

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
MOHAMED BOUDIAF UNIVERSITY - M'SILA

Faculty of Mathematics and Computer Science
Computer Science Department

N° :



FIELD : Mathematics and Computer Science

FILIERE : Computer Science

OPTION : SIGL

Thesis presented for the obtention
Of Academic Master's degree

By: MEKKI Ala eddine

Entitled

**Study and design an electoral system based on
blockchain technology**

Presented to the jury composed of :

Dr. MOKHTARI Rabah	Université de M'sila	President
Dr. MOUSSAOUI Adel	Université de M'sila	Supervisor
Ms. BAYA Chalabi	Université de M'sila	Examiner

Academic year: 2022 / 2023

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
MOHAMED BOUDIAF UNIVERSITY - M'SILA

Faculty of Mathematics and Computer Science
Computer Science Department

N° :



FIELD : Mathematics and Computer Science

FILIERE : Computer Science

OPTION : SIGL

Thesis presented for the obtention
Of Academic Master's degree

By: MEKKI Ala eddine

Entitled

Study and design an electoral system based on
blockchain technology

Presented to the jury composed of :

Dr. MOKHTARI Rabah	Université de M'sila	President
Dr. MOUSSAOUI Adel	Université de M'sila	Supervisor
Ms. BAYA Chalabi	Université de M'sila	Examiner

Academic year: 2022 / 2023

Acknowledgement

First of all, Alhamdulillah, Alhamdulillah, Alhamdulillah.

I would like to express my deep gratitude to my supervisor, Moussaoui Adel, for his invaluable help and patience throughout this thesis. His expert guidance and dedication have been of paramount importance to the success of this thesis.

I would like to express my heartfelt gratitude to my mother and father for their unwavering support throughout my studying journey. Their love, encouragement, and belief in me have been the driving force behind my accomplishments.

I would like to extend my heartfelt gratitude to my sisters and my brothers for their constant support and encouragement during my studying journey. I am truly blessed to have such incredible sisters and brothers who have stood by me, cheered me on, and celebrated my achievements.

Ala eddine.

Contents

1	Introduction générale	1
1	Chapter 1 Blockchain	2
1.1	Introduction	2
1.2	Blockchain 1.0 - Bitcoin	2
1.3	Creation and Governance of bitcoin	3
1.4	Double Spending Problem	3
1.5	Solution of double spending problem with bitcoins	4
1.6	Byzantian Generals Problem	5
1.7	Solution of Byzantian Generals Problem With POW	5
1.8	Understanding Blockchain Technology	6
1.9	Blockchain Characteristics	7
1.9.1	Functional Characteristics	7
1.9.2	Emergent Characteristics	8
1.10	Blockchain Layers	8
1.11	Blockchain Architecture	10
1.12	Explaining Blockchain Elements	12
1.12.1	Data inside the block	12
1.12.2	Block hashing	14
1.12.3	Generating signature	15
1.13	Blockchain functionality	15
1.14	Consensus	18
1.14.1	Consensus mechanism	18
1.14.2	Consensus Working	19
1.14.3	Proof of work	19
1.14.4	proof of stake	20
2	Chapter 2 - Technical principals of blockchain	22
2.1	Centralized, Decentralized And Distributed Networks	22
2.1.1	Centralized Network	22

2.1.2	Decentralized Network	22
2.1.3	Distributed Network	22
2.2	Distributed Ledger	23
2.3	Peer-to-peer Network	24
2.3.1	Traditional Client-Server Network	24
2.3.2	Client-Server vs Peer-to-peer Networks	24
2.4	Cryptography in Blockchain	25
2.4.1	Cryptography Definition	25
2.4.2	Encryption and Decryption	25
2.4.3	Cryptography Asymmetric	25
2.4.4	Cryptography Symmetric	26
2.4.5	Digital Signature	27
2.4.6	Hashing	28
2.5	Permissionless versus Permissioned Blockchains	28
2.6	Blockchain types	29
2.6.1	Public Blockchain	29
2.6.2	Private Blockchain	29
2.6.3	Hybrid Blockchain	29
2.7	Blockchain Actors	30
2.8	Blockchain use cases	31
2.9	Smart Contracts	33
2.9.1	Definition Of Smart Contracts	33
2.9.2	Smart contracts characteristics	33
2.9.3	Smart contracts Life cycle	34
2.9.4	Consensus Mechanisms and Smart Contracts	35
2.9.5	Smart Contracts use cases	35
3	Chapter 3 - Electronic voting	38
3.1	Introduction	38
3.2	The importance of secure and efficient voting systems	38
3.3	Electronic voting systems	39

3.3.1	Definition of Electronic voting systems	39
3.3.2	Characteristics of Electronic voting systems	39
3.3.3	Types of electronic voting systems	39
3.3.4	Considerations for electronic voting	40
3.3.5	Advantages and disadvantages of electronic voting systems	42
3.3.6	Challenges and concerns associated with electronic voting	43
3.4	Electronic Voting with blockchain	43
3.4.1	Motivation for using blockchain in electronic voting systems	44
3.4.2	Benefits of integrating blockchain technology into voting processes	44
3.4.3	Blockchain Addressing the Challenges of Electronic Voting	45
4	Chapter 4 - Design and implementation of voting system	47
4.1	Introduction	47
4.2	Ethereum Blockchain	47
4.2.1	Definition	47
4.2.2	Ethereum Virtual Machine	47
4.2.3	Ethereum Transaction	48
4.2.4	Transaction: Contract Creation	48
4.2.5	Smart contracts in the Ethereum blockchain	49
4.2.6	Solidity language	49
4.2.7	Ethereum Tokens	49
4.2.8	Wallets	50
4.3	ERC20 Token contract	50
4.3.1	ERC20 Token Standard (Ethereum Request for Comments 20)	50
4.3.2	Purpose of ERC-20 Token	51
4.3.3	Design of ERC-20 Token	52
4.3.4	ERC-20 Token Information	53
4.3.5	ERC-20 Token Mappings and Events	53
4.3.6	ERC-20 Token Initialization	55
4.3.7	Token Transfer	56
4.3.8	Approvals and Allowances	57

4.3.9	Token TransferFrom	58
4.3.10	Conclusion	59
4.4	Voting Smart Contract	60
4.4.1	Electoral systems types	60
4.4.2	Key Components of Voting Smart Contract	61
4.4.3	Voting Smart Contract State variables	63
4.4.4	Modifier in the Voting contract	63
4.4.5	Voting contract Initialization	64
4.4.6	ERC20 Token Integration	64
4.4.7	Voter Registration	66
4.4.8	Voting Process	67
4.4.9	Winner Determination	68
4.4.10	Additional functions present in your smart contract	69
4.4.11	Summary	69
4.5	Development	70
4.5.1	Development tools	70
4.5.2	Deploying the smart contracts using truffle and ganache (Local)	71
4.5.3	Smart Contracts deployment on Testnet	76
4.5.4	Screenshots from the user interface of the application	86
5	General conclusion	88

List of Tables

1	Comparison of Client-Server and Peer-to-peer Network	24
2	registeredVoters Mapping representation	62
3	Voters Mapping representation	62
4	Voting Smart Contract State variable details	63
5	Voting Smart Contract Instruction details	64

List of Figures

1	Double spending problem [6]	4
2	Byzantian Generals Problem[9]	5
3	Blockchain Layers[11]	9
4	Blockchain Architecture [12]	10
5	Block Structure [14]	12
6	Blocks on the blockchain [15]	13
7	Signature link of the blocks [15]	13
8	Small change of data in the blocks [15]	13
9	Block hashing [15]	14
10	Signature link of the blocks [15]	15
11	Blockchain functionalit [13]	16
12	Bitcoin transaction [16]	18
13	Proof of work demonstration[19]	20
14	Proof of stake demonstration [19]	21
15	Different networks [23]	23
16	Asymmetric Encryption[29]	26
17	Symmetric Encryption [29]	26
18	Digital Signature Process[31]	27
19	Hashing Process[32]	28
20	Blockcahin Types[34]	30
21	Smart contracts Life cycle Major phases[41]	35

22	Ethereum Virtual Machine[78]	48
23	ERC20 Class diagram	52
24	ERC20 Use case diagram	52
25	ERC20 Initialization process that occurs within the constructor	55
26	Sequence diagram of Token transfer	56
27	Sequence diagram of Approval	57
28	Sequence diagram of TransferFrom	59
29	Candidate struct	61
30	IERC20 Interface	64
31	Sequence diagram of Voter Registration	66
32	Sequence diagram of Voting Process	67
33	Sequence diagram of Winner result	68
34	Truffle initialisation	71
35	Project directory	71
36	Truffle network configuration	72
37	Compiling the smart contracts	73
38	Ganache Started	73
39	Code to deploy the Smart Contracts	74
40	migrating the Smart Contracts	74
41	Smart Contracts Deployed	75
42	Getting accounts	75
43	Interacting with smart contract	76
44	Switching to Sepolia test network	77
45	Getting some ETH	77
46	Account have eth	78
47	Deploying ERC20 Smart Contract	79
48	ERC20Contract Deployed	80
49	Timestamp Conversion	80
50	Deploying Voting SM With Setting Times ans ERC20Token Address	81
51	Account hve eth	82

52	Admin Registring Voter	83
53	Voting Process	84
54	Functions And Calls to interact with Voting Smart Contract	85
55	Admin Dashboard	86
56	Voting Page	86
57	Results Page	87

INTRODUCTION GÉNÉRALE

The fast advancement of technology has transformed our world at an incredible speed. Today, new technologies are being developed all the time, as countries compete to develop their systems to stay at the top of the technological field. Technology has entered all domains of human society, and one of them, the voting system, is considered to be one of the most important foundations of global democracy.

In this context, blockchain technology has become more popular and is being used in a variety of fields. The blockchain concept is based on a decentralized, secure network, where transactions are recorded transparently and immutably. This innovative technique offers many potential benefits for the voting process, thus offering an exciting chance to enhance our democracy.

The implementation of blockchain technology within the electoral process, commonly known as e-voting on blockchain, could bring about considerable improvements. The use of blockchain would make it possible to ensure the transparency, security and integrity of votes, and reduce the risk of electoral fraud. Each individual vote is permanently registered and available to all authorized participants in a decentralized ledger, making manipulation and falsification less possible.

In addition, blockchain technology would boost public trust in the electoral process. Voting results would be independently auditable, thus helping reduce concerns and complaints around the correctness of the outcome. Voters will also be given a transparent way of tracking their vote, ensuring privacy and maintaining traceability.

Implementing e-voting on blockchain could help governments to simplify and speed up the electoral process. Conventional voting systems at present are usually complicated and expensive systems to set up, demanding a lot of infrastructure and logistics. By utilizing blockchain technology, it would be much simpler to build e-voting solutions that are accessible to all, reducing the challenges linked to electoral participation.

Concerns and discussions regarding the possible effects on democracy are being raised by the use of blockchain technology in the voting process. While blockchain-based electronic voting has many potential advantages, there may also be problems that need to be resolved.

CHAPTER 1

BLOCKCHAIN

1 Introduction

Blockchain has recently gained popularity in the world of technology because it provides ways and solutions to numerous issues in various fields involving a type of value exchange, and because of the many features of blockchain that enable this technology to be adjusted as necessary. Research to develop digital currencies has been ongoing since the 1980s, and over the years many proposals have been published by many scientists in the field of computer and cryptography using cryptographic techniques and proposed intelligently designed ideas. However, they have always faced difficulties and problems, such as the problem of trust, the problem of double spending, a computational problem, a consensus mechanism between nodes, and some security issues. A paper by Satoshi Nakamoto titled "Bitcoin: A peer-to-peer Electronic Cash System" was published in 2008. A peer-to-peer electronic cash system without a middleman bank to handle transactions was the original concept. Based on earlier studies and proposals in hashing, cryptography, digital signatures, Merkle trees, cryptographic time stamping, and proof of work, all of these concepts helped lead to the creation of this digital currency, which successfully solved the famous Byzantine Generals' Problem and the double spending problem, and It introduced the term chain of blocks. [1]

2 Blockchain 1.0 - Bitcoin

Since bitcoin was the first actual blockchain implementation, it is necessary to start there in order to understand blockchain. In 2008, Satoshi Nakamoto published a white paper titled "Bitcoin: A Peer-to-Peer Electronic Cash System" in which he introduced bitcoin, a decentralized cryptocurrency that doesn't require a third party or central authority to perform online payment immediately. Instead, it uses a peer-to-peer network to avoid the issue of double spending and a digital signature to ensure the system's high level of security. [2]

Due to multiple important factors that contribute to an increase in bitcoin's value:

- No need of third trusted party, which makes it decentralized [3]
- Based on the Peer-to-Peer Network [3]
- Privacy by making identities anonymous and transactions is public [3]
- Secure because it's hard to hack [3]
- Open Source [3]
- Can't be changed or reversed [3]

3 Creation and Governance of bitcoin

The Genesis block is the first bitcoin block created in early 2009 with a reward size of 50 bitcoin, then bitcoin is expected to be halved exactly every 210,000 blocks, to date, the reward is halved three times, equal to 6.25 bitcoins per minted block, this process continues until the 21 million bitcoins are finally minted.[3] Bitcoin network controlled by all the users of it requiring that every one of them to adhere to the same rules of the system which guarantee a strong consensus in the system. [4]

4 Double Spending Problem

One of the biggest challenges facing digital currencies is the double spending problem, as digital currencies are just digital information so it can be duplicated and used more than one time in two parallel transactions in two different processes in parallel. To explain the problem, we will proceed to a demonstration with the help of a simple possible transaction in the real world, if someone wants to buy fruits, he will pay directly with the cash of 500 da, this amount will be direct in the cash box, then the salesman will confirm the transaction and will give you your purchase, so you cannot spend the same cash several times, the same thing with big money transactions those operations hoes through the banks where they play the role of observing and assuring that the transactions are not manipulated. [5]

On the other hand, in the case of digital currencies in the absence of a third party to verify the transaction, let's say that user "A" wants to pay user "B" 10 coins and transaction can take time to

be verified, and during that the user "A" can use the same 10 coins to user "C" in parallel payment and performing the double spending problem. [6]

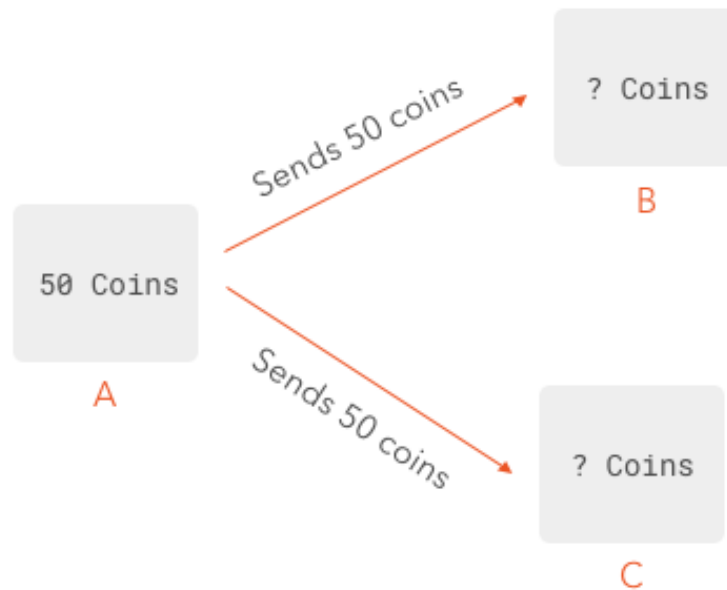


Figure 1: Double spending problem [6]

5 Solution of double spending problem with bitcoins

The solution to this problem is carried out by bitcoin. By using a mechanism called proof of work that ensures transactions are correct by using mining, miners validate and secure transactions and earn new bitcoins as a reward for their work of verifying transactions, As more miners join the network, the difficulty of mining increases, making this mechanism more secure and trustworthy, and also eliminates the need for an authority and maintaining the integrity of the bitcoin network.[7]

6 Byzantian Generals Problem

This is represented by the existence of two armies situated in different positions and there is an enemy city in common between these armies, this city can resist the attack of a single army but not more than one, so these armies must come to an agreement to attack in the same moment to succeed in winning against the enemy or it will be chaos and they will lose against their enemy, the communication is done only by messengers sent by each army. However, the messengers in their mission of delivering the message from one side to the other can change the message, lie or even get caught up in the enmity and replace it with a fake messenger who will spread a false message to attack in the wrong time which means the absence of consensus between the two armies.[8]

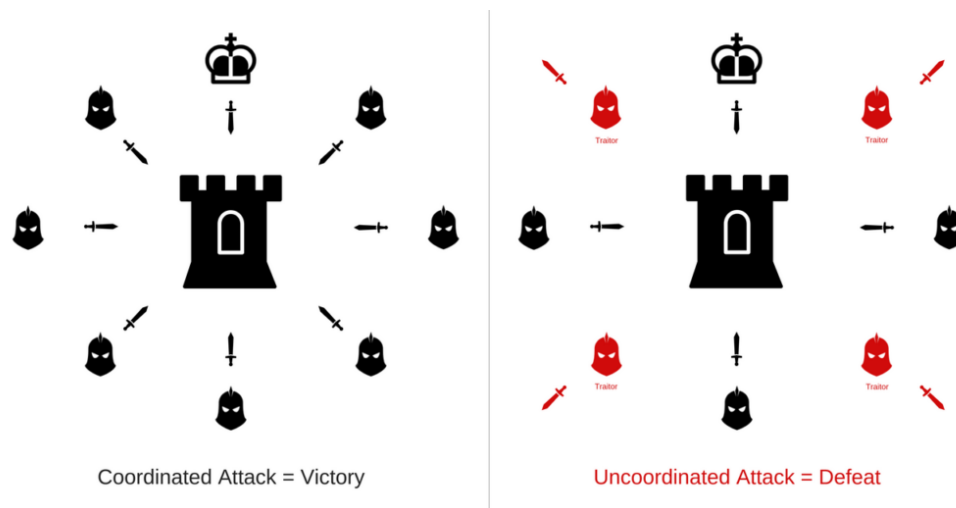


Figure 2: Byzantian Generals Problem[9]

7 Solution of Byzantian Generals Problem With POW

In the bitcoin system there is a similar case to this problem which can be seen in the lack of trust between the nodes in this decentralized system, bitcoin offers an effective solution that will allow the arrival of a consensus between all parties in the system without another third party or central authority all this using the proof of work which means solving a complex mathematical problem that takes a lot of time and make it easy for other parties to validate. [8]

In our general scenario the solution is as follows:

1. The armies must solve the Pow problem to create a block and broadcast it in the network so that all armies receive it. [8]
2. Each army must now verify the validity of the block's profile of work and this allows it to be added to the chain or not if it's invalid. [8]

Due to the amount of energy and work consumed to produce a block, Invalid or empty blocks are less likely to be created by miners because of the work involved, because it will be rejected by others and will not get any reward for their effort. [8]

8 Understanding Blockchain Technology

To understand the blockchain we must first define it, in simple way "Blockchain is an ever-growing, secure, shared record keeping system in which each user of the data holds a copy of the records, which can only be updated if all parties involved in a transaction agree to update." [1]

And in another more technical definition "Blockchain is a peer-to-peer, distributed ledger that is cryptographically-secure, append-only, immutable (extremely hard to change), and updateable only via consensus or agreement among peers". [1]

In view of this definitions a peer-to-per network allow the users to communicate with each other directly with no extern coordinator or a third-party, additionally to that the data saved in the ledger is spread in the network among all the users holding a duplicated version of it, another thing that the ledger is secured using cryptography methods allowing to be resistant to any attempt of manipulation or falsifying, also the records in the blockchain are time-stamped in chronological order and immutable making it hard to modify it. [1]

9 Blockchain Characteristics

There are several characteristics of blockchain that contribute to its effective operation and give benefits to apps that use and rely on this technology. These characteristics include functional characteristics that govern how blockchain works and emergent characteristics that arise as a result of the first characteristics.

