Transparency, Security

**LIST OF TABLES**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | |  | | | | Page no | | | |
| Table 2.2 | | | | Key findings and observations from literature | | | | 8-9 | | | |
| Table 5.1 | | | | Test Case 1 | | | | 29 | | | |
| Table 5.2: | | | | Test Case 2 | | | | 29 | | | |
| Table 5.3: | | | | Test Case 3 | | | | 30 | | | |
| Table 5.4 | | | | Test Case 4 | | | | 31 | | | |
| Table 5.5 | | | | Test Case 5 | | | | 32 | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |

**LIST OF FIGURES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | | Page no | |
| Figure 4.2.1: | | Transfer contract terms into code | | 14 | |
| Figure 4.2.2: | | The code is stored in a blockchain and replicated between participants | | 15 | |
| Figure 4.2.3: | | When a term is satisfied, computers in the network verify its correctness. | | 16 | |
| Figure 4.5.1: | | Architecture of Purposed system | | 21 | |
| Figure 4.5.2 | | Blockchain Voting System working flow Design | | 22 | |
| Figure 4.5.3 | | Sequence Diagram | | 24 | |
| Figure 4.5.4 | | Flowchart | | 25 | |
| Figure 4.7.1 | | Level 0 Data Flow Diagram | | 27 | |
| Figure 4.7.2 | | Level 1 Data Flow Diagram | | 27 | |
| Figure 4.7.3 | | Level 2 Data Flow Diagram | | 28 | |
| Figure 4.8 | | Use Case Diagram | | 28 | |

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | Page No. |
| CERTIFICATE | | | II |
| DECLARATION | | | III |
| ACKNOWLEDGEMENTS | | | IV |
| ABSTRACT | | | V |
| LIST OF TABLES | | | VI |
| LIST OF FIGURES | | | VII |
|  | | | |
| 1 | INTRODUCTION | | 1-5 |
|  | 1.1 | Introduction | 1 |
|  | 1.2 | What is Blockchain? | 1 |
|  | 1.3 | Three parts of Blockchain | 2 |
|  | 1.4 | Aim and Objective | 3 |
|  | 1.5 | Existing System | 4 |
|  | 1.6 | Problem Statement | 4 |
|  | 1.7 | Proposed System | 4 |
| 2 | LITERATURE SURVEY | | 6-9 |
|  | 2.1 Detailed Review of Literature | | 6 |
| 3 | SYSTEM SPECIFICATION | | 10-13 |
|  | 3.1 Functional Requirements | | 10 |
|  | 3.2 Non- Functional Requirements | | 11 |
|  | 3.3 Requirement Analysis | | 12 |
|  | 3.4 Project Planning | | 13 |
| 4 | SYSTEM DESIGN AND ARCHITECTURE | | 14-28 |
|  | 4.1 Software Design | | 14 |
|  | 4.2 Smart Contract | | 14 |
|  | 4.3 System Architecture | | 16 |
|  | 4.4 Modules | | 18 |
| 5 | TESTING | | 29-32 |
| 6 | CONCLUSION AND FUTURE ENHANCEMENTS | | 33-34 |
|  | SCREENSHOTS | | 35-38 |
|  | REFERENCES | | 39 |

CHAPTER 1

INTRODUCTION

1. **Introduction**

Internet is the greatest thing invented humanity. But there are some flaws on the internet. Consider a situation where you are depositing money or casting a vote, there is a single point of authority, and we are supposed to believe him/her with our data/money/vote. The limitation of the present system is a single point of control/failure. The Authority may or may not Authority be telling the truth or corrupted. The solution to this is to employ a decentralised and distributed system where the consensus of the users/peers is used to evaluate the transactions/votes/data.

1. **What is Blockchain?**

A blockchain is a collection of blocks linked together with chains using cryptography. Blockchain is one of the emerging technologies with strong cryptographic foundations enabling applications to leverage these abilities to achieve resilient security solutions. Here the data is divided into blocks and linked together. Each block is associated with a hash value (which represents the block), and the (which represents the block), and the link is made link is made possible by listing the possible by listing the hash of the previous block into the current block. To summarise a block consists of the data section, hash, section, hash, previous hash.

### Now the created chain of blocks doesn’t get stored in a single computer in a single computer. All the users have there own copy of the blockchain which known as Distributed Ledger.

If someone tries to tamper with the data, the hash value gets changed, hash value gets changed, and the link is broken and the link is broken. To make the attempt successful, the attacker needs to change and recalculate the hashes of subsequent blocks. Each block, when created, is curated by the users and based on their consensus, and the block may be added or rejected. Hence the blockchains provide Security, Immutability and Transparency.

There are three main types of blockchains in practice; they are Public, Private and Consortium Blockchains.

1. **Three Parts of Blockchain**

A blockchain can be studied as a database that A blockchain can be studied as a database that is distributed across its users. The essential Distributed across its users. The essential requirements of a requirements of a blockchain are Peer blockchain are Peer to Peer networking, Asymmetric to Peer networking, Asymmetric Cryptography, Hashing Cryptography, Hashing.

**Ethereum**

Ethereum is a decentralized, open-source blockchain featuring smart contract functionality. Ether (ETH) is the native cryptocurrency of the platform. It is the second-largest cryptocurrency by market capitalization, after Bitcoin. Ethereum is the most actively used blockchain. Ethereum was proposed in 2013 by programmer Vitalik Buterin. Development was crowdfunded in 2014, and the network went live on 30 July 2015, with 72 million coins.

The Ethereum Virtual Machine (EVM) can execute Turing-complete scripts and run decentralized applications. Ethereum is used for decentralized finance, and has been utilized for many initial coin offerings. A smart contract is a piece of code lying on the block, which is used to make decisions/transactions. Ethereum uses Solidity Programming Language to write its Smart Contracts.

Ethereum has a native cryptocurrency called Ether (ETH) which is similar to a Bitcoin. Being a programmable blockchain, many developers can use this blockchain service in their applications.

**Smart Contracts**

Smart Contracts are tools that can automatically execute transactions if certain conditions are met without requiring the help of an intermediary company or entity. They are often associated with Ethereum, a blockchain that was designed to accommodate smart contracts, but the idea isn’t restricted to any particular platform or network. So to perform or deploy the contract, a cost is associated with it, called Gas.

In general, it is expensive and slow to execute on the shared network than to perform network in a traditional setup.

**Election Process**

The election is a formal way of making decisions. A democratic society has its foundations from voting. Elections are powerful as they are the deciding factors for the fate of an organisation/country organisation/country. The question of Transparency and Security is still unanswered.

* Administrator – Manages and conducts the election
* Candidate – Participant in the election
* Voter – Person who is entitled to vote

Traditional elections use a centralised system where a body is trusted to conduct and manage the whole process. Some problems with this structure are administrative authority may be compromised, tampered may be occurred.

#### Aim and objective

The main objective of this project is to build a web application using blockchain technology where people can vote from anywhere if he/she possess a valid Citizenship of respective country where he/she wants to vote and protect each and evry vote to ensure that each and every vote matters.

The vast majority of the ongoing work discusses security, exactness, respectability, quickness, protection, and review capacity however existing frameworks are powerless for assaults at some degree.

Disadvantages of Existing System

* Centralized architecture.
* Attack prone.
* Not trustable.
* Non-transparent vote casting process.

The existing systems are prone to attacks and are either easily hackable or very difficult to maintain. Data integrity and security are the major concerns and the proposed solution should be able to address all the short comings of the existing systems.

#### Existing System

Voting is an integral part of a democratic society. It is a decision making mechanism and security plays an important role in voting. The existing systems are:

* Ballot System: In India, before 2004 there was a paper-based voting system. This is called as ballot Paper system. It is placed in the election booth and is used the voters.
* Electronic Control System : In order to overcome duplication and damage of ballot problems Electronic Voting Machines Were introduced. It stores and assembles votes, used by poll workers.
* Current Digital Voting Systems: A number of digital voting systems are currently in use in countries around the world. We researched some of these systems to familiarise ourselves with current implementations, particularly Estonia.

Estonia has had electronic voting since 2005 and in 2007 was the first country in the world to allow online voting. In the 2015 parliamentary election 30.5% of all votes were made though the nation’s i-voting system (Vabariigi Valimiskomisjon, 2016). The bases of this system is the national ID card that all Estonian citizens are given. These cards contain encrypted files that identify the owner and allows the owner to carry out a number of online and electronic activities including online banking services, digitally signing documents, access their information on government databases and i-voting. (Electronic ID Card, no date)

#### Problem Statement

Several studies have been done on using computer technologies to improve elections. These studies tell about the risks of adopting electronic voting system, because of the software challenges, insider threats, network vulnerabilities, and the challenges of auditing.

