N° d'ordre :PFE_RTIC 8/RTIC/M2/2022

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Réseaux et Technologies de l'Information et de la Communication (RTIC)**

Blockchain-Based Application For Untampered Voting

Par :
BENNADJI ABDELALI

Soutenu le 19/06/2023 devant le jury composé de :

Chighoub Rabiaa	MCB	Président
Guerrouf Fayçal	MCB	Rapporteur
Hamida Ammar	MAA	Examinateur

Année universitaire 2022-2023

Abstract

The process of conducting fair and transparent elections is a critical aspect of any democratic society. However, traditional voting systems have faced numerous challenges, including concerns about tampering, fraud, and lack of transparency. Blockchain technology offers a promising solution to address these issues and enhance the integrity of voting systems. This abstract presents an overview of a blockchain-based application designed to facilitate untampered voting.

The proposed application leverages the decentralized nature of blockchain to ensure transparency, security, and trust throughout the voting process. By using a distributed ledger, the application enables the creation of an immutable record of each vote, preventing unauthorized modifications or tampering. The transparency of the blockchain allows for real-time verification and auditing of the voting results by all stakeholders, including voters, election officials, and independent observers.

The blockchain-based application also addresses the issue of voter authentication. By implementing a decentralized identity management system, it ensures that only eligible voters can participate in the election. This system can integrate various identification methods, such as biometrics or government-issued digital identities, to establish a robust and secure authentication process.

In conclusion, a blockchain-based application for untampered voting provides a promising solution to enhance the integrity and trustworthiness of electoral processes. By leveraging the decentralized and transparent nature of blockchain, this application ensures secure and verifiable voting, protecting the principles of democracy in the digital age.

Keywords: Blockchain, E-voting, Smart contract, Ethereum, Crypto-currency.

Résumé

L'organisation d'élections équitables et transparentes est un aspect essentiel de toute société démocratique. Cependant, les systèmes de vote traditionnels ont été confrontés à de nombreux défis, notamment des préoccupations concernant la falsification, la fraude et le manque de transparence. La technologie blockchain offre une solution prometteuse pour résoudre ces problèmes et renforcer l'intégrité des systèmes de vote. Ce résumé présente une vue d'ensemble d'une application basée sur la blockchain conçue pour faciliter le vote non falsifié.

L'application proposée tire parti de la nature décentralisée de la blockchain pour garantir la transparence, la sécurité et la confiance tout au long du processus de vote. En utilisant un registre distribué, l'application permet la création d'un enregistrement immuable de chaque vote, empêchant toute modification ou altération non autorisée. La transparence de la blockchain permet la vérification et l'audit en temps réel des résultats du vote par toutes les parties prenantes, y compris les électeurs, les fonctionnaires électoraux et les observateurs indépendants.

L'application basée sur la blockchain aborde également la question de l'authentification des électeurs. En mettant en œuvre un système décentralisé de gestion de l'identité, elle garantit que seuls les électeurs éligibles peuvent participer à l'élection. Ce système peut intégrer diverses méthodes d'identification, telles que la biométrie ou les identités numériques émises par le gouvernement, afin d'établir un processus d'authentification robuste et sécurisé.

En conclusion, une application basée sur la blockchain pour le vote non falsifié fournit une solution prometteuse pour améliorer l'intégrité et la fiabilité des processus électoraux. En tirant parti de la nature décentralisée et transparente de la blockchain, cette application garantit un vote sécurisé et vérifiable, protégeant ainsi les principes de la démocratie à l'ère numérique.

Mots-clés: Blockchain, E-voting, Smart contract, Ethereum, Crypto-currency.

DEDICATION

To my beloved family, you are the heart and soul that sustains me. With gratitude overflowing, I dedicate these words to you.

Through the highs and lows, your unwavering love has been my anchor.

Your support and encouragement have fueled my dreams, and your presence has filled my life with joy.

In times of darkness, you've been my guiding light.

Together, we've laughed, cried, and triumphed, creating unforgettable memories.

With profound love and appreciation, I dedicate this humble tribute to the rock-solid foundation you provide.

Forever grateful for the blessing of our extraordinary bond.

ACKNOWLEDGEMENTS

In the Name of Allah, the Most Gracious, the Most Merciful
(and you have been given only a little knowledge).

Praise be to Allah, and may Allah bless our master Muhammad and his family and companions.

Praise be to Allah, who has blessed us with completing the work and completing the project within its deadline.

Then thanks to everyone who contributed to his process from near or far.

To **Dr. Guerrouf Fayçal**, all thanks and gratitude for your patience and patience with us, and for your guidance that illuminated the path of action.

To the honorable family that spared no effort to provide a suitable atmosphere for work and research.

To my mother who prays to God for me and all my colleagues in her prayers.

To my colleagues at work, each in his name, for their support and encouragement.

This humble work is dedicated to you

Contents

List of Figures

General Introduction

Elections play a crucial role in upholding democratic societies as they enable citizens to actively participate in selecting their representatives and shaping the government. However, traditional election systems, such as paper and electronic votes with centralized processes, often encounter issues regarding transparency, potential fraud, and limited accessibility.

In response to these challenges, authorities are seeking innovative solutions. This project aims to address the limitations of conventional voting systems. We propose an approach that utilizes a decentralized system to ensure immutability, robust encryption, and transparency.

This solution harnesses the power of blockchain technology to deliver these services and more. The decentralized nature of blockchain makes it an appealing option for enhancing election security and integrity, as it provides resistance against tampering and manipulation.

When applied to elections, blockchain technology offers several advantages. Firstly, it ensures transparency by creating an unchangeable and auditable record of all transactions or votes. Each vote cast on the blockchain is encrypted and tied to a unique identifier, guaranteeing its authenticity and preventing any tampering. This transparency fosters trust in the election process and enables independent verification of the results.

Secondly, blockchain enhances security. By cryptographically linking each transaction or vote to the previous block, the entire chain becomes highly resistant to modification without detection. Additionally, the decentralized nature of blockchain eliminates reliance on a single point of failure, thereby minimizing the risk of hacking or tampering with election data.

Furthermore, blockchain technology can improve accessibility and convenience in elections. Through the use of digital identities and cryptographic keys, voters can securely cast their votes remotely from any location. This eliminates the need for physical polling stations and allows individuals who may face obstacles in participating in traditional elections, such as overseas citizens or people with disabilities, to engage in the democratic process.

This master thesis is structured as follows: Chapter One presents the current state of blockchain technology, including its description, functionalities, benefits, and drawbacks.

Chapter Two focuses on voting systems worldwide, discussing their history, types, constitutional laws, the process of their creation, and the challenges they encounter.

Chapter Three outlines the design of our approach. It begins with an overview of the system's global design, describing each component and its interconnection, followed by detailed designs for each component.

Chapter Four covers the implementation of the proposed system. The master thesis concludes with a comprehensive summary in the conclusion section.

Chapter 1

Blockchain Technology

1.1 Introduction

Blockchain technology is a revolutionary concept that has gained significant attention in recent years. It is a distributed ledger technology that allows secure and transparent transactions between parties without the need for intermediaries.

In a traditional transaction, intermediaries such as banks or financial institutions are involved to validate and process the transaction. With blockchain, the need for intermediaries is eliminated, as the transaction is validated and processed by a network of computers connected to the blockchain.

The basic principle behind blockchain technology is that it allows for the creation of a decentralized, secure, and tamper-proof ledger. This ledger is maintained by a network of participants, with each participant having a copy of the ledger. Transactions that take place on the blockchain are recorded in the ledger and cannot be altered or deleted, creating a permanent record of the transaction.

The technology has several applications beyond just financial transactions, including supply chain management, digital identity, voting systems, and more. The potential uses of blockchain technology are vast, and it is considered to be one of the most significant technological innovations of recent times.

In this chapter, we will explore the underlying technology behind blockchain, its history, how it works, its potential uses and benefits, as well as some of the challenges and limitations of the technology.

1.2 Blockchain Definition

Blockchain is a chain of blocks-based distributed data system. Blockchain keeps track of all transactions on a network by acting as a distributed database or a global ledger. The blocks that include the time-stamped transactions are recognized by their cryptographic hashes.

The blocks line up in a straight line, with each block referencing the hash of the one before it to create the chain of blocks known as the Blockchain. Every node in a network that maintains a blockchain executes and records the same transactions.

The nodes in the Blockchain network duplicate the Blockchain. The transactions can be read by any node in the network [22]

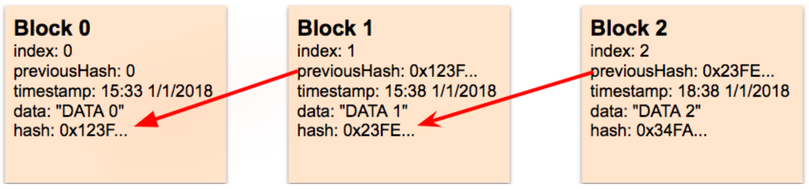


Figure 1.1: Blockchain Structure

1.3 Blockchain History

1991-2008: Early Years of Blockchain Technology
Stuart Haber and W. Scott Stornetta had the idea for blockchain in 1991. They began by constructing a chain of blocks that were encrypted. The documents' timestamps could not be altered. In 1992, Merkle trees were added to the system. As a result, the system became more effective and could accommodate more documents in a block [62].

2008-2013: The Emergence of Bitcoin
Bitcoin was created in 2008 as the first application in the history of blockchain technology. Since the Bitcoin blockchain was gaining popularity at this point, many people confuse Bitcoin and blockchain as the same thing [47].

In 2009, Satoshi Nakamoto published the first article on blockchain technology. In this article, they explained blockchain as a peer-to-peer electronic system. Nakamoto created the first block, or Genesis block, from which other blocks were mined. These blocks were connected together, creating one of the largest blockchains that contains various information [78].

2013-2015: The Development of Ethereum Bitcoin's limitations worried a programmer named Vitalik Buterin a little too much. He explained that Bitcoin has not been able to take full advantage of blockchain technology. So he set about developing a blockchain that involved more than just peer-to-peer technology. A brand new public blockchain called Ethereum was launched in 2013. She had additional functions [60].

2018: The Future

In recent years, blockchain technology has undergone many changes. In order to use this technology, numerous new initiatives have been developed. The first decentralized open-source blockchain platform, called NEO, was introduced in China. The nation has banned cryptocurrencies, but it is still making significant investments in the advancement of blockchain technology. The Chinese version of Ethereum is called NEO [7] [56].

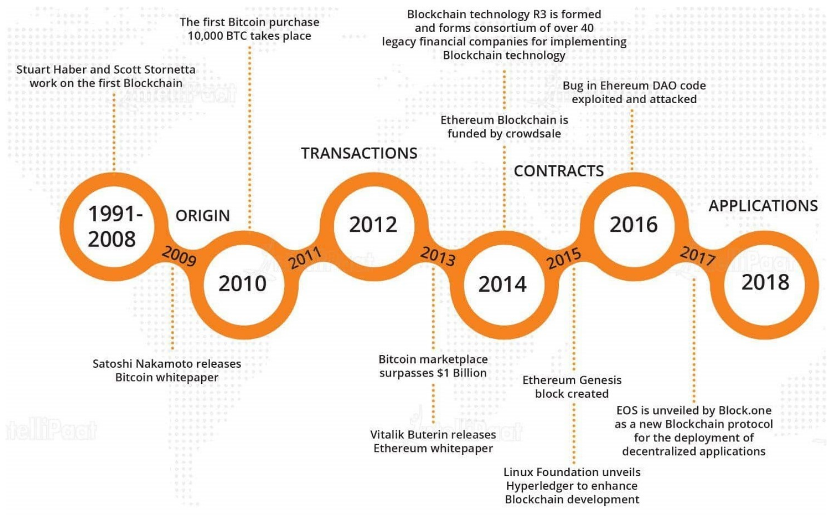

Stuart Haber and Scott Stornetta work on the first Blockchain
The first Bitcoin purchase 10,000 BTC takes place
Blockchain technology R3 is formed and forms consortium of over 40 legacy financial companies for implementing Blockchain technology
Bug in Ehereum DAO code exploited and attacked
Ethereum Blockchain is funded by crowdsale
TRANSACTIONS
CONTRACTS
APPLICATIONS
ORIGIN
1991-2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
Satoshi Nakamoto releases Bitcoin whitepaper
Bitcoin marketplace surpasses $1 Billion
Ethereum Genesis block created
EOS is unveiled by Block.one as a new Blockchain protocol for the deployment of decentralized applications
Vitalik Buterin releases Ethereum whitepaper
Linux Foundation unveils Hyperledger to enhance Blockchain development

Figure 1.2: History Of Blockchain

1.4 Types of Blockchain Architecture

Private and public blockchains are the two main categories of blockchains. However, there are many variations including consortium and hybrid blockchains. First, let's explore what the different blockchain types have in common before delving into the specifics of each type. Each blockchain consists of a cluster of peer-to-peer (P2P) nodes. Each node on the network maintains a copy of the shared registry, which is updated immediately.Each node has the ability to create blocks, send or receive transactions, and verify transactions [3].

1. ***Public Blockchain***:
   Public blockchain is a concept where anyone can participate and participate in the main activities of the blockchain network. The public blockchain is fully decentralized, has access and control over the ledger and its data is not limited to individuals, it is always available and a central authority manages all blocks on the chain [21]. Anyone can set up their own node or block in the network/chain. Once a node or block is established on the blockchain, all blocks are connected as peer-to-peer connections.

2. ***Private Blockchain***:
   To access the private blockchain, miners require authorization. The other parties engaged will not have access to it, and only the subjects participating in the transaction will be aware of it [79]. Unlike public blockchains, private blockchains are run by the organization that owns the network.

3. ***Consortium Blockchain:***
   Something like personal blockchains linked to the block chain of governmental institutions. Organizations created them, and nobody outside of those groups is allowed access. All businesses between corporations collaborate equally in consortium blockchains. They don't permit access from the consortium's or organizations' network [59].

The following table provides a detailed comparison among these three blockchain systems:

Property	**Public blockchain**	**Consortium blockchain**	**Private blockchain**
Consensus determination	All miners	Selected set of nodes	Within one organization
Read permission	Public	Public or restricted	Public or restricted
Immutability level	Almost impossible to tamper	Could be tampered	Could be tampered
Efficiency (use of resources)	Low	High	High
Centralization	No	Partial	Yes
Consensus process	Permissionless	Needs permission	Needs permission

Table 1:Comparison between the three types of blockchain

1.5 Blockchain Structure

Blockchain is a Distributed Ledger Technology. It is a distributed and decentralized database and it is secured ever as compared to other technologies [3].

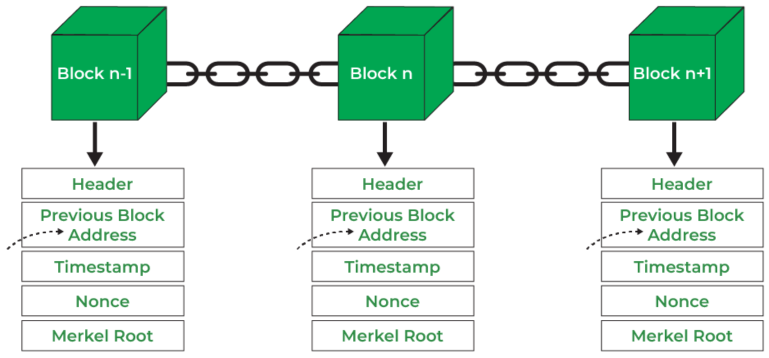


Figure 1.3: Blockchain Structure

1. ***Header***:
   It is used to identify a specific block in the overall blockchain. It supports all blockchain blocks. The block header is regularly hashed by miners by changing the nonce as part of normal mining. There are also three sets of block metadata in the block header [48].

2. ***Previous Block Address/ Hash***:
   Used to combine the i + 1 block with the i th block using a join. In short, it's a reference to the hash of the previous (parent) block in the chain.

3. ***Timestamp***:
   It is a mechanism that checks block data and gives digital documents a creation time or date. A timestamp is a string of characters that both uniquely identifies and notates the creation date of a document or an event.

4. ***Nonce***:
   a nonce that is only used once. It is a crucial component of the block's proof of work. If it is less than or equal to the current target, it is compared to the live target. Individuals who mine, test, and remove a large number of once every second until they discover a legitimate instance of a valuable once.

5. ***Merkel Root:***
   It is a frame-like data structure made up of various data blocks. A Merkle Tree creates a digital fingerprint of each transaction and saves them all together in

a block. It enables users to confirm if a transaction may or cannot be included in a block.

1.5.1 Blockchain core architecture

These are the core blockchain architecture components [54]:

- ***Node*** - user or computer within the blockchain architecture (each has an independent copy of the whole blockchain ledger).
- ***Transaction*** - smallest building block of a blockchain system (records, information, etc.) that serves as the purpose of blockchain.
- ***Block*** - a data structure used for keeping a set of transactions which is distributed to all nodes in the network.
- ***Chain*** - a sequence of blocks in a specific order.
- ***Miners*** - specific nodes which perform the block verification process before adding anything to the blockchain structure.
- ***Consensus (consensus protocol)*** - a set of rules and arrangements to carry out blockchain operations

1.6 Blockchain Layers

The layered architecture of blockchain is categorized into six layers [1] [23].

- ***Hardware infrastructure layer***
  The content of the blockchain is hosted on a server at a data center somewhere on this beautiful planet. The client-server model is used when users go online or use any applications to request content or data from application servers.

  Now, clients can communicate with one another and exchange data. A vast network of computers sharing data is referred to as a peer-to-peer (P2P) network. Blockchain is a peer-to-peer computer network that computes, validates, and accurately records transactions in a shared ledger. As a result, all of the data, transactions, and other important data are stored in a distributed database. A node in a P2P network is a computer [48].

- ***Data layer***
  The data structure of a blockchain is described as an ordered linked list of blocks. Pointers and a linked list are the two main components of the blockchain's

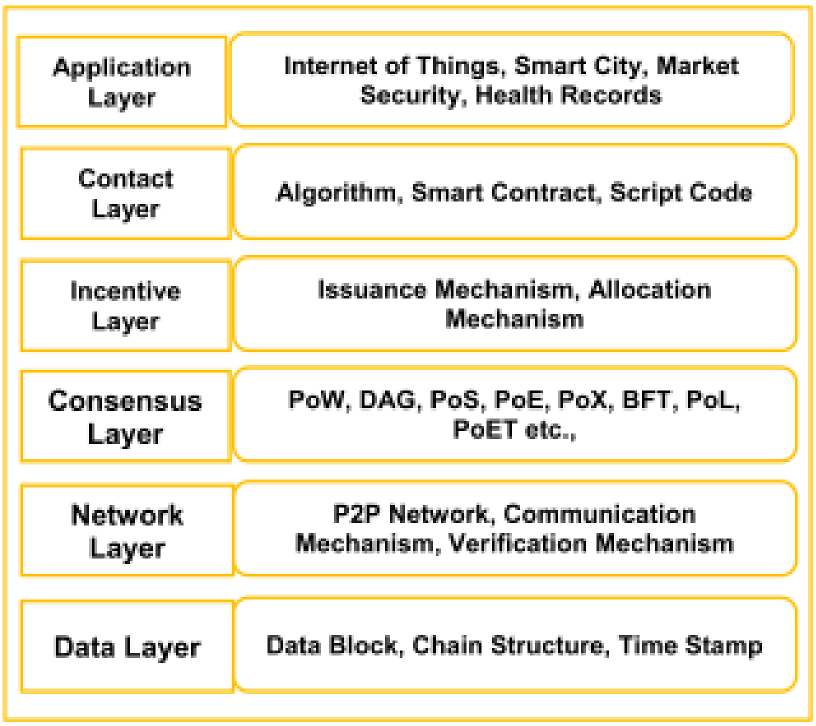

Application Layer
Internet of Things, Smart City, Market Security, Health Records
Contact Layer
Algorithm, Smart Contract, Script Code
Incentive Layer
Issuance Mechanism, Allocation Mechanism
Consensus Layer
PoW, DAG, PoS, PoE, PoX, BFT, PoL, PoET etc.,
Network Layer
P2P Network, Communication Mechanism, Verification Mechanism
Data Layer
Data Block, Chain Structure, Time Stamp

Figure 1.4: Blockchain Layers

data structure. A linked list is a collection of linked blocks that each contain data and a pointer to the block before it.