9.1 Functional Characteristics

- a) **Decentralized Network** : Among the most important features of this technology is that a decentralized network contains nodes referred to as peers; each peer has a specific role and permission assigned by the application; this feature allows this network to operate without a central authority, so these participants will cooperate to handle transactions and communication in this network.[10]
- b) **Immutable**: The immutable property signifies that when data is appended to the blockchain, it will always remain in the chain and very difficult to change or modify, as this data is in the form of blocks linked to each other using hashes, any attempt to modify in one block will affect all blocks will cause a problem in the chain and peers in the network will be alerted.[10]
- c) **Distributed Shared Ledger** It is a ledger in which records are maintained that are indistinguishably replicate and shared among network nodes, allowing participants to access, monitor, analyze, and evaluate the status of a transaction throughout its life cycle.[10]
- d) **Consensus** : Another key feature that is considered a technique to validate any transactions that are taking place is that members of the network must come to an agreement and approval so that the transaction can be completed, this is what we call the consensus mechanism. Appending a block is dependent on whether or not consensus has been reached, thus the block is successfully added or considered orphaned, this technique aids in improving trust between the nodes.[10]
- e) **Security** : Blockchain is based on cryptographic algorithms to secure operations on it, making the network consistent and ensuring integrity, using a hashing function that returns a fixed length output regardless of the input length, and there are certain rules that must be followed to approve any transactions and in the generation of blocks.[10]

9.2 Emergent Characteristics

- a) Transparency : While blockchain uses a peer-to-peer protocol, this protocol makes all participants equal in terms of power, permissions, and access to the network, allowing all participants to view and verify ongoing transactions. Furthermore, all nodes have the same replicate of the ledger, making it easily accessible, and operations on the blockchain are transparent as a result.[10]
- b) Anonymity : Due to the lack of a central authority, the identity of blockchain participants is anonymous, and transactions are conducted anonymously, anonymity degree varies depending on the type of blockchain application.[10]
- c) Auditability : This technology enables audibility thanks to the way it operates by making records accessible and unmodifiable throughout their lifespan, as well as employing time-stamping to maintain track of the sequence of transactions.[10]
- d) Trust : While the blockchain eliminates the need for a central authority, trust is spread across the network so that all nodes are responsible for it and work together to approve transactions and achieve the required consensus. [10]

10 Blockchain Layers

As shown below in the accompanying graphic, blockchain may be viewed as a layer of a peer-to-peer distributed network operating on top of the internet. It is comparable to TCP/IP-based SMTP, HTTP, or FTP:

- a) TCP/IP Network : The first layer in the Blockchain is the TCP/IP system which is the internet, and the way it simply works with all its protocols. The internet offers the possibility to distribute applications and communication to the blockchain.[11]
- b) Peer-to-peer Protocols: On the top of the Network layer there's peer-to-peer layer which includes information-propagation techniques like broadcasting blocks with flooding protocols in unstructured peer-to-peer.[11]
- c) Cryptography Algorithms After this comes the Cryptography layer which plays an important role in the foundation of blockchain, this layer incorporates important cryptographic proto-

cols used to maintain the security and the integrity of data, such as public and private keys encryption, hash function and digital signature[11]

- d) Consensus Algorithms : The next level is where consensus algorithms are found, which are another fundamental idea underlying the blockchain. Consensus algorithms either use traditional fault tolerance-based mechanisms that rely on publishing and verifying signed messages or proof-based consensus mechanisms that require a lot of work to ensure agreement between all nodes in the network.[11]
- e) Execution : Then there's The Blockchain's execution layer serves as a simple interface between developers and the real Blockchain's capabilities. This layer is in charge of carrying out the fundamental functions of a Blockchain network, including transactions, storing data in the network, virtual machines, and carrying out various smart contracts.[11]
- f) Applications : Finally, we have the application layer that includes decentralized applications, Smart contracts, this layer contains the business logic of blockchain implementations where the blockchain is accessed via applications known as decentralized applications.[11]

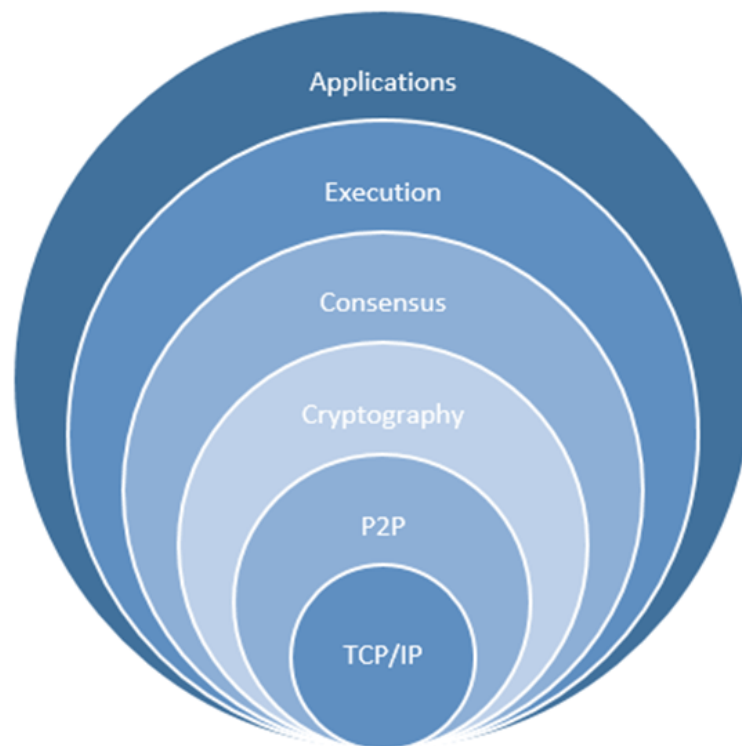


Figure 3: Blockchain Layers[11]

11 Blockchain Architecture

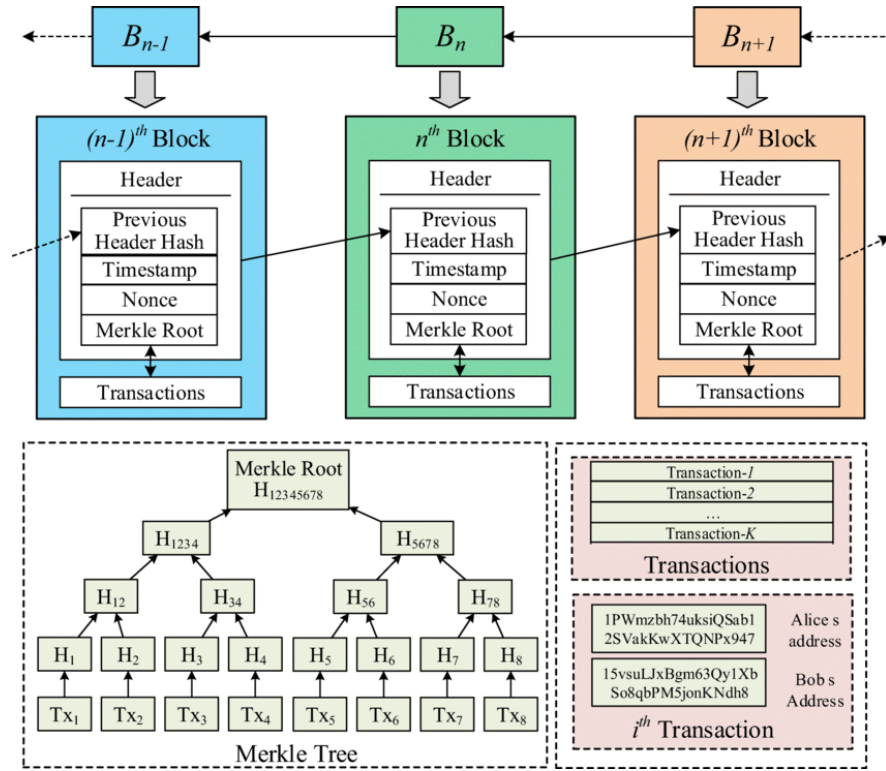


Figure 4: Blockchain Architecture [12]

These are the elements that we will run to when it comes to blockchain. The elements of a typical blockchain are shown above, though the data within each block might vary depending on the technology used:

- Address** : Addresses are unique IDs used to identify the parties involved in a transaction made using blockchain technology. In general, an address is a key that is available to everyone or one that is obtained from a public key.[13]
- Transaction** : Any exchange of a data or value between to addresses in the network, this is considered a key and fundamental component of the blockchain.[13]
- Block** : A block is made up of many transactions as well as other things like the nonce, timestamp, and preceding block hash (hash pointer). A block is made up of a block header and plenty of transactions that are grouped altogether and properly ordered.[13]

Here's elements in the block:

- Reference to previous Block: the first thing included in the head of the block is the value of hashing the preceding block header linked to the actual block, this value is unique and bring a certain level of security to the chain and blocks validation, all blocks include this value except the genesis block which is the first block in the chain the value will be given with the creation of the blockchain.[13]
 - Nonce : another important value that appears in a block, is an ever-generating, one-time-use number. To offer authentication, and encryption, a nonce is widely employed in many cryptographic processes. For transaction replay protection and Pow consensus.[13]
 - Timestamp : this value indicates the time when the block is generated and help in the keep a sequence of the blocks clear.[13]
 - Merkle root : its value is the hash of every single node in the Merkle tree, this tree is mainly used to verify and secure data and transactions in an efficient and secure way, root of this tree spares us from verifying all Merkle tree transactions because it's sufficient to only verify the root of the tree.[13]
 - Transaction : One more thing to added to the block is transactions that constitute the body of the block, any operation or event of transferring of value is recorded as a transaction.[13]
- d) Node : Depending on their goals in the blockchain and the kind of blockchain, nodes are not exclusively committed to performing the same job. Instead, they perform a variety of tasks. A node can keep the blockchain's transaction record, propose and approve transactions, and carry out mining to aid in consensus and protect the blockchain via a consensus process, The presence of various nodes throughout the global network ensures the network's ability to withstand centralized attacks, thus enhancing its resilience, there are two general types of the: full nodes and light nodes.[13]

The following structure is a simple block diagram that represents a generic block:

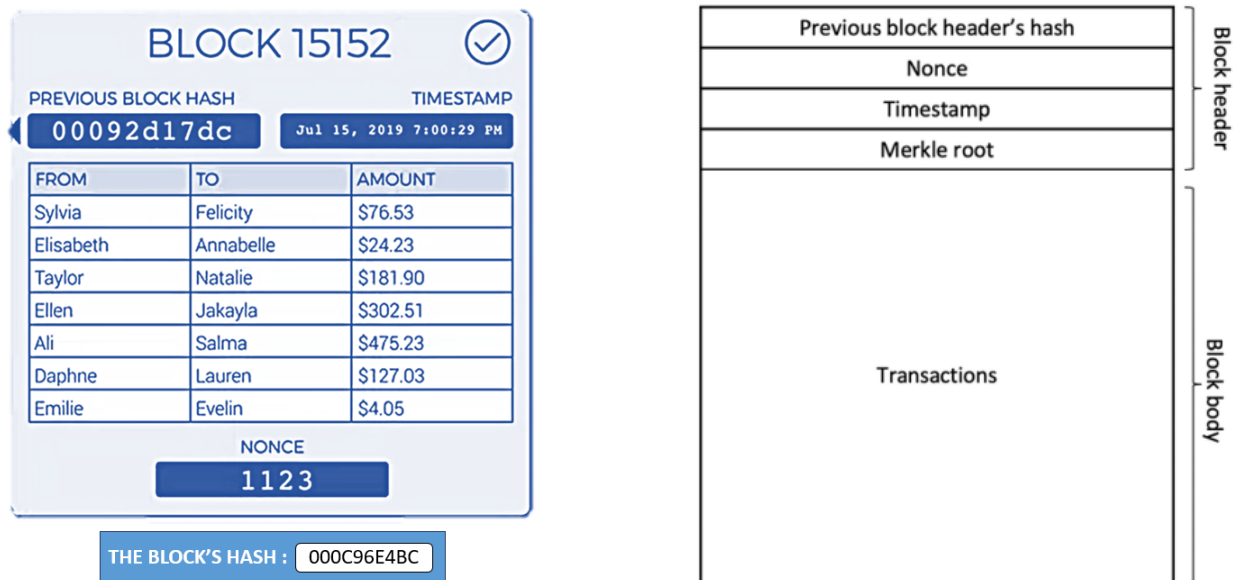


Figure 5: Block Structure [14]

12 Explaining Blockchain Elements

Here we will discuss the block element in the blockchain :

12.1 Data inside the block

Based on the technology that's being utilized, the data within each block varies. The blocks on the Bitcoin blockchain include the sender's and receiver's IDs as well as the total quantity of bitcoins being exchanged. Transaction information from every bitcoin transaction makes up the bulk of the data on the blockchain.[15]

Block 1 displays the first transaction, whereas blocks 2 and onwards explain subsequent transactions. In order to be chained together, each block is given a special digital signature that corresponds to the string of data in that block. New digital signatures are created via hashing whenever a block is altered.[15]

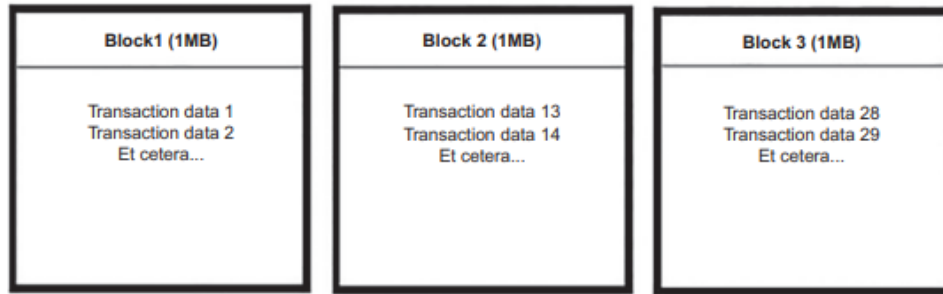


Figure 6: Blocks on the blockchain [15]

Block 1 records two transactions and receives the "X32" signature for this string of data. Here is an illustration of that, According to this structure, block 2's signature is dependent upon block 1's signature, the signature contribute in linking blocks and constructing the blockchain.[15]

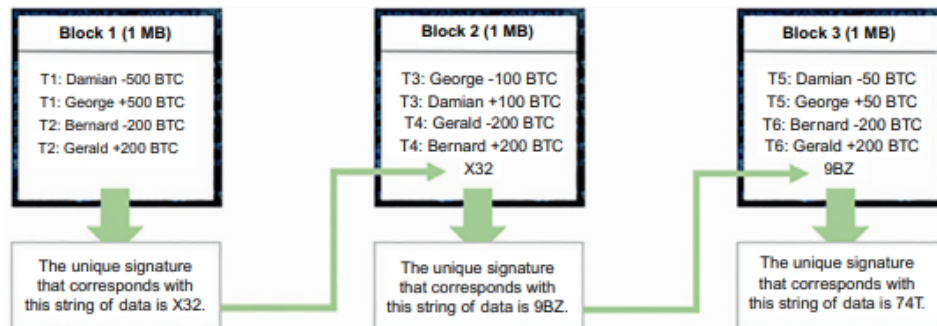


Figure 7: Signature link of the blocks [15]

And because the the dependency between the blocks in signature any modification in first block will affect the signature used in the next block and the linking between them will fail.[15]

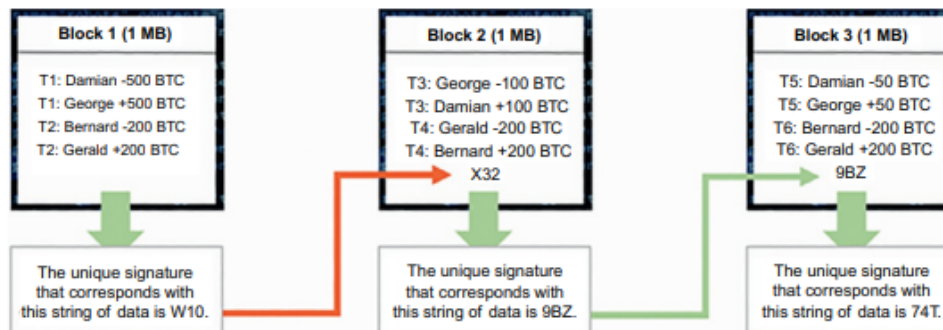


Figure 8: Small change of data in the blocks [15]

12.2 Block hashing

A block's hash serves as its digital signature. thus, the hashing algorithm is crucial to blockchain, and If a hash is changed, the block's original state is lost. The chain connecting the blocks is formed by the hash of the preceding block, providing a reliable proof of concept and a safeguard against manipulation.[15]

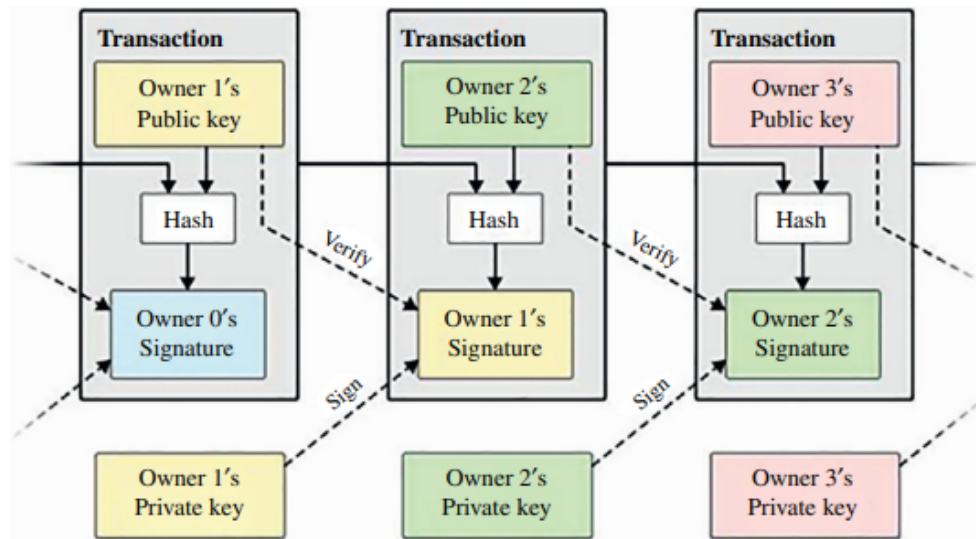


Fig. 5 Hash of a block.