#### Proposed System

We’ve proposed to design the existing online voting system which is integrated with the Blockchain technology. The proposed system has the following advantages as compared to the existing system as discussed on last page

* Users’ can vote from anywhere in the world until he possess a citizenship of the
* country.
* The voting is stored in the Blockchain which makes it tamper proof.
* We have worked the following ideas by having the two different set of modules: election commission and the voter(s). Election Commission creates elections and adds registered candidates along with the parties for contesting the election. Using an election’s REST API
* hosted on Ethereum’s Blockchain, the details are shown at the front-end of the voter for casting the vote. Then, while polling the vote is stored on our blockchain framework of which the Election Commission fetches the vote count. The limitation which we have faced due to not using the traditional way of smart contracts is that the blockchain framework which we have coded cannot run on the main net as it needs to be hosted and a separate web3 provider have to be used for interacting with it and not having a public API of voter ID creates a drawback of not having authentication of a voter.
* The objectives for developing the project are as follows:
* To improve the existing online voting system using Blockchain technology.
* To reduce the workload of setting up an election booth and conducting elections in physical form.
* Non-Resident Indian can cast their votes as it is totally online.
* We are supposed to learn the concept of Blockchain and how it can be utilized to work on different sectors.

CHAPTER 2

LITERATURE SURVEY

**2.1 Detailed Review of Literature**

1. Smith, J. (2017) – "Evolution of Electronic Voting Systems"-This paper explores the progression of electronic voting systems from early Direct Recording Electronic (DRE) machines to internet-based models. The author emphasizes the limited transparency and security flaws that have historically affected voter trust.

* **Contribution:** Highlights the shortcomings of existing electronic voting infrastructure and establishes a need for more secure alternatives.
* **Relevance:** Sets the stage for blockchain as a credible solution to long-standing problems in e-voting.

**2.** Jones, M., & Brown, A. (2019) – "Challenges in Traditional Electronic Voting Systems"- This study critically evaluates traditional e-voting models, focusing on vulnerabilities such as malware threats, vote alteration, and the lack of public verifiability.

* **Contribution:** Provides a comprehensive understanding of security and integrity issues in centralized systems.
* **Relevance:** Justifies the transition toward decentralized voting mechanisms with immutable ledgers.

3. Nakamoto, S. (2008) – "Bitcoin: A Peer-to-Peer Electronic Cash System"

-Although not directly about voting, Nakamoto's whitepaper introduces the core technology underpinning blockchain systems—decentralized ledgers and cryptographic consensus.

* **Contribution:** Introduces blockchain as a method for secure, trustless record-keeping.
* **Relevance:** Forms the theoretical foundation for all blockchain-based applications, including decentralized voting.

4. Smith, A., & Johnson, B. (2020) – "Blockchain in E-Voting: A Comprehensive Analysis"-This paper provides an in-depth review of blockchain-based voting systems, specifically focusing on Ethereum smart contracts and cryptographic voter privacy techniques.

* **Contribution:** Discusses Ethereum’s suitability for deploying smart contracts that handle voting logic and verification.
* **Relevance:** Offers practical insight into implementing voting systems with tamper-resistant and automated features.

5. Çabuk, U. C., Adiguzel, E., & Karaarslan, E. (2020) – "A Survey on Feasibility and Suitability of Blockchain Techniques for E-Voting Systems"-The authors evaluate the application of blockchain across different voting models, comparing public and private chains.

* **Contribution:** Analyses Ethereum alongside other platforms, pointing out its transparency strengths and gas fee limitations.
* **Relevance:** Helps in selecting appropriate blockchain configurations for scalable voting systems.

6. Tarasov, P., & Tewari, H. (2017) – "The Future of E-Voting"- This work takes a visionary approach, exploring how emerging technologies like blockchain, biometrics, and digital IDs can converge to create more robust e-voting systems.

* **Contribution:** Recommends hybrid systems combining blockchain with authentication technologies for improved security.
* **Relevance:** Supports the idea of integrating Ethereum with external identity verification mechanisms.

7. Hjálmarsson, F. Þ., et al. (2018) – "Blockchain-Based E-Voting System"- Presented at an IEEE conference, this research proposes and implements an Ethereum-based e-voting prototype. It leverages smart contracts to automate voting, ensure transparency, and provide public verifiability.

* **Contribution:** Demonstrates a working implementation of decentralized voting using Ethereum.
* **Relevance:** Offers a real-world example and blueprint for building Ethereum-based election systems.

# 2.2 Key Findings and Observations from Literature

|  |  |  |  |
| --- | --- | --- | --- |
| S. No. | Reference | Key Findings | Observations |
| 1 | Smith, J. (2017) | Traditional electronic voting systems lack transparency, are prone to tampering, and often exclude robust audit mechanisms. | Emphasizes the need for trustworthy, transparent alternatives such as blockchain-based voting. |
| 2 | Jones, M., & Brown, A. (2019) | Centralized e-voting systems are vulnerable to attacks and manipulation; limited voter confidence in digital systems. | Supports decentralization as a strategy to eliminate single points of failure. |
| 3 | Nakamoto, S. (2008) | Introduces the foundational concept of blockchain: a secure, decentralized, peer-to-peer system that prevents double-spending without a central authority. | Forms the theoretical basis for blockchain-enabled voting systems with tamper-proof ledgers. |
| 4 | Smith, A., & Johnson, B. (2020) | Smart contracts on Ethereum can enforce voting rules, increase transparency, and automate result tabulation. | Confirms Ethereum’s applicability for decentralized voting due to its programmability and public verifiability. |
| 5 | Çabuk, U. C., Adiguzel, E., & Karaarslan, E. (2020) | Analyzes public vs private blockchain suitability for voting systems; Ethereum offers openness but faces scalability and fee challenges. | Highlights trade-offs in choosing a blockchain platform; Ethereum is viable but may not be optimal for large-scale elections. |
| 6 | Tarasov, P., & Tewari, H. (2017) | Advocates for hybrid systems combining blockchain with digital identity and biometrics to enhance security and voter verification. | Suggests that blockchain alone is insufficient—needs to be part of a broader secure ecosystem. |
| 7 | Hjálmarsson, F. Þ., et al. (2018) | Demonstrates a working Ethereum-based e-voting prototype using smart contracts with public audit trails and immutability. | Provides practical evidence that Ethereum can support real-world decentralized voting applications. |

CHAPTER 3

SYSTEM SPECIFICATION

### Functional Requirements

Functional Requirement defines a function of software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. The functional requirements of the project are one of the most important aspects in terms of entire mechanism of modules.

**The functional requirements here are:**

* **Maintaining user:** interface responsiveness: If the application needs to perform a time- consuming task, multiple threads can be used to prevent user interface from becoming unresponsive while the task is in progress. If the program is downloading information from the Internet, this will keep the user-interface running at nearly full-speed while the download is in progress.
* **Simple Multitasking:** Multitasking allows to execute multiple instances of a process quit easily. The downloading routine just mentioned can be extended so that the program can transfer multiple files simultaneously and still keep the user interface well behaved. All that is needed is to create another thread for each file to download.
* **Building Multi-user Applications:** Multithreading is often used when building server applications. Server applications wait for request to arrive and then establish conversations with the requester.
* **Multiprocessing:** Many operating systems support machines with multiple processors. Most of these systems are unable to break a single thread of execution for execution on different processors. By breaking an application into different Threads, it is possible to make the best use of processing power.

### Non-Functional Requirements

### Reliability: The framework ought to be dependable and solid in giving the functionalities. When a client has rolled out a few improvements, the progressions must be made unmistakable by the framework. The progressions made by the Programmer ought to be unmistakable both to the Project pioneer and in addition the Test designer.

#### Security: Aside from bug following the framework must give important security and must secure the entire procedure from smashing. As innovation started to develop in quick rate the security turned into the significant concern of an association. A great many dollars are put resources into giving security. Bug following conveys the greatest security accessible at the most noteworthy execution rate conceivable, guaranteeing that unapproved clients can't get to imperative issue data without consent. Bug following framework issues diverse validated clients their mystery passwords so there are limited functionalities for all the clients.

#### Maintainability: The framework observing and upkeep ought to be basic and target in its approach. There should not be an excess of occupations running on diverse machines such that it gets hard to screen whether the employments are running without lapses.

#### Performance: The framework will be utilized by numerous representatives all the while. Since the framework will be facilitated on a solitary web server with a solitary database server out of sight, execution turns into a noteworthy concern. The framework ought not succumb when numerous clients would be utilizing it all the while. It ought to permit quick availability to every last bit of its clients. For instance, if two test specialists are all the while attempting to report the vicinity of a bug, then there ought not to be any irregularity at the same time.

* **Portability**: The framework should to be effectively versatile to another framework. This is obliged when the web server, which s facilitating the framework gets adhered because of a few issues, which requires the framework to be taken to another framework.
* **Scalability**: The framework should be sufficiently adaptable to include new functionalities at a later stage. There ought to be a typical channel, which can oblige the new functionalities.

#### Flexibility: Flexibility is the capacity of a framework to adjust to changing situations and circumstances, and to adapt to changes to business approaches and rules. An adaptable framework is one that is anything but difficult to reconfigure or adjust because of diverse client and framework prerequisites. The deliberate division of concerns between the trough and motor parts helps adaptability as just a little bit of the framework is influenced when strategies or principles change.