- *Network layer*
  Inter-node communication is handled by the network layer, often known as the P2P layer. The network layer is in charge of transactions, block propagation, and discovery. This layer is also known as the propagation layer.

  This P2P layer guarantees that nodes may connect with each other and communicate, share, and synchronize to maintain the integrity of the blockchain network. A peer-to-peer (P2P) network is a computer network where nodes are dispersed and share the workload of the network to accomplish a shared goal. Nodes are responsible for processing transactions on the blockchain.

- *Consensus layer*
  For blockchain platforms to function, the consensus layer is necessary. Whether it's an Ethereum, Hyperledger, or other blockchain, the consensus layer is the most important and required layer. The blocks are ordered and validated by the consensus layer, which also makes sure that everyone is in agreement.

- *Application layer*
  The application layer is composed of smart contracts, chaincode, and decen-

tralized apps (DApps). The application and execution layers are further separated into the application layer protocols. The software that end users use to interact with the blockchain network is included in the application layer. It includes user interfaces, frameworks, application programming interfaces (APIs), and scripts.

1.7 Blockchain Working

As the name implies, a blockchain is a shared, unchangeable record that divides data into units or blocks, whereas a database organizes data into tables [63].

A succession of blocks makes up a blockchain. Blocks are attached to one another after they have been filled with data. Let's use an example to better understand how blockchain functions:

Consider Jack and Phil as two nodes on the bitcoin blockchain network who wish to transact with one another. [19]

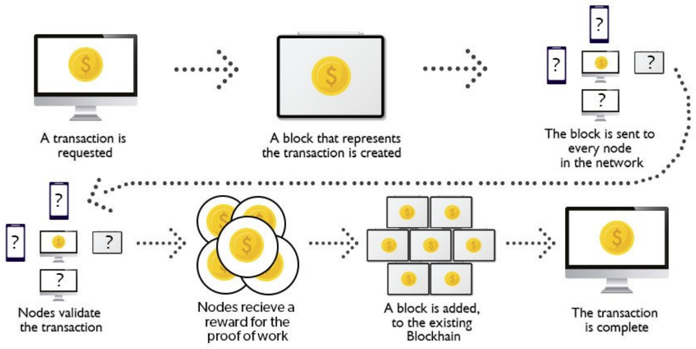

A transaction is requested
A block that represents the transaction is created
The block is sent to every node in the network
Nodes validate the transaction
Nodes recieve a reward for the proof of work
A block is added, to the existing Blockhain
The transaction is complete

Figure 1.5: Blockchain Works

Step 1: Facilitating the transaction: Jack wants to send **20 BTC** to Phil via the Blockchain network.

Step 2: Verification of transaction: All nodes on the network will get the message for verification. All nodes will verify the crucial transactional characteristics, such as whether Jack has enough balance—at least 20 BTC—to complete the transaction. Jack, a node, is he registered? The node Phil is he registered? The transaction is validated when the parameters have been checked.

Step 3: Formation of a new block: A new block is created when several validated transactions accumulate in mempools and are kept there. The block will additionally contain this validated transaction.

Step 4: Consensus algorithm: The Proof-of-Work consensus mechanism will be utilized for block verification since this discussion is about bitcoins. In proof-of-work, the system gives a node the goal hash value; as a result, it is required to generate a hash for the new block. The hash value for the new block that is below the intended value must be determined by the node. The block that presents the greatest challenge is chosen when two or more miners are mining the same block simultaneously. The remainder are referred to as stale blocks. Typically, mining pays out in blockchain currency. Bitcoin serves as the blockchain currency in this instance.

Step 5: Addition of the new block in the blockchain: Only once the freshly produced block has been added to the network and authenticated using proof-of-work will the transaction be marked as complete. Jack will give Phil 20 Bitcoin. The new block will be connected to the blockchain's open end.

Step 6: Transaction complete: The transaction will occur and 20 BTCs will be transferred from Jack's wallet to Phil's wallet as soon as the block is uploaded to the blockchain. The blockchain permanently secures the transactional facts.

1.8 Consensus Protocols

The exact way to determine if a transaction is genuine or fraudulent is provided by a consensus protocol at the heart of the blockchain network [72]. It provides a way to verify and confirm information that should be in a blockchain record. Blockchain networks often lack a central authority to decide who is right and who is wrong, so all blockchain nodes must agree on the state of the network by adhering to established protocol rules [50].

There are some common protocols of consensus:

1.8.1 Proof of Work (PoW)

Bitcoin, the first blockchain, makes use of PoW. The nodes, or "miners," use their computers to solve mathematical or cryptographic puzzles to authenticate transactions on the Bitcoin network. Bitcoin is given to miners in exchange for their efforts in validating and enabling a block record [37].

1.8.2 Proof of Stake (PoS)

Instead of miners in PoS, there are "forgers". These forgers stake a certain amount of bitcoin, giving them a chance to become a block validator based on likelihood. The necessary block transaction costs are given to the successful forger as compensation. A forger is discouraged from trying to fool the network by staking their own cryptocurrency on a block because they risk losing the stake if their fraudulently added transactions are found to be on the network [36].

1.9 Cryptography in Blockchain

In the presence of malevolent outside parties, or adversaries, cryptography enables safe communication. An algorithm and a key are used in encryption to convert an input (plaintext) into an encrypted output (i.e., ciphertext) [15].

Cryptography focuses on four different objectives [4]:

1. ***Confidentiality***: Just the individual for whom the information is meant can access it; no one else is permitted.

2. ***Non-repudiation***: The information's author or sender cannot retract his intent to send additional information in the future.

3. ***Integrity***: Information cannot be changed either in storage or during transmission from the sender to the intended recipient without the addition of new information being noticed.

4. ***Authenticity***: It is established who sent the message and who received it. The information's origin and destination are both verified.

1.10 Types of Cryptography

Understanding the different types of encryption is essential to understanding blockchain cryptography [25]. Hash functions, symmetric key cryptography, and asymmetric key cryptography are the three main methods used to run cryptographic algorithms.

1.10.1 Symmetric-Key Cryptography

Symmetric cryptography [31], commonly referred to as secret key cryptography, is the sharing of encrypted data between parties using a single shared secret. Because you use the same key to encrypt and decrypt the data, ciphers in this category are referred to be symmetric [17].

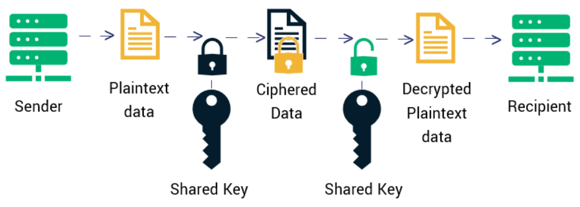


Figure 1.6: Symmetric Encryption

1.10.2 Asymmetric-Key Cryptography

Asymmetric cryptography employs two distinct keys for encryption and decryption. Each user has a private key in addition to a public key. The public key can be freely shared, but the private key must always remain a secret [26].

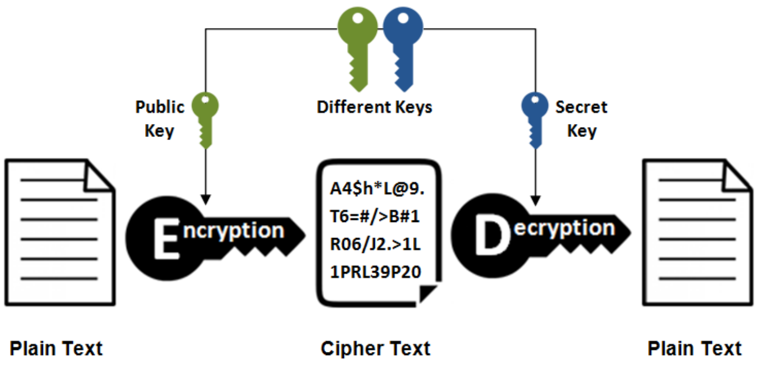


Figure 1.7: Asymmetric Encryption

1.10.3 Hash Functions

An algorithm called a hash function converts input data of any size into an output of a specific size [71]. The result is a hashed text known as a hash value. A cryptographic hash function's primary goal is to confirm the validity of data [11].

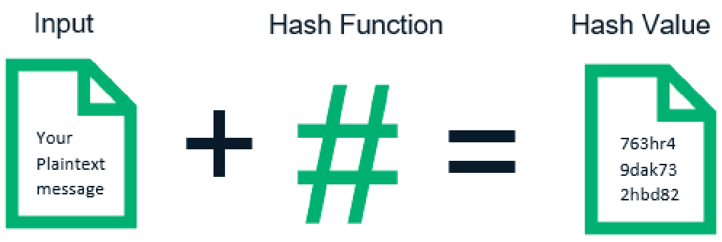


Figure 1.8: Hash Encryption

1.11 Uses of blockchain

There is a lot of wide usage of Blockchain beyond finance, We mention some of them:

1.11.1 Electronic Voting

The numerous issues that these early attempts at internet voting have revealed can all be resolved by blockchain. Because any hacker with access to the terminal cannot influence other nodes, a blockchain-based voting application is not concerned with the security of its Internet connection. Without disclosing their identities or political inclinations to the public, voters can nevertheless cast their ballots efficiently. The ability to attribute each ID to a single vote, prevent the creation of fakes, and prevent tampering allows officials to count ballots with complete confidence [8] [53].

1.11.2 Supply Chain Management

Blockchain enables businesses to conduct transactions directly and without the involvement of a third party, increasing the efficiency of global supply chains. It also enables more data sharing between stakeholders and supports increased integration of financial and logistical services [14] [69].

1.11.3 Healthcare

Blockchain's distributed ledger technology makes it easier to transfer patient medical records securely, improves the security of healthcare data, controls the flow of medications, and aids genetic code discovery in the medical field [2] [44].

1.11.4 Peer-to-Peer Transactions

Although convenient, P2P payment services like Venmo have some limitations. Certain services limit transactions depending on location. Others demand payment in exchange for use. Also, a lot of them are hacker-prone, which deters users from disclosing their financial information. With all of the aforementioned advantages, blockchain technology could remove these obstacles [12] [65].

1.12 Smart Contracts

It is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain. smart contracts are a type of Ethereum account. However, they are not controlled by the user, they are deployed across the network and work as designed.User accounts can then interact with the smart contract by sending transactions that perform the function defined in the smart contract [73]. Smart contracts can define rules like a standard contract and apply them automatically via code. By default, smart contracts cannot be deleted and interactions with them are irreversible [9].

1.13 Advantages and Disadvantages of Blockchain

Every technology has it's advantages and disadvantages that we need to know. we present it as follows [40]:

1.13.1 The advantages of the Blockchain

- ***Decentralized system***: Dealing with the third-party organization or the central administrator is not necessary. This shows that all Blockchain users are involved in decision-making and that there is no intermediary in the system [64].

- ***Trusty***: The belief of two or more participants—who do not know one another—is the foundation of the Blockchain's trust. The fundamental premise is the genuine, non-worthless exchanges between these unidentified parties.

- ***Transparency***: The transaction copying procedure makes the Blockchain transparent. Each transaction, as stated above, gets duplicated to either computer in the Blockchain network. Every participant has access to all transactions, therefore every activity is visible to everyone using the Blockchain.

- ***Security***: Blockchain technology's high level of security is attained at the point of network entrance for each user. because every user of the Blockchain receives a special ID that is connected to their account. The trustworthy chain of the cryptographic hash is blockchain security.

1.13.2 The Blockchain disadvantages

- ***Energy***: The Blockchain's high energy consumption is its biggest drawback. Power usage is required to maintain a real-time ledger. Every time a new node is created, it connects with every other node at the same time [39].

- ***Cost***: A lot of energy is required for each cryptocurrency transaction. Very little likelihood exists that technological development will be able to remedy this problem. Also, there's a chance that unresolvable energy problems will overlap with the storage problem.

- ***Immaturity***: Since blockchain technology is still relatively new, people do not place a lot of trust in it and are not yet willing to invest in it. Nevertheless, a number of blockchain-based applications are flourishing across numerous industries, but more people's trust in the technology is still needed before it can be fully utilized.

1.14 Blockchain today

1.14.1 Cryptocurrencies

There are 1,833 cryptocurrencies listed with a market cap of $200 billion, and they have been around since 2009. Cryptocurrencies are an alternative to the conventional central banking system because they can be used to exchange electronic cash while using strong encryption to ensure the security of funds and transactions. Let's learn more about a few of them [34].

1.14.1.1 Bitcoin

With the introduction of bitcoin, which has shown to be very safe and robust from a network and protocol point of view, decentralization of currency was made feasible for the first time. It is the first use of blockchain technology that combines a peer-to-peer network, according to this definition. The Bitcoin protocol is used by nodes in this peer-to-peer network to communicate with one another [75]. With Bitcoin, the double spending issue is overcome through a distributed ledger where each transaction is permanently recorded [74].

1.14.1.2 Ethereum

The Ethereum idea was first presented by Vitalik Buterin in November 2013. Ethereum is a decentralized platform that employs blockchain technology and smart contracts to handle money in order to avoid downtime and outside influence. In May 2015, Ethereum's initial release was made. It is an application that functions precisely as intended, with no chance of fraud, censorship, or downtime [28].

1.15 Conclusion

During this chapter, we outlined in detail the blockchain, its principles, and its usefulness as it became clear why many people are interested in this new technology and are thinking of developing it in order to achieve efficiency. In the next chapter, we will see an area in which we apply the blockchain, as it is completely different to digital currencies.

Chapter 2

Electoral Systems

2.1 Introduction

In the previous chapter, we represented the background of Blockchain technology, and how the characteristics made it supported by many domains and applications, one of which the Elections or Voting.

In this chapter, we will see how blockchain offers a platform for elections, and how it makes it easier, faster, and cheaper and the main aim is safer.

2.2 Definition of elections

The electoral system is a set of laws governing the procedures for casting votes in elections to the representative assembly and converting those ballots into seats in that legislative chamber. The term electoral code, which it uses to mean a broader set of rules for assessing the validity and validity of elections, denotes the process by which the electoral system decides the composition of the Parliament or Assembly, taking into account the totality of the votes cast [35].

2.3 History of Elections

Elections, the process of appointing individuals to leadership positions or making decisions on behalf of a group, have a long and varied history around the world. Here is a brief overview of the history of elections in different parts of the world [27]:

1. ***Ancient Greece***: Most male Athenians had the right to vote by the fifth century BC. Voting became very popular as many other Greek nations imitated the democratic model of Greece. Even when the Macedonian dynasty took control

of the city-states, it persisted. Even after their democratic constitutions had been curtailed, the Greek city-states still chose their own magistrates [18].

2. ***Roman Empire***: Roman tribes frequently held elections. Rome created an extremely intricate voting system that combined direct and indirect democracy. They also pioneered the secret ballot, which is today recognized as being crucial to conducting free and impartial elections. Elections in ancient Rome were sometimes violent and bloody [10].

3. ***Medieval Europe***: In certain European nations throughout the Middle Ages, elections were used to choose kings or other leaders, although wealthy aristocratic families frequently had a significant impact on the outcome. The Magna Carta, which was ratified in England in 1215 and laid the groundwork for contemporary democratic values, established the notion that even rulers were subject to the law [66].

4. ***Modern Era***: Elections are now a crucial component of democratic societies all across the world in the modern period. Modern democratic systems were made possible by the late 18th century American and French Revolutions, and several nations have subsequently adopted them. Elections are now used to choose representatives at all levels of government, including municipal, state/provincial, and national levels. In many nations, the right to vote is seen as a basic human right [18].

2.4 Features and Characteristics of Electoral Systems

A democratic electoral system can be said to be one where:

- elections are regular and fair.
- votes are of equal value.
- the will of the majority is achieved.
- the interests of minorities are taken into consideration.
- there is a high level of participation by the electorate.
- voting is accessible.

There are three main characteristics of any electoral system that determine how it works [33]:

- ***District Magnitude***: Here, the term "representatives" refers to the number of candidates who won in the district or riding. They might be either single- or multi-member ridings.
- ***Ballot Structure***:Number refers to the number of voting preferences that a voter is permitted to mark on a ballot. A single preference for a party or candidate, multiple preferences amongst parties and candidates, and ranking candidates in order of preference weighting are all options.
- ***Electoral Formula***: Given the size of the district and the chosen ballot format, this refers to the process through which votes are converted into seats. It could have cutoff points that specify the number of votes required to win an election.

2.5 Types of Electoral System

As any system in the world, the electoral system can be divided into a single winner and multiple winners [38].

2.5.1 Single winner systems

To elect a single winner is to elect the candidate with the most votes, this guides us to other types of single-winner systems like:

1. **First-past-the-post voting (FPTP)**: Each voter may select just one candidate, and the one receiving the most votes wins. The winner may get a plurality of votes, an absolute majority of votes (greater than 50%), or neither. The system is often referred to as a simple plurality or a single member plurality (SMP). This type of system is also known as a simple majority system when there are just two candidates. FPTP is used in the United Kingdom, Canada, India, the United States, Algeria and many other countries [67].

2. **Two round system (TRS)**: Every voter selects one candidate. A candidate is deemed the winner if they earn more than 50% of the votes. Otherwise, the other candidates are removed, and the top two vote-getters are selected as the only contenders for a second round of voting. Each voter in the second round has the opportunity to select one of the two surviving candidates, and the winner is determined by receiving the most votes. The method, often referred to as run-off voting and double-ballot, is frequently referred to as 2RS. The TRS is used in countries such as France, Mali, Togo, Egypt, Iran, Belarus and Ukraine [24].

2.5.2 Multiple winners systems

To elect multiple winners, one could of course simply extend the first-past-the-post system and elect multiple candidates with the highest number of votes. There are many systems in multiple winner systems like [52]:

1. **Block vote (BV)**: A voter may select n candidates. The top n candidates are then proclaimed the victors after the candidates are ranked according to the amount of votes they have received. Multiple non-transferable votes and plurality-at-large voting are other names for the procedure (MNTV) [61].
   The BV is used in Lebanon, Bermuda, Laos, Thailand, Kuwait, the Philippines and other countries.

2. **Single non-Transferable Vote (SNTV)**: method, in which voters are only permitted to select one candidate, can be viewed as a blockvote system. The fundamental advantage of SNTV is that it facilitates the representation of minority parties. The system gets more proportional the more seats there are in the constituency. The SNTV system is used for parliamentary elections in countries such as Jordan, Taiwan, Japan [41].