Figure 9: Block hashing [15]

12.3 Generating signature

Block 1 is a record of a single, signed transaction in which A sent B 100 bitcoin. A cryptographic hash algorithm was used to generate this signature, which transforms any input text to a distinctive 64-digit string of result. Supposed we have this text of block 1 transaction as input and we use hashing function to hash it, [15] and then we will add this value to the next block we will get:

$$\begin{aligned} & \text{SHA-256}(\text{Thomas -100 David +100}) \\ & = \\ & 1d49afa70970585fe0ea2d29e81f654958b88f31dd0ad9049884315648f97c19[15] \end{aligned}$$

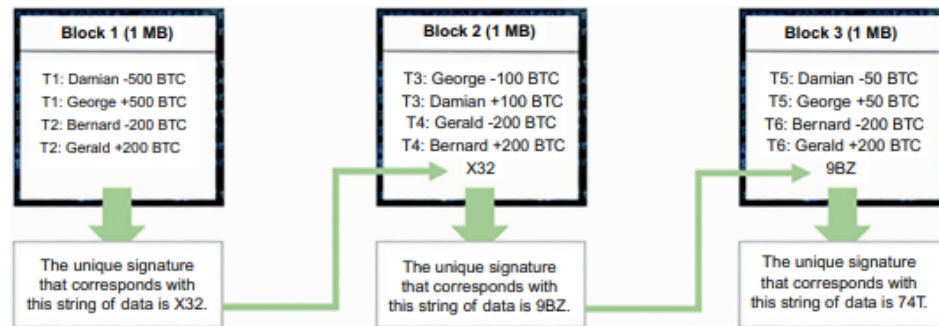


Figure 10: Signature link of the blocks [15]

Next, to get the signature of the 2nd block :

$$\begin{aligned} & \text{SHA-256} \left(\begin{array}{c} \text{David -100 Jimi +100} \\ 1d49afa70970585fe0ea2d29e81f654958b88f31dd0ad9049884315648f97c19 \end{array} \right) \\ & = \\ & a12b210e848a583710b7f8e66dd5d16a743fce72c0784b4256d03927c59367f6 \end{aligned}$$

13 Blockchain functionality

We will discover how the mechanism of blockchain operates. In the blockchain network, there are nodes these nodes can play either create new blocks or node that validate and digitally sign the transactions, these nodes are usually in competition to create a block and append it to the

blockchain to determine which node get the right to do that a decision must be taken using a consensus mechanism which is the process of coming to a settlement on the final outcome [13]

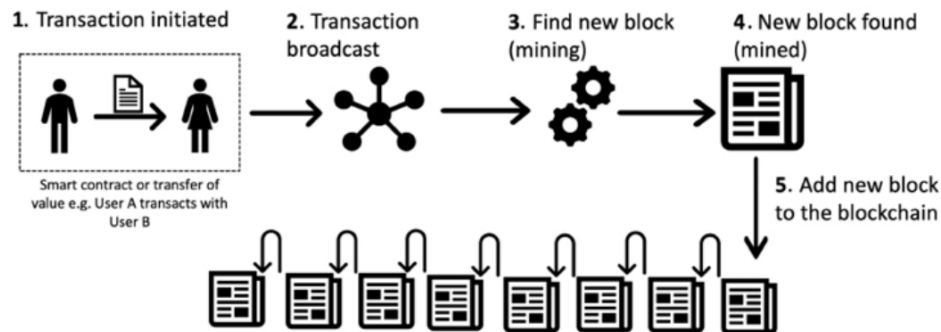


Figure 11: Blockchain functionalit [13]

The procedure below must be taken in order to produce and add blocks to the blockchain, as well as to verify transactions in the blockchain:

- i) Transaction is initiated : A node begins a transaction by initially creating it, and then using its private key to sign it digitally, usually the transaction is an exchange of value between participants, the source and destination addresses, applicable rules, and other validation data are often included in a transaction.[13]
- ii) Transaction is validated and broadcast : Typically, data-dissemination methods are used to propagate (broadcast) a transaction to other peers who then verify it in accordance with pre-determined validity criteria. A transaction is also validated to make sure it is legitimate before it is propagated.[13]
- iii) Find new block : The transaction is added to a block and mining begins as soon as it is received and verified by particular users known as miners on the blockchain network. Here, a mechanism known as mining is used by nodes referred to as miners to compete to complete the block they have constructed.[13]
- iv) New block found : The transaction is added to a block and mining begins as soon as it is received and verified by particular users known as miners on the blockchain network. Here, a mechanism known as mining is used by nodes referred to as miners to compete to complete the block they have constructed.[13]

- v) Add new block to the blockchain Following the validation of the newly produced block and the execution of any associated transactions or smart contracts, it is spread to other peers. Peers verify and carry out the block as well. The block is now included in the blockchain (ledger), and the next block cryptographically ties back to this one. [13]

In order to illustrate the procedure of a blockchain transaction, we will present a demonstration involving the transfer of bitcoin between two individuals following these steps:

- When Alice intends to transfer coins from her wallet X to Bob's wallet Y, she initiates a transactional data request labeled as "t" and sends it to Bob. This request is then broadcasted across the entire network, both Alice and Bob have keys (public key It is a combination of characters and numbers that functions much like an email address and private key this key is used as a "password" to the account and that approves the digital currency transaction.) [16]
For the transaction, Alice sign using her private key this transaction and all of its accompanying parameters are sent to the blockchain: Source transaction of coins, the quantity of coins to be sent and Bob's public address.[16]
- The request is acknowledged by the distributed nodes, which proceed to incorporate the transactional details involving Alice and Bob into their respective ledgers, thereby updating them.[16]
- Once the ledgers have been updated, Alice calculates a digital signature and disseminates it across the network.[16][16]
- The transaction is confirmed and authenticated by a miner node. It works out the proof-of-work computations to make sure the received digital signature matches by getting the right hash by resolving a mathematical challenge created using hashing cryptography, the first miner to create a hash with a certain number of leading 0s wins that block and may add it to the network.[16]
- The result is sent to all nodes for confirmation and verification whether the proof-of-work successfully aligns with the s digital signature.[16]
- The remaining miner nodes verify both the digital signature and the proof-of-work. The

miner node responsible for calculating the proof-of-work is properly rewarded, often in monetary terms, if the verification procedure is successful.[16]

- The block is added to the chain that has already been successfully verified when it has been completed, and the transaction is then broadcast to the whole blockchain network.[16]
- The bitcoins are put into Bob's wallet Y by using the verified transaction designated as "t".[16]
- After decrypting the information using Alice's paired public key (PUK), Bob sends a confirmation back to her.[16]
- As soon as Alice gets a confirmation for the transaction, the transaction is complete.[16]

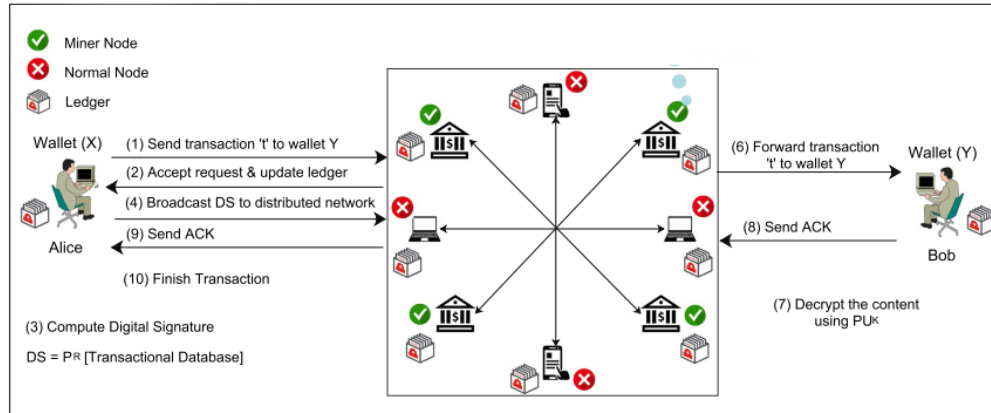


Figure 12: Bitcoin transaction [16]

14 Consensus

14.1 Consensus mechanism

We can define consensus mechanism as "A method of authenticating and validating a value or transaction on a Blockchain or a distributed ledger without the need to trust or rely on a central authority. Consensus mechanisms are central to the functioning of any blockchain or distributed ledger".[17]

14.2 Consensus Working

When a transaction is produced and transmitted to the network, it is signed with the signature of the initiator, whose approval is required to spend money, create a contract, or transfer data parameters related to the transaction. The transaction is legitimate and includes all the information required to be carried out if it is signed.[17]

A node linked to the blockchain network receives the transaction and verifies it using a set of rules called consensus rules. Valid transactions are transmitted to other connected nodes, whereas invalid transactions are rejected. This flooding strategy ensures that a legitimate transaction will quickly reach the entire network. As long as the senders utilize many nodes to make sure the transaction propagates, they are not required to trust the nodes they employ to broadcast the transactions. The transactions are signed or don't include any sensitive data, so the receivers don't have to trust the senders either.[17]

A transaction spreads to the network once it has been approved and included to a block. The new block gets linked into the blockchain after the whole network has reached consensus and all of the peers of the network approve it. All nodes in the blockchain network will accept a transaction as genuine after it has been published on the blockchain and sufficiently validated by future blocks to become a lifelong component of the public ledger.[17]

There are many different mechanisms that can build this consensus. Common consensus mechanisms such as proof-of-work (PoW), proof-of-stake (PoS), delegated proof-of-stake (DPoS), practical Byzantine fault tolerance (PBFT), proof-of-capacity (PoC), proof-of-activity (PoA), proof-of-publication (PoP), proof-of-retrievability (PoR), proof-of-importance (PoI), proof-of-burn (PoB), proof-of-elapsed time (PoET), and proof-of-ownership (PoO).[17]

14.3 Proof of work

is a consensus mechanism used in the Bitcoin, in the proof of work (PoW) model, a user who is the first to correctly solve a challenging computational puzzle publishes the following block. The puzzle is designed to be challenging yet checkable, making it simple for all other complete nodes to verify any suggested following blocks. The requirement must be satisfied for a block to be accepted. The requirement is that the block header's hash digest be smaller than a given number is a typical

puzzle technique. In an effort to locate a hash digest that satisfies the requirement, publishing nodes make minor adjustments to their block header. It becomes computationally expensive to repeatedly hash the block header, and the goal value may be changed over time to change the level of difficulty. As we said bitcoin uses this consensus model, the puzzle here is that the first miner to produce a hash with a certain amount of leading 0s will be the winner, The number of needed leading zeros is effectively increased or decreased depending on how tough the problem is to solve, the network's security is maintained by the network's nodes and the growing computational power over time, which also causes a general rise in puzzle complexity. A publishing node sends a block to all full nodes in the blockchain network along with a valid nonce and gets a reward with cryptocurrency for the all the computational work and therefore solving the puzzle, after confirming that the new block satisfies the puzzle requirement, the recipient full nodes add the block to their copy of the blockchain and broadcast it again to their peer nodes. The block is quickly spread around the network of active nodes, as only one hash is required to determine if the nonce solves the puzzle. [18]

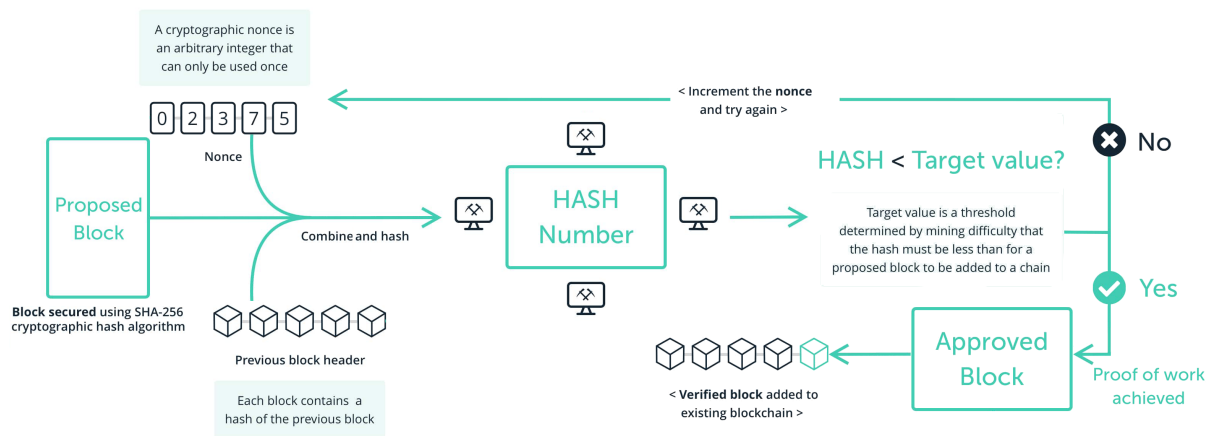


Figure 13: Proof of work demonstration[19]

14.4 proof of stake

Based on the notion that users are more likely to be committed to a system to work properly and more unlikely to want to harm it the more stake they have in it, the proof of stake (PoS) paradigm was developed. A stake is often a certain amount of bitcoin that a blockchain network user has bet on the network; once bet, the digital currency is no more usable. The percentage of a stakeholder's

stake in relation to the total quantity of staked bitcoin on the blockchain network determines the possibility that they would publish a new block on the network. Blockchain networks that employ proof of stake use a user's stake to determine when to publish new blocks. Some blockchain networks have chosen to operate without a block creation reward because the consensus model uses fewer resources than proof of work. Instead, all of the digital currency is divided among the users, and the payment for block creating is often made from transaction fees that users supply. The stake may be used in a variety of ways by the blockchain network. Here, we go through four strategies: delegate systems, multi-round voting, coin aging systems, and random selection of staked users. Users who have more stake are actually more probable to publish new blocks regardless of the specific method. Proof of stake systems allow the "rich" to stake more digital assets, but it is often expensive to collect the majority of digital assets. [18]



Figure 14: Proof of stake demonstration [19]

Participants stake their tokens in "Delegated Proof of Stake" and elect delegates to build new blocks. In Proof of Authority, consensus is based on reputation, whereas in Practical Byzantine Fault Tolerance, blocks are approved by special "ordering" or "leader" nodes. These controls are intended for private collaborative blockchains with partly trustworthy participants.[20]

CHAPTER 2

TECHNICAL PRINCIPALS OF BLOCKCHAIN

1 Centralized, Decentralized And Distributed Networks

1.1 Centralized Network

In a centralized network, all users and devices are linked by only one network controller. The main server may be set up as a connection for the client nodes to submit data queries. Major data processing is controlled and maintained current by the central server. Due to the need for a single network owner to connect all other satellite users and devices, this type of network is the most prevalent on the internet, creating a point of failure that could be exploited by malicious actors.[21, 22]

1.2 Decentralized Network

Instead of depending on a single central server, a decentralized network spreads information processing duties among a number of devices. Even if one of the master nodes fails or is hacked, the remaining servers may continue to provide users access to data, and the whole network will continue to function without interruption. Each device works as a miniature central unit that communicates autonomously with other nodes. Decentralized networks are possible due to technology advances, but still rely on primary servers and do not distribute data processing and storage evenly.[21, 22]

1.3 Distributed Network

A distributed network resembles a decentralized network in that it does away with an individual core master server. Since distributed networks are made up of alike, linked nodes, information possession and computing capacity are dispersed equally throughout the whole network. On a distributed network, decisions are made by each node participating to switch to a different state, and the system behaves differently depending on the overall outcomes of the decisions. Due to concerns about security, data storage, and privacy as well as the requirement for enhanced efficiency, systems

that are distributed are growing in popularity.[21, 22]

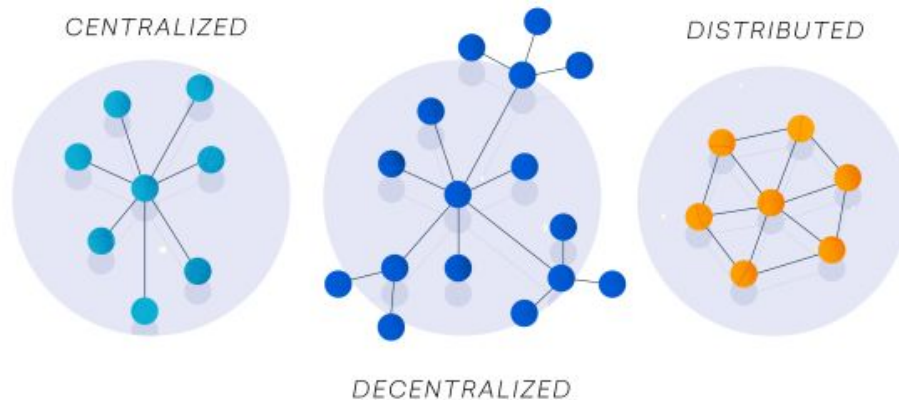


Figure 15: Different networks [23]

2 Distributed Ledger

A distributed ledger system keeps the data simultaneously at multiple places it does not have a central location to store information, in contrast to conventional databases. It's based on a peer-to-peer (P2P) network. Decentralization is made possible with the use of a consensus algorithm. The easiest way to describe DLT is as a distributed ledger that is replicated, synced, and distributed.[24]

There's some key characteristics of this technology:

- **Immutable:** Cryptography is used to secure data stored in a distributed ledger, ensuring that it cannot be modified or changed.[24]
- **Append only:** Distributed ledgers are append-only, meaning they provide the entire transactional history.[24]
- **Distributed:** The distributed architecture of the ledger is a fundamental trait, since each peer possesses a duplicate of the ledger in most DLTs.[24]
- **Shared:** The ledger is distributed across nodes, with some responsible for maintaining an exact copy, while others only have the information needed to function efficiently.[24]

There are three main types of distributed ledger systems available: Permissioned, Hybrid and Permissionless.