1. **Requirement Analysis**

In order to effectively design and develop a system, it is important to understand and document the requirements of the system. The process of gathering and documenting the requirements of a system is known as requirement analysis. It helps to identify the goals of the system, the stakeholders and the constraints within which the system will be developed. The requirements serve as a blueprint for the development of the system and provide a reference point for testing and validation.

###  Hardware Requirements

o Processor – 2 GHz or moreo RAM – 4 GB or moreo Disk Space – 100 GB or more

###  Software Requirements

* Node.js (version – 18.14.0) o Web3.js (version – 1.8.2) o Truffle (version – 5.7.6) o Solidity (version – 0.5.16) o Ganache (version – 7.7.3) o Metamask
* Python (version – 3.9) o FastAPI o MySQL Database (port – 3306)

#### Project Planning

Project Planning is the most essential thing in developing a project. It sets out the phases, activities and task needed to deliver a project. The timeframes required to deliver the project, along with the resources and milestones are also shown on the project plan.

Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. Project planning is often used to organize different areas of a project, including project plans, workloads and the management of teams and individuals. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path. Project planning is inherently uncertain as it must be done before the project is actually started. Therefore, the duration of the tasks is often estimated through a weighted average of optimistic, normal, and pessimistic cases. The critical chain method adds “buffers”; in the planning to anticipate potential delays in project execution. Float or slack time in the schedule can be calculated using project management software. Then the necessary resources can be estimated and costs for each activity can be allocated to each resource, giving the total project cost. At this stage, the project schedule may be optimized to achieve the appropriate balance between resource usage and project duration to comply with the project objectives. Once established and agreed, the project schedule becomes what is known as the baseline schedule. Progress will be measured against the baseline schedule throughout the life of the project. Analyzing progress compared to the baseline schedule is known as earned value management.

## CHAPTER 4

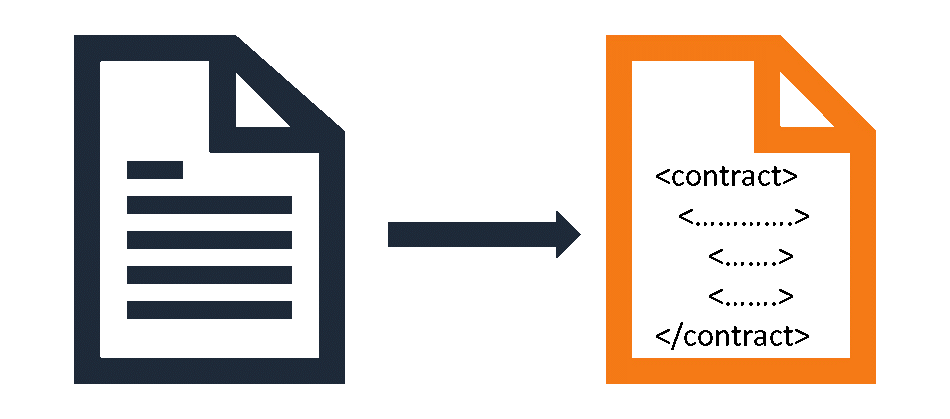
## SYSTEM DESIGN AND ARCHITECTURE

### Software Design

A project plan is a model of the process that the project team intends to follow to realize the project objectives. It brings together a number of important aspects of this process including its scope, timing and associated risks. The project plan can be viewed as a type of “contract” between the project team members and the reviewers. It defines the process by which objectives will be achieved, and the responsibilities in carrying out this process. It also underpins a number of other key project management functions including estimating and forecasting, options analysis and decision-making, and performance monitoring and control.

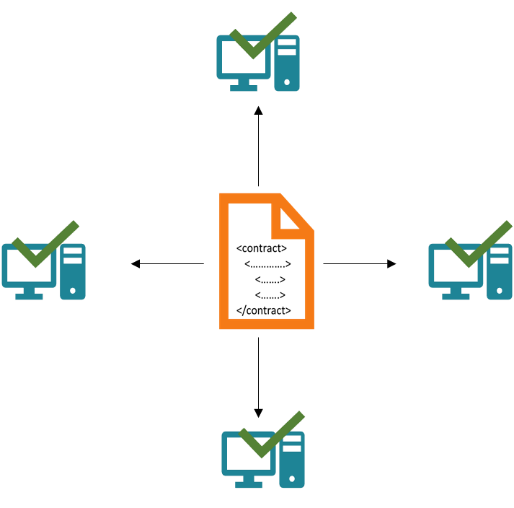
1. **How do smart contracts work?**

First, the contractual parties should determine the terms of the contract. After the contractual terms are finalized, they are translated into programming code. Basically, the code represents a number of different conditional statements that describe the possible scenarios of a future transaction.



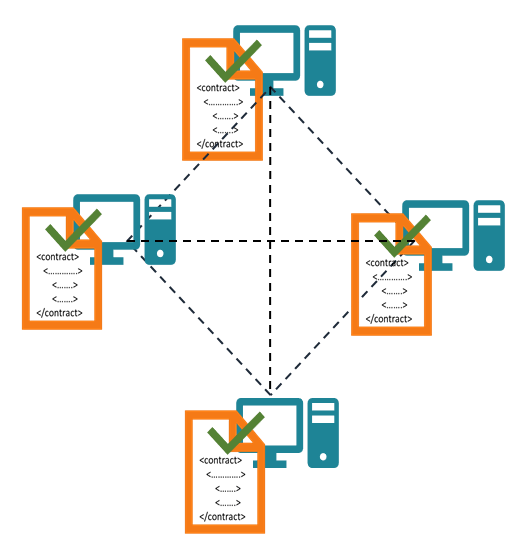
**Fig 4.2.1 : Transfer contract terms into code**

When the code is created, it is stored in the blockchain network and is replicated among the participants in the blockchain.



#### Fig 4.2.2 : The code is stored in a blockchain and replicated between participants

Then, the code is run and executed by all computers in the network. If a term of the contract is satisfied and it is verified by all participants of the blockchain network, then the relevant transaction is executed.



**Fig 4.2.3 : When a term is satisfied, computers in the network verify its correctness.**

**4.3 MetaMask**

MetaMask is a popular cryptocurrency wallet and browser extension that enables users to interact with Ethereum-based decentralized applications (D Apps) and manage their digital assets securely. It acts as a bridge between users and the Ethereum blockchain, providing a user-friendly interface for accessing and interacting with blockchain-based applications.

Here are some key features and aspects of MetaMask:

* 1. Wallet Functionality: MetaMask serves as a digital wallet that allows users to store, manage, and transfer Ethereum and other ERC-20 tokens. Users can create multiple accounts within MetaMask and securely store their private keys, which provide access to their funds.
  2. Browser Extension: MetaMask is primarily available as a browser extension for popular web browsers like Google Chrome, Firefox, and Brave. Once installed, the MetaMask extension adds a small icon to the browser's toolbar, providing quick access to the wallet functionality.
  3. Ethereum Network Access: MetaMask connects users to the Ethereum blockchain, enabling them to send and receive Ether (ETH) and interact with Ethereum-based smart contracts and D Apps. It supports both the Ethereum and various test networks, such as Kovan, allowing users to test and experiment with D Apps in a development environment.
  4. Seamless D App Integration: MetaMask simplifies the process of interacting with D Apps. When a user visits a website that has integrated MetaMask, the extension automatically detects the D App and displays a popup window. This window allows users to authorize transactions, sign messages, and interact with smart contracts directly from their MetaMask wallet.
  5. Enhanced Security: MetaMask provides a secure environment for managing digital assets. It stores private keys locally on the user's device and encrypts them with a user-defined password. MetaMask also supports hardware wallets like Ledger, which offer additional layers of security by keeping private keys offline.
  6. Custom Network Configuration: MetaMask allows users to configure custom Ethereum networks, enabling them to connect to private or alternative networks beyond the Ethereum. This feature is useful for developers and organizations that operate on their own blockchain networks.
  7. Mobile App: In addition to the browser extension, MetaMask offers a mobile app for iOS and Android devices. The mobile app provides similar wallet functionality and allows users to access their accounts and interact with D Apps on the go.

**4.4 Ganache**

Ganache (formerly known as Test RPC) is a local blockchain development tool that provides developers with a simulated Ethereum environment for testing and development purposes. It allows developers to create private, local blockchain networks that behave similarly to the Ethereum mainnet or testnets, but with faster block confirmations and no need for real Ether. Here are some key features and characteristics of Ganache:

1. Local Blockchain: Ganache creates a local Ethereum blockchain on your development machine. This blockchain operates locally, meaning it runs on your computer rather than being connected to the actual Ethereum network. This provides developers with a sandboxed environment to test and debug their decentralized applications (DApps) without incurring any costs or affecting the live network.
2. Fast Block Confirmations: Unlike the Ethereum mainnet or public testnets, which have varying block confirmation times, Ganache provides near instantaneous block confirmations. This allows developers to quickly observe the effects of their transactions, test different scenarios, and iterate on their code more rapidly.
3. Pre-funded Accounts: Ganache automatically generates a set of pre-funded accounts for development purposes. These accounts come pre-loaded with test Ether (fake Ether), which can be used to simulate transactions and interactions with smart contracts on the local blockchain. Each account has a private key associated with it, allowing developers to sign and authorize transactions.
4. Smart Contract Deployment: Ganache provides an interface for deploying and interacting with smart contracts on the local blockchain. Developers can compile their Solidity smart contracts using tools like Remix IDE or Truffle, and then deploy and test them on Ganache. This allows for quick iteration and debugging of smart contract logic and functionality.
5. Network Customization: Ganache offers customization options for the local blockchain network. Developers can configure parameters such as the gas limit, block time, and network ID to mimic specific network conditions or test different scenarios. This flexibility enables developers to simulate a wide range of network conditions and edge cases during development and testing.