2.6 Advantages and disadvantages of Electoral systems

As every system [49] any election system has some advantages and disadvantages, and here are some of them:

2.6.1 Advantages of Elections:

- **Voice of the people:** The best method for individuals to express themselves and have a say in who governs them is via elections. It empowers people to decide who will lead them and what policies will have an impact on their daily life.

- **Peaceful transfer of power:** A peaceful transition of power from one government to another is made possible through elections. This aids in preserving stability and averting disputes that can result from governmental changes.

- **Legitimacy:** Elections give the government and the policies it enacts legitimacy. People are more inclined to recognize and view as legitimate leaders those who were chosen via free and fair elections.

- **Accountability:** Politicians are held to a higher standard by elections. Politicians that break their promises or engage in dishonest behavior are more likely to lose support from the electorate.
- **Encourages political participation:** Elections promote involvement in and commitment to politics. They provide individuals the chance to learn more about political problems and engage in politics on a deeper level.

2.6.2 Disadvantages of Elections

- **Campaign financing:** Elections can be expensive and require significant funds. This can create a system where participants who have access to the most money are more likely to win, even if they don't have the best rules.
- **Low voter turnout:** Election turnout is low, which might result in under representation of several societal groups. As a result, policies may be created that do not benefit all people equally.
- **Negative campaigning:** Political campaigns can involve harsh attacks between candidates rather than a focus on substantive matters. This may result in a decline in voter turnout and a lack of confidence in the political system.
- **Manipulation and fraud:** Elections may be rigged or influenced, producing unjust results. Political instability may result from this, which might undermine the legitimacy of the administration.
- **Polarization:** Political polarization, in which people grow increasingly divided and less amenable to compromise, can be exacerbated by elections. This may result in deadlock and a lack of advancement on crucial policy matters.

So, to reduce the costs and improve voter participation, as well as to make the election process more secure and legitimate, the solution is to use high-tech that can handle the perfect security and transparency for the voters: an E-voting system based on blockchain technology [45].

2.7 Electronic voting

To reduce the impact of the disadvantages of traditional elections, high technology offers a solution: electronic voting, where anyone can cast a vote, anywhere, any time, and more secure.

2.7.1 Definition of Electronic voting

Electronic voting or e-voting allows electors to vote electronically and/or count votes in elections. Electronic voting is an alternative to traditional forms of voting, such as postal voting or ballot papers. The term e-voting encompasses several forms of voting using modern information technology. On the one hand, this term refers to the use of electronic devices for casting and counting votes, such as voting machines or vote counters. On the other hand, electronic voting also includes electronic voting methods in which voting takes place outside the polling station, such as online voting [6].

There are some countries around the world how apply the E-voting: India, Brazil, Estonia, Netherlands, Ireland, Paraguay and USA [57].

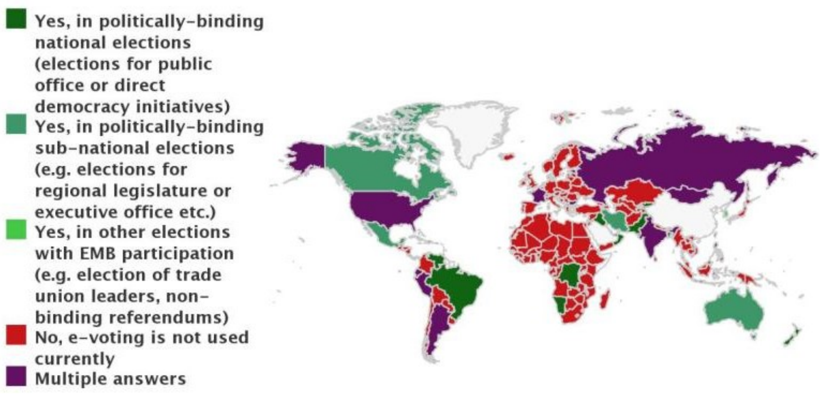

Yes, in politically-binding national elections (elections for public office or direct democracy initiatives)
Yes, in politically-binding sub-national elections (e.g. elections for regional legislature or executive office etc.)
Yes, in other elections with EMB participation (e.g. election of trade union leaders, non-binding referendums)
No, e-voting is not used currently
Multiple answers

Figure 2.1: Electronic Voting around the world

2.7.2 E-voting Stranght Points

Here some stranght points of E-voting[77]:

- Faster vote count and tabulation.
- Reduction of spoilt ballot papers as voting systems can warn voters about any invalid votes (although consideration should be given to ensuring that voters are able to cast a blank vote should they so choose).

- Increased convenience for voters.
- Cost savings by using Internet voting.

2.7.3 E-voting Weaknesses Points

Here some weak points of E-voting [77]:

- Possible breach of the vote's confidentiality, particularly in systems that handle both voter authentication and voting.
- System certification is needed, but there aren't any generally accepted certification standards.
- Potential of fraud including extensive manipulation on the part of a small number of insiders.

2.8 Blockchain-based E-Voting system

The use of blockchain in voting allows votes to be tracked and counted while being visible to everyone, assuring voters of a safe and untampered voting system [45]. This using has some benefits to mentions :

1. Blockchain ensures the security of votes by introducing a seemingly impenetrable system.
2. Anonymity is provided to the voters while voting to encourage more people to take part in the voting system.
3. Tackling the processing time in traditional system blockchain can gather and process votes quickly and efficiently.
4. In case of manipulation of the system such as stealing or change of votes, the changed data could be identified easily as other connected nodes are in sync.
5. Providing the solution to a DDoS attack, this attack might result in a few nodes becoming online but once the nodes are brought back and synced the system will work avoiding any single point failure, ensuring consistency and availability.
6. Considering long-run use, the system is considered to be cheaper and safer than any other standard database applications in the long run.
7. Online voting provides an increase in the participation of voters by giving accessibility to the elderly, disabled, and reluctant youth.

2.9 Examples of Blochchain in Voting systems

In this section, we present some of the states of the art relevant e-voting systems that use blockchain as a service.

2.9.1 Agora

Is a complete and auditable blockchain-based voting solution for governments and institutions [13]. Agora uses its tokens on the blockchain for elections that governments and institutions buy them for from every eligible voter [43] [68].

2.9.2 Netvote

Is a blockchain-based decentralized voting network runs on the Ethereum blockchain [68]. Netvote uses decentralized optimized applications (dApps) for the system user interface. The Admin dApp allows election administrators to set up elections rules, create ballots, set and open registration rules and vote closed. Voter dApp is used by individual voters for registration, voting and can be integrated with other devices (like biometric readers) for voter identification [43].

2.9.3 Votem

Votem is a blockchain-based mobile voting technology that enables voters from around the world to vote online conveniently and securely. Votem simply believes that voting should be easy and impenetrable to cheating. Votem is doing everything in its power to change the way we vote and believes that mobile voting would make a huge difference if they brought modern voting to the world [16].

2.9.4 Voatz

Voatz is a privately held American business that generates revenue by offering a blockchain-based private mobile voting application. One of Voatz's advantages is that their system will provide three records so users can confirm their vote and ensure it was tallied. The ballot receipt, paper ballot, and blockchain's data are all included. An "Anonymous ID" will encrypt all of these records. In contrast, the verification procedure may be carried out with confidence in these three documents[55].

2.9.5 Horizon State

Is a corporation that performs the function of creating a transparent, verifiable, and reliable digital voting platform that is tamper-resistant utilizing blockchain technology (Horizon State, 2022). Each customer receives a dedicated server infrastructure from them. As the market is not highly confidence in e-voting, this creates a competitive advantage. The security and customizability capabilities have unquestionably risen thanks to the special server cluster[55].

2.10 Conclusion

In conclusion, election systems are crucial for the functioning of democratic societies. As they provide a mechanism for citizens to choose their representatives and hold them accountable. There are various types of election systems, and it is important to ensure that election systems are designed to be fair, transparent, and accessible to all eligible voters.

Blockchain technology has been proposed as a solution to some of the challenges faced by traditional election systems. By providing a secure and transparent way to record and count votes, blockchain could help to prevent fraud and increase trust in the electoral process. Additionally, blockchain-based systems could potentially increase voter accessibility and participation, particularly for those who are unable to physically attend a polling station.

Chapter 3

Design of blockchain-based e-voting

3.1 Introduction

Decentralized apps (DApps) are computer programs that run on a distributed computing system. These have been popularized recently by the distributed ledger technologies underlying projects such as Bitcoin and Ethereum. Unlike the client-server architecture that powers most internet apps, DApps interact with app logic deployed on a blockchain, enabling transparent, verifiable, and immutable records of each transaction. When built on blockchain networks, DApps can contain their own suite of associated smart contracts that are used to encode business logic and allow persistent storage of state.[46].

The state of the art that we have established in the previous chapters has allowed us to understand and clearly situate the basic concepts for the design and realization of our project. Thus, in the process of developing our system, we propose a design that will describe it in an unambiguous way.

In this chapter, we will present the global design, which explains the components of this system. On the other hand, we have made detailed design modeling with the UML (using case diagrams to give a global vision of the functional behavior of our system and sequence diagrams). to make a scenario of our system). Then we made a data flow diagram to explain how an activity works. We end this chapter with a conclusion.

3.2 Global Architecture

In this section, we propose an overall blockchain-based system architecture that helps solve the problems of electronic voting, as shown in Figure 3.1. The overall system architecture consists of a private blockchain with a public interface, a server, and a blockchain. It's also consists of users (voters, and candidates), and the administrator of the system.

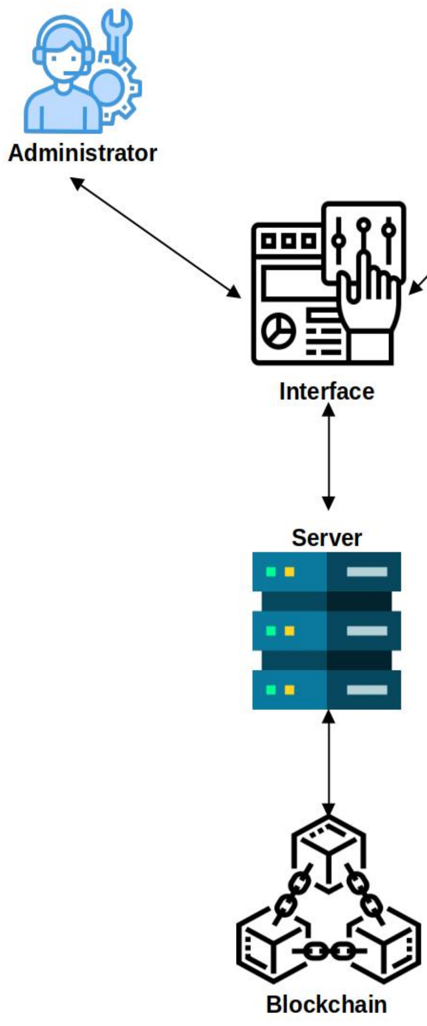


Figure 3.1: Global Architecture

3.2.1 Interaction of user Sign in

If any citizen went to vote, a form had to be filled out by entering the details through an interface, such as first and last name and telephone number. These information's are sent to the server, where it asks the blockchain to check for the existence of any preview accounts; if there is not, a new account will be created; otherwise, a notification will be sent to the user that the account exists, with a suggestion to update its own information.

3.2.2 Interaction to cast a vote

After the user gets an account, he can display the list of candidates. After choosing the candidate, a casting vote is done, which is stored in the blockchain as a confirmation of choice. After the confirmation, a notification will be sent to the user by the interface.

3.2.3 Interaction of administrator with the system

The administrator verifies the list of voters by requesting a blockchain query to find information about a specific voter. Also, the administrator has the authority to add candidates, update their information, and delete them from the database if any of them violate any rule of the election system. After the deadline for the election, the admin can display the results to know the winners.

3.3 Use case System

The purpose of using use cases in software development is to capture and describe the functional requirements of a system from the perspective of its users or external entities. Use cases provide a high-level view of the system's behavior and help stakeholders understand how the system will be used to achieve specific goals or tasks.

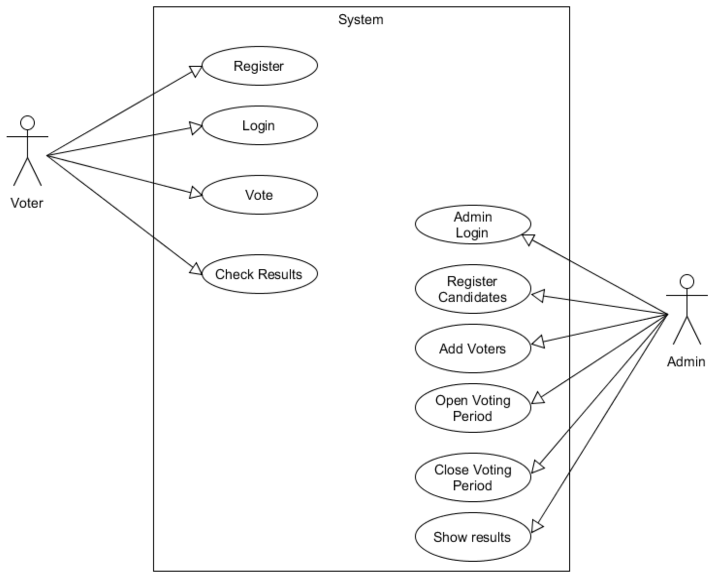


Figure 3.2: Use case of the system

As shown in Figure 3.2, our proposed system contains two main actors: the administrator, who is the system's master ruler, and the voter, who casts the vote. Each actor in the system has some operations (use cases) to do. The administrator has six functions to perform:

1. Login to the system as an admin.
2. Register the candidates for the vote.
3. Add voters by approving their demand for registration.
4. Open the vote period to give people the opportunity to vote.
5. Close the vote period after the delay is over.
6. Finally, show the results of the votes cast.

And the voter, the second actor, how has four functions to perform:

1. Register to the vote.
2. After the registration, login into the system to cast the vote.
3. Make the vote by choosing one of the candidates.
4. Check the results after the close of the vote period.

3.4 Detailed architecture

Sequence diagrams are a type of interaction diagram in software engineering and systems analysis. They depict the flow of messages and interactions between various objects or components within a system over a specific period of time. The purpose of sequence diagrams is to visualize the dynamic behavior of a system and understand how different objects collaborate to achieve a particular functionality or use case.

3.4.1 User Sequence Diagram

The user sequence diagram in Figure 3.3 shows all user interactions with system scenarios, with registration occurring first. The user has to fill out the form by entering information such as his full name, Ethereum address, and phone number through the interface. The server collects data about the voter that will be approved by the administrator of the system. After the approval, the voter can now login, and he can display the list of candidates, cast a vote, and view the vote result after the end of the voting period. If he is a candidate, the same scenario applies as for a voter.

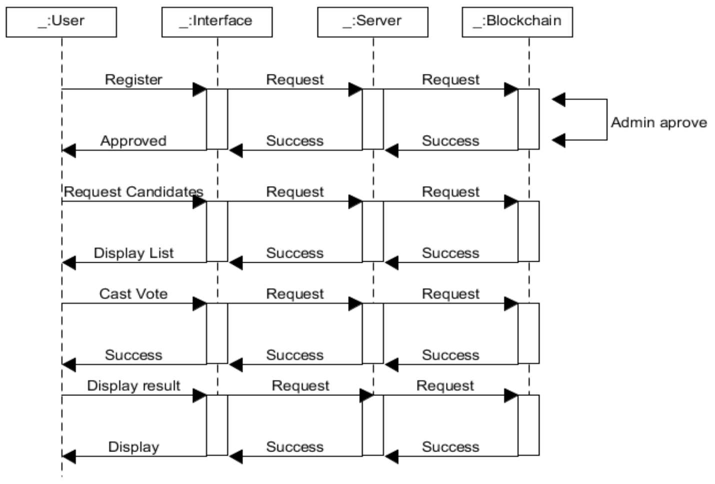


Figure 3.3: User Sequence Diagram

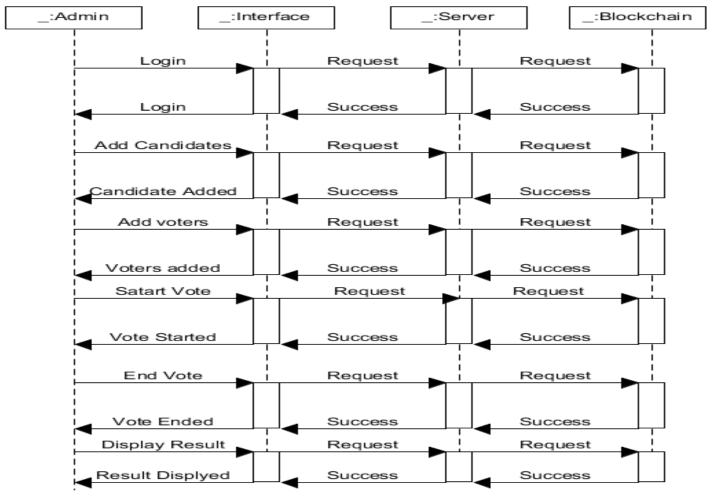


Figure 3.4: Admin Sequence Diagram

3.4.2 Administrator Sequence Diagram

The admin sequence diagram in Figure 3.4 shows the admin's interactions with the system. To access the system, the administrator must first authenticate himself by entering the necessary information in a form such as his full name, e-mail, etc. After that, he can add the candidates, approve or deny the voters, and start and end the vote period. At the end, the admin can display the results.

3.4.3 Data Flow Diagram

The purpose of data flow diagrams (DFDs) is to visually represent the flow of data within a system or process. DFDs provide a clear and concise way to depict how data moves from one component to another, highlighting the inputs, outputs, and transformations that occur.

The Figure 3.5 represent the flow-data of the system, The process starts with checking the registration of the voter; after confirming the registration, he can login and display the candidate list; after that, he can cast the vote; after the successful casting of the vote, a transaction to the blockchain is made; after that, the voter can view the result of the vote and logout of the system.

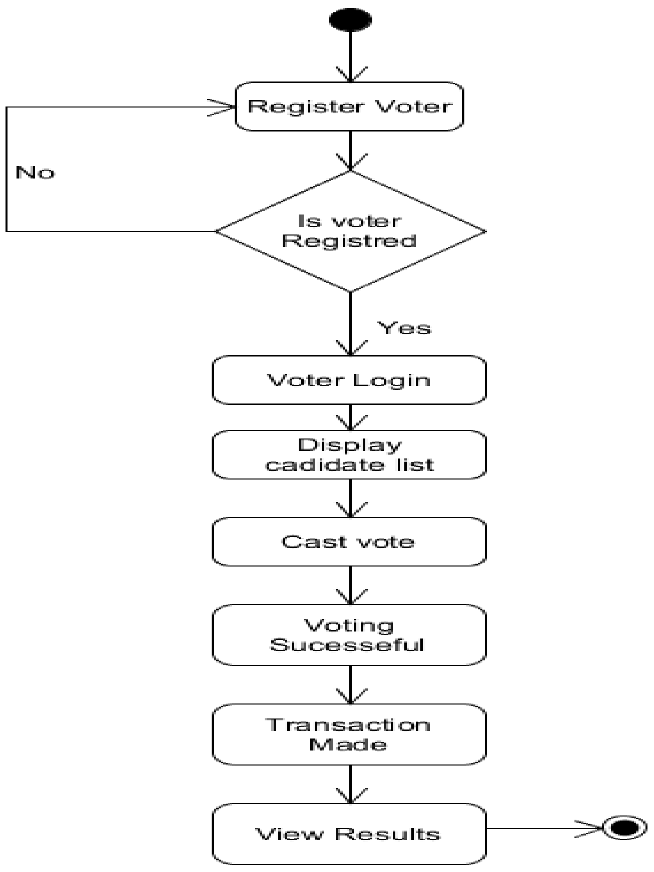


Figure 3.5: Data Flow Diagram

3.5 Conclusion

In this chapter, we first presented the overall design and components of the system, and second, we presented the detailed design using Unified modeling language (UML). The next chapter is devoted to the implementation phase of our systems.