3 Peer-to-peer Network

Peer-to-Peer technology enables network users to communicate directly with one another, peers are nodes that have identical capability and carry out the same functions and they're also open, decentralized, and strong because they reciprocally provide and consume services. Nodes function as servers for downloading and sharing files, enabling a P2P network that is quicker, more secure, and more effective, The efficiency of blockchain technology increases with the number of active peers.[25]

3.1 Traditional Client-Server Network

A client-server network architecture is one in which the every node (computer or device) on a given network is one of two things: a client or a server. It is built on the consumer model, which states that clients get whatever is accessible on the servers. Servers are powerful machines that manage network activity and shared resources while also offering services through protocols such as HTTP and HTTPS. [26]

3.2 Client-Server vs Peer-to-peer Networks

Client-Server	Peer-to-peer Network
To access services as well as data, several client computers connected to one or more servers over a network connection.[26]	In order to share data and resources, a group of peer devices connects.[26]
The processing power of the server affects performance. It is a well-known fact that performance degrades if a greater number of client devices are connected to a given server.[26]	As a greater number of peer devices or nodes are joining the shared network, the network's functionality elevates.[26]
There is no shared resource between the inter-connected clients, and mostly just one server is responsible for handling everything.[26]	Each peer connecting to the network shares the workload.[26]
When a server fails, the service is absent, and all clients have connection issues til the server comes back up.	No major interruption of functionality among any of the current peers occurs in the event that one or more nodes cease to exist.[26]

Table 1: Comparison of Client-Server and Peer-to-peer Network

4 Cryptography in Blockchain

The blockchain uses cryptography to shield the data from any attempt to access from individual that They are not allowed to. It incorporates some concepts like encryption and hashing. It is used to safeguard participants, secure transactions, and stop double-spending. By guaranteeing that only the individuals who are meant for the transaction can access, look at, and handle them, it contributes to the security of blockchain network transactions.[27]

4.1 Cryptography Definition

The Greek words *kryptos* and *graphein*, which mean "hidden" and "write," are the building blocks of the process or system known as cryptography, that is utilized to secure data from outsiders.[27]

4.2 Encryption and Decryption

A communication that can be read is converted to an unreadable state via a technique of encryption. The process of returning an encrypted information to its original (readable) format is known as decryption. While the ciphertext message is the encrypted communication, the plaintext message is the original conversation.[27]

4.3 Cryptography Asymmetric

Asymmetric encryption, is a newer technology that encrypts plain text using a pair of keys. Secret keys are sent and received through the vast network, preventing dangerous individuals from misusing the keys. This type of encryption employs two associated keys to increase security, with anyone who wishes to communicate with you having free access to your public key and the other one private key being hidden to ensure nobody but you is aware. A private key is required to decode a public key communication, while a private key message may also be decoded employing a key that is publicly accessible. The key that is available to everyone is not safe since it is openly accessible and may be shared over the internet. An asymmetric key has a significantly greater potential for maintaining the safety of the data sent.[28]

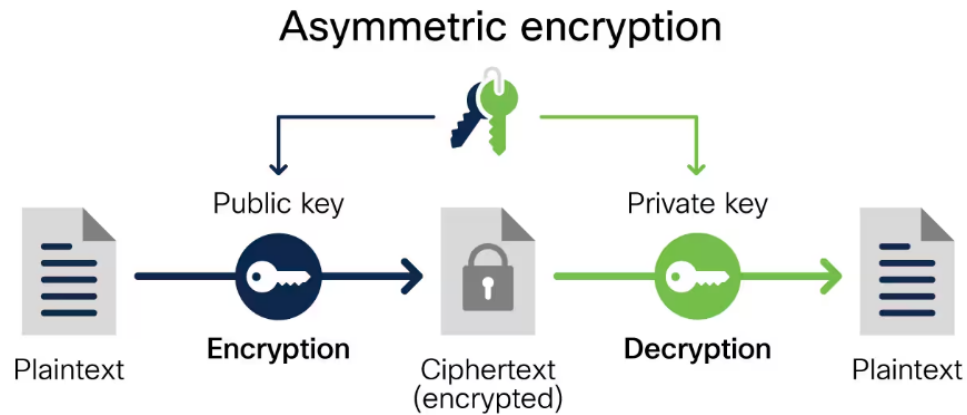


Figure 16: Asymmetric Encryption[29]

4.4 Cryptography Symmetric

Is a time-tested and widely used approach. This kind of encryption is the simplest since it just uses a single secret key to encode and decode data. Both the sender and the recipient must be aware of the secret key used to encrypt and decode all conversations. It is a message that is blended with plain text in order to change the information. The most extensively used algorithm is symmetric encryption, with AES-128, AES-192, and AES-256 being the most popular.[30]

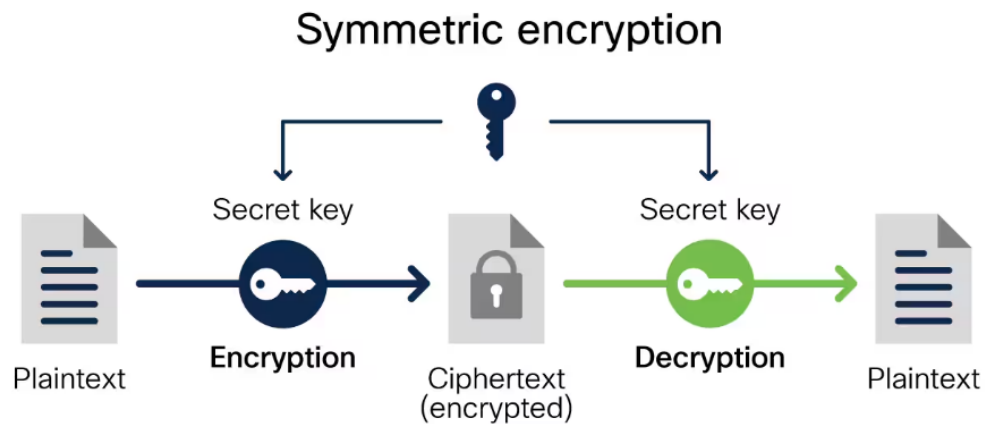


Figure 17: Symmetric Encryption [29]

4.5 Digital Signature

The purpose of digital signatures is equivalent to that of handwritten ones in that they assure that a message was created by the signer, that it wasn't played around with, and that the signer of a message should not be able to deny signing it. Digital signatures make use of public-key cryptography. The use of digital signatures is part of the Bitcoin protocol. In essence, bitcoin addresses are public keys. Each Bitcoin address is uniquely identified by the private key that corresponds to each public key.[31]

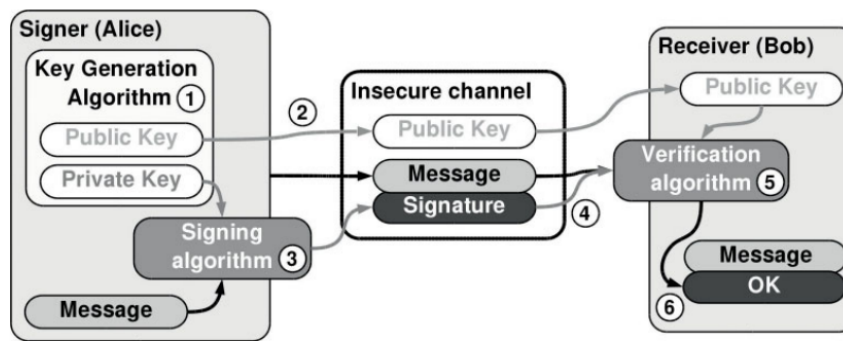


Figure 18: Digital Signature Process[31]

4.6 Hashing

It is possible to create a fixed-length "hash" for any given piece of information, regardless of its size or length, using a hashing algorithm. Hash functions and algorithms used for this purpose include SHA-1, SHA-256, MD5, and many others. While the other two methods of cryptography require keys, this form of encryption does not. Hashing is the key to preserving a true and lasting record of blockchain transactions and data, as any modification of the source information can result in changes to the block's hash value, which in this case will affect the hashes of successive blocks, rendering the blockchain invalid. Cryptographic hash functions encrypt any data and make it very difficult to decode, except by chance guesswork on the data until the outcome of the hash is matched. It's a fundamental notion to guarantee the reliability and immutability of the blockchain.[32]

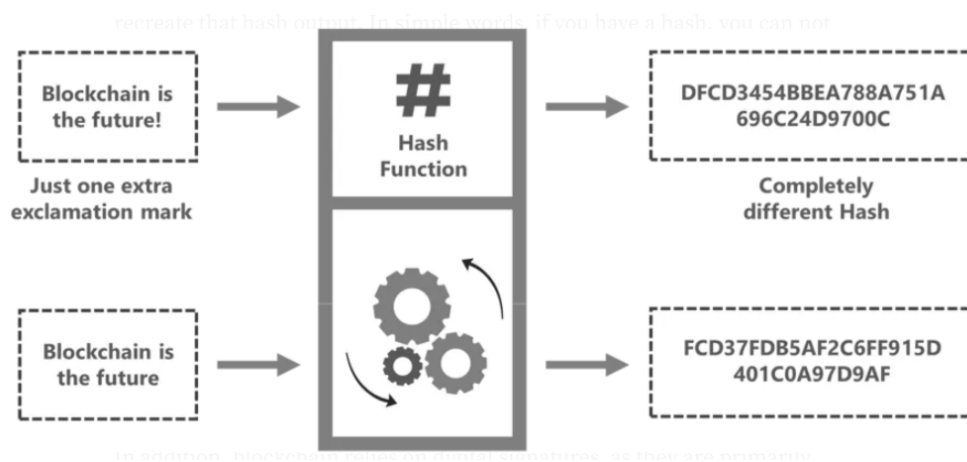


Figure 19: Hashing Process[32]

5 Permissionless versus Permissioned Blockchains

Permissionless, permissioned, or both may be used to categorize all different kinds of blockchains. Permissioned blockchains provide limited privileges and access to the nodes, with the users' identities being disclosed and it's more efficient due to the fewer number of nodes, resulting in less time spent on transactions. in the other hand Given the many participants and the massive amount of transactions, permissionless blockchains are more likely to be safer than permissioned blockchains. However, with numerous nodes and a huge amount of transactions, the time it requires to process

transactions is significant.[33]

Permissioned blockchains have both advantages and disadvantages. The concentration of permissioned blockchains with a single authority makes them less secure and more vulnerable to attack. Because the small number of nodes on a blockchain makes it easier for malicious parties to collaborate, administrators of private blockchains must ensure that the nodes that contribute and validate blocks are highly trustworthy.[33]

6 Blockchain types

When it comes to creating a blockchain-based system or solutions an important thing we have to consider is the type of blockchain used that we need in our application, hence we need a good comprehension of each type.

6.1 Public Blockchain

Public blockchains are open to all and are not centrally managed, giving all nodes similar access to the blockchain, development of new data blocks and approval of existing data blocks. Public blockchains, such as Bitcoin, Ethereum and Litecoin, are used for trading and mining cryptocurrencies. Nodes "mine" crypto-currency by solving complex calculation to create blocks for network transactions. The mining nodes receive a fee in return. Miners work similarly to modern bank tellers, who create a transaction and are paid for their work.[33]

6.2 Private Blockchain

Private blockchains are authorized blockchains that are managed by a single entity. The central authority decides who gets to be a particular blockchain node and does not always provide every node with equal access to functions. Ripple and Hyperledger are two examples of blockchains with limited public access. Because public access is limited, this blockchains are only somewhat decentralized.[33]

6.3 Hybrid Blockchain

A hybrid blockchain is one which brings together both public and private blockchain networks. With a hybrid blockchain network, users may control who has permission to see their data. Not all

blockchain data is made public. The person in charge of the data has the choice of making it public or keeping it entirely secret and available only to a small number of individuals.[34]

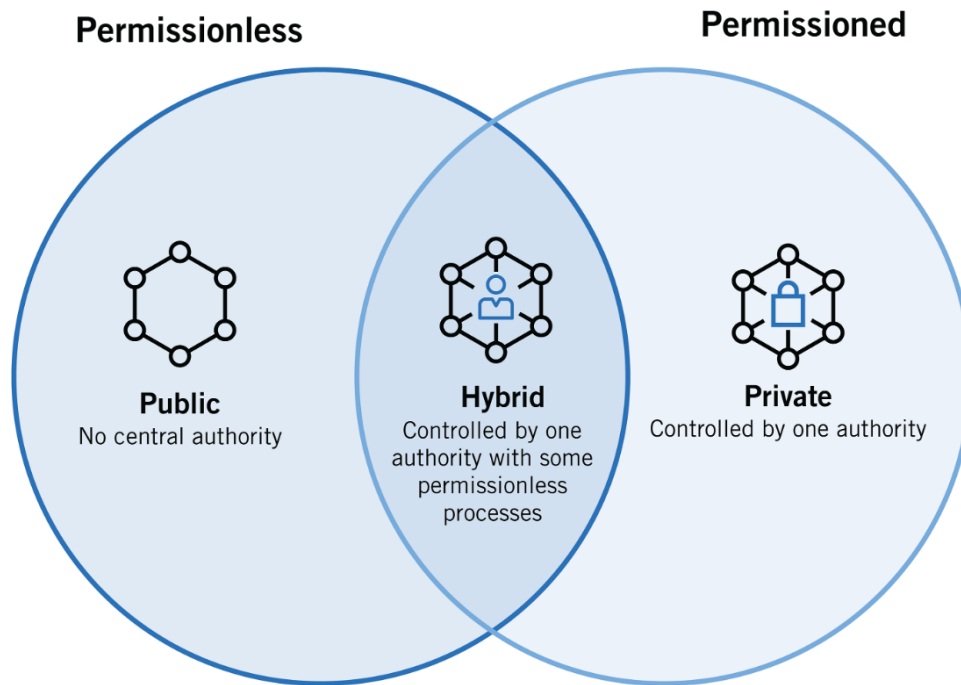


Figure 20: Blockchain Types[34]

7 Blockchain Actors

To make the blockchain in work it needs some operators or actors that contribute to creating the blockchain depending on the solution needed :

- **Blockchain architect** : A blockchain architect must be aware of peers, consensus, security, chaincode, integration, ledgers and applications, as well as business and technical analysis. He will design and establish how the group and its proposed blockchain based applications will employ consensus methods, norms, and best practices.[35]
- **Blockchain developer** : Developers working on blockchain networks focus on code and how to integrate blockchain code with other applications, data sources, events, and registries. They need to know the basics of blockchain, create and implement smart contracts, use APIs in the code to communicate with blockchain networks, and create, implement, and manage decentralized applications (dApps) and user interface[35]

- Blockchain user : End users can transact against smart contracts while remaining transparent to them through dApps web apps, which allow them to access blockchain-based smart contracts through an interactive interface or mobile app. [35]
- Blockchain regulator: They provide access limits and grant participants licenses (certificates). They make sure the right players have access to the appropriate areas of the blockchain network thanks to their practical understanding of the procedures.[35]
- Blockchain operator : They concentrate on the operational aspect of the blockchain business network, which includes peers, consensus, and security.[36]
- Traditional data sources : The blockchain can employ data from regular systems to drive the behavior of smart contracts.

8 Blockchain use cases

Besides the initial use of blockchain in crypto-currencies, there are plenty of areas of possible applicability for this technology due to all characteristics:

- Financial services and banking : An un-falsifiable tamper-proof record of sensitive activities can be created easily and securely using blockchain technology. This makes it ideal for the movement of funds and making payments abroad also enabling the adoption of automation for manual processing, thus reducing transaction costs. While blockchain secure behavior and features provides a good application for accounting and auditing, reducing human error and guaranteeing accuracy it can be used in Regulatory compliance and audit.[37]
- Healthcare : Une blockchain partagée peut être utilisée pour préserver les informations médicales, en les rendant accessibles à de nombreux utilisateurs sans soulever de problèmes de confidentialité. Ces informations contiennent des détails généraux tels que l'âge, le sexe et des précisions sur vos premiers problèmes de santé, monitoring health epidemics and diseases.[37]
- Government : The initial use case for blockchain in the public sector is presented here. Blockchain can be used to secure this data, control access, facilitate identity management,

and use it to store personal information about individuals, such as birth and death dates, marital status, and property transfers. Blockchain can also be used to implement voting, making it more accessible, improving security, streamlining organization, preventing tampering and maintaining anonymity throughout the process.[37]

- **Cybersecurity** : Blockchains are very secure because of their immutability, transparency, and decentralized structure. Blockchain storage doesn't have a one stop failure location. Thieves cannot access information because blockchains are decentralized and the data contained in each block cannot be altered. [37]
- **Supply chain**: Blockchain is well-suited for activities which include real-time tracking of items that travel and exchange throughout the process, due to its immutable record. Using a blockchain expands the possibilities for companies that deliver these goods. Events related to a supply chain can be queued up using records on a blockchain, such as the assignment of recently received items to various shipping containers. The blockchain offers a novel and powerful way to organize and use tracking data. [37]
- **Art** : A controversial subject, non-fungible tokens (NFTs) have been used to create works of art that have sold at auction for lot of money. NFTs have made the creation of crypto-art and digital collectibles possible, allowing creators such as musicians, painters and influencers to earn more money from their unique works. [38]

9 Smart Contracts

Contracts are used in many important areas as engagements to formally establish a business partnership, and are also relevant in the political field because of the ideas of "social contract" and their application or in professional and family relationships.[39]

Currently, with the advancement of modern technological innovations, there are new forms of contracts referred to as Smart Contracts, which are based on the use of cryptographic mechanisms and algorithms that offer a new approach to efficiently designing and executing contracts. Well, the smart contracts are running through the power of the blockchain.