### 4.5 System Architecture

Within our proposal we have tried to design a service and system that minimises the size of attack vectors to prevent potential malicious attacks. We have tried to evaluate and analyse our design from various perspectives to make sure we have thought about each step of the voting process. This section of the report discusses the potential risks associated with our proposal and suggests actions that can be taken to help mitigate them. One risk is if a voter were to forget their ID, password or polling card on the day of voting. In this case the voter will be unable to cast their vote as they cannot enter the system. Possible risk mitigations include the voter returning later that day with the correct information or the implementation of a backup authentication service such as by phone. Alternatively, a forgotten password system could be added to the voter registration website; this could work in much the same way as recovering a password works on other websites. However, this increases the risk of a hacker attempting to change a voter’s password without their knowing. A 51% attack is a potential threat to our proposed design.

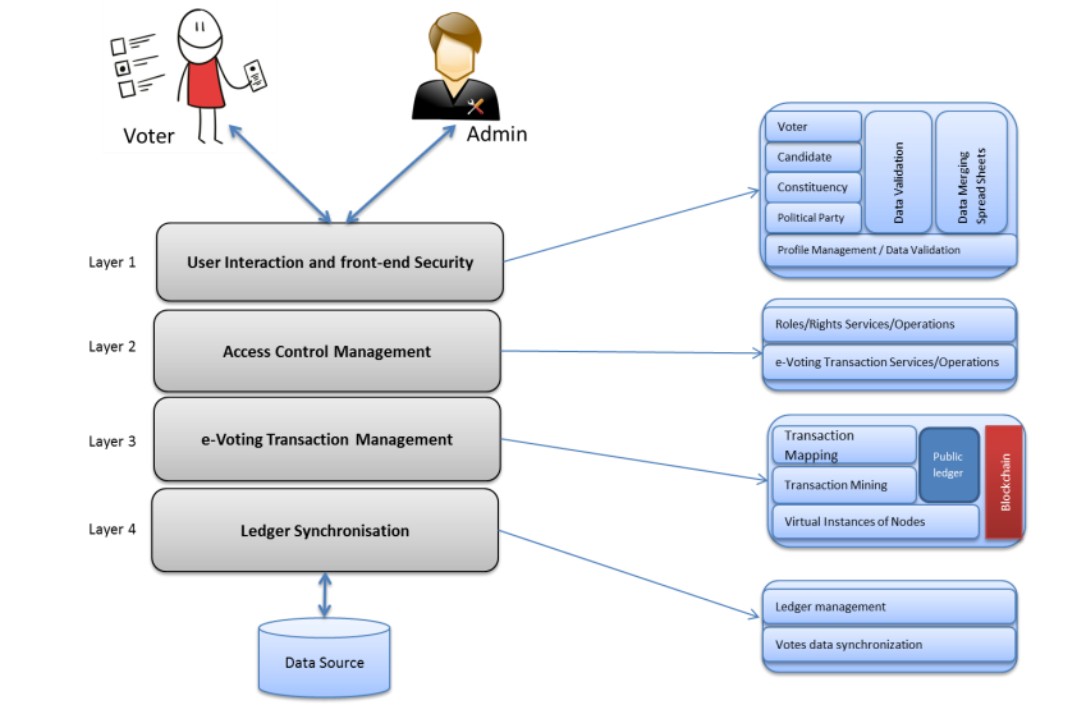
The basis of the attack being that someone could theoretically control a majority of the digital voting mining hash-rate, leading to them being able to manipulate the public ledger. The chances of this type of attack occurring are slim due to the immense cost needed to purchase hardware capable of this scale of processing. We also have the added security of an auditor who checks and keeps track of people connecting to the network and the locations of each node. This is a feature that current systems such as bitcoin lack. (Learncryptography.com, 2016) The online aspect of the voting within our system is the largest attack vector for hackers as they could potentially exploit voters through their own devices in a host of ways. To combat this software could be developed that could be downloaded onto the clients device to establish a secure connection to the polling station.

When it is time to vote, authentication of a user requires three distinct pieces of evidence; their identification number (e.g. UK citizens have national insurance numbers), the password supplied on registration, their ballot card which contains a QR code. As there are two methods of voting (web browser, physical polling station) the way the user will input the authentication details shall differ; however, in order to vote they are required to provide all three pieces of information. It is also important to note that each user will have been registered at a certain constituency so they will only be able to vote at a local polling station within that constituency or via the internet at the URL provided on the ballot card. (Each constituency is to be equipped with its own web server and URL to ensure votes are aggregated within the right network.) Behind the scenes the polling station will consult the voter blockchain to ensure the voter has not already used up their vote. If the user does have a vote, then the station will then allow the user to continue to the voting screen. If not, then system will respond to the user appropriately. See diagram Appendix B Figure 6 to see the process.

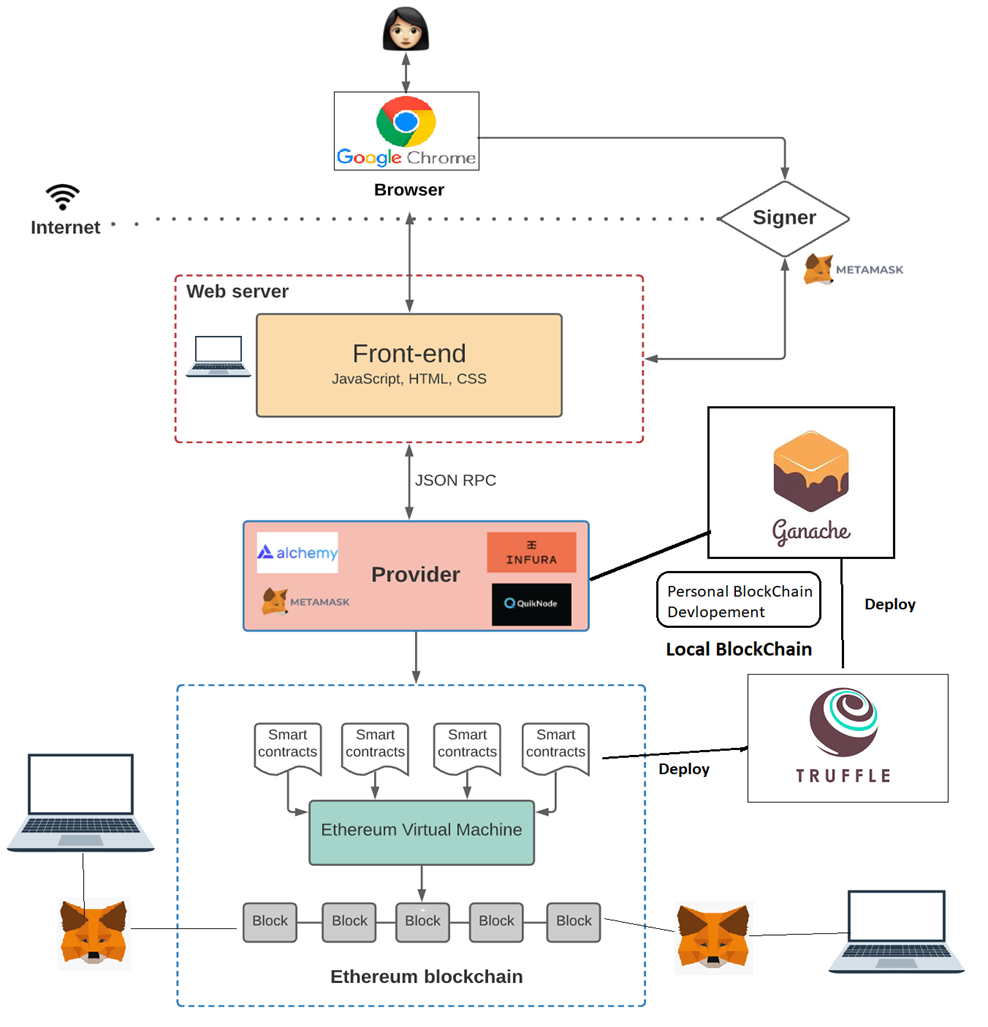
After selecting their vote (from the selection of options including abstention) and then confirming the submission, the vote will become a transaction, it will be encrypted with the relevant constituency’s public key. This transaction is then passed to the constituency node where it is added to a block and the update is then pushed to all other nodes connected to that particular constituency node. The connected nodes then pass the data on to their peers until the whole network is updated.