Chapter 4

Implementation

4.1 Introduction

An implementation is a transformation of the system design to an application that could run on hardware. Thus, in this chapter, we first present the different tools used in this transformation, and second, we present the result of the implementation.

4.1.1 System Configuration and Operating System

The project is run on an Intel(R) Core(TM) i5-6300U processor running at 2.40GHz or 2.50GHz and using 8 Go of memory. The project is carried out with Windows 10 Pro version 22H2.

4.1.2 Node.js

A JavaScript runtime environment that is open-source and cross-platform is called **Node.js**. It is based on the V8 JavaScript engine in Chrome, which runs and parses JavaScript code. Because of its event-driven, non-blocking I/O approach, Node is quick and light. In order to set up our environment for creating smart contracts, we first install the Node Package Manager **(NPM)** that comes with Node.js [51].

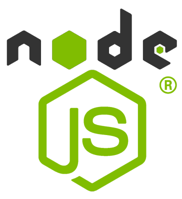


Figure 4.1: Node.js Logo

4.1.3 Ganache

Developers frequently utilize Ganache, an open source software, to build local blockchain networks for testing and development. By using Ganache to simulate a blockchain network on their local PC, developers can efficiently test various situations and troubleshoot their blockchain applications. Ganache facilitates the rapid creation of distributed apps utilizing Ethereum [70].

Figure 4.2: Ganache Logo

4.1.4 Metamask

MetaMask is a crypto wallet that can be used on Chrome, Firefox, and Edge browsers as a plugin. Metamask enables users to access the Web3 ecosystem of decentralized applications (dapps) [30].

Figure 4.3: Metamask Logo

4.1.5 Remix IDE

Remix IDE is an open-source program with flexible access to many useful plugins and the convenience of user-friendly graphical user interfaces. With the Solidity programming language, it can be a suitable partner for engineers throughout the smart contract life-cycle. More importantly, Remix can also serve as a viable training environment for mastering Ethereum. Solidity newbies should also note that Remix IDE is available as a desktop, online, or VS-Code extension [76].

Figure 4.4: Remix-IDE Logo

4.1.6 Truffle

Truffle Framework is a popular development environment for Ethereum, a decentralized contract platform. These contracts are apps that work exactly as programmed without the possibility of downtime, censorship, or third-party interference [29].

Figure 4.5: Truffle Logo

4.1.7 Web.js

Interacting with the Etherem Blockchain using a developing client. It is a set of libraries that enables you to carry out operations like sending ether between accounts, reading and writing data from smart contracts, creating smart contracts, and many other things [58].

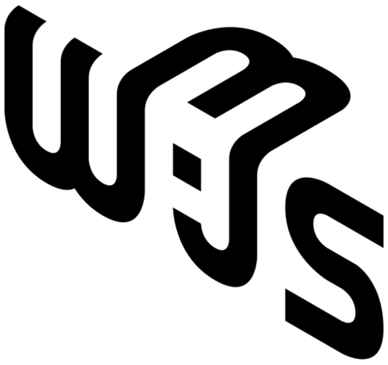


Figure 4.6: Web3.js Logo

4.1.8 React.js

The React or React.js Framework is an open source JavaScript framework and library developed by Facebook. It is used to quickly and efficiently create interactive user interfaces and web applications with much less code than regular JavaScript [42].

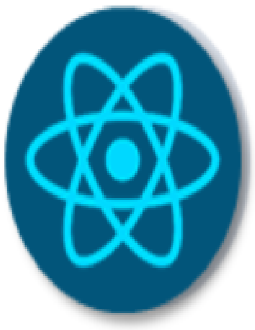


Figure 4.7: React.js Logo

4.1.9 Visual Studio Code

On the desktop, VS-Code or Visual Studio Code is a quick yet effective source code editor that runs on Windows, macOS, and Linux. In addition to a robust ecosystem of extensions for other languages and runtimes (such C++, C#, Java, Python, React, PHP, etc.), it provides built-in support for JavaScript, TypeScript, and Node.js [5].

Figure 4.8: Visual Studio Code Logo

4.1.10 JavaScript

A language for writing scripts that generate dynamic web page content. It creates things like dropdown menus, animated graphics, and dynamic backdrop colors to enhance how website visitors interact with online sites [20].

Figure 4.9: JavaScript Logo

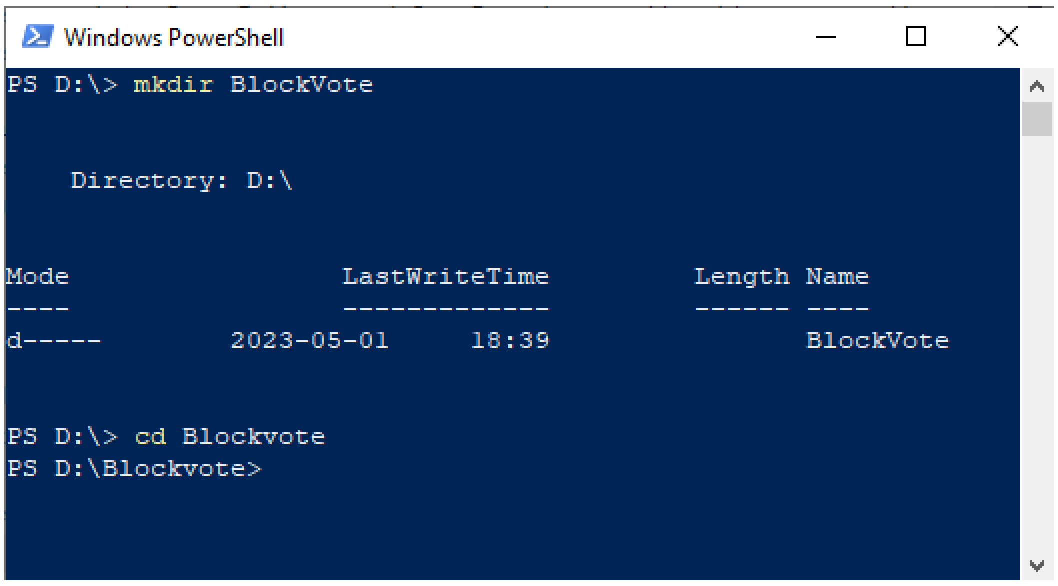
4.1.11 Solidity

A high-level, object-oriented language used to build smart contracts, which are computer programs that control how accounts behave within the Ethereum state.C++, Python, and JavaScript are all influences on the Ethereum Virtual Machine (EVM)-specific programming language known as Solidity. In addition to supporting inheritance, libraries, and advanced user-defined types, Solidity is statically typed [32].

Figure 4.10: Solidity Logo

4.2 Environment configuration

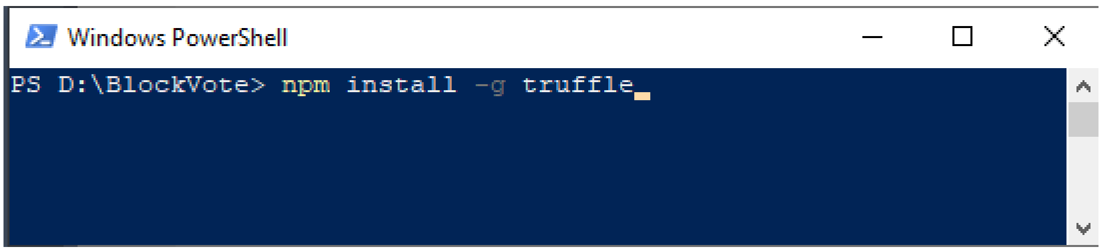
First, using Power-Shell at the command prompt, we will create a directory that will house our project files Fig 4.11:



```
PS D:\> mkdir BlockVote


    Directory: D:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2023-05-01     18:39                BlockVote


PS D:\> cd Blockvote
PS D:\Blockvote>
```

Figure 4.11: Create Project Directory

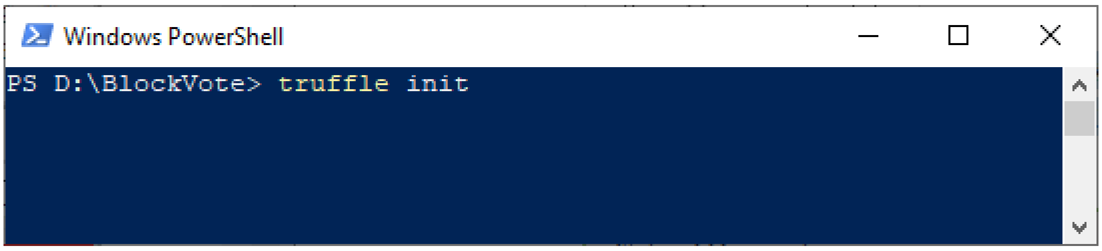
Before starting, Truffle must be installed on your computer (make sure you have NodeJs already installed). To do this, open a terminal or power-shell terminal and run this command Fig 4.12:



```
PS D:\BlockVote> npm install -g truffle_
```

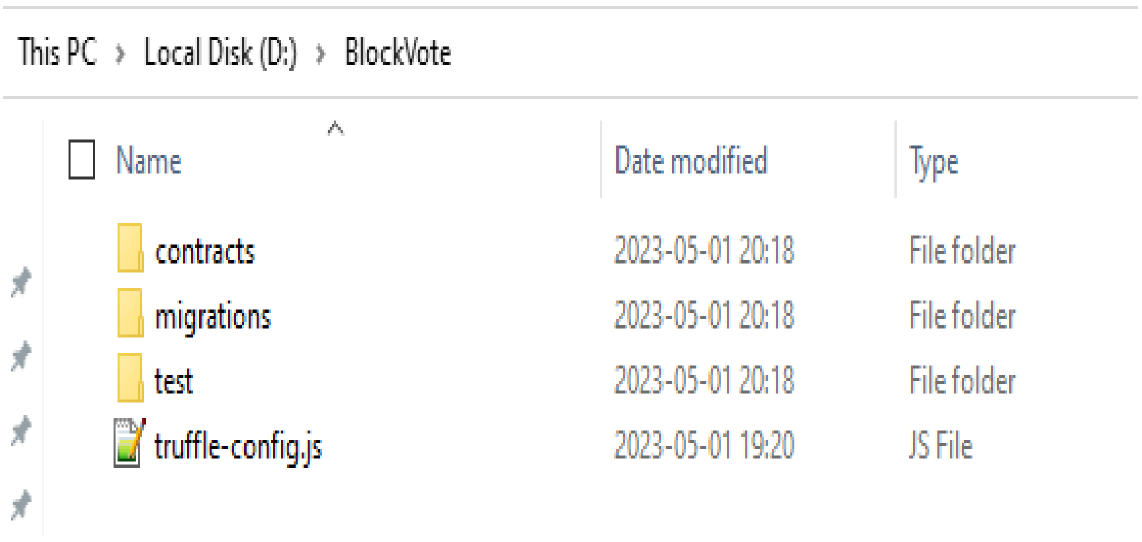
Figure 4.12: Install truffle compiler

Next let's create our back-end project using Truffle, we run **truffle init**, this will set up the following basic structure in our directory shown in Fig 4.13 and Fig 4.14



```
PS D:\BlockVote> truffle init
```

Figure 4.13: truffle init command



This PC › Local Disk (D:) › BlockVote

Name	Date modified	Type
contracts	2023-05-01 20:18	File folder
migrations	2023-05-01 20:18	File folder
test	2023-05-01 20:18	File folder
truffle-config.js	2023-05-01 19:20	JS File

Figure 4.14: Directory structure of project

The figure 4.14 present the basic structure of directory witch has:

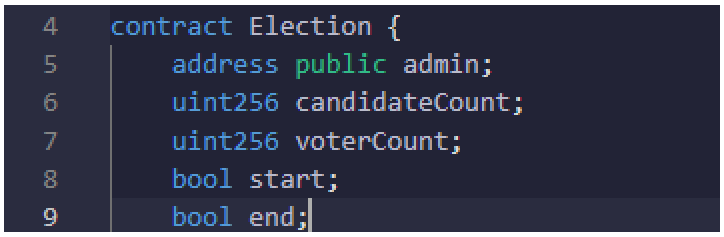
- **contract**: This folder contains the smart contracts of the project.
- **migrations**: The folder here is where the scripts for contract deployments are.
- **test**: This folder contains the files to test the application and the smart contracts.
- **truffle-config.js** : This is Truffle's configure file. For example, here we will put the network that we will use to deploy our application (localhost, testnet ... etc).

4.3 Writing Smart Contract

To build the decentralized application, the first thing to do is create an Ethereum smart contract named **Election.sol** in the contracts directory. The Solidity compiler here is the default version (the latest one). We use **Remix-IDE** in this step to write and compile the smart contract, before implement it in **vscode**.

Five state variables were defined in the smart contract as Fig 4.15 :

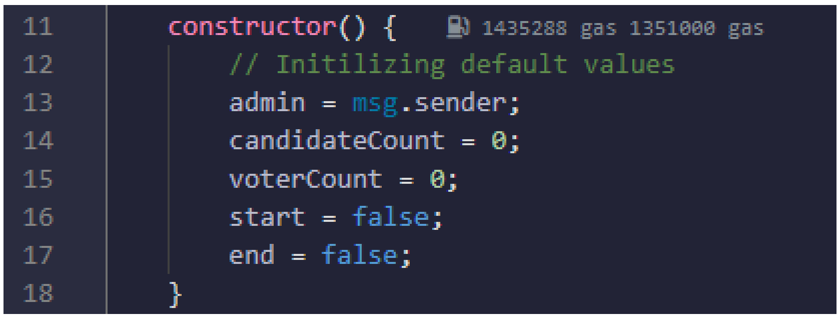
1. **address public admin:** Address of the administrator of the contract, which is the account for deploying the smart contract.
2. **uint256 candidateCount:** Candidate count with an unsigned integer of 256 bits in size to store the number of added candidates to the contract.
3. **uint256 voterCount:** : Votercount with an unsigned integer of 256 bits in size to store the number of voters added to the contract.
4. **bool start:** A boolean variable Start by defining the start of the election.
5. **bool end**: A boolean variable End by defining the end of the election.



```
contract Election {
    address public admin;
    uint256 candidateCount;
    uint256 voterCount;
    bool start;
    bool end;
```

Figure 4.15: State Variables of the smart contract

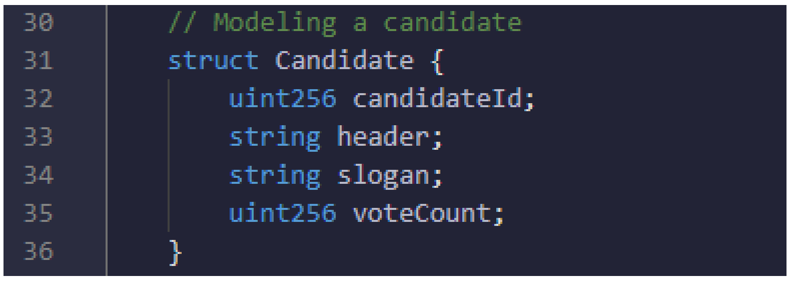
Because the Solidity language is influenced by language-oriented programming, a constructor is included in the smart contract to initialize the state variables, as Fig 4.16.



```
constructor() {    1435288 gas 1351000 gas
    // Initilizing default values
    admin = msg.sender;
    candidateCount = 0;
    voterCount = 0;
    start = false;
    end = false;
}
```

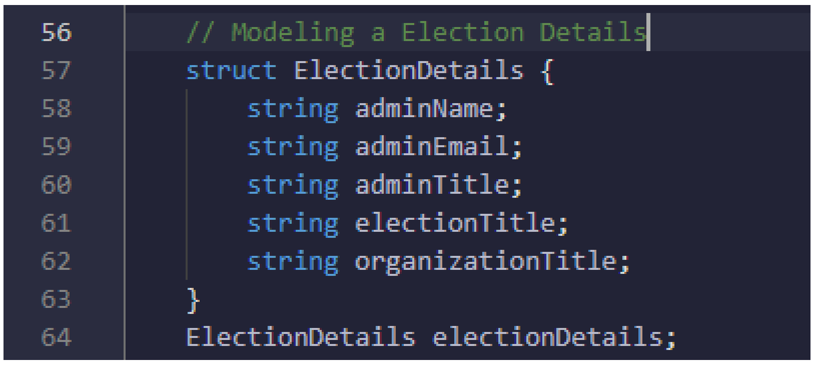
Figure 4.16: Constractor of the smart contract

Data structures were created with attributes by creating a **candidate struct**, an **election details struct**, and a **voter struct**. These structures store all the needed attributes. as Fig 4.17, Fig 4.18, and Fig 4.19.