9.1 Definition Of Smart Contracts

In the simplest form, smart contracts are blockchain-based algorithms which run when certain criteria are fulfilled. They are frequently employed to automate the implementation of an agreed-upon plan so that all parties can be sure of the conclusion immediately, without the need for third parties. Additionally, they can automate a process so that when certain criteria are fulfilled, then the following act is initiated. Smart contracts are created using a coding approach that can be easily shared across a network via a blockchain, they offer the potential to allow for the automation of a variety of tasks and across different fields.[40] Assuming two parties have reached an agreement on a contract fine for a violation of agreement, the penalty is taken off the violator's deposit.[41]

9.2 Smart contracts characteristics

- **Autonomy:** they're auto executable, they are run on the basis of specific conditions which are respected all without the slightest need for intervention by anybody.[42]
- **Transparency:** As the smart contracts are held on the so-called blockchain, which is itself initially a public ledger, smart contracts implementation are made visible to all involved actors.[42]
- **Immutability:** so that once put into operation, they can no longer be tweaked or corrupted. They can only be dropped if some sort of functionality was previously Implemented within the contract's code.[42]

- **Efficiency:** By being automated and lacking human involvement, smart contracts are quicker and better than ordinary ones. As a result, a change of ownership, for example, can be accomplished quickly in minutes instead of several days.[43]
- **Cost-effective:** By suppressing the necessity of intermediaries or some other third entity to audit and execute contracts smart contracts can save time and money. The cost of hiring a lawyer to draft and maintain a conventional contract, for example, can be eliminated when a smart contract is established to transfer asset ownership.[43]

9.3 Smart contracts Life cycle

To begin with, the smart contract is put into effect, according to the need of the agreement, whether it is a commercial or an administrative matter, all involved parties have to negotiate in order to come up with an agreement, which might be drafted by legal experts or advisers. Then, these natural language agreements are translated into meaningful and computer-based contracts by the developers. The conversion of smart contracts, as is the process of building computer applications, is an ongoing, multi-cycle, iterative process of development, revision, and modification. It typically requires conception, implementation, validation.[41]

Afterwards, The implementation of smart contracts is deployed in platform that is built upon the blockchain may then host the validated smart contracts. As a result of the blockchain's immutability, the resulting contracts that are stored there cannot be corrupted and can be accessed by all parties through blockchains.[41]

After deployment, smart contracts will be eventually executed if the terms of and conditions of the agreement are met. The logic, all of which is expressed as declarative statements, is immediately executed, and any such event which complies with the conditions of the contract, the associated action is executed and the transaction is recorded and then stored in the blockchain. Finally, now the smart contract is executed the new states are updated for all affected players. As a matter of fact, the operations made during the process of performing the smart contracts and the updated states are recorded in the blockchains.[41]

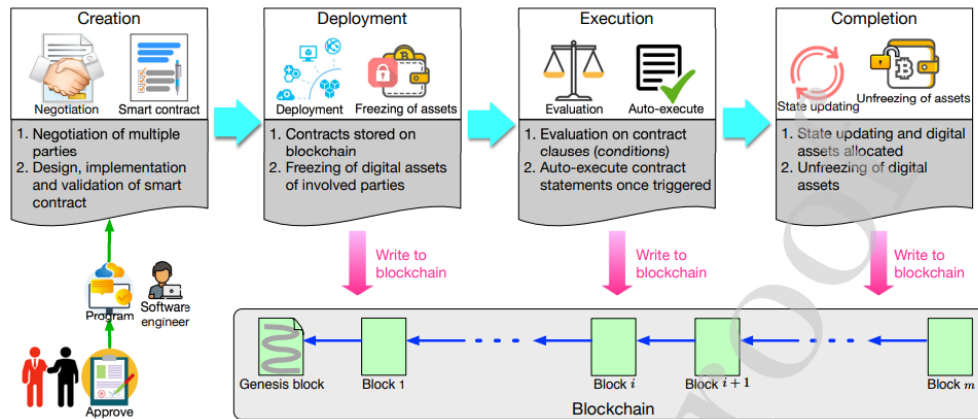


Figure 21: Smart contracts Life cycle Major phases[41]

9.4 Consensus Mechanisms and Smart Contracts

Previously, we referred to certain popular consensus mechanisms and the manner in which such mechanisms impact the way network members cooperate to validate transactions yet maintain network security and correctness, and thus this mechanism influences the behavior of the smart contracts in their execution. PoW is extremely computationally intensive and requires large amounts of energy, which can affect the execution of smart contracts and make them slow, PoS, which speeds up operations due to its mode of operation not requiring as much energy and making transactions faster than Pow, as well as smart contract execution, and so on. We find that the execution of the smart contract is affected by the consensus mechanism used.[44]

9.5 Smart Contracts use cases

Smart contracts provide a trusted way to carry out business with no intermediate parties. The Creation of Smart Contracts has the further capability to support and assist a wide range of industries. It can be used in the field of finance by globally automating transactions and helping to facilitate processing, minimize fees and speed up settlements, while ensuring transparency.[45]

- **Financial services:** Smart contracts came along as a transformative technology in the financial services industry, transforming various processes and bringing efficiency and transparency to traditional systems. Now, one notable application of smart contracts is in clearing and settlement of transactions. It's possible for smart contracts to effectively accomplish the

automation of approval workflows between parties, accurately calculate settlement amounts for transactions, and facilitate the transfer of funds directly from one party to another without the aid of middlemen. In the field of insurance, they enable automatic error checking, routing and approval workflows, making claims processing smooth and seamless. They also accurately calculate payments based on the type of claim and policy terms, reducing time and resources and improving customer experience.[45]

- **Life sciences and health care :** Smart contracts also are capable of transforming the life sciences and medical care sector by providing simpler procedures and better results for patients and health care. They can be employed in electronic medical records by relying on blockchain to ease the secure transfer and access of medical records. With multi-signature approvals between patients and healthcare services, patients can have greater control over their personal health data while ensuring its confidentiality and integrity. Also, they can enable access to personal health information stored in decentralized databases, giving researchers the capacity to analyze aggregated data while respecting the privacy of individuals. On top of that, micro-payments can be transferred to patients for their part in research efforts, which encourages patient involvement and data sharing. Smart contracts can improve personal health tracking by using IoT devices, such as wearables and sensors, to monitor vital signs, exercise and medication adherence. Smart contracts can generate rewards based on completed goals, encouraging healthy habits and sticking to treatment plans.[45]
- **Public Sector :** Smart contracts can create greater effectiveness and transparency in the public sectors by automating processes, such as record keeping for private companies. This allows for the ready access to accurate and up-to-date information about owners and their holdings, improving data integrity, eliminate middlemen, and increase transparency by providing an unfalsifiable and verifiable record of transactions.[45]
- **Cross-industries:** Smart contracts have many benefits in multi-sector applications, such as supply chain and trade finance business documentation. They enable transparent payment transfers and aid the chain-of-custody process by allowing custodial parties to record evidence of the product, ensuring its authenticity and mitigating the risk of counterfeiting or tampering. They contribute to the trust and integrity of the supply chain.[45]

- Voting : Smart contracts have the power to completely revolutionize the voting procedure through transparency, security, and efficiency. They automate the validation of voter criteria, record each vote on the blockchain, and can initiate to take particular actions based on the final results of the majority vote. By embracing smart contracts, the voting process itself is getting more trustworthy, accountable, and accessible to all participants.[45]

Smart contracts offer lot of advantages such, but they also have downsides like scalability, legal issues, and complexity. Scalability is a major challenge due to blockchain networks having limited flow and high latency. Legal issues are due to the fact that most legal systems do not recognize smart contracts and there is lack of a clear structure for resolving conflicts and performing obligations. These are limits that need further development and research to address them successfully.[46, 47]

CHAPTER 3

ELECTRONIC VOTING

1 Introduction

Voting is of key thing to a stable and democratic nation. It is the mechanism by which citizens can express their views and take part in political decision-making. Voting allows people to choose the leaders and representatives who will rule the country and make decisions that will impact everybody's life. It is an indispensable method of defending their rights and interests. Every vote matters and can play a role in the future of the country. It is thus essential to recognize the importance of voting and to be a part of the democratic and electoral process.[48]

2 The importance of secure and efficient voting systems

A free and fair society requires secure and effective voting methods, which are also a pillar of any democracy in the world, this kind of methods allow citizens to participate in the political process by voting and expressing their opinions. They also help to ensure that elections are fair and that the results are accepted by the public. This, in turn, helps to build trust in democratic institutions. At present, traditional paper-based voting systems are commonly adopted worldwide, but these methods suffer from some limitations. Paper-based voting methods have drawbacks that affect the efficiency and integrity of the electoral process. These include slowness, hand counting and verification, errors and fraudulent activities. In addition, they oblige voters to actually go to the assigned polling places, which can create an obstacle and prevent them from taking part in the vote. [49, 50]

One way to improve the security and efficiency of voting systems is to use electronic voting machines. These machines can be programmed to protect against voting tampering and to accurately count votes.[51]

3 Electronic voting systems

3.1 Definition of Electronic voting systems

Electronic voting can typically be defined as a type of computer-based election in of which the voters use a computer to submit their votes. Voters typically make their choice through a touch screen, although there may be the option of using voice input for voters with special visual difficulties. [52]

3.2 Characteristics of Electronic voting systems

Electronic voting systems in the general election system offer many potential improvements, such as faster and more accurate results, less chance of being miscounted or mistakenly counted. In addition, electronic voting systems improve accessibility, serving the physical needs of people with special needs by providing easier access to voting and allowing the creation of ballots in more diverse forms of language. Furthermore, they allow access to more information about voting choices, enabling voters to make informed decisions and to control ineligible voters, such as minors or those over the specified age limit.[53]

3.3 Types of electronic voting systems

E-voting systems differ in a number of different types, all having specific benefits and drawbacks. The most widely adopted electronic voting systems today are as follows:

- **Punch card voting systems** : this system utilize a punch card to mark the votes. Electors punch holes into the cards with a punching device that is provided next to the candidate or their chosen ballot, after which the punched cards get processed by a machine that calculates the number of votes cast. Punch-card voting systems are relatively inexpensive and easy to use. Two different types of punched-card voting system have been developed: Votomatic and Datavote. Whereas Datavote cards show the nominated candidate's name or a short text explaining an option displayed on the ballot next to the punched hole location, Votomatic cards give the holes numbers to signify votes, this type of voting gradually disappeared.[54]
- **Optical scan voting** : The use of a combined system of paper ballots and computerized voting

machines allows voters the option of voting on a paper ballot, whereas the process of counting can be carried out rapidly and centrally. Votes can also be manually counted if there are any malfunctions in the electronic scanning equipment. This systems deliver greater accuracy compared with punch card voting systems, but can be more expensive and more difficult to operate.[54]

- **Direct-recording electronic (DRE) :** DRE devices come into operation at voting centers as either advanced voting equipment or for election-day voting. The voting ballot is visualized via a display screen, where the voter simply selects the option or candidate they prefer by pushing a button. After the election, the voting records are compiled by the system which can be printed out or stored in removable memory. The results can then be collected in one place. Until they were found unacceptable by the German Constitutional Court in 2006, DRE voting machines were largely adopted in the Netherlands, the USA, Europe and other.[54]
- **Internet voting :** Internet voting involves posting or distributing a ballot paper online. It can be used in a variety of ways, both uncontrolled (remote Internet voting) and controlled (Internet voting at polling stations, kiosk voting), and is the most practical type of electronic voting system. The main advantages of remote Internet voting are for voters, but it also raises security issues, including doubts about the potential of the network to transmit sensitive data, concerns about hacker attacks and undue influence on the electorate.[54]

Besides the above mentioned types of electronic voting systems, a certain number of other types are currently also in development and being tested.

3.4 Considerations for electronic voting

- **Auditing :** An e-voting audit consists of a detailed inspection of the procedures applied to the vote collection, counting as well as the recounting of votes, with the aim of ensuring the correctness and accuracy of the voting results. In fact, this is an indispensable factor in establishing the trust and verifiability of electronic voting systems.[54]It is essential to ensure that electronic voting machines are checked prior as well as during and after the election.[55]The audit takes place through independent verification methods, typically a paper register or voter-verified paper, and also through the random selection of a portion of the

votes. [56]

- **Voter Authentication :** This step within the e-voting process consists of checking to make sure everyone who casts an e-vote is indeed an eligible person, and not someone different, Different authentication techniques are possible for voters like Electronic ID cards Simply place the card in the slot and type in the PIN to be authenticated, using known Username and password to login in the platform and vote or using personal information like date of birth, social security number, voter ID's with the password and username.[57]
- **Voting confidentiality :** Voting confidentiality in electronic voting refers the principle that the voter's identity and vote cast cannot be linked by anyone, including the voter himself, this plays a role in Integrity and legitimacy aspects of electronic voting systems,[58] there's some methods used to achieve the secrecy like encrypting the votes where votes are encrypted into a code which is only decrypted by a trusted party or authority. The encryption keys are kept secure and separate from the voter's identification details.[59]
- **Establishing Trust :** Unlike traditional voting methods, electronic voting systems can be more tricky to monitor and understand. For example, it is easier to monitor the counting of regular papers than it is to monitor the counting of electronic votes. This complexity creates concerns about the reliability of e-voting systems, and calls for measures to reinforce and guarantee trust. Among the measures introduced is the use of a Voter Verified Paper Audit Trail (VVPAT) in Direct Recording Electronic Voting (DRE) machines. A VVPAT generates a paper ballot for each vote cast electronically. It acts as an independent audit tool, allowing electors to check that their vote was registered in a correct manner, to detect potential fraud or malfunctions, and to serve as an audit tool for electronic results. General requirements that help build trust in different e-voting systems include maximizing transparency, verifiability, public debate and a test phase. Improving transparency by providing comprehensive information on how the system works can boost confidence. Verification mechanisms, such as VVPAT and end-to-end verification, enable voters to confirm the accuracy of their vote. Sufficient public debate prior to the implementation of e-voting systems helps to address concerns and improve understanding, which in turn strengthens confidence in the technology. Carrying out a test phase provides valuable information on system performance, poten-

tial problems and vulnerabilities, helping to address concerns and build confidence before full-scale implementation.[58]

3.5 Advantages and disadvantages of electronic voting systems

Electronic voting systems They offer a number of advantages as well as disadvantages compared to conventional paper-based voting. Some of the advantages are as follows :

- The ability to place your vote from anywhere via an Internet connection, Voters can exercise their voting rights by using an online platform to submit their votes remotely, using their Internet-connected devices.[60]
- Time and money saved due to avoidance of traditional voting equipment expenses. [60]
- More flexible participation over a longer period the implementation of measures enabling voters to cast their vote at different times over a longer period. it offers flexibility to people, giving them a chance to take part in the voting the way they want.[61]
- Better accessibility for people with reduced mobility or the aged people.[61]
- Higher participation rates based on voter motivation and profile, this helps to create a more democratic system where many voices and viewpoints are represented in the results of elections. [61]

And here are some of the disadvantages :

- The possibility of hacking and electoral fraud, Election hacking is clearly the main downside of e-voting systems. Like any electronic device, there is always a possibility that somebody might want to change the election results in some illegal way, whether by physical manipulation or by remote attack over the Internet. Allowing citizens to vote using their own devices could also present significant risks, as a malicious actor could alter millions of electronic votes undetected, whereas it would be impossible to go undetected if the same number of paper ballots were altered.[62]
- The influence of the electronic voting system manufacturer, Bias on the behalf of the company that creates and distributes electronic voting systems can affect voting. As e-voting

systems source code is usually kept secret, it is unclear whether the company has built in any biases. This low transparency gives reason for concern about the accuracy and fairness of electronic voting systems.[62]

- The initial costs of digital voting are high, Despite the possibility of making long-term savings, the initial investment in digital voting is quite significant. Indeed, it is necessary to invest in voting machines, cover maintenance and installation costs, test the infrastructure and ensure the security of the installations. These initial costs far exceed those associated with traditional paper-based voting. [62]

3.6 Challenges and concerns associated with electronic voting

From what discussed above we found that there's some challenges and concerns associated with electronic voting that need to be considered and can be crucial in the process:

- Security: it is necessary to ensure that the voting system is protected against cyberattacks, frauds, errors or failures, and that it respects the principles of sincerity, uniqueness and independence of the vote.[62]
- Transparency: it is necessary to make sure that the voting process is understandable, verifiable and auditable by voters, candidates and the relevant authorities, and that any potential problems can be identified and rectified.[62]
- Privacy: the secrecy of the vote and the protection of voters' personal data, avoiding any possibility of traceability or exposure of the choices made.[62]
- Trust: it is necessary to encourage voter support and participation, giving voters reasonable guarantees of the reliability, neutrality and credibility of the voting system, and taking into account their expectations.[62]

4 Electronic Voting with blockchain

Blockchain has the potential to transform the way we vote. By storing votes on a blockchain, it would be possible to create a secure and transparent voting system, safe from fraud and manipulation. Electronic voting with blockchain is a solution that uses blockchain technology as a transparent, immutable register to record and verify the votes cast by voters in an electronic voting

system Where every single transaction corresponds to one vote cast by a voter for a specific candidate. Electronic voting with blockchain is about providing a decentralized, secure, transparent and auditable process for voting that can help bypass some of the difficulties and issues associated with conventional electronic voting systems.[63]

4.1 Motivation for using blockchain in electronic voting systems

What motivates blockchain's use in e-voting systems is to make a solution available that can meet some of the challenges and concerns associated with traditional e-voting systems, such as security, transparency, confidentiality and trust. [64]

Blockchain technology presents many benefits suited to an electronic voting system, including:

- **High availability:** the system does not rely on a centralized server and can operate even if some nodes fail or are attacked.[65]
- **Immutability:** the transactions recorded on the blockchain cannot be altered or deleted, ensuring the integrity and verifiability of the votes.[65]
- **Decentralization:** the system does not depend on a single authority or intermediary to control or manage the voting process, reducing the risk of fraud or manipulation .[65]
- **Privacy:** the system can use cryptographic techniques to protect the identity and the choices of the voters, ensuring the secrecy and anonymity of the vote.[65]

4.2 Benefits of integrating blockchain technology into voting processes

The overall value of bringing blockchain technology to the voting process is in creating a voting solution that increases efficiency, security, transparency and accessibility. [66]

So this technology can enable the following benefits for voting processes:

- **Efficiency :** By using a digital token that can be transmitted directly to the chosen candidate's wallet, voters have an easy way of exercising their right to vote via mobile devices or computers. Blockchain can therefore lower both the financial and time costs associated with the stages of the traditional voting process, as well as eliminating the need for physical polling stations and intermediaries.[67]

- **Security** : Blockchain technology provides a powerful means of combating electoral fraud and vote tampering by securely recording every vote as an encrypted transaction in a decentralized ledger. This immutable, traceable and verifiable ledger ensures that any attempt to manipulate or falsify recorded votes is easily detected. As a result, blockchain can also guarantee the integrity and auditability of votes, so that voters, candidates and authorities are able to verify the overall fairness of the process and results. [68]
- **Transparency** : Blockchain strengthens the trust and credibility of the voting process by guaranteeing openness, fairness and independent verifiability. It enables voters to verify their votes and candidates to track the progress and results of elections. Blockchain facilitates public inspection and surveillance of the voting process by providing universal accessibility to the ledger, allowing any individual to review and judge its information. [69]
- **Accessibility** : By offering secure, remote voting alternatives, the accessibility of the voting process can be improved, leading to potential increases in voter participation. Eliminating geographical constraints, blockchain technology makes it possible to vote securely from any location, which can particularly benefit people with disabilities, remote workers and citizens living abroad.[68]

The above benefits of integrating blockchain technology into voting systems have the potential to bring about a meaningful change in the way voting is performed.