Once the vote has been confirmed the polling station will then 12 generate a transaction to remove the user’s vote within the voter blockchain. It is important to note that there are two distinct blockchains being held; one which contains transactions relating to which users have registered and which users still have a vote, the second containing the contents of the vote (such as what party was voted for.).

Through the use of these two distinct blockchains we ensure voter anonymity when selecting their vote.



(4.5.1 Architecture of Purposed system)



**(Fig: 4.5.2 Blockchain Voting System working flow Design)**

Preceding the introduction to our voting system, it merits mentioning that the Ethereum protocol utilized as part of our system has not been modified in any way. Our system, DigiVote, uses existing functionality and features provided by Ethereum to provide the ability for creating and voting on ballots. Our implementation consists of three smart contracts coded in

Ethereum’s Solidity language, two scripts written in JavaScript, and one HTML page. DigiVote is an open source project and the entirety of the code is available for public use . We assume the administrator, creators, and voters have the MetaMask plugin downloaded in their browser or running an Ethereum node to create and manage Ethereum accounts as well as interact with our system. We utilize Ethereum’s Web3 framework internally, this allows our users to easily manage signed transactions and interactions with the Ethereum blockchain. Using MetaMask and Web3 eliminates the need for users to download full or even partial Ethereum blockchains on their local machines in order to broadcast transactions. The only action required of users when registering, voting, or creating ballots is to use their passwords to unlock their Ethereum accounts in the MetaMask plugin and securely interact with the blockchain. If the user decides not to utilize the Meta mask plugin then they are responsible for running a node on their local machine and syncing it with the blockchain to interact with our system using Web3.

### ADVANTAGES

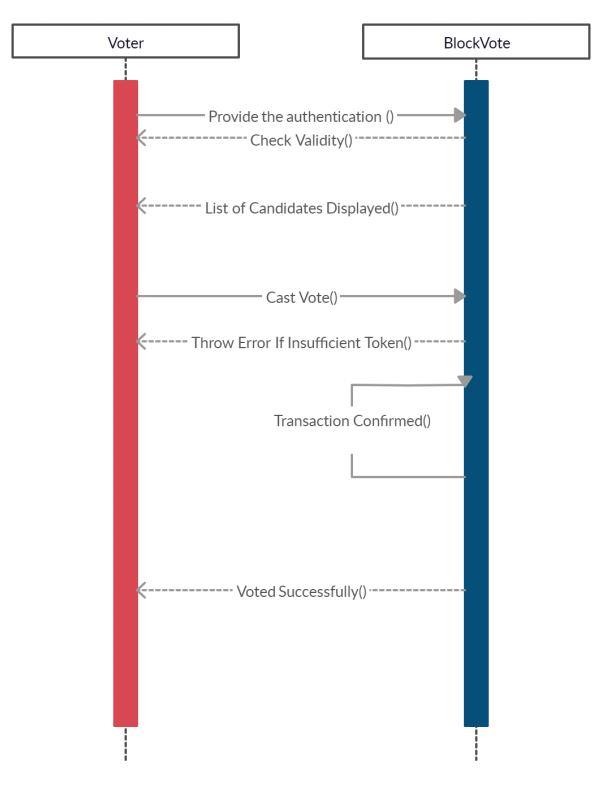
* Accurate results and speed in vote count
* Low cost of setup because just internet connection cost is required to vote across all the available e-voting platforms
* Enhanced security as voting take place over secure communication channels
* Accessibility from any corner of the world just by having an internet connection
* Fraud prevention due to less human intervention therefore avoiding the fraud that could possibly take place at the polling stations
* Reduced influence by family members or peers as voters can change their opinion until end of the voting day several times as only the last vote will be considered.

### DISADVANTAGES

### In many developing countries internet access is not available to everyone, example: In rural areas low wage workers could not afford internet also many people don’t know how to use and access the web.

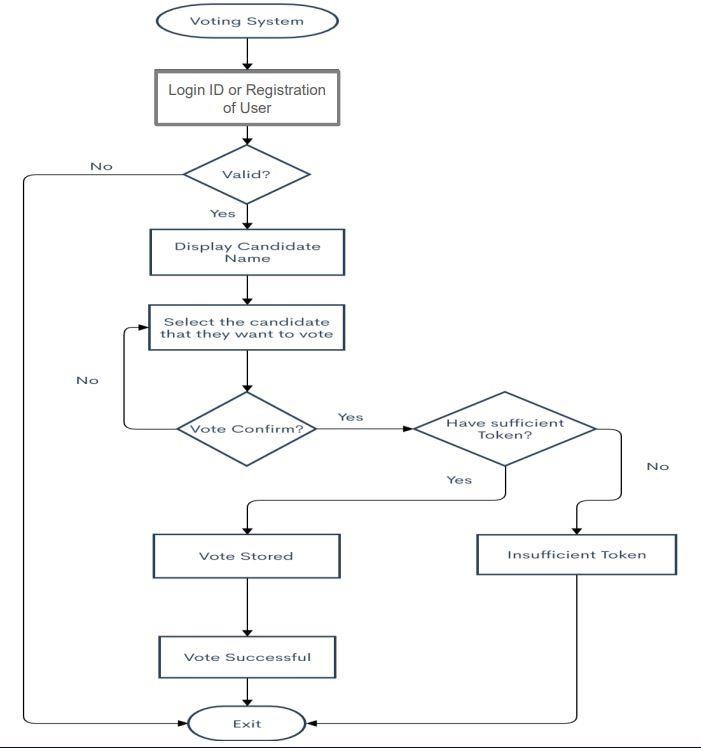
### E-voting machines use software to register the vote and it is built by a company, general public don’t know how a software works that might lead to fraudulent results being generated, vendors could also be bribed and in return they could tweak the software to work in their favour.

**4.5.3 Sequence Diagram**



(Fig: 4.5.3 Sequence Diagram)

#### 4.5.4 Flowchart



#### (Fig: 4.5.4 Flowchart)

#### 4.6 Modules

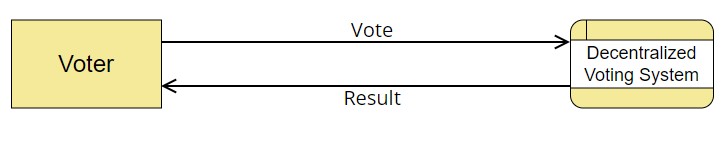
The project has been divided into many modules in which for every functionality we have designated modules. Any software comprises of many systems which contains several subsystems and those sub-systems further contains their sub-systems. So, designing a complete system in one go comprising of each and every required functionality is a hectic work and the process can have many errors because of its vast size.

Effective modular design can be achieved if the partitioned modules are separately solvable, modifiable as well as compliable. Following are the project modules:

* **Candidate:** The candidates should be a set of list. A candidate in a organization/ community who stand for election should submit the details to RA. Candidate The candidates should be a set of list For each candidate to vote can be defined as Ci.
* **Registration Authority:** The voter should register in RA to get ready to vote. The candidate should register in RA with his information and the RA will give him the id of candidate.
* **Voter:** The voters should be a set of list For each voter to vote can be defined as Vi The voter should transfer his public key (PKI/Token) to EA**:** In this module, voters who have been provided with the personal ETH wallet will import onto the voting portal using the Metamask extension and cast their vote. Voter registers in our system with a valid student/employee ID and e-mail address to vote on given ballot ID numbers.
* **Election Authority (Admin):** The EA is responsible for starting the election, creating a vote, limiting the voter numbers of the voting, paying the voting fees for the Transaction generated automatically in the backend. In this module, an entity named Election Commission will be responsible to setup the smart contract and register candidates, parties and start off an election. **Administrator** is responsible for deploying the initial Registrar and Creator smart contracts. The administrator also has the ability to grant or revoke ballot creation permission for registered voters/creators.
* **Solidity Programming:** It acts as the record and gate keeper. It keeps track of all registered voters and creators, ballot IDs, voting contract addresses, and whitelisted email domains. The information regarding the voter and different ballots are linked together in the contract. This allows the contract to perform voter verification, permission modification, and Voting address retrieval. The owner of this contract is the administrator.