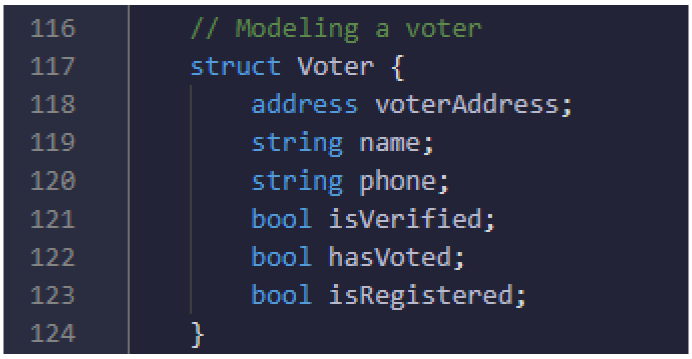
```
// Modeling a candidate
struct Candidate {
    uint256 candidateId;
    string header;
    string slogan;
    uint256 voteCount;
}
```

Figure 4.17: Struct Candidate



```
// Modeling a Election Details
struct ElectionDetails {
    string adminName;
    string adminEmail;
    string adminTitle;
    string electionTitle;
    string organizationTitle;
}
ElectionDetails electionDetails;
```

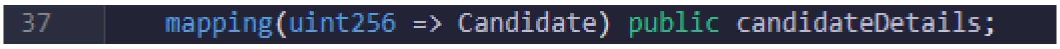
Figure 4.18: Struct Election Details



```
// Modeling a voter
struct Voter {
    address voterAddress;
    string name;
    string phone;
    bool isVerified;
    bool hasVoted;
    bool isRegistered;
}
```

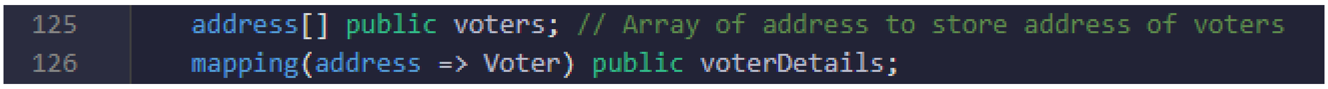
Figure 4.19: Struct Voter

Solidity's mappings are a hash table that stores information as key-value pairs, where a key can be any of the built-in data types (aside from reference types) and the value can be of any type. as Fig 4.20 and Fig 4.21.



```
mapping(uint256 => Candidate) public candidateDetails;
```

Figure 4.20: Mapping candidate details

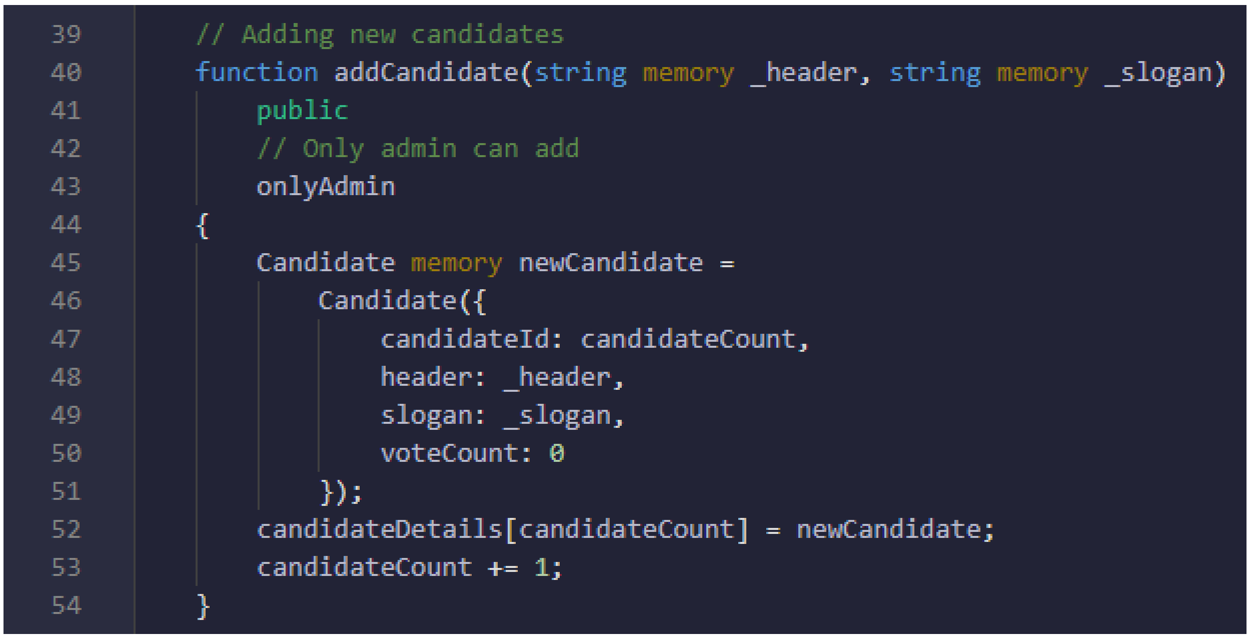


```
address[] public voters; // Array of address to store address of voters
mapping(address => Voter) public voterDetails;
```

Figure 4.21: Mapping Voters

Now, let's talk about the functions in the Solidity smart contract. There are two kinds of them: functions and callback functions, the same as setters and getters in other languages.

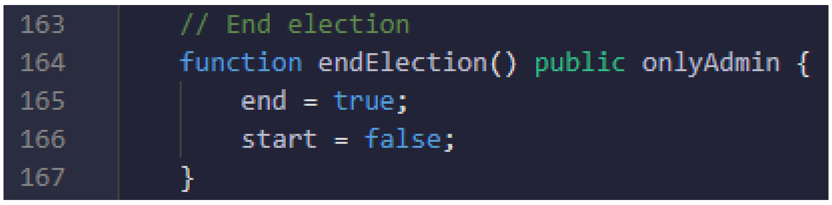
In the smart contract of the project, six (06) functions are set as setters :



```
// Adding new candidates
function addCandidate(string memory _header, string memory _slogan)
    public
    // Only admin can add
    onlyAdmin
{
    Candidate memory newCandidate =
        Candidate({
            candidateId: candidateCount,
            header: _header,
            slogan: _slogan,
            voteCount: 0
        });
    candidateDetails[candidateCount] = newCandidate;
    candidateCount += 1;
}
```

Figure 4.22: Function Add Candidate

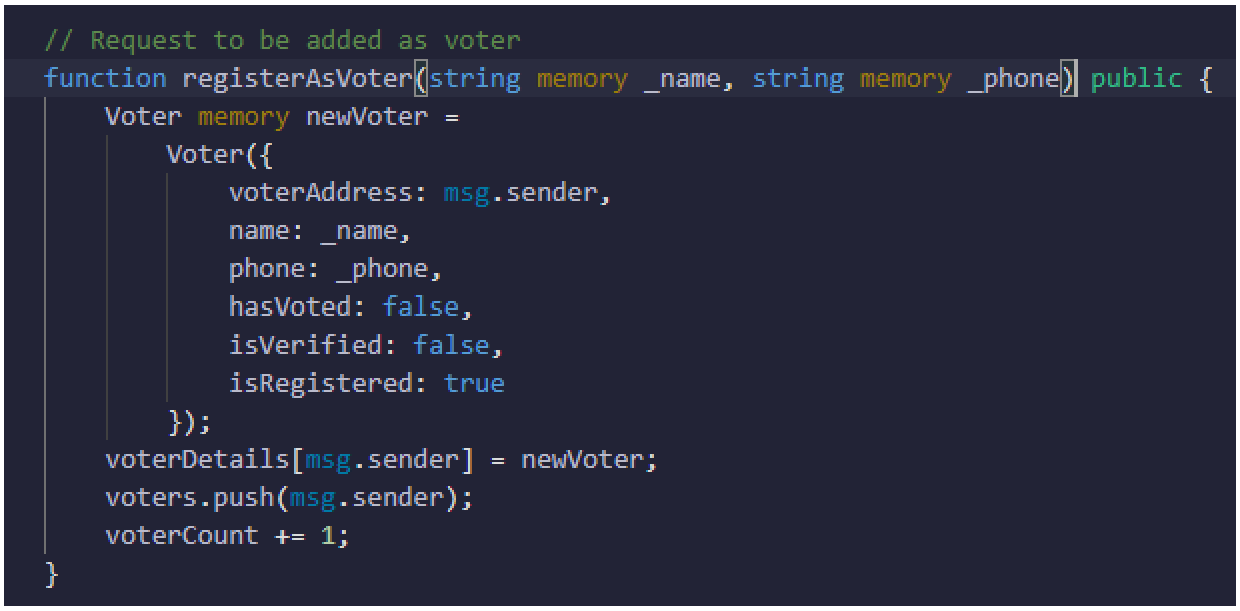
In this Fig 4.22, int the function **addCandidate**, only the administrator has the right to add new candidates to the application, the main information about the candidate is the full name, and the slogan (Why does he present himself as a candidate?).



```
// End election
function endElection() public onlyAdmin {
    end = true;
    start = false;
}
```

Figure 4.23: Function End Election

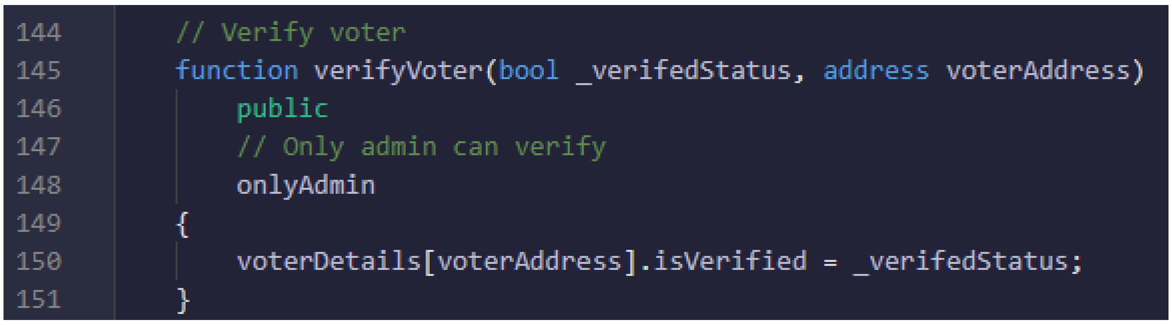
In this Fig 4.23, the function **endElection** has two Boolean variables, end with true, and start with false.



```
// Request to be added as voter
function registerAsVoter(string memory _name, string memory _phone) public {
    Voter memory newVoter =
        Voter({
            voterAddress: msg.sender,
            name: _name,
            phone: _phone,
            hasVoted: false,
            isVerified: false,
            isRegistered: true
        });
    voterDetails[msg.sender] = newVoter;
    voters.push(msg.sender);
    voterCount += 1;
}
```

Figure 4.24: Function Register as Voter

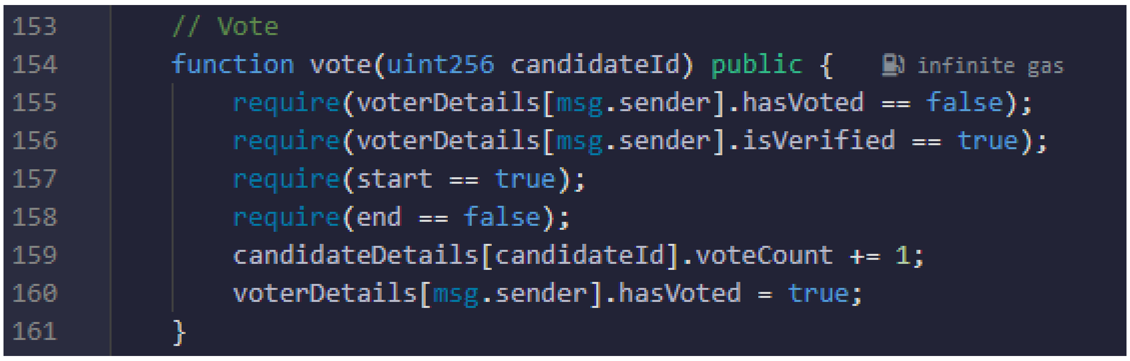
The figure Fig 4.24, present the function **registerAsVoter**, which is a request from the voter to be accepted into the election application after the registration with the electronic address, name, and phone number.



```
    // Verify voter
    function verifyVoter(bool _verifedStatus, address voterAddress)
        public
        // Only admin can verify
        onlyAdmin
    {
        voterDetails[voterAddress].isVerified = _verifedStatus;
    }
```

Figure 4.25: Function verify voter

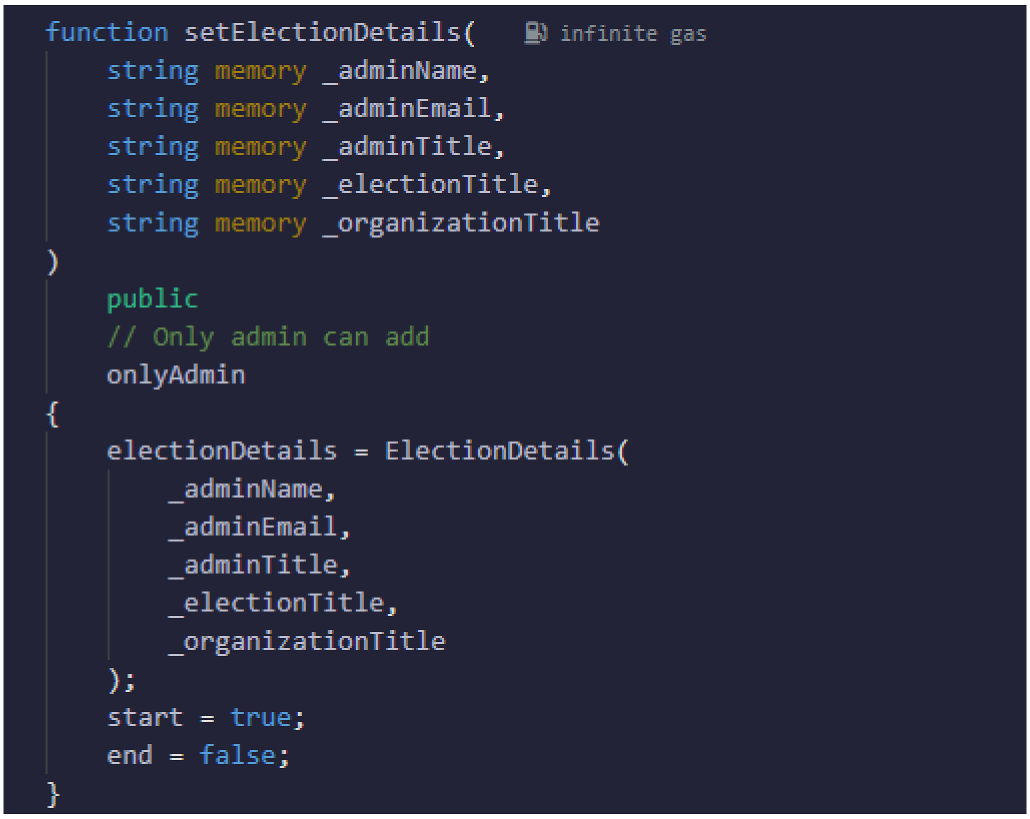
The function **verifyVoter** in Fig 4.25 Verify the state of a voter by his etherium address; does he make the vote or not? The output of this is a boolean.



```
    // Vote
    function vote(uint256 candidateId) public {    infinite gas
        require(voterDetails[msg.sender].hasVoted == false);
        require(voterDetails[msg.sender].isVerified == true);
        require(start == true);
        require(end == false);
        candidateDetails[candidateId].voteCount += 1;
        voterDetails[msg.sender].hasVoted = true;
    }
```

Figure 4.26: Function Vote

The function vote in Fig 4.26, verify the state of the voter, if he does make the vote cast or not, by his **candidateId**. A vote count increment occurs each time a voter casts the vote; here, voting is only for ones.

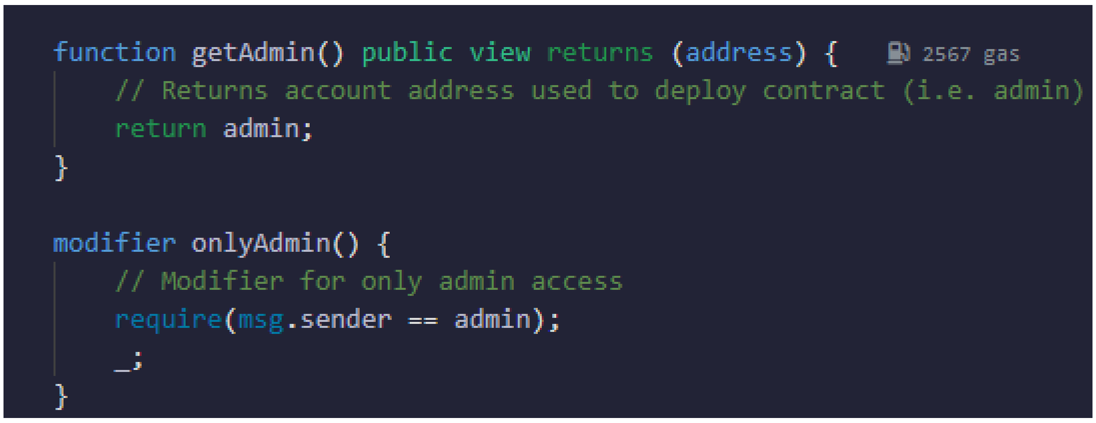


```
function setElectionDetails(    infinite gas
    string memory _adminName,
    string memory _adminEmail,
    string memory _adminTitle,
    string memory _electionTitle,
    string memory _organizationTitle
)
    public
    // Only admin can add
    onlyAdmin
{
    electionDetails = ElectionDetails(
        _adminName,
        _adminEmail,
        _adminTitle,
        _electionTitle,
        _organizationTitle
    );
    start = true;
    end = false;
}
```

Figure 4.27: Function set election details

The function **setElectiondetails**, with four local variables: **adminName**, **adminEmail**, **adminTitle**,**electionTitle**, and **prganisationTitle**. After full fill the details the vote can be start. Note here that only the administrator can manage these details.

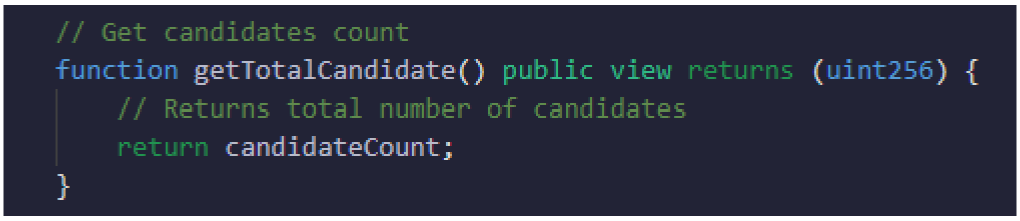
Now, let's see the call back functions:



```
function getAdmin() public view returns (address) {    2567 gas
    // Returns account address used to deploy contract (i.e. admin)
    return admin;
}

modifier onlyAdmin() {
    // Modifier for only admin access
    require(msg.sender == admin);
    _;
}
```

Figure 4.28: Function getAdmin

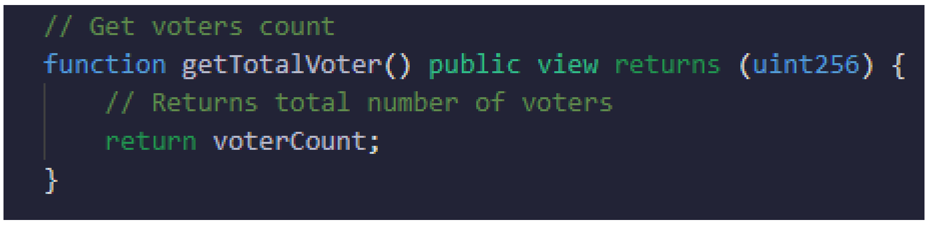
The function **getAdmin** in figure Fig 4.28 return the etherium address used to deploy the contract and the address of the admin, and there is a **modifier** that only the admin has the right to access.



```
// Get candidates count
function getTotalCandidate() public view returns (uint256) {
    // Returns total number of candidates
    return candidateCount;
}
```

Figure 4.29: Function Get Total Candidates

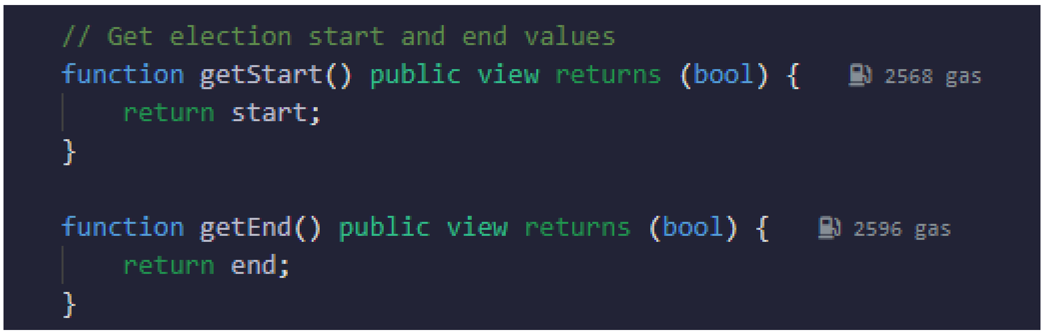
The function **geTotalCandidate** in Fig 4.29, return the number of registered candidates to make the stats after the close of the vote period.



```
// Get voters count
function getTotalVoter() public view returns (uint256) {
    // Returns total number of voters
    return voterCount;
}
```

Figure 4.30: Function Get Total Voters

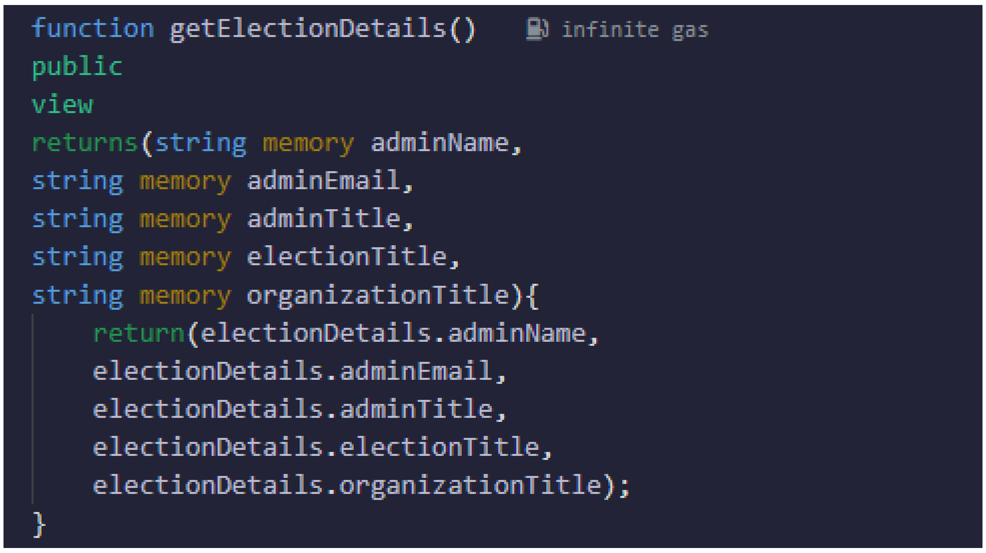
The function **getTotalVoter** in Fig 4.30, return the number of voters who cast the vote.



```
// Get election start and end values
function getStart() public view returns (bool) {    2568 gas
    return start;
}

function getEnd() public view returns (bool) {    2596 gas
    return end;
}
```

Figure 4.31: Functions Get Start and End vote

The functions **getStart** and **getEnd** are two functions that return the boolean state of the vote.