4.3 Blockchain Addressing the Challenges of Electronic Voting

Blockchain technology can provide solutions to the issues faced in electronic voting systems. Blockchain has the potential to address these challenges :

- **Security** : Blockchain is a decentralized, distributed ledger that stores data in a series of interconnected blocks. Each block contains a set of transactions, and once a block is added to the chain, it cannot be altered. [70]This immutability makes it difficult for hackers to tamper with the voting data. Additionally, blockchain uses cryptographic techniques to ensure the anonymity of voters and protect their privacy.[71]
- **Transparency**: In a blockchain-based e-voting system, all transactions (votes) are recorded on the public ledger, visible to all. As a result, public auditability is achieved, meaning that

anyone can monitor the entire voting process and the election results. Such transparency contributes both to preventing fraud and to ensuring the electoral process's integrity. [72]

- Trust : The decentralized nature of blockchain removes the necessity for a central entity to oversee the voting procedure. Instead, the system operates through a network of interconnected nodes that collaborate to authenticate and document transactions.[73] so it increases the trust in the system.
- Auditing and recounting: However, conventional electronic voting systems do not always allow votes to be verified or recounted. The transparent and immutable nature of blockchain makes verification and recounting of votes easy, guaranteeing the integrity of the electoral process.[74]
- Cost reduction: Blockchain-based voting systems can reduce the costs associated with traditional voting methods, from paper voting ballots to centralized electronic systems. By automating various aspects of the voting process through smart contracts and thereby minimizing the need for dedicated physical infrastructure, blockchain can help reduce the overall cost of running elections. [75]

CHAPTER 4

DESIGN AND IMPLEMENTATION OF VOTING SYSTEM

1 Introduction

Based on a smart contract, this system is designed to facilitate the voting process. It enables registered voters to vote for candidates, keeps track of the vote count for each candidate and determines the winning candidate according to the highest number of votes. The system which is blockchain-based features its own functionalities to provide a transparent and decentralized way to organize elections, using an ERC20 token as the voting mechanism.

2 Ethereum Blockchain

2.1 Definition

From the official Ethereum definition: “Ethereum is a blockchain with a computer embedded in it. It is the foundation for building apps and organizations in a decentralized, permissionless, censorship-resistant way.” Ethereum has two important integral components, the first one is the crypto-currency “Ether” which serves as the basis for Ethereum transactions on the Ethereum blockchain, the Ethers are stored in accounts of the users using electronic wallets and the second one is the “gas” fee is a unit that represents the computational effort required for Ethereum transactions. All Ethereum operations are charged a “gas” fee, no matter what the result. [76, 77]

2.2 Ethereum Virtual Machine

The Ethereum Virtual Machine (EVM) serves as a shared computer on which all network participants agree. Each Ethereum node keeps a copy of the EVM’s state. Participants can request the EVM to perform any calculation, with the other participants in the network verifying and executing the computation. This results in a state shift in the EVM, and this is then spread across the entire network.[77]

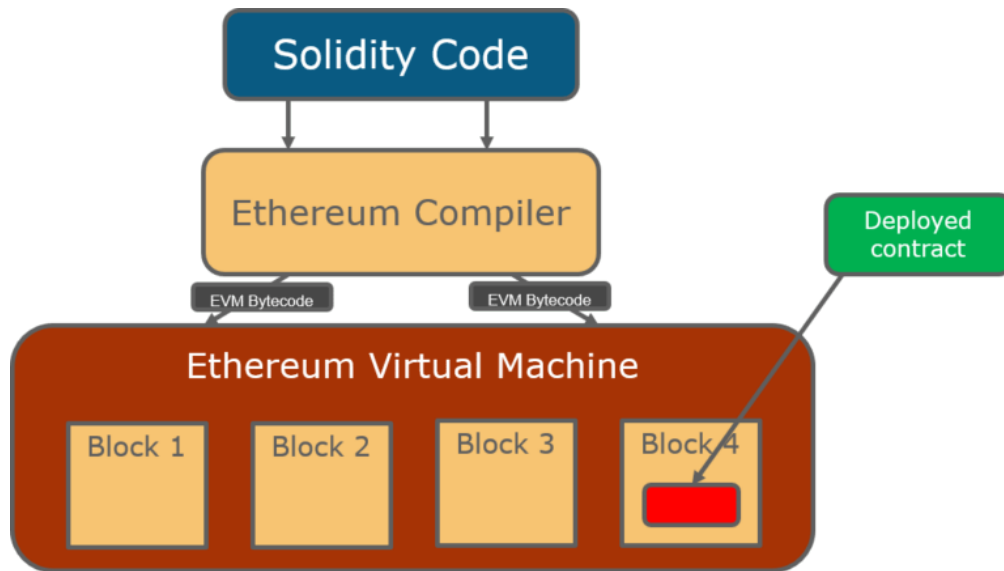


Figure 22: Ethereum Virtual Machine[78]

2.3 Ethereum Transaction

A communication called an Ethereum transaction is delivered between accounts and contains instructions for specific operations like transferring money or calling a smart contract function. It is then broadcast to the Ethereum network, signed with the account owner's private key, verified, and added to a blockchain block. Once verified, a transaction is irreversible and unchangeable, barring changes or reversals.[79]

The address of the intended recipient, the value of the transaction, optional data or instructions, and a signature (v, r, s) derived from the sender's private key make up the structure of an Ethereum transaction. This structure also includes a unique value called "Nonce" that safeguards against replay attacks, a gas price showing the sender's intention to pay for every bit of gas used, a gas limit determining the maximum amount of gas available, and other information. These elements make guarantee that transactions on the Ethereum network are handled appropriately and only once.[79]

2.4 Transaction: Contract Creation

An additional smart contract is introduced to the network via a contract creation transaction on the Ethereum blockchain. It is sent to the network and executed by nodes there. It includes

important information, such as the bytecode of the smart contract. Parameters like the Gas Limit, Gas Price, No., Address, and Value control the transaction. Participating nodes run the bytecode, launching a fresh instance of the contract on the Ethereum network. Users can communicate with the contract, perform functions, or send Ether by using the special contract address that is produced when it has been published and verified by consensus.[79]

2.5 Smart contracts in the Ethereum blockchain

”A Smart contract is simply a program that runs on the Ethereum blockchain. It’s a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain.”

Smart contracts in Ethereum correspond to autonomous accounts with a balance amount, with which they can be targeted by transactions. Unlike user-controlled accounts, smart contracts are deployed and executed programmatically, applying predefined rules. Interactions with smart contracts are irreversible, and they can never be removed. [80]

2.6 Solidity language

“Solidity is a high-level object-oriented programming language specifically designed for building smart contracts on the Ethereum Virtual Machine (EVM). It is influenced by other languages like C++, Python and JavaScript, offering features such as static typing, inheritance, libraries and complex data types. With Solidity, it is possible to create contracts for various applications”.[81]

2.7 Ethereum Tokens

Tokens can be used for a lot more than just digital currency. Tokens are capable of serving multiple functions simultaneously. A token, can stand for voting rights, access rights and ownership of a certain resource. These multi-purpose tokens go beyond traditional money, offering a variety of functionalities in different contexts[82], there’s different two types of tokens Non-fungible tokens and Fungible tokens they can be used based on what they represent. In addition it exist several popular token standards: ERC-20, ERC-721, ERC-777, ERC-1155 and ERC-4626, and ERC20 is the most popular token standard applicable with fungible assets.[83]

2.8 Wallets

A wallet is a tool for managing personal digital identities, represented by digital keys, necessary for blockchain operations like using dapps, transacting cryptocurrencies and exchanging Tokens. When a crypto-currency wallet is created, a unique and confidential "seed phrase" is generated, serving as the password. This base phrase is used to derive public addresses and private keys for each account. Public addresses identify you on the blockchain, while private keys provide access to your account.[84]

Smart contracts may need to give or receive cryptocurrency or tokens from a different party. This is the point at which your system needs to integrate cryptocurrency wallets, there's a large of Wallets we can chose from like "Argent", "Portis", "Coinbase Wallet", but the most famous one which used for smart contract is "Metamask" Wallet.[85]

3 ERC20 Token contract

3.1 ERC20 Token Standard (Ethereum Request for Comments 20)

ERC-20 is a standard for fungible tokens, meaning that each token is absolutely identical in terms of type and value to all other tokens, this standard implements an API for tokens within Smart Contracts.

A smart contract is considered an ERC-20 Token Contract if a set of functions are implemented[86], there's required functions:

```
function totalSupply() public view returns (uint256)
function balanceOf(address _owner) public view returns (uint256 balance)
function transfer(address _to, uint256 _value) public returns (bool success)
function transferFrom(address _from, address _to, uint256 _value) public
    returns (bool success)
function approve(address _spender, uint256 _value) public returns (bool
    success)
function allowance(address _owner, address _spender) public view returns (
    uint256 remaining)
```

And optional functions[86], that can be not implemented but it can be used to improve usability:

```
function name() public view returns (string)
function symbol() public view returns (string)
function decimals() public view returns (uint8)
```

ERC-20 Token is the most popular token standard and can be used for voting purposes[87], and it's used as the foundation for the vast majority of tokens. [82]

3.2 Purpose of ERC-20 Token

The purpose of this ERC20 token contract is to define and implement a standard interface for fungible tokens on the Ethereum blockchain, it represents a special token that will be used as voting token in the election smart contract which will interact with this ERC20 token contract to perform token transfers and balance inquiries following the ERC20 standard rules.

3.3 Design of ERC-20 Token

The ERC20Token contract can be seen as like a class that defines the structure, properties and actions of an ERC20 token. It includes the token name, symbol, total supply, balance information and functions to transfer tokens, approve, allowances, and retrieve allowances.

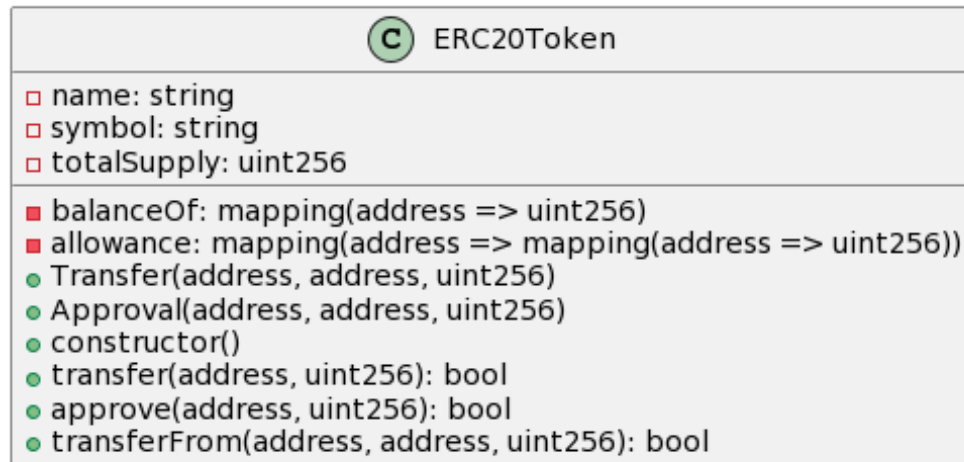


Figure 23: ERC20 Class diagram

In the Use case diagram shown below, the main actor is the "user" who interacts with the ERC20Token contract. Users can initiate a token transfer, approve a spend allowance for a specific address, and transfer from the approved allowance.

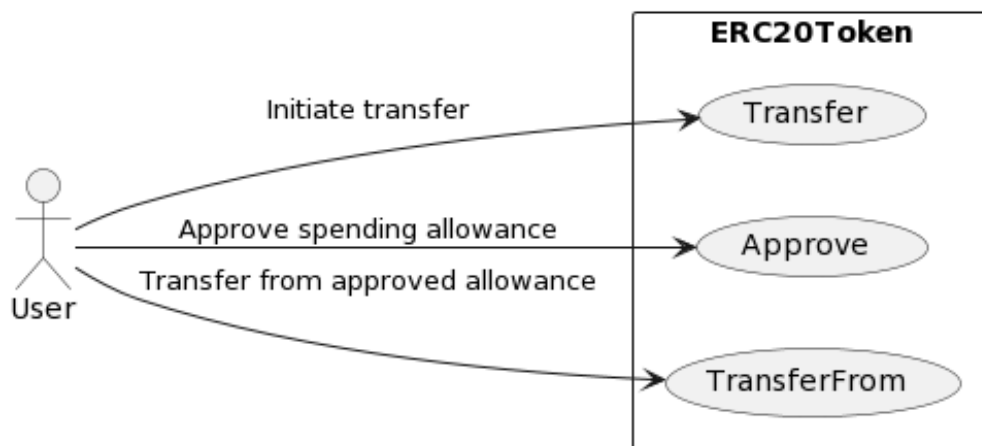


Figure 24: ERC20 Use case diagram

3.4 ERC-20 Token Information

While the name and the symbol are optional and we have the option to define them or not but we defined them for better usability and understanding:

- name: Represents the name of the ERC20 token ("evotingdz").
- symbol: Represents the symbol or ticker of the token ("EVTDZ"). And the other one optional decimals we are not going to define it because we do not intend to send fractions of the token but only 1 token.
- totalSupply : Represents the total supply of our token which signify in the context of our application the maximum number of voters that will participate in the vote this number will be defined by the authorities responsible of counting the eligible voters number, Here just for demonstration purposes we have initialized it to 1000 tokens which means 1000 voter.

3.5 ERC-20 Token Mappings and Events

The smart contract has Mappings and events used a follow:

- a) ERC20Token Mapping Types : in solidity every mapping has a key and value associated with it We can think of mappings as hash tables.[88]

balanceOf Mapping:

- This mapping allows us keep track of the token balance of every single address in the contract
- The key of this mapping is an address, representing an account.
- The value associated with each address is a uint256, representing the token balance of that address.

balanceOf Mapping represented in table format :

Key (address)	Value (uint256)
address 1	balance 1
address 2	balance 2
...	...

allowance Mapping:

- This mapping keeps track of the approved spending allowance for a spender by an owner.
- The key of this mapping is another mapping, where the outer key is the owner's address, and the inner key is the spender's address.
- The value associated with each pair of owner and spender addresses is a uint256, representing the approved spending allowance given to spender from the token owner.

allowance Mapping represented in table format :

Key (address)	Value (uint256)	Additional Column
Key (Owner address)	Key (Spender address)	Value (uint256)
owner address 1	spender address 1	allowance 1
owner address 2	spender address 2	allowance 2
...	...	...

- b) ERC20Token Events : Events is solidity Events allow logging to the Ethereum blockchain.
[89]

Transfer event:

- This event is released whenever tokens are transferred from one address to another it helps us to monitor the tokens transfers.
- It provides information about the sender (from), recipient (to), and the number of tokens transferred (value).

Approval event:

- This event is released whenever the spending allowance is approved by the owner for a specific spender helping us to observe the approval to spend tokens allowances.
- It displays information on the owner (owner), the spender (spender) and the approved spending allowance (value).

3.6 ERC-20 Token Initialization

Initialization is done with a constructor is a function that is executed upon contract creation when deployed in the blockchain initializing the state variables of the contract.[90]

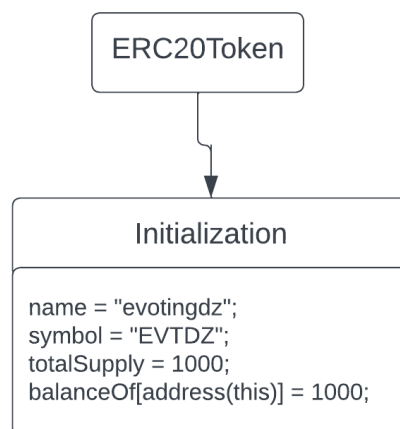


Figure 25: ERC20 Initialization process that occurs within the constructor

In our contract we consider the state variables as follow:

- name: The name of the token is set to "evotingdz".
- symbol: The symbol of the token is set to "EVTDZ".
- totalSupply: The total supply of tokens is set to 1000.
- The constructor also initializes the token balance for the contract itself, the contract address (address(this)) is assigned a token balance of 1000 tokens

3.7 Token Transfer

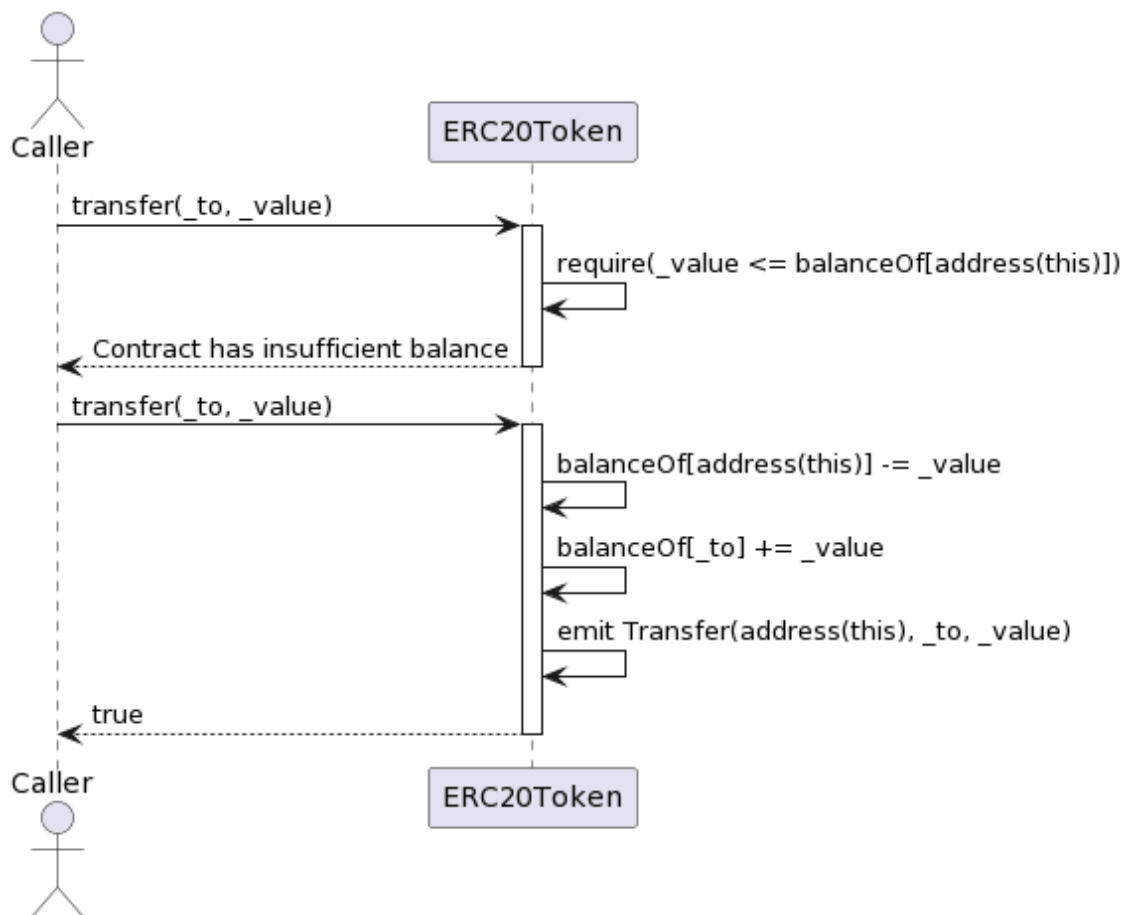


Figure 26: Sequence diagram of Token transfer

This sequence diagram illustrates how the caller and the ERC20Token contract interact when the transfer function is executed.