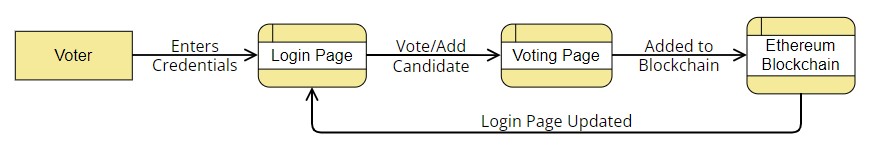
**4.7 Data Flow Diagram**

 **Level 0 data flow diagram**



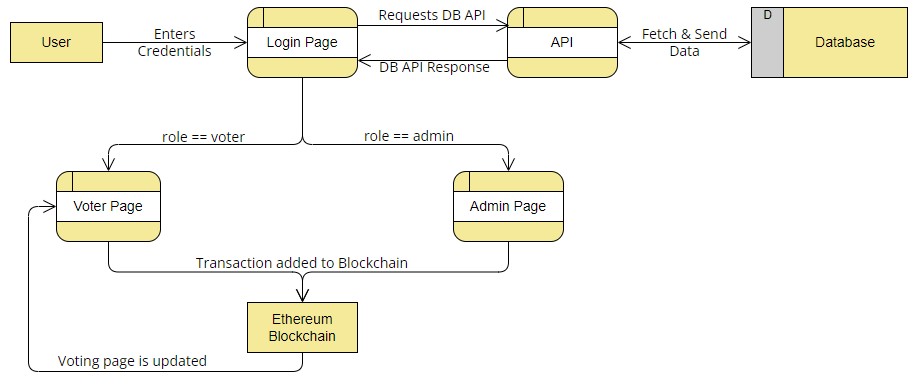
**Figure 4.7.1** Level 0 Data Flow Diagram

###  Level 1 data flow diagram



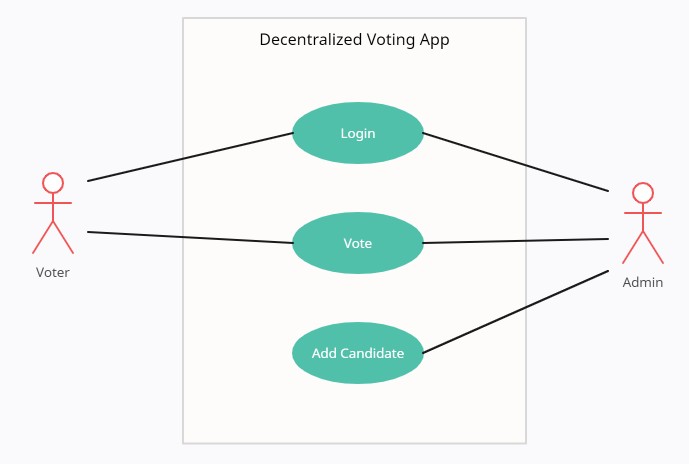
**Figure 4.7.2** Level 1 Data Flow Diagram

###  Level 2 data flow diagram



**Figure 4.7.3** Level 2 Data Flow Diagram

#### 4.8 Use Case Diagram



**Fig 4.8**

CHAPTER 5

TESTING

**Test Results**

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It includes a set of techniques and methods to identify defects, bugs, performance issues and providing a reliable and quality product. The goal is to identify issues as early as possible and improve the overall quality of the system.

#### 5.1 Test Case 1

|  |  |
| --- | --- |
| **Test Case No.** | 1 |
| **Test Type** | Unit Test |
| **Name of Test** | Checking JWT Authorization |
| **Test Case Description** | The objective of this test case is to check jwt authorization. |
| **Input** | Login and Password |
| **Expected Output** | User should not be able to login without proper authorization. |
| **Actual Output** | User cannot access voting or admin page without authorization. |
| **Result** | Pass |
| **Comments** | Working properly. |

**5.2 Testcase 2**

|  |  |
| --- | --- |
| **Test Case No.** | 2 |
| **Test Type** | Functional Test |
| **Name of Test** | Verify user login |
| **Test Case Description** | The objective of this test case is to verify that user can login to the voting portal. |
| **Input** | Voter\_id and password |
| **Expected Output** | User must be able to login if credentials match the database, else unauthorized error is shown. |
| **Actual Output** | User is able to login with correct credentials only. |
| **Result** | Pass |
| **Comments** | Working properly. |

**5.3 Testcase 3**

|  |  |
| --- | --- |
| **Test Case No.** | 3 |
| **Test Type** | Unit Test |
| **Name of Test** | Verify candidate registration |
| **Test Case Description** | The objective of this test case is to verify that candidate can be registered by admin. |
| **Input** | Candidate name and party. |
| **Expected Output** | Registration transaction should be successful. |
| **Actual Output** | Registration transaction is successful. |
| **Result** | Pass |
| **Comments** | Working properly. |

#### 5.4 Test Case 4

|  |  |
| --- | --- |
| **Test Case No.** | 4 |
| **Test Type** | Unit Test |
| **Name of Test** | Verify date registration |
| **Test Case Description** | The objective of this test case is to verify that date of voting can be specified by admin. |
| **Input** | Starting and ending date |
| **Expected Output** | Date transaction should be successful. |
| **Actual Output** | Date transaction is successful. |
| **Result** | Pass |
| **Comments** | Working properly. |

#### 5.5 Test Case 5

|  |  |
| --- | --- |
| **Test Case No.** | 5 |
| **Test Type** | Functional Test |
| **Name of Test** | Verify voting |
| **Test Case Description** | The objective of this test case is to verify that voter is able to cast their vote. |
| **Input** | Select a candidate and click “Vote” button. |
| **Expected Output** | Vote transaction should be successful. |
| **Actual Output** | Vote transaction is successful. |
| **Result** | Pass |
| **Comments** | Working properly. |
|  |  |

## CHAPTER 6

## CONCLUSION AND FUTURE ENHANCEMENT

**CONCLUSION:**

A reliable and truthful voting system is crucial for any democratic society. Democracies depend on trusted elections and citizens should trust the election system for a strong democracy. However traditional paper-based elections do not provide trustworthiness. The idea of adapting digital voting systems to make the public electoral process cheaper, faster and easier, is a compelling one in modern society.

Making the electoral process cheap and quick, normalizes it in the eyes of the voters, removes a certain power barrier between the voter and the elected official and puts a certain amount of pressure on the elected official. It also opens the door for a more direct form of democracy, allowing voters to express their will on individual bills and propositions.

This project has been developed to a blockchain-based electronic voting system that utilizes smart contracts to enable secure and cost-efficient election while guaranteeing voters privacy.

In the next build of this application, it has been proposed to create separate client designs for various roles such as one for election commission and one for candidates registered to a certain party with the existing voting client design. Also, the current versions lack authentication as we don’t have access to current Aadhar’s or Voter SDK to integrate in our application. Also, it is planned that in the next build notification prompt will be given on the day of voting to all the voters to cast their vote so that the voter turnout is maximum for that election.

**FUTURE ENHANCEMENTS:**

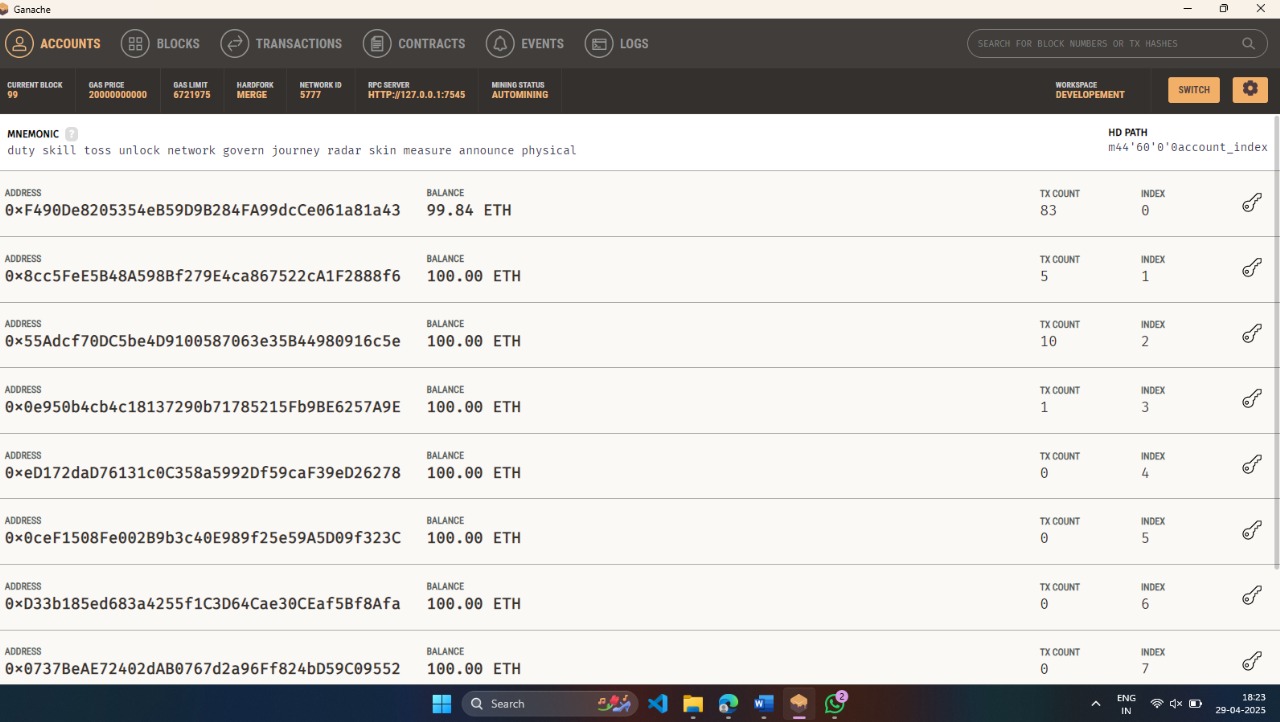
In future work, we will investigate the possibility of implementing Paillier cryptosystem as a library in Solidity. With the system we currently have, moving the cryptography to a library in Solidity could largely improve our individual ballot verifiability. Having the Paillier library in