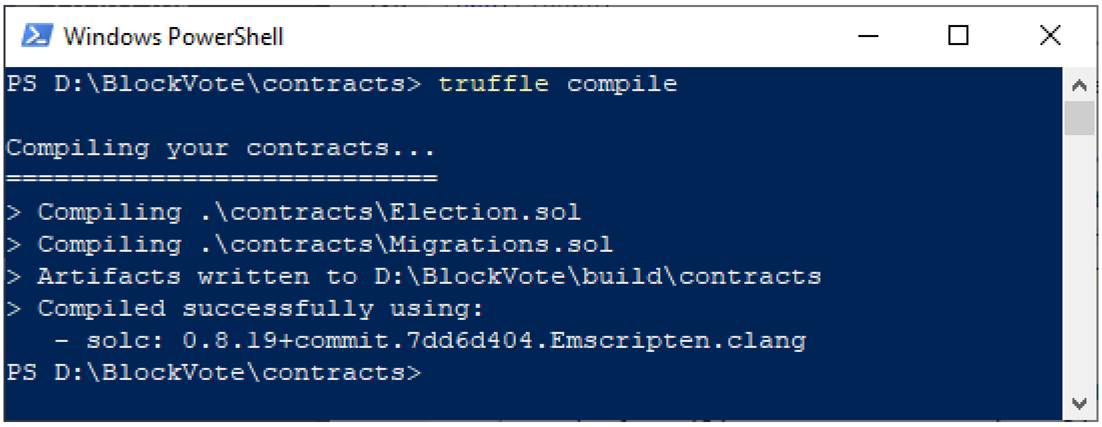
```
function getElectionDetails()    infinite gas
public
view
returns(string memory adminName,
string memory adminEmail,
string memory adminTitle,
string memory electionTitle,
string memory organizationTitle){
    return(electionDetails.adminName,
    electionDetails.adminEmail,
    electionDetails.adminTitle,
    electionDetails.electionTitle,
    electionDetails.organizationTitle);
}
```

Figure 4.32: Function Get Election Details

The function **getElectionDetails** return after the deployment, and by the etherium address the information of the admin.

4.4 Compiling the Smart Contract

After the compilation is successfully run, a **JSON** file generated automatically by the Solidity compiler under the name "**(.../build/contracts/election.json)**", which provides details about the functions in the smart contract that can be exposed to external clients as in Fig 4.33.



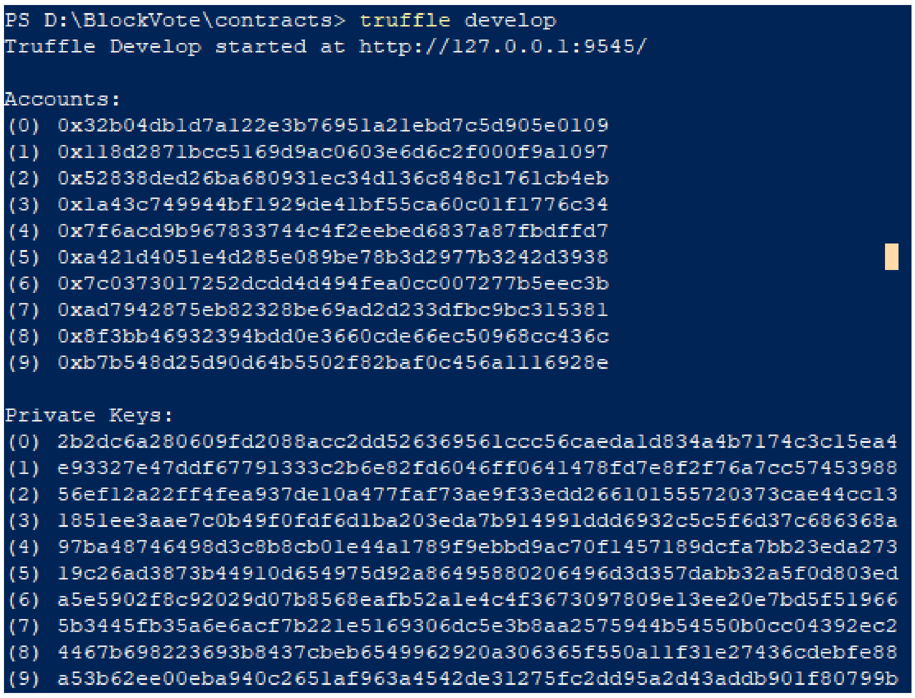
Windows PowerShell

```
PS D:\BlockVote\contracts> truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to D:\BlockVote\build\contracts
> Compiled successfully using:
   - solc: 0.8.19+commit.7dd6d404.Emscripten.clang
PS D:\BlockVote\contracts>
```

Figure 4.33: Compiling Smart Contract in Solidity

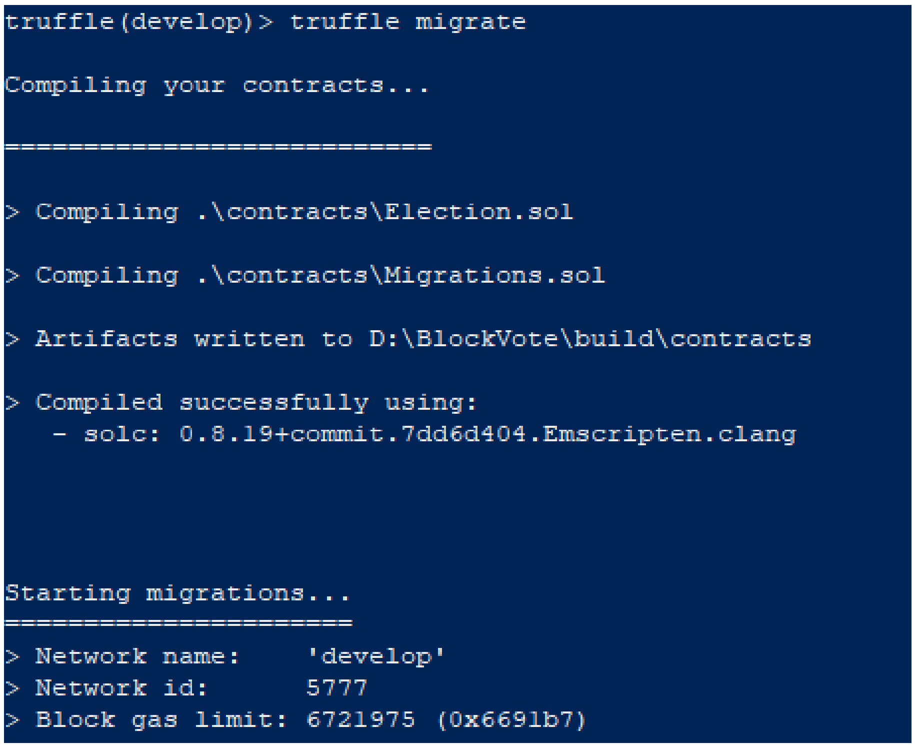
4.5 Migration and Deployment of the smart contract

Now, the smart contract is ready for deployment. and for this, a connection to the blockchain for deployment is needed. Truffle provides a virtual blockchain for testing purposes under the two commands "**truffle develop**" and "**truffle migrate**" in the terminal. The steps in Fig 4.34, Fig 4.35, Fig 4.36 and Fig 4.37.



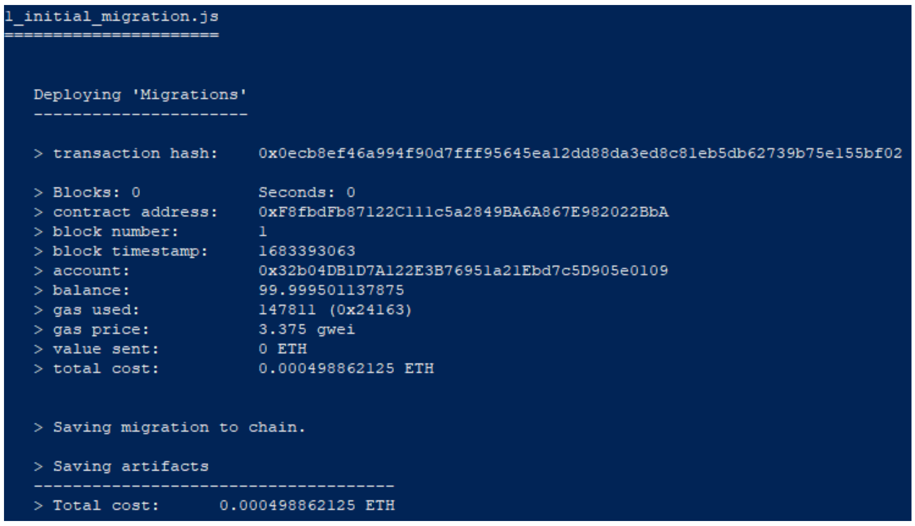
```
PS D:\BlockVote\contracts> truffle develop
Truffle Develop started at http://127.0.0.1:9545/

Accounts:
(0) 0x32b04db1d7a122e3b76951a21ebd7c5d905e0109
(1) 0x118d2871bcc5169d9ac0603e6d6c2f000f9a1097
(2) 0x52838ded26ba680931ec34d136c848c1761cb4eb
(3) 0x1a43c749944bf1929de41bf55ca60c01f1776c34
(4) 0x7f6acd9b967833744c4f2eebed6837a87fbdffd7
(5) 0xa421d4051e4d285e089be78b3d2977b3242d3938
(6) 0x7c0373017252dcdd4d494fea0cc007277b5eec3b
(7) 0xad7942875eb82328be69ad2d233dfbc9bc315381
(8) 0x8f3bb46932394bdd0e3660cde66ec50968cc436c
(9) 0xb7b548d25d90d64b5502f82baf0c456a1116928e

Private Keys:
(0) 2b2dc6a280609fd2088acc2dd52636956lccc56caeda1d834a4b7174c3c15ea4
(1) e93327e47ddf67791333c2b6e82fd6046ff0641478fd7e8f2f76a7cc57453988
(2) 56ef12a22ff4fea937de10a477faf73ae9f33edd26610155572037 3cae44cc13
(3) 1851ee3aae7c0b49f0fdf6d1ba203eda7b914991ddd6932c5c5f6d37c686368a
(4) 97ba48746498d3c8b8cb01e44a1789f9ebbd9ac70f1457189dcfa7bb23eda273
(5) 19c26ad3873b44910d654975d92a86495880206496d3d357dabb32a5f0d803ed
(6) a5e5902f8c92029d07b8568eafb52a1e4c4f3673097809e13ee20e7bd5f51966
(7) 5b3445fb35a6e6acf7b221e5169306dc5e3b8aa2575944b54550b0cc04392ec2
(8) 4467b698223693b8437cbeb6549962920a306365f550a11f31e27436cdebfe88
(9) a53b62ee00eba940c2651af963a4542de31275fc2dd95a2d43addb901f80799b
```

Figure 4.34: Smart contract Beginning of Deployment



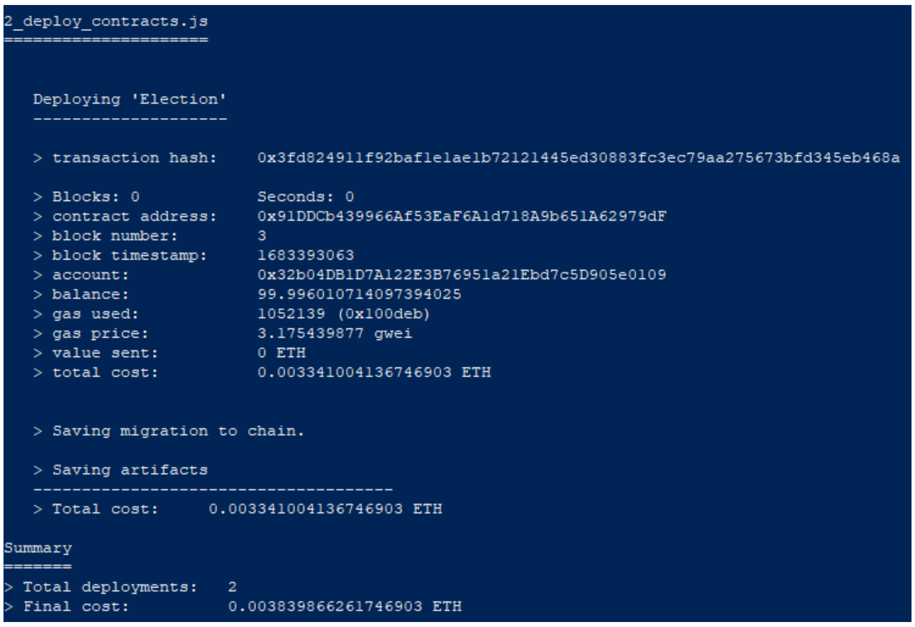
```
truffle(develop)> truffle migrate

Compiling your contracts...

===========================

> Compiling .\contracts\Election.sol

> Compiling .\contracts\Migrations.sol

> Artifacts written to D:\BlockVote\build\contracts

> Compiled successfully using:
   - solc: 0.8.19+commit.7dd6d404.Emscripten.clang



Starting migrations...
======================
> Network name:    'develop'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)
```

Figure 4.35: Smart contract Beginning of Migration



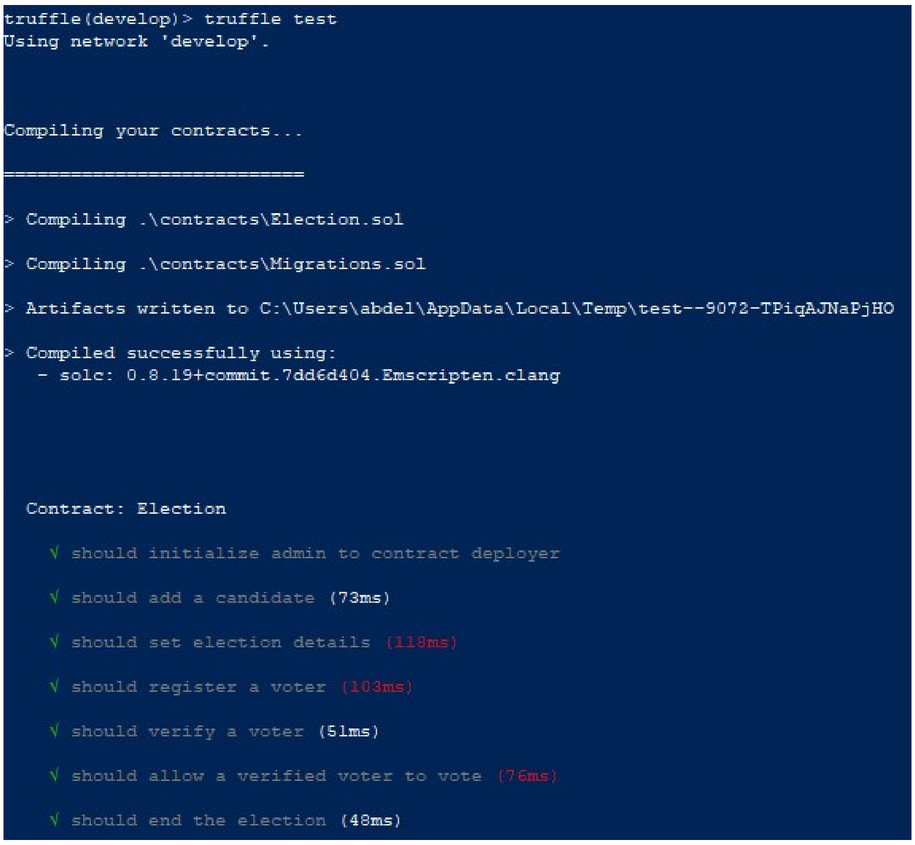
```
1_initial_migration.js
======================


   Deploying 'Migrations'
   ----------------------

   > transaction hash:    0x0ecb8ef46a994f90d7fff95645ea12dd88da3ed8c81eb5db62739b75e155bf02

   > Blocks: 0            Seconds: 0
   > contract address:    0xF8fbdFb87122C111c5a2849BA6A867E982022BbA
   > block number:        1
   > block timestamp:     1683393063
   > account:             0x32b04DB1D7A122E3B76951a21Ebd7c5D905e0109
   > balance:             99.999501137875
   > gas used:            147811 (0x24163)
   > gas price:           3.375 gwei
   > value sent:          0 ETH
   > total cost:          0.000498862125 ETH


   > Saving migration to chain.

   > Saving artifacts
   -------------------------------------
   > Total cost:      0.000498862125 ETH
```

Figure 4.36: Smart contract Starting Migration



```
2_deploy_contracts.js
=====================


   Deploying 'Election'
   --------------------

   > transaction hash:    0x3fd824911f92baf1e1ae1b72121445ed30883fc3ec79aa275673bfd345eb468a

   > Blocks: 0            Seconds: 0
   > contract address:    0x91DDCb439966Af53EaF6A1d718A9b651A62979dF
   > block number:        3
   > block timestamp:     1683393063
   > account:             0x32b04DB1D7A122E3B76951a21Ebd7c5D905e0109
   > balance:             99.996010714097394025
   > gas used:            1052139 (0x100deb)
   > gas price:           3.175439877 gwei
   > value sent:          0 ETH
   > total cost:          0.003341004136746903 ETH


   > Saving migration to chain.

   > Saving artifacts
   -------------------------------------
   > Total cost:     0.003341004136746903 ETH

Summary
=======
> Total deployments:   2
> Final cost:          0.003839866261746903 ETH
```

Figure 4.37: Smart Contract Deployment



```
truffle(develop)> truffle test
Using network 'develop'.


Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to C:\Users\abdel\AppData\Local\Temp\test--9072-TPiqAJNaPjHO
> Compiled successfully using:
   - solc: 0.8.19+commit.7dd6d404.Emscripten.clang



  Contract: Election
    √ should initialize admin to contract deployer
    √ should add a candidate (73ms)
    √ should set election details (118ms)
    √ should register a voter (103ms)
    √ should verify a voter (51ms)
    √ should allow a verified voter to vote (76ms)
    √ should end the election (48ms)
```

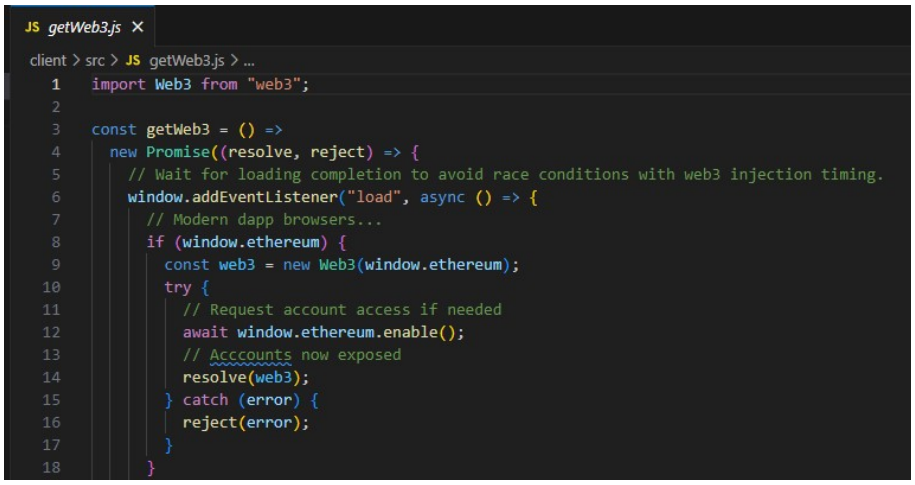
Figure 4.38: Testing the Smart Contract

Now, after the deployment has been successfully completed, a test is run using a JavaScript file named **test.js**,as shown in Fig 4.38.