- The Caller initiates the transfer function by providing the `_to` address and `_value` amount as arguments and send the transfer message to ERC20Token contract.
- The ERC20Token contract receives the transfer message.
- First of all, the contract checks the balance constraint using the "require" instruction.
- If the condition is satisfied, the contract reduces the contract balance by the `_value` specified and increases the balance of `_to` accordingly.
- Then the contract emits a Transfer event to indicate the token transfer from the contract's address to the specified `_to` address with the transferred `_value`.
- Finally, the ERC20Token contract returns true to the Caller, indicating a successful transfer.

3.8 Approvals and Allowances

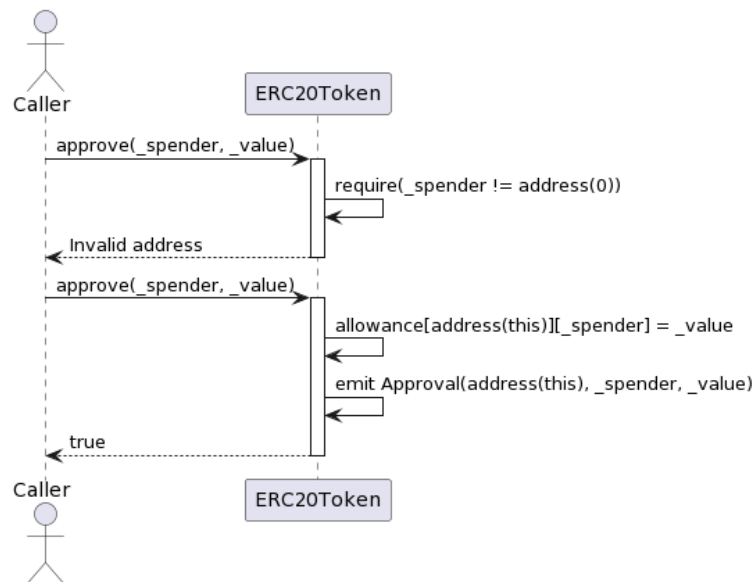


Figure 27: Sequence diagram of Approval

This sequence diagram illustrates how the caller and the ERC20Token contract interact when the approve function is executed.

- The TokenOwner initiates the approve function by providing the `_spender` address and `_value` amount as arguments and send the transfer message to ERC20Token contract.
- The ERC20Token contract receives the approve message.
- First of all, the contract uses the require instruction to check that `_spender`'s address is a valid address and not null address.
- If the address is valid, the "TokenContract" updates the allowance of the `msg.sender` (TokenOwner) to itself (`msg.sender`) with the specified `_value`.
- The contract emits an Approval event to indicate the approval of spending allowance from TokenOwner to spend tokens on behalf of itself.
- Finally, the ERC20Token contract returns true to the TokenOwner, indicating a successful approval.

3.9 Token TransferFrom

This sequence diagram below illustrates how the caller and the ERC20Token contract interact when the approve function is executed.

- The Caller initiates the transferFrom function by providing the `_from` address , `_to` address and `_value` amount as arguments and send the transfer message to ERC20Token contract.
- The ERC20Token contract receives the approve message.
- First of all, the contract use the require instruction to check if both that sender and receiver addresses are a valid addresses and not null address.
- Also it checks if the `_value` amount is less than or equal to the balance of `_from` and the allowance from `address(this)` to `_to`.
- Once all the conditions are satisfied, the contract takes the specified `_value` from the balance of `_from` and adds it to the balance of `address(this)`, to the contract itself, in the same time it removes the allowance.
- The contract emits an Transfer event indicating the token transfer.
- Finally, the ERC20Token contract returns true to the Caller, indicating a successful transfer.



Figure 28: Sequence diagram of TransferFrom

3.10 Conclusion

We have implemented a Token smart contract using ERC20 Standard and essential functionalities that are required for secure and transparent management of tokens and balance tracking which will facilitate token transfers in our election system where tokens are used as voting rights where each voter must have to be able to cast his vote, we will need the address of this contract to be able to call its functions via an interface in our second smart contract that handle the voting process.

4 Voting Smart Contract

Now we are going to see the Voting smart contract that will be used to implement a voting system where registered voters can participate in ballot and cast their votes for their chosen candidates, the focus in this contract to make the voting process easier including voters' registration, voting, vote counting and results processing Where voting rights are represented by an ERC20 token, along with specific permission conditions to guarantee the beginning of voting. The contract also contains the prerequisites and validations necessary to guarantee the security and validity of the voting process.

4.1 Electoral systems types

Before heading to the election smart contract, it's important to take a look about most popular electoral systems types, Majority, proportional and mixed electoral systems form the three principal types of electoral systems.

- **Majoritarian systems:** The winners represent the parties or candidates who get the highest number of electoral votes, while plurality systems simply require more votes than anyone else, majority systems require the winning candidate or party to obtain a complete majority of the votes cast. Most majoritarian systems, including first-past-the-post (FPTP), can be characterized by the number of votes per voter and seats per constituency.[91]
- **Proportional systems:** Proportional representation (PR) is a system in which the number of seats obtained by a party is based on the number of votes that party won. Two main forms exist: closed-list PR and open-list PR. In closed-list PR, voters vote for a list of candidates according to party, while in open-list PR, voters vote for a specific candidate according to candidate.[91]
- **Mixed systems:** Mixed systems use a combination of majority and proportional elements, with two main types: the parallel system and the mixed proportional system (MPS). In parallel systems, voters cast their vote twice for a candidate in a majority voting district and once for a party in a proportional voting district, while in mixed systems, voters cast their vote once for a candidate in a majority voting district and once for a party in a proportional voting district. Parties' seats are determined by the number of seats won in both majoritarian and proportional districts.[91]

In the system we designed the electoral system type is based on is the Majoritarian system and exactly the Plurality system where the candidate with more votes wins.

4.2 Key Components of Voting Smart Contract

a) Candidate structure :

In solidity “Structs are custom defined types that can group several variables”, in our contract we used the struct to define and represent a candidate that will participate in the election with An ID, voteCount and other personal information as the struct fields.[92]

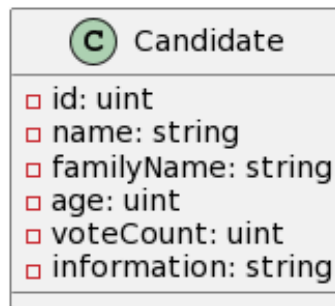


Figure 29: Candidate struct

b) Voting events :

votedEvent event:

- This event is released whenever tokens are transferred from one address to another address it helps us to monitor the tokens transfers.
- It displays information about which candidate the voter has given his vote to.

Approval event:

- This event is released for every voter has been registered by the admin to be eligible for voting.
- It displays to everyone the information that of a new voter is registered in this election.

These events are created in order to inform other systems or user interfaces of certain activities taking place, events are broadcast inside the contract functions. Applications on the front end may keep an eye out for these events and adjust the user experience appropriately.

c) Voting Mappings :

These mappings are used in the data changes of the election process, like as tracking the voting status, registration status, and token usage status of individual voters. They facilitate extraction and updating of voter information during the electoral process.

registeredVoters Mapping :

- This mapping allows us keep track of the addresses that have been registered as voters by the contract admin.
- Where the key of this mapping is an address, representing an voters Address and the value associated with it type bool, representing whether the voter is a registered voter.

registeredVoters Mapping represented in table format :

Key (address)	Value (bool)
address voter1	True/false
address voter2	True/false
...	...

Table 2: registeredVoters Mapping representation

voters Mapping :

- This mapping allows us keep register the addresses of every person who already voted to prevent them from voting more than they are allowed.
- Where the key of this mapping is an address, representing an voters Address and the value associated with it type bool, indicating whether the voter did cast his vote or not.

voters Mapping represented in table format :

Key (address)	Value (bool)
address voter1	True/false
address voter2	True/false
...	...

Table 3: Voters Mapping representation

4.3 Voting Smart Contract State variables

The smart contract has state variables work together to administrate and run the voting process, register voters, monitor voting and determine the winners based on the vote count.

State Variable	Type	Role
admin	address	This variable stores the address of the contract administrator who has privileged access to perform certain operations.
tokenAddress	address	stores the address of the ERC20 token contract
tokenAmount	uint	stores the amount of ERC20 tokens required for a voter to cast a vote.
votingStartTime	uint	This variable stores the timestamp (Unix time) when the voting period starts.
votingEndTime	uint	This variable stores the timestamp (Unix time) when the voting period ends.
candidatesCount	uint	Total number of candidates participating in the election.
candidates	Candidate[]	Dynamic array storing information of all candidates participating in the election.

Table 4: Voting Smart Contract State variable details

4.4 Modifier in the Voting contract

Function Modifiers in solidity language “They Can be used to change the behaviour of functions in a declarative way as inheritable properties of contracts”[93]

onlyAdmin Modifier:

We created this modifier to use it to restrict the registerVoter function execution just by the smart contract admin only hence the admin is the only one can register voters during the registration phase, and it can be used in another functions that we want to do the same for it.

```
modifier onlyAdmin() {
    require(msg.sender == admin, "Only admin can perform this operation");
    _;
}
```

The modifier verifies that the sender’s address, `msg.sender`, matches the admin address. If they do, the function’s code that employs the modifier will be executed. Or an exception is thrown.

4.5 Voting contract Initialization

The administrator's address, the ERC20 token contract address, the necessary token amount for voting, the start and finish times for voting, and the starting candidates for the election are all initialized using the specified values in the constructor. The election contract's initial setup is determined by these settings so once the contract is deployed it is ready to work.

The tokenAddress is passed like a parameter so the election contract will work with the ERC20 Token.

Instruction	Role
admin = msg.sender;	Assigns the address of the contract deployer as the administrator of the contract.
tokenAddress = _tokenAddress;	Sets the address of the ERC20 token contract to be used for voting.
tokenAmount = 1;	Defines the required number of ERC20 tokens for each vote.
votingStartTime = 1686372000;	Sets the start time of the voting phase as a timestamp.
votingEndTime = 1686372060;	Sets the end time of the voting phase as a timestamp.
candidates.push(Candidate());	Adds a candidate to the candidates array with the specified details.
candidatesCount = candidates.length;	Sets the candidatesCount variable equal to the length of the candidates array, representing our candidate's total number.

Table 5: Voting Smart Contract Instruction details

4.6 ERC20 Token Integration

Like we seen above in the ERC20 Token smart contract, here we will use an interface called IERC20 to use the functions and events of the ERC20 Token in the voting process.

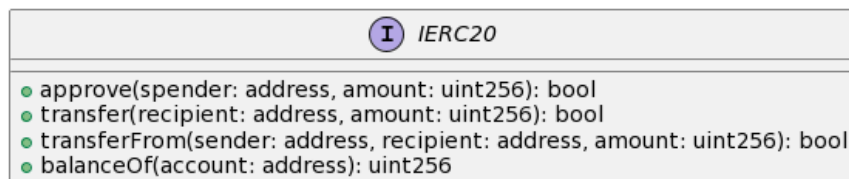
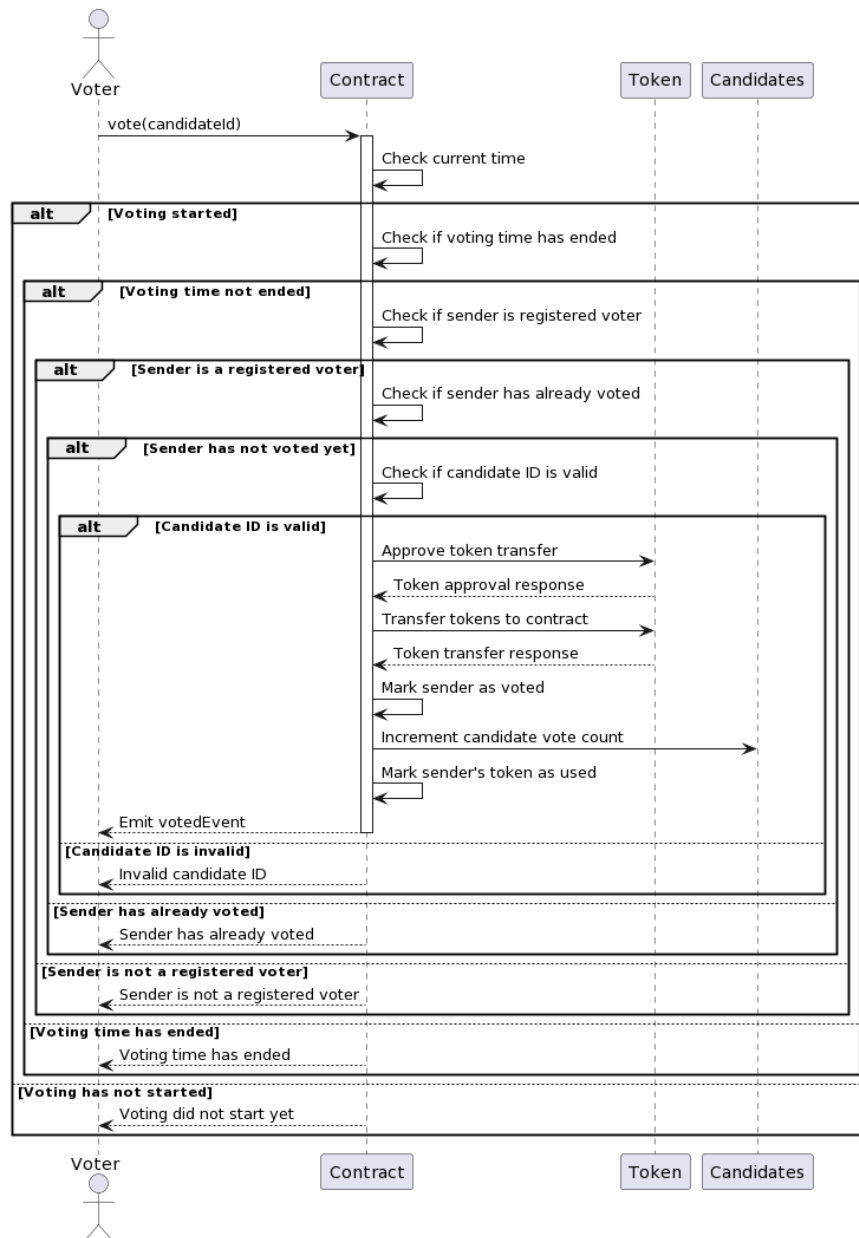


Figure 30: IERC20 Interface

- **Approve function:** This function allows the election contract to approve a certain amount of tokens to be spent by another address (spender), in the election process this function is called when the voter is going to vote so the voter gives approval to the election contract spent his token after he commit his vote.
- **Transfer function:** This function responsible of transferring the tokens between to addresses, so when the voter cast his vote, the token will be taken from his account and transferred to the contract.
- **transferFrom function:** This function allows the election contract to transfer a certain number of tokens (1 Token) from the voter's address to another the provided contract's address on behalf of the voter.
- **balanceOf function:** This function is for checking the token balance of a registered voter. It assures the voter has the necessary number of tokens to take part in the ballot. Insufficient tokens will prevent the voter from voting.



The sequence begins with the voter calling the vote function. The contract checks that the conditions are met before authorizing the vote. It checks the time, whether voting has started and not ended, whether the voter is registered and has never voted before, and whether it validates the candidate's identity. If all conditions are met, the contract interacts with the token contract to approve and transfer the voter's tokens to the contract. Next, the contract updates the vote count

for the chosen candidate, marks the voter as having voted and releases the votedEvent. If any of the above requirements are not met, error messages are sent back to the voter.

4.9 Winner Determination

Starting from invoking the contract's `getWinner` function, which checks the current time and retrieves the results of the candidates' votes. It then determines the winning candidate by iterating over all the candidates and comparing their votes. If the vote is not completed, an error message is returned to the voter.

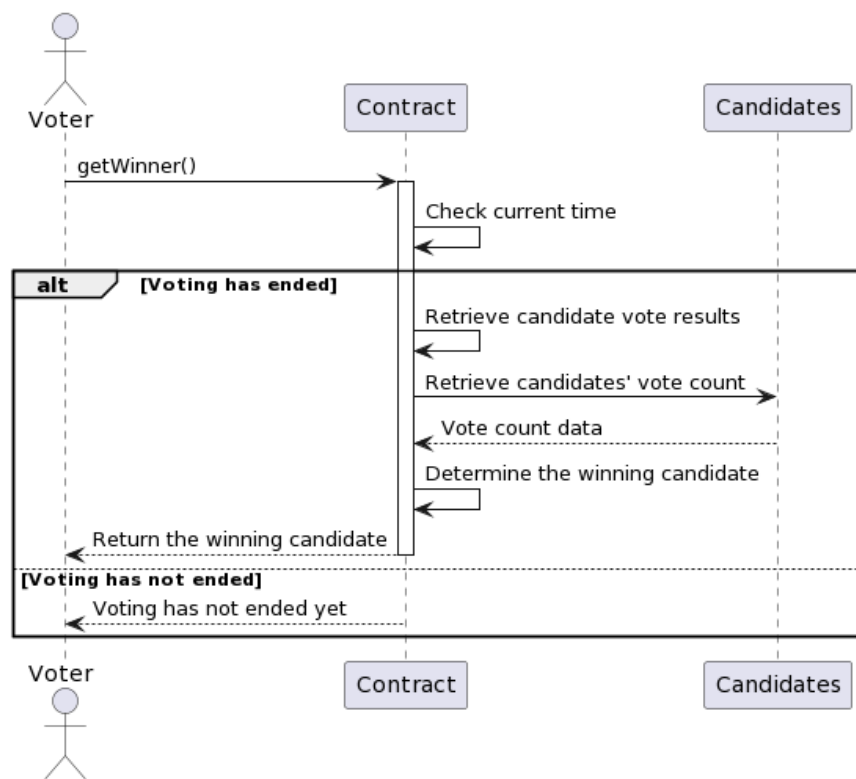


Figure 33: Sequence diagram of Winner result

4.10 Additional functions present in your smart contract

There are other functions implemented in the smart contract used to make data accessible and enable more transparency to the process.