Solidity would help us generate new private and public key for each ballot. This will help us achieve individual voter audit on different ballots without compromising the other ballots. To increase user accessibility, we will also look into integrating the Ethereum Lightwallet into our system will allow users to unlock their accounts in our UI without needing to run a node or plugin.. In continuation of this work, we are focused at improving the resistance of blockchain technology to ‘double spending’ problem which will translate as ‘double voting’ for e-voting systems. Although blockchain technology achieves significant success in detection of malleable change in a transaction however successful demonstration of such events have been achieve which motivates us to investigate it further. To this end, we believe an effective model to establish trustworthy provenance for e-voting systems will be crucial to achieve an end-toend verifiable e-voting scheme. The work to achieve this is underway in the form of an additional provenance layer to aid the existing blockchain based infrastructure

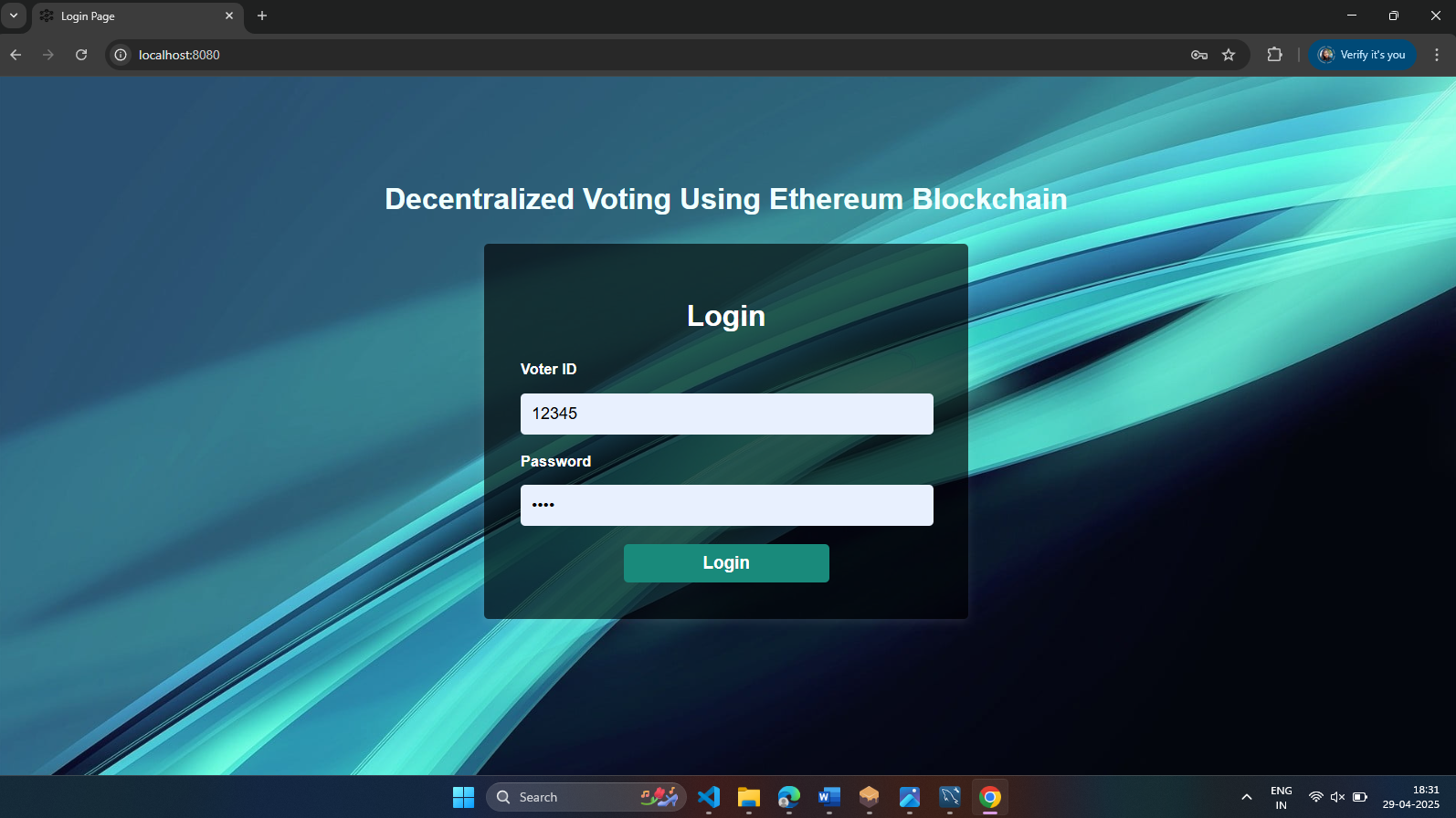
Finally, to help with voter verification, we will try to integrate an API/process that will allow us to check the validity of all e-mails used to register into our system.

SCREENSHOTS

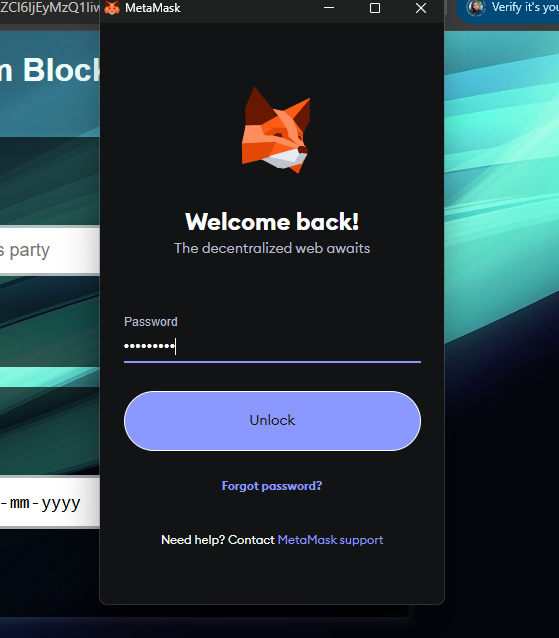
**Ganache : Smart Contract Owner Account**

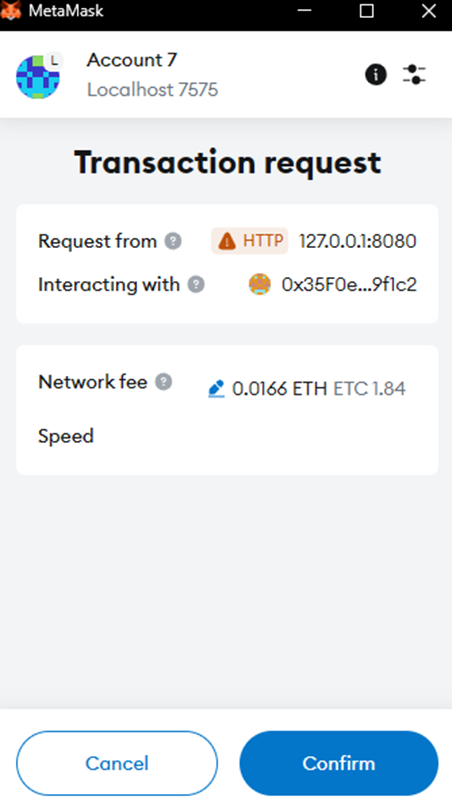
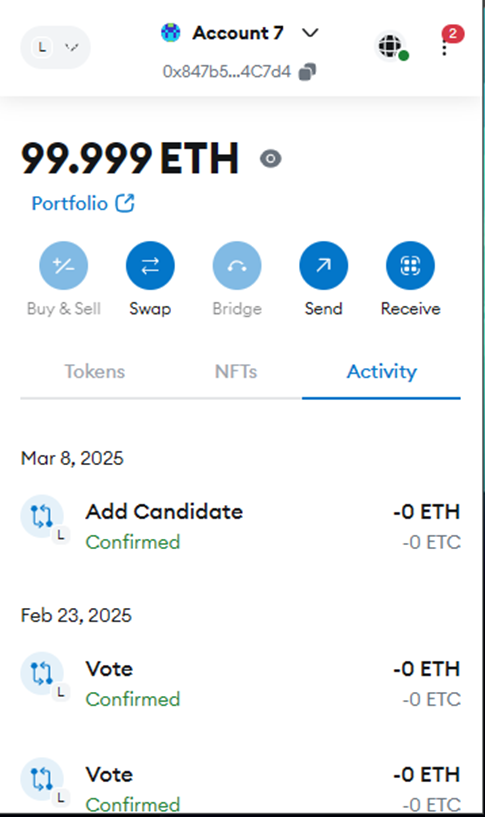