4.6 Front-end/User Interface

After completing all preceding measures, we proceeded to construct the client side in order to establish interaction with the Voting smart contract. We imported a number of accounts from Ganache to Metamask and linked our web browser to the blockchain via Metamask, which in turn connected to our personal Ganache blockchain. This facilitated our capacity to function on behalf of our service users.

Importing web3js into our application by creating a file named **getWeb3.js** as shown in figure is one of the primary steps in connecting our client-side application to the blockchain.

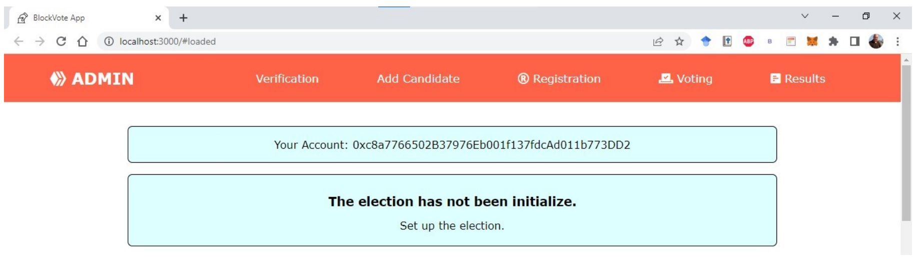


```javascript
import Web3 from "web3";

const getWeb3 = () =>
  new Promise((resolve, reject) => {
    // Wait for loading completion to avoid race conditions with web3 injection timing.
    window.addEventListener("load", async () => {
      // Modern dapp browsers...
      if (window.ethereum) {
        const web3 = new Web3(window.ethereum);
        try {
          // Request account access if needed
          await window.ethereum.enable();
          // Acccounts now exposed
          resolve(web3);
        } catch (error) {
          reject(error);
        }
      }
```

Figure 4.39: Web3js file in application

4.7 System Interface

After the import of some account Metamask in the main project folder, precisely the client folder, type the command **$ npm start**, and the server will run on port 3000 and show the main application page in the default web browser in Figures:
Fig 4.40, Fig 4.41 and Fig 4.42 .

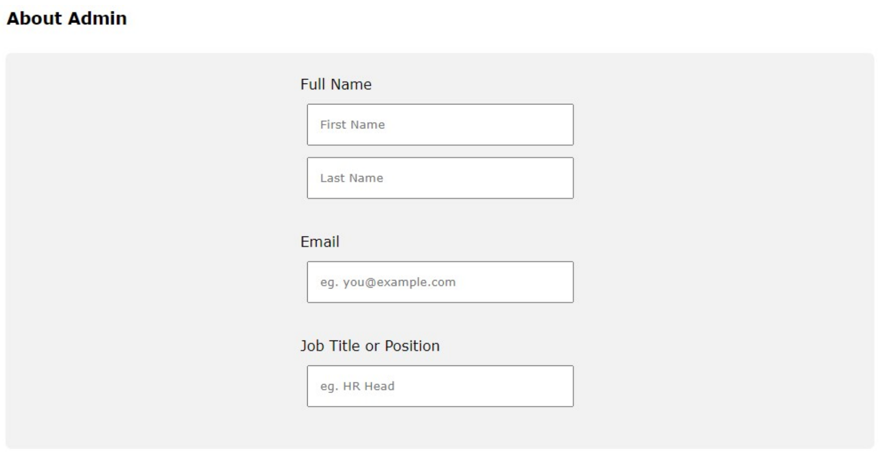


Your Account: 0xc8a7766502B37976Eb001f137fdcAd011b773DD2

The election has not been initialize.

Set up the election.

Figure 4.40: The main application page -Admin Eth Adresse-



About Admin

Full Name

First Name

Last Name

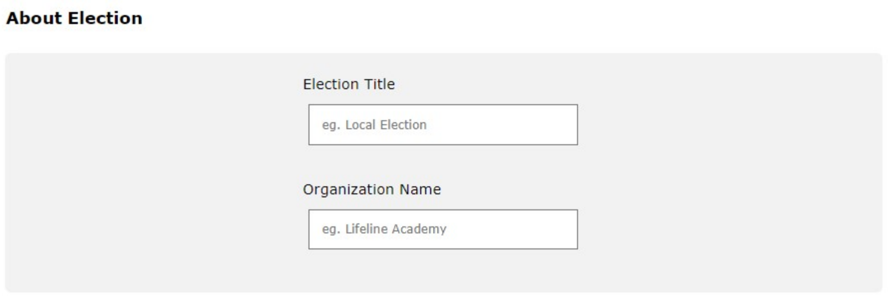
Email

eg. you@example.com

Job Title or Position

eg. HR Head

Figure 4.41: The main application page -About Admin-



About Election

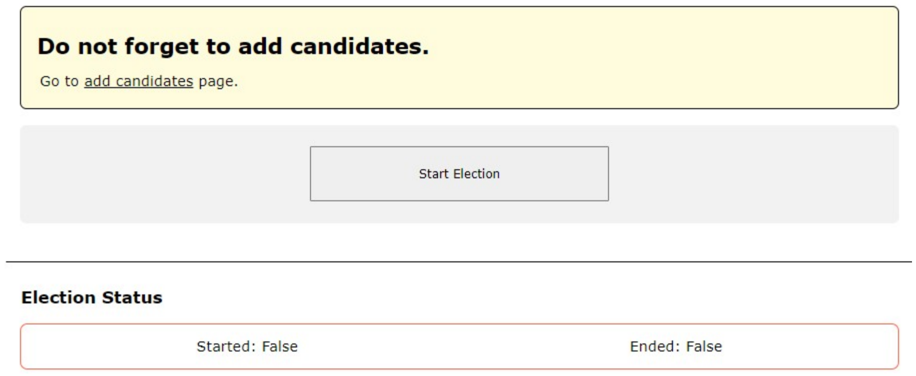
Election Title

eg. Local Election

Organization Name

eg. Lifeline Academy

Figure 4.42: The main application page -About Election-



Do not forget to add candidates.

Go to add candidates page.

Start Election

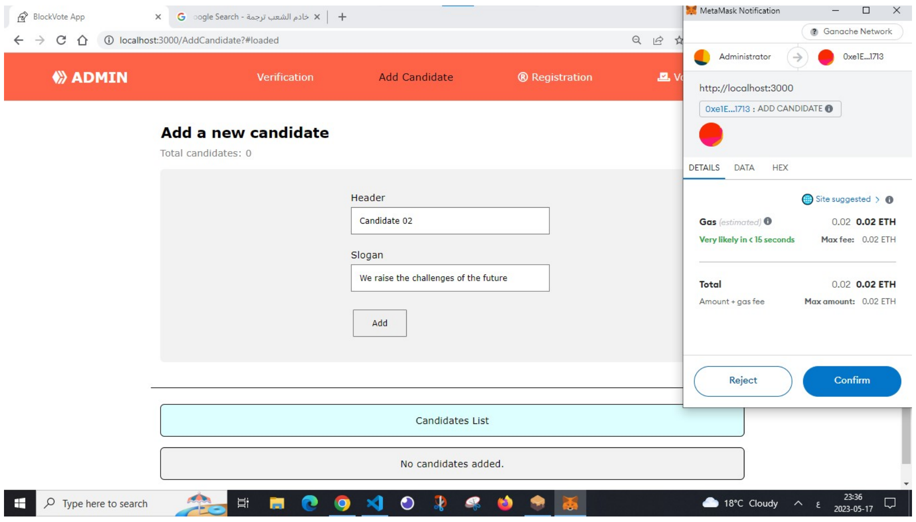
Election Status

Started: False Ended: False

Figure 4.43: The main application page -Start Election-

The administrator must fill out the form with his own information, such as his full name, email, job title, and information about the nature of the election, and start the process.

After the process starts, the admin adds the candidates, and we note here that every addition needs a transaction with Metamask, as the Fig 4.44 show:



Add a new candidate

Total candidates: 0

Header

Candidate 02

Slogan

We raise the challenges of the future

Add

No candidates added.

http://localhost:3000

0xe1E...1713 : ADD CANDIDATE

Gas (estimated) 0.02 **0.02 ETH**

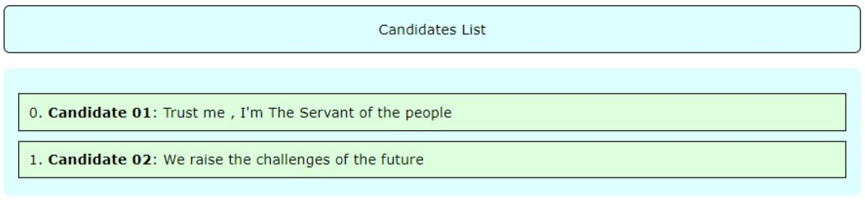
Very likely in < 15 seconds Max fee: 0.02 ETH

Total 0.02 **0.02 ETH**

Amount + gas fee Max amount: 0.02 ETH

Figure 4.44: Add Candidates to election

The list of candidates after adding the complete as Fig 4.45 :



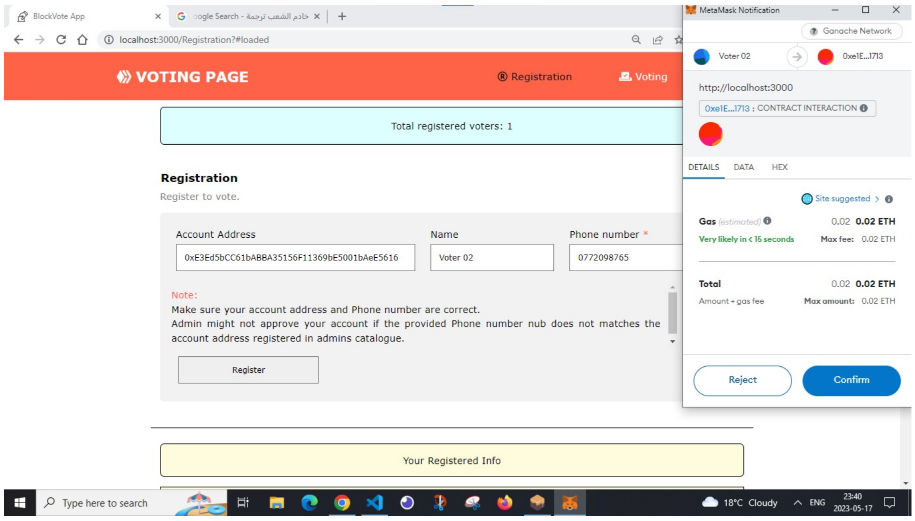
Candidates List

0. **Candidate 01**: Trust me , I'm The Servant of the people

1. **Candidate 02**: We raise the challenges of the future

Figure 4.45: Candidates List

Now, the voters can register to cast their votes, and we have to mention here that every imported Metamask account is simulated as a voter account. Note that a transaction is made after every registration as show Fig 4.46 and Fig 4.47. After the registration, the administrator approves the demands for voter registrations as shown in Fig 4.47.



Total registered voters: 1

Registration

Register to vote.

Account Address	Name	Phone number *
0xE3Ed5bCC61bABBA35156F11369bE5001bAeE5616	Voter 02	0772098765

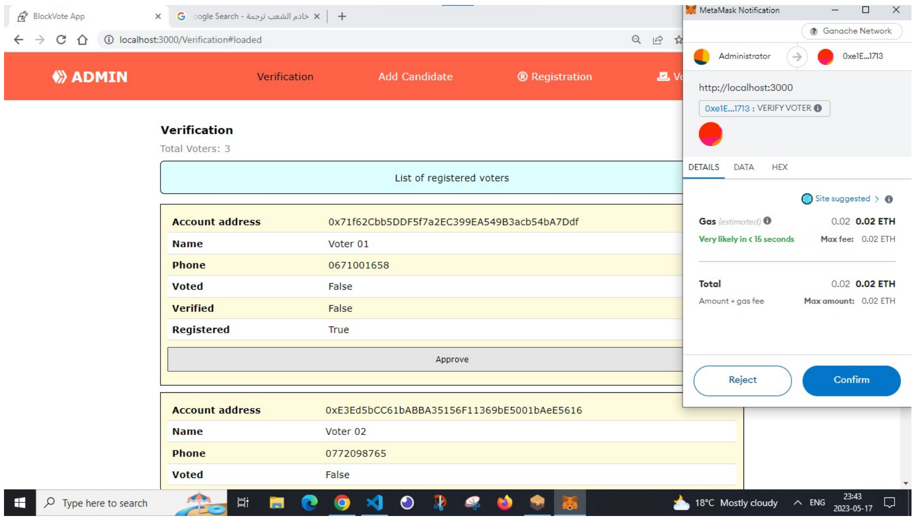
Note:
Make sure your account address and Phone number are correct.
Admin might not approve your account if the provided Phone number nub does not matches the account address registered in admins catalogue.

Your Registered Info

http://localhost:3000
0xe1E...1713 : CONTRACT INTERACTION

Gas (estimated)	0.02	0.02 ETH
Very likely in < 15 seconds	Max fee:	0.02 ETH
Total	0.02	0.02 ETH
Amount + gas fee	Max amount:	0.02 ETH

Figure 4.46: Voter registration



Verification

Total Voters: 3

List of registered voters

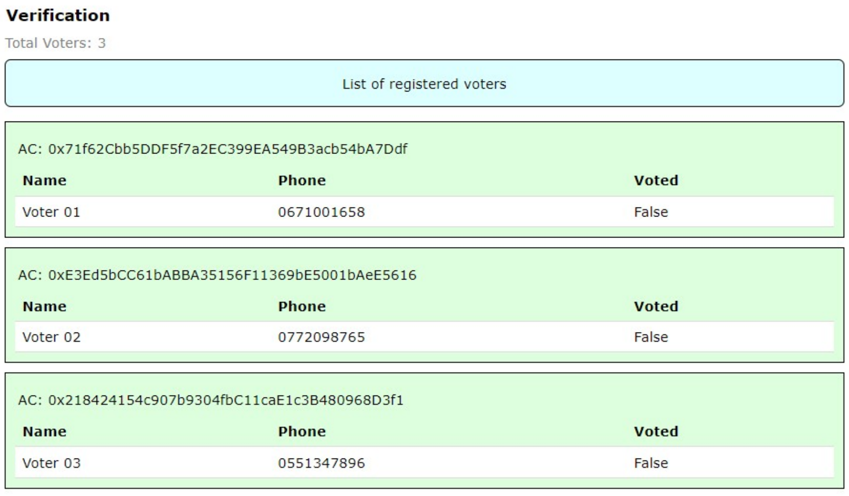
Account address	0x71f62Cbb5DDF5f7a2EC399EA549B3acb54bA7Ddf
Name	Voter 01
Phone	0671001658
Voted	False
Verified	False
Registered	True

Account address	0xE3Ed5bCC61bABBA35156F11369bE5001bAeE5616
Name	Voter 02
Phone	0772098765
Voted	False

http://localhost:3000
0xe1E...1713 : VERIFY VOTER

Gas (estimated)	0.02	0.02 ETH
Very likely in < 15 seconds	Max fee:	0.02 ETH
Total	0.02	0.02 ETH
Amount + gas fee	Max amount:	0.02 ETH

Figure 4.47: Appove voters



Verification

Total Voters: 3

List of registered voters

AC: 0x71f62Cbb5DDF5f7a2EC399EA549B3acb54bA7Ddf

Name	Phone	Voted
Voter 01	0671001658	False

AC: 0xE3Ed5bCC61bABBA35156F11369bE5001bAeE5616

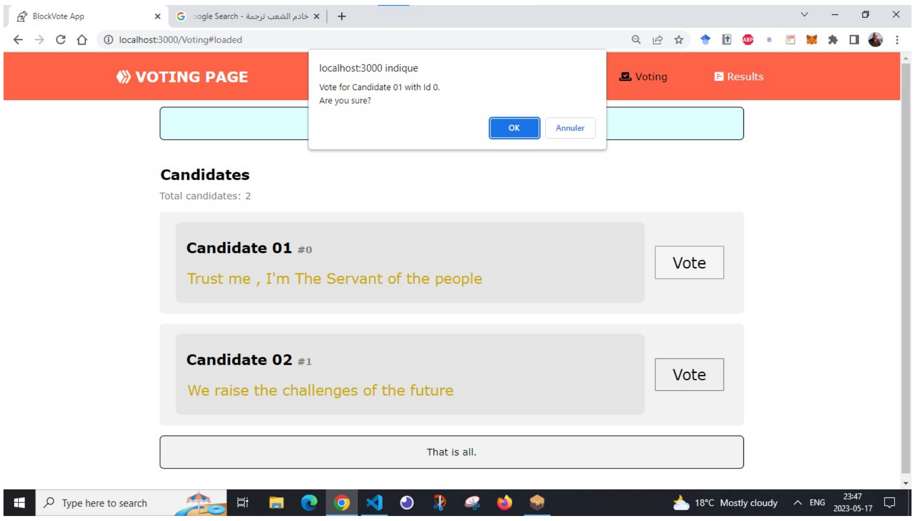
Name	Phone	Voted
Voter 02	0772098765	False

AC: 0x218424154c907b9304fbC11caE1c3B480968D3f1

Name	Phone	Voted
Voter 03	0551347896	False

Figure 4.48: List of approved voters

After the approval of voters, they can cast their vote by choosing one of the candidates. Note that there is always a transaction to confirm the action,as shown in Figures Fig 4.49 and Fig 4.50.