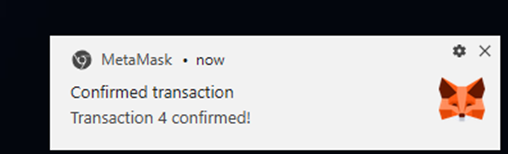
**Login Page:**



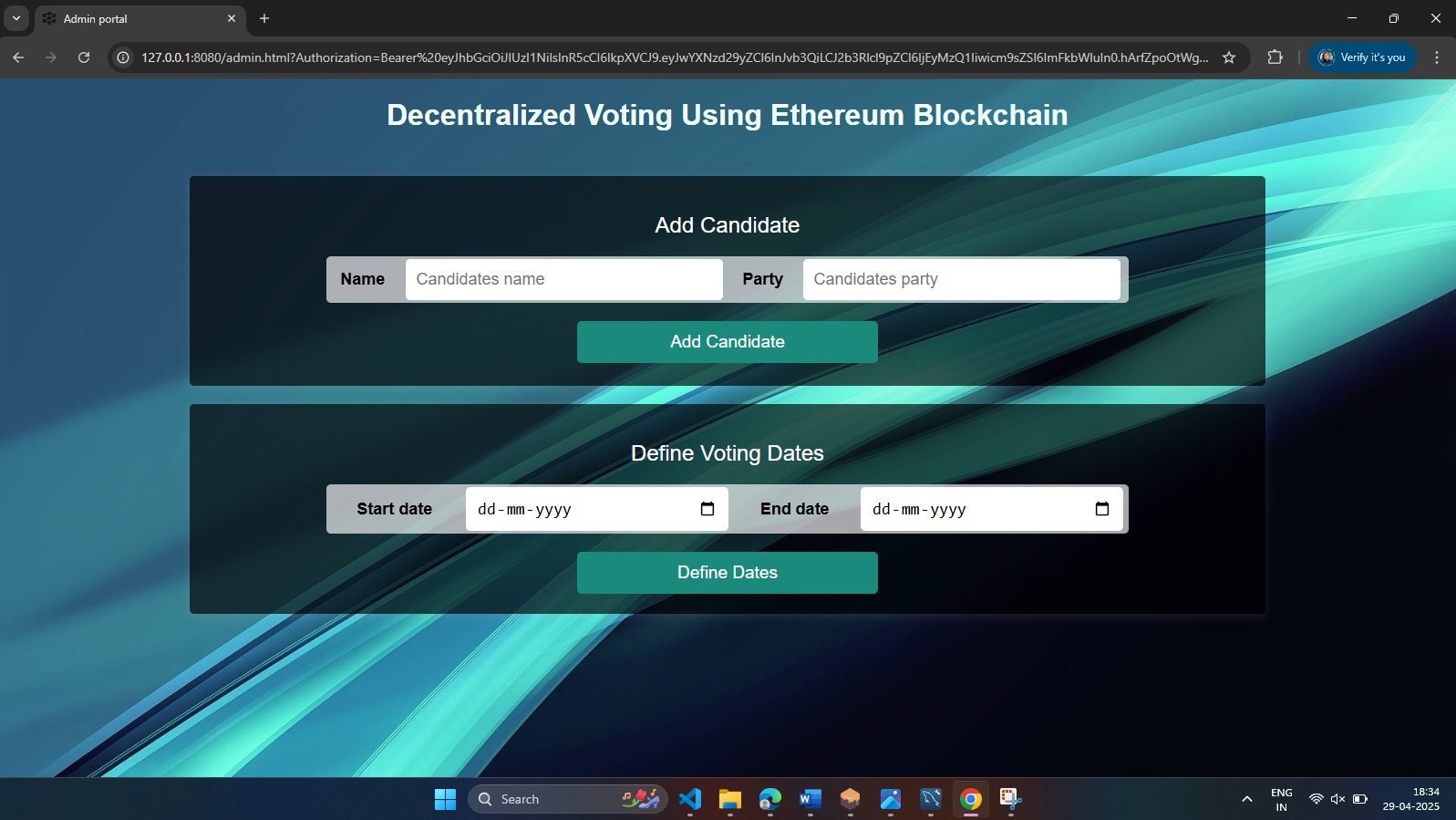
**Metamask Account:**



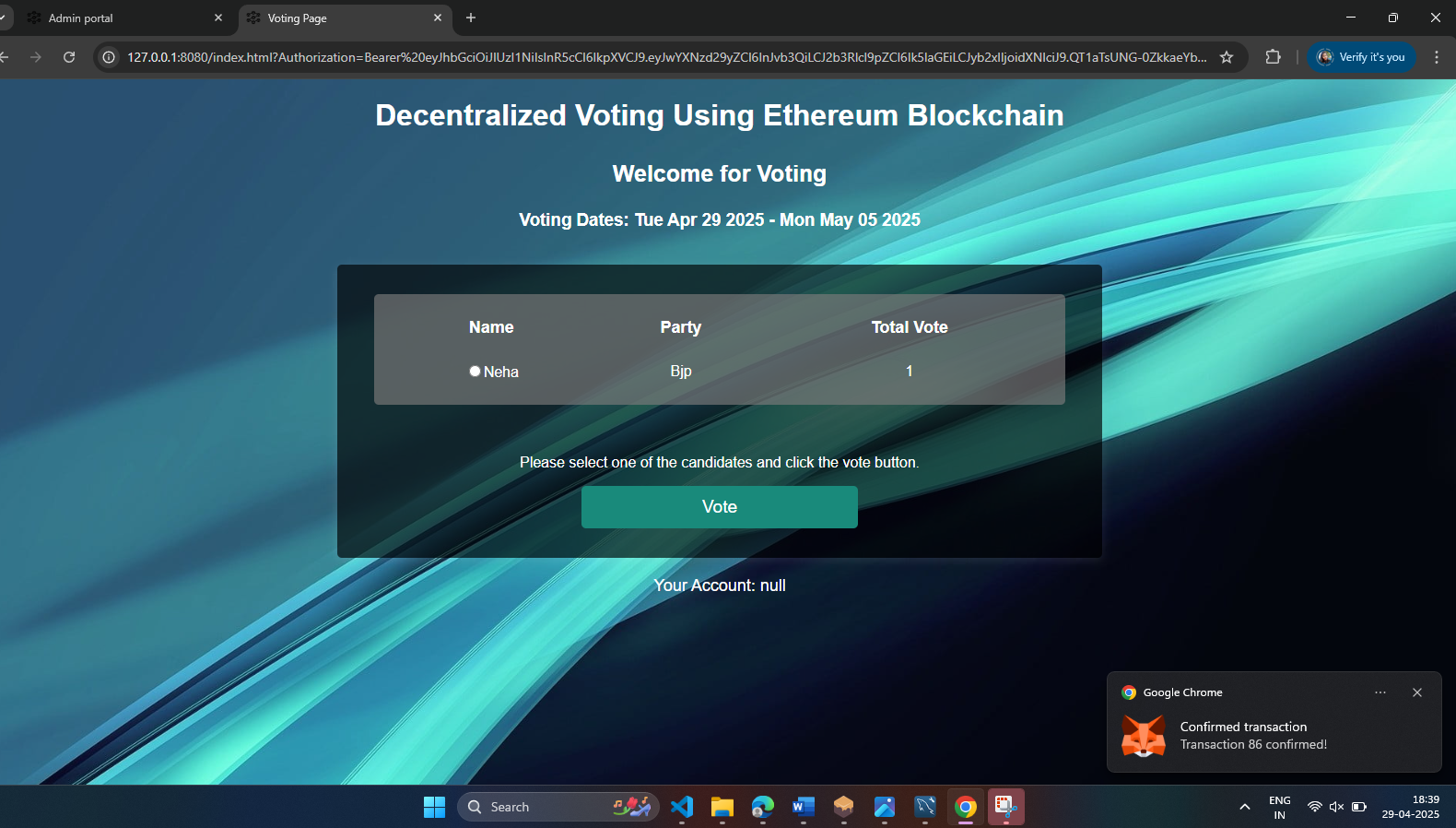




**Admin Page:**



**Voting Page:**



**REFERENCES**

1. Smith, J. (2017). "Evolution of Electronic Voting Systems." Journal of Elections and Technology, 12(3), 45-62.
2. Jones, M., & Brown, A. (2019). "Challenges in Traditional Electronic Voting Systems." Cybersecurity Journal, 8(2), 112-130.
3. Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System." Bitcoin.org.
4. Smith, A., & Johnson, B. (2020). "Blockchain in E-Voting: A Comprehensive Analysis." International Journal of Digital
5. Democracy, f), 221-240.
6. Çabuk, U. C., Adiguzel, E., & Karaarslan, E. (2020). A survey on feasibility and suitability of blockchain techniques for the e-

voting systems. arXiv preprint arXiv:2002.07175.

1. Tarasov, P., & Tewari, H. (2017). The future of e-voting. 12, 148-165.
2. F. Þ. Hjálmarsson, G. K. Hreiðarsson, M. Hamdaqa and G. Hjálmtýsson, "Blockchain-Based E-Voting System," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2018, pp. 983-986, doi: 10.1109/CLOUD.2018.00151.