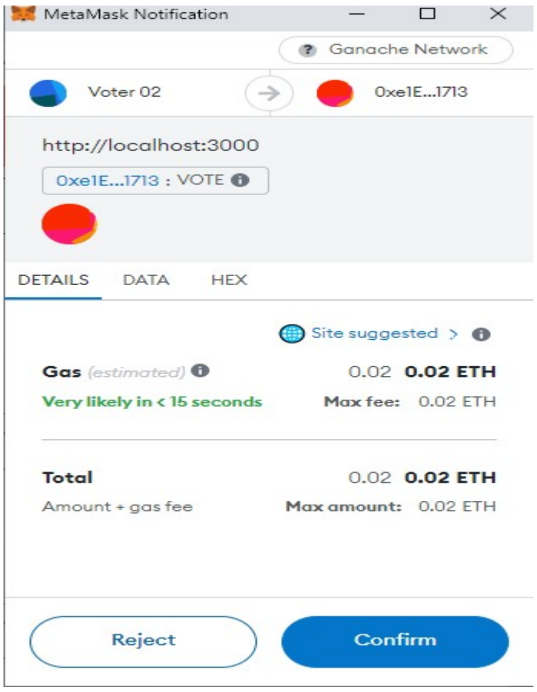
Candidates
Total candidates: 2

Candidate 01 #0
Trust me , I'm The Servant of the people
Vote

Candidate 02 #1
We raise the challenges of the future
Vote

That is all.

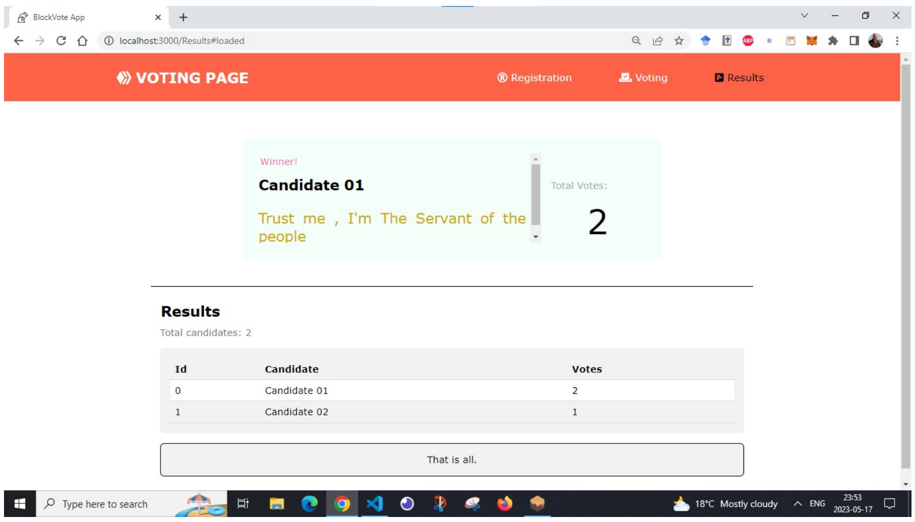
Figure 4.49: Cast vote to candidate



Gas (estimated)	0.02 **0.02 ETH**
Very likely in < 15 seconds	Max fee: 0.02 ETH
Total	0.02 **0.02 ETH**
Amount + gas fee	Max amount: 0.02 ETH

Figure 4.50: Voter cast transaction

Now that all the voters have cast their votes, the administrator ends the voting process to display the result for the voters, as shown in Figure.



Winner!

Candidate 01

Trust me , I'm The Servant of the people

Total Votes:

2

Results

Total candidates: 2

Id	Candidate	Votes
0	Candidate 01	2
1	Candidate 02	1

That is all.

Figure 4.51: Election Results

4.8 Conclusion

Through a series of screenshots, we described the development tools we employed to implement the project, the smart contract, and the key user interfaces of our application in this chapter.

General Conclusion

In conclusion, elections serve as the cornerstone of democratic societies, allowing citizens to actively participate in the democratic process by choosing their representatives and shaping the government. However, traditional election systems often face challenges such as a lack of transparency, potential fraud, and limited accessibility. As a response to these drawbacks, there is a growing interest in exploring new solutions that can address these issues.

This project proposes the use of a decentralized system and blockchain technology to overcome the limitations of traditional voting systems. By leveraging the decentralized nature of blockchain, this approach aims to provide immutability, high levels of encryption, and transparency in the electoral process.

Blockchain technology offers several potential benefits for elections. Firstly, it ensures transparency by creating an immutable and auditable record of all transactions or votes. This transparency increases trust in the election process and allows for independent verification of the results. Secondly, blockchain enhances security by making it extremely difficult to modify the recorded data without detection. The decentralized nature of blockchain also reduces the risk of hacking or tampering with the election data.

Moreover, blockchain technology can increase accessibility and convenience in elections by enabling secure remote voting through digital identities and cryptographic keys. This opens up opportunities for individuals who may face barriers in participating in traditional elections, such as overseas citizens or people with disabilities, to engage in the democratic process.

In summary, the use of blockchain technology in elections has the potential to address the shortcomings of traditional voting systems. It offers transparency, security, and accessibility, thereby enhancing the integrity and inclusiveness of the electoral process. As further research and development in this field continue, the adoption of blockchain-based voting systems may pave the way for more democratic and trustworthy elections in the future.

Bibliography

[1] A beginner's guide to understanding the layers of blockchain technology — cointelegraph.com. `https://http://bit.ly/41rHZ1t`. [Accessed 15-Feb-2023].

[2] blockchain in healthcare: 17 examples to know. `http://bit.ly/3Z7jlB5`, journal=Built In, year=2022,note = [Accessed 18-Feb-2023] .

[3] Blockchain Structure - GeeksforGeeks. [Accessed 10-Feb-2023].

[4] Cryptography-and-its-types. `https://bit.ly/3xVzoqc`. [Accessed 16-Feb-2023].

[5] Documentation for Visual Studio Code — code.visualstudio.com. `https://code.visualstudio.com/docs`. [Accessed 24-Apr-2023].

[6] Electronic voting. `https://www.polyas.com/election-glossary/e-voting`, journal=polyas.com, year=2016, month=Oct, note = [Accessed 27-Mar-2023].

[7] History of Blockchain, url = `https://rb.gy/oblqtf`, note = [Accessed 08-Feb-2023].

[8] How blockchain technology can prevent voter fraud. `http://bit.ly/40jBFHY`, journal=Investopedia, year=2023. [Accessed 18-Feb-2023].

[9] Introduction to smart contracts | ethereum.org. `http://bit.ly/3xUInrQ`. [Accessed 18-Feb-2023].

[10] A short history of voting in the ancient world. `http://bit.ly/3IR4h47`. [Accessed 03-Mar-2023].

[11] Understanding-hashing-in-cryptography. `http://bit.ly/3Z5WSoO`. [Accessed 17-Feb-2023].

[12] Use-cases-of-blockchain-technology-in-business-and-life. `http://bit.ly/41sWIcv`. [Accessed 18-Feb-2023].

[13] What is agora ? https://www.agora.vote/about, journal=Agora. [Accessed 03-Mar-2023].

[14] What is blockchain in supply chain management? https://bit.ly/3LGy1UE, journal=GetSmarter Blog, author=Jara, Anitta, year=2022, month=Feb,note = [Accessed 18-Feb-2023] .

[15] What-is-cryptography. https://bit.ly/3Z0aAcP. [Accessed 16-Feb-2023].

[16] What is votem?. https://positiveblockchain.io/database/votem/, journal=PositiveBlockchain.io, language=en. [Accessed 25-Mar-2023].

[17] Chapter 4 - encrypting private data. In Mark Burnett and James C. Foster, editors, *Hacking the Code*, pages 153–204. Syngress, Burlington, 2004.

[18] History of elections. https://http://bit.ly/3SWOBRQ, 2017. [Accessed 03-Mar-2023].

[19] How does a blockchain work. http://bit.ly/3Z0cJ8f, Aug 2018. [Accessed 15-Feb-2023].

[20] Jordana A. What Is JavaScript? A Basic Introduction to JS for Beginners — bit.ly. https://bit.ly/3LpdS4K, Jan 31, 2023. [Accessed 24-Apr-2023].

[21] P Aithal, P Saavedra, Sreeramana Aithal, and Surajit Ghoash. Blockchain technology and its types-a short review. *International Journal of Applied Science and Engineering*, 9:189–200, 12 2021.

[22] Arshdeep Bahga and Vijay Madisetti. Blockchain platform for industrial internet of things. *Journal of Software Engineering and Applications*, 09:533–546, 01 2016.

[23] Muhammad Nasir Mumtaz Bhutta, Amir Khwaja, Adnan Nadeem Al Hassan, Hafiz Ahmad, Khurram Khan, Moataz Hanif, Houbing Song, Majed Alshamari, and Yue Cao. A survey on blockchain technology: Evolution, architecture and security. *IEEE Access*, PP:1–1, 04 2021.

[24] Sarah Birch. Two-round electoral systems and democracy. *Comparative Political Studies*, 36:319–344, 4 2003.

[25] Christian Cachin, Marko Vukolic Sorniotti, and Thomas Weigold. Blockchain, cryptography, and consensus. *IBM Res., Zürich, Switzerland, Tech. Rep*, 2016, 2016.

[26] Sourabh Chandra, Smita Paira, Sk Safikul Alam, and Goutam Sanyal. A comparative survey of symmetric and asymmetric key cryptography. In *2014 international conference on electronics, communication and computational engineering (ICECCE)*, pages 83–93. IEEE, 2014.

[27] Josep M Colomer. The strategy and history of electoral system choice. *The Handbook of electoral system choice*, pages 3–78, 2004.

[28] Chris Dannen. *Introducing Ethereum and solidity*, volume 1. Springer, 2017.

[29] Juisti Davis. Truffle Ethereum: The Definitive Guide - Iglu — iglu.net. `https://iglu.net/truffle-ethereum/`, FEBRUARY 12, 2023. [Accessed 24-Apr-2023].

[30] Daniel Phillips Decrypt / Matt Hussey. What is MetaMask? How to Use the Top Ethereum Wallet - Decrypt — decrypt.co. `https://decrypt.co/resources/metamask`, May 3, 2022. [Accessed 24-Apr-2023].

[31] Hans Delfs and Helmut Knebl. *Symmetric-Key Cryptography*, pages 11–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[32] docs.soliditylang.org. Solidity &x2014; Solidity 0.8.20 documentation — bit.ly. `https://bit.ly/3AoeUaO`. [Accessed 24-Apr-2023].

[33] Carribean Elections. Types of electoral systems. `https://bit.ly/3mlZJv4`, 2019. [Accessed 03-Mar-2023].

[34] Elad Elrom. *The Blockchain developer: A practical guide for designing, implementing, publishing, testing, and securing distributed Blockchain-based projects*. Apress, 2019.

[35] Michael Gallagher. *The Politics of Electoral Systems*. Oxford University Press, USA, 2006.

[36] Peter Gaži, Aggelos Kiayias, and Dionysis Zindros. Proof-of-stake sidechains. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 139–156. IEEE, 2019.

[37] Arthur Gervais, Ghassan O Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 3–16, 2016.

[38] Matt Golder. Democratic electoral systems around the world, 1946–2000. *Electoral Studies*, 24(1):103–121, 2005.

[39] Julija Golosova and Andrejs Romanovs. The advantages and disadvantages of the blockchain technology. In *2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE)*, pages 1–6. IEEE, 2018.

[40] Jūlija Golosova and Andrejs Romānovs. The advantages and disadvantages of the blockchain technology. *2018 IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*, pages 1–6, 2018.

[41] Bernard Grofman, Sung-Chull Lee, Edwin Winckler, Brian Woodall, et al. *Elections in Japan, Korea, and Taiwan under the single non-transferable vote: The comparative study of an embedded institution*. University of Michigan Press, 1999.

[42] David Herbert. What is React.js? (Uses, Examples, & More) — blog.hubspot.com. `https://blog.hubspot.com/website/react-js`, June 27, 2022. [Accessed 24-Apr-2023].

[43] Friorik P. Hjalmarsson, Gunnlaugur K. Hreioarsson, Mohammad Hamdaqa, and Gisli Hjalmtysson. Blockchain-based e-voting system. volume 2018-July, pages 983–986. IEEE Computer Society, 9 2018.

[44] Marko Hölbl, Marko Kompara, Aida Kamišalić, and Lili Nemec Zlatolas. A systematic review of the use of blockchain in healthcare. *Symmetry*, 10(10):470, 2018.

[45] Irjet.net. Blockchain_based_e-voting. `https://bit.ly/3yt7DWi`, year=2022,note =[Accessed 04-Mar-2023] .

[46] Matthew Johnson, Michael Jones, Mark Shervey, Joel T Dudley, and Noah Zimmerman. Building a secure biomedical data sharing decentralized app (dapp): tutorial. *Journal of medical Internet research*, 21(10):e13601, 2019.

[47] H Kent Baker, Ehsan Nikbakht, and Sean Stein Smith, editors. *The emerald handbook of blockchain for business the emerald handbook of blockchain for business*. Emerald Publishing, Bingley, England, March 2021.

[48] Amos Kibet, Demeke Gebresenbet Bayyou, and Rosanna Esquivel. Blockchain: It's structure, principles, applications and foreseen issues. 04 2019.

[49] The Electoral Knowledge Network. Electoral systems —. `http://bit.ly/3IWFrQo`, 2019. [Accessed 04-Mar-2023].

[50] Melanie Kramer. What are consensus protocols? `http://bit.ly/42ONic1`, Apr 2019. [Accessed 30-Mar-2023].

[51] Pankaj Kumar. Node.js for Beginners: How to Get Started. `https://bit.ly/3AnWS8P`, 2023. [Accessed 23-Apr-2023].

[52] Martin Lackner and Piotr Skowron. *Multi-Winner Voting with Approval Preferences*. Springer International Publishing, 2023.

[53] Anita A. Lahane, Junaid Patel, Talif Pathan, and Prathmesh Potdar. Blockchain technology based e-voting system. *ITM Web of Conferences*, 32:03001, 7 2020.

[54] Anastasiia Lastovetska. Blockchain architecture basics: Components, structure, benefits & creation. `http://bit.ly/40uIabu`, Jan 2018. [Accessed 16-Feb-2023].

[55] Chia Hao Lee, Han Foon Neo, and Chuan Chin Teo. Secure e-voting system based on blockchain technology. *Journal of System and Management Sciences*, 12:121–138, 2022.

[56] Daniel Levis, Francesco Fontana, and Elisa Ughetto. A look into the future of blockchain technology. *Plos one*, 16(11):e0258995, 2021.

[57] Conny Mccormack. Electronic voting around the world. `https://bit.ly/3FVNi0a`. [Accessed 27-Mar-2023].

[58] Gregory McCubbin. Intro to Web3.js · Ethereum Blockchain Developer Crash Course | Dapp University — dappuniversity.com. `https://www.dappuniversity.com/articles/web3-js-intro`, March 30, 2023. [Accessed 24-Apr-2023].

[59] Prof. Gayathri Naidu and Prof. Reeta Mishra. Blockchain technology architecture and key characteristics - ijariie. `https://shorturl.at/vCZ12`, journal=ijariie.com, note = [Accessed 10-Feb-2023].

[60] Michele Pasqua, Andrea Benini, Filippo Contro, Marco Crosara, Mila Dalla Preda, and Mariano Ceccato. Enhancing ethereum smart-contracts static analysis by computing a precise control-flow graph of ethereum bytecode. *Journal of Systems and Software*, 200:111653, 2023.

[61] Thomas Quinn. Block voting in the labour party: A political exchange model. *Party Politics*, 8(2):207–226, 2002.

[62] K Varaprasada Rao, Mutyala Sree Teja, P Praneeth Reddy, and S Saikrishna. Building permissioned blockchain networks using hyperledger fabric. In *Bitcoin and Blockchain*, pages 193–235. CRC Press, August 2020.

[63] Simanta Shekhar Sarmah. Understanding blockchain technology. *Computer Science and Engineering*, 8:23–29, 2018.

[64] Simanta Shekhar Sarmah. Understanding blockchain technology. *Computer Science and Engineering*, 8(2):23–29, 2018.

[65] Rashmi Sarode, Manoj Poudel, Shashank Shrestha, and Subhash Bhalla. Blockchain for committing peer-to-peer transactions using distributed ledger technologies. *International Journal of Computational Science and Engineering*, 24:215, 01 2021.

[66] Jessica Savage. Elections in the middle ages. [Accessed 03-Mar-2023].

[67] Siamak F Shahandashti. Electoral systems used around the world. In *Real-World Electronic Voting*, pages 93–118. Auerbach Publications, 2016.

[68] Jahnvi Sharma, Poorvi Maheshwari, and Kaushlendra Singh. Digital voting system using blockchain, 2022.

[69] Chetanpal Singh, Rahul Thakkar, and Jatinder Warraich. Blockchain in supply chain management. *European Journal of Engineering and Technology Research*, 7:60–69, 10 2022.

[70] ONKAR SINGH. How to use Ganache for blockchain project development. `https://bit.ly/3oDhAyV`, APR 17, 2023. [Accessed 24-Apr-2023].

[71] Rajeev Sobti and Ganesan Geetha. Cryptographic hash functions: a review. *International Journal of Computer Science Issues (IJCSI)*, 9(2):461, 2012.

[72] B. Sriman, S. Kumar, and Shamili Prabakaran. *Blockchain Technology: Consensus Protocol Proof of Work and Proof of Stake*, pages 395–406. 09 2020.

[73] Hamed Taherdoost. Smart contracts in blockchain technology: A critical review. *Information*, 14(2):117, Feb 2023.

[74] Harald Vranken. Sustainability of bitcoin and blockchains. *Current opinion in environmental sustainability*, 28:1–9, 2017.

[75] Dejan Vujičić, Dijana Jagodić, and Siniša Ranđić. Blockchain technology, bitcoin, and ethereum: A brief overview. In *2018 17th international symposium infoteh-jahorina (infoteh)*, pages 1–6. IEEE, 2018.

[76] Georgia Weston. Solidity for Beginners- Remix IDE Tutorials — 101blockchains.com. `https://101blockchains.com/remix-ide-tutorials/`, August 30, 2022. [Accessed 24-Apr-2023].

[77] Peter Wolf, Rushdi Nackerdien, and Domenico Tuccinardi. *Introducing electronic voting: essential considerations*. Policy paper. International Institute for Democracy and Electoral Assistance, Stockholm, 2011.

[78] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. Blockchain technology overview. *arXiv preprint arXiv:1906.11078*, 2019.

[79] Jesse Yli-Huumo, Deokyoon Ko, Sujin Choi, Sooyong Park, and Kari Smolander. Where is current research on blockchain technology?—a systematic review. *PLOS ONE*, 11:e0163477, 10 2016.