- **getRegisteredCandidates ():** This function refers to an array of Candidate structures representing registered candidates it shows the list of candidates that voter can give his vote.
- **getRegisteredVotersCount():** This function returns the count of registered voters.
- **getCurrentPhase():** This function determines the current phase of the voting process. It compares the current timestamp with the votingStartTime and votingEndTime values to determine whether the registration phase, voting phase, or the voting has ended. It returns a string representing the current phase and the corresponding start and end timestamps.

4.11 Summary

We have seen the design of voting smart contract that integrate ERC20 Token Standard for handling the tokens transfer, followed with different phases and their functions and the security measures from the the registration phase passing by the voting phase and finally getting the result of the vote.

5 Development

5.1 Development tools

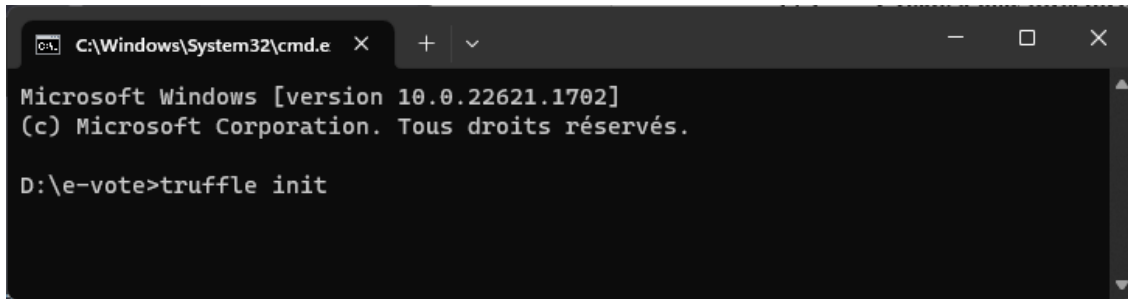
- i) **Remix - Ethereum IDE** : Remix IDE is a open source web and desktop application that promotes rapid development and offers a wide range of plugins featuring user-friendly graphical interfaces. It serves as a comprehensive platform for creating Solidity-based smart contracts, facilitating the entire development process, and also functions as a versatile environment for exploring and instructing Ethereum.[94]
- ii) **Code Visual Studio** : Visual Studio Code, a lightweight yet robust source code editor, operates on desktop platforms including Windows, macOS, and Linux. It offers native assistance for JavaScript, TypeScript, and Node.js, and boasts an extensive collection of extensions that support additional languages and runtimes such as C++, Java, Python, PHP, Go, and .NET.[95]
- iii) **Truffle** : “A world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier. Remix IDE offers a complete set of tools for smart contract development, including compilation, deployment, and binary management. It provides advanced debugging capabilities with breakpoints and variable analysis, along with console.log support. Deployments and transactions are secure with MetaMask and Truffle Dashboard integration.”[96]
- iv) **Ganach** : “Ganache is a convenient personal blockchain solution designed for fast development of Ethereum distributed applications (dApps). It provides a safe and deterministic environment for the entire development cycle, allowing developers to create, deploy, and test their dApps effectively. Ganache offers two versions: a user-friendly UI desktop application and a command-line interface (CLI) option. The UI version supports both Ethereum and Filecoin technologies, providing a versatile platform for dApp development.”[97]
- v) **MetaMask** : “MetaMask is a user-friendly and highly secure tool that allows seamless connectivity to blockchain-based applications. It provides a simple yet robust interface for interacting with the decentralized web. With MetaMask, users have complete control over their actions and data when engaging with blockchain applications. It offers a secure environ-

ment for conducting transactions and managing digital assets. MetaMask acts as a gateway, enabling users to effortlessly connect and interact with the new decentralized web, while ensuring their privacy and security.”[98]

5.2 Deploying the smart contracts using truffle and ganache (Local)

Step 1: Set up the project

- Create a new directory for your project.
- Open a terminal and navigate to the project directory.
- Initialize a new Truffle project by running the following command:



```
C:\Windows\System32\cmd.e x + v
Microsoft Windows [version 10.0.22621.1702]
(c) Microsoft Corporation. Tous droits réservés.

D:\e-vote>truffle init
```

Figure 34: Truffle initialisation

- After that you will find those folders and a file in the project directory

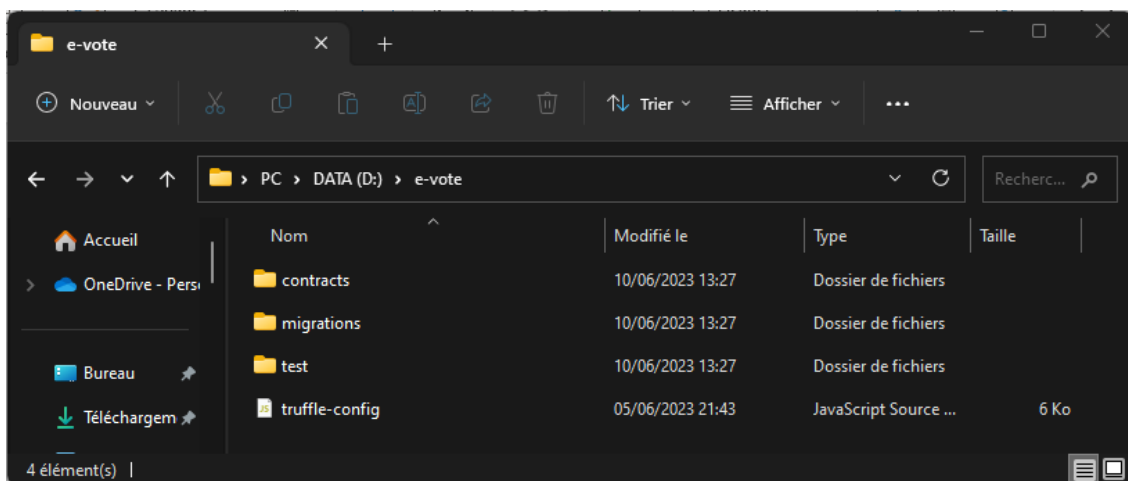


Figure 35: Project directory

Step 2: Update the Truffle configuration

- Update the configuration to connect to your Ganache network. By default, Truffle assumes the network is running on `http://127.0.0.1:7545`

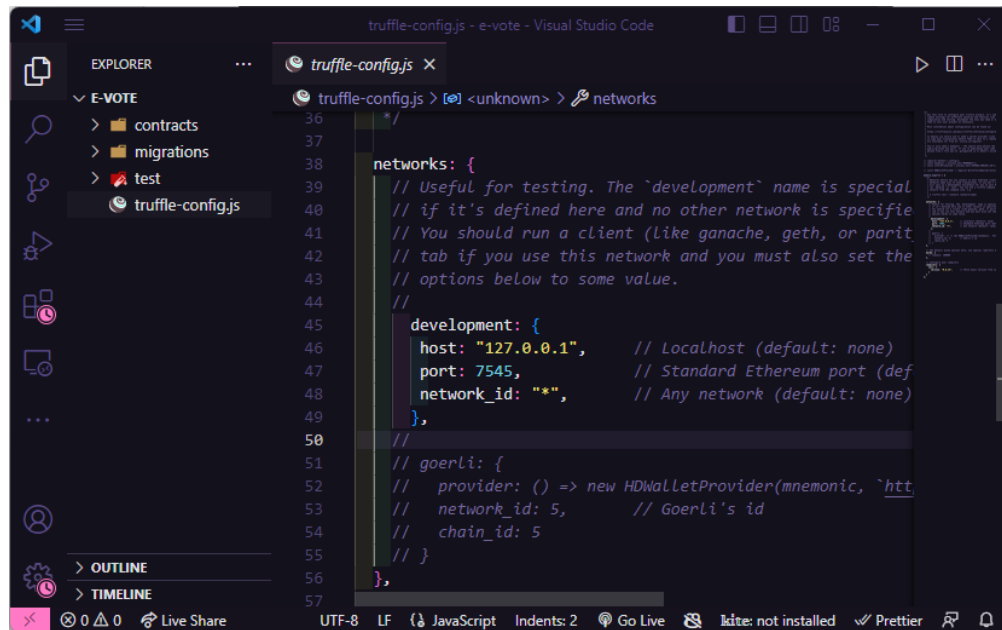
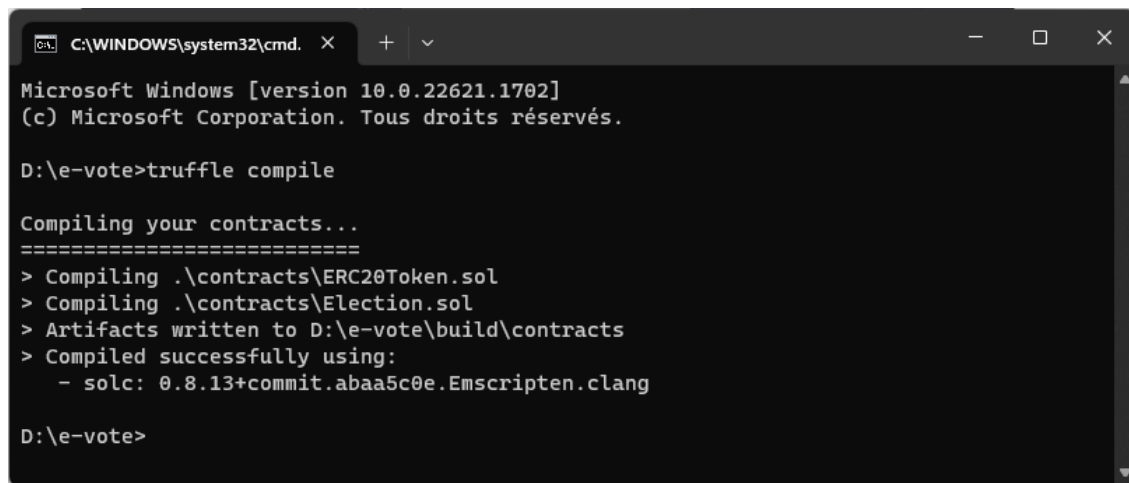


Figure 36: Truffle network configuration

Step 3: Compile the smart contracts

- Place the ERC20Token.sol and Election.sol files in the contracts directory of your Truffle project.
- Run the following command to compile the smart contracts:



```
C:\WINDOWS\system32\cmd. X + v
Microsoft Windows [version 10.0.22621.1702]
(c) Microsoft Corporation. Tous droits réservés.

D:\e-vote>truffle compile

Compiling your contracts...
=====
> Compiling .\contracts\ERC20Token.sol
> Compiling .\contracts\Election.sol
> Artifacts written to D:\e-vote\build\contracts
> Compiled successfully using:
  - solc: 0.8.13+commit.abaa5c0e.Emscripten.clang

D:\e-vote>
```

Figure 37: Compiling the smart contracts

Step 4: Start Ganache

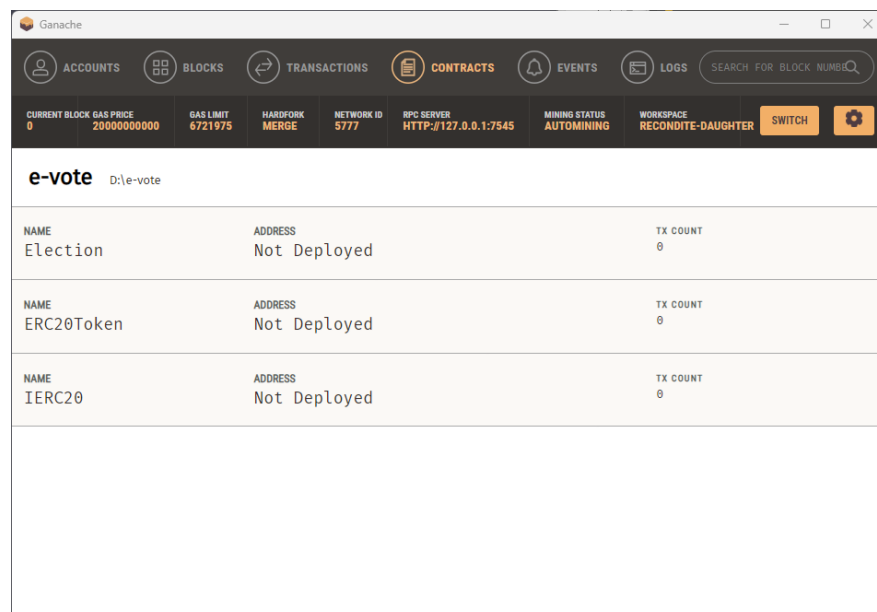


Figure 38: Ganache Started

Step 5: Migrate the smart contracts to Ganache

- Open the created migration file in the migrations directory (`1_deploy_contracts.js`).
- Added the the following code to deploy the contracts and then save :

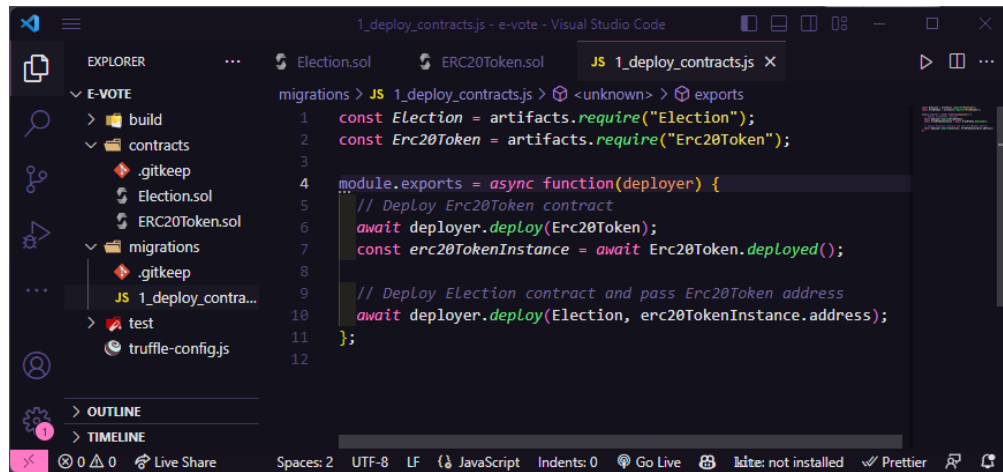


Figure 39: Code to deploy the Smart Contracts

The code above will deploy the ERC20Token contract first and then deploys the Election contract, passing the address of the ERC20Token contract as an argument.

Step 7: Deploy the contracts

- In the terminal, run the following command to deploy the contracts to your Ganache network:
- This command will deploy the contracts using the development network configuration specified in the `truffle-config.js` file.

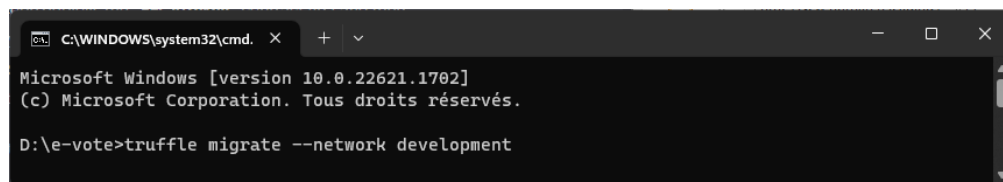


Figure 40: migrating the Smart Contracts

This command will deploy the contracts using the development network configuration specified in the `truffle-config.js` file.

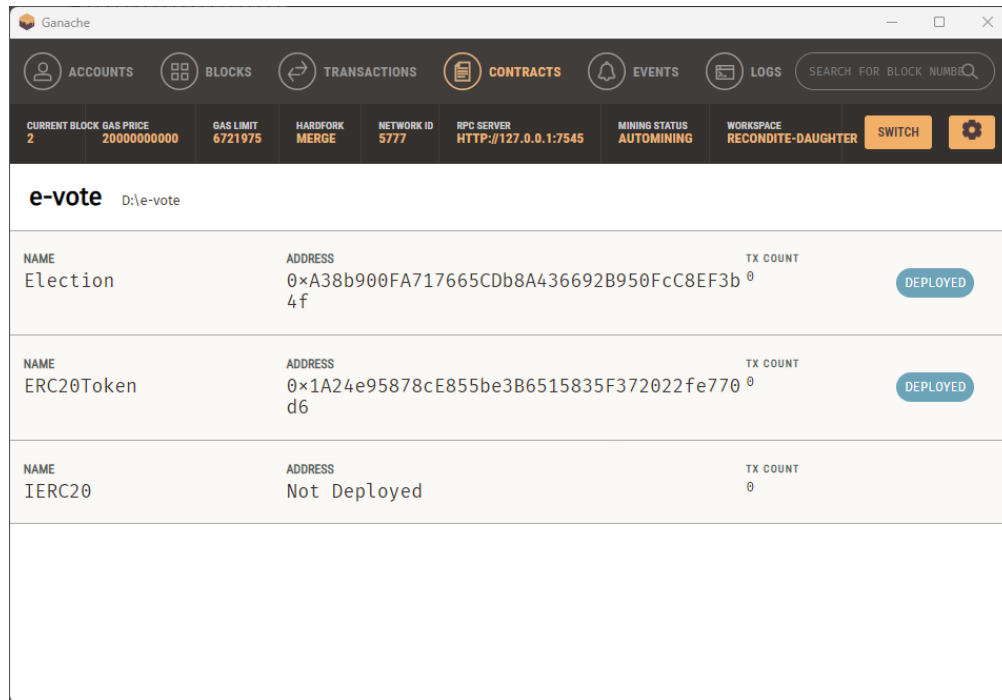


Figure 41: Smart Contracts Deployed

Step 8: Interact with the deployed contracts

- To interact with the contract, you can use the Truffle console using javascript, for example we can get the accounts provided for the test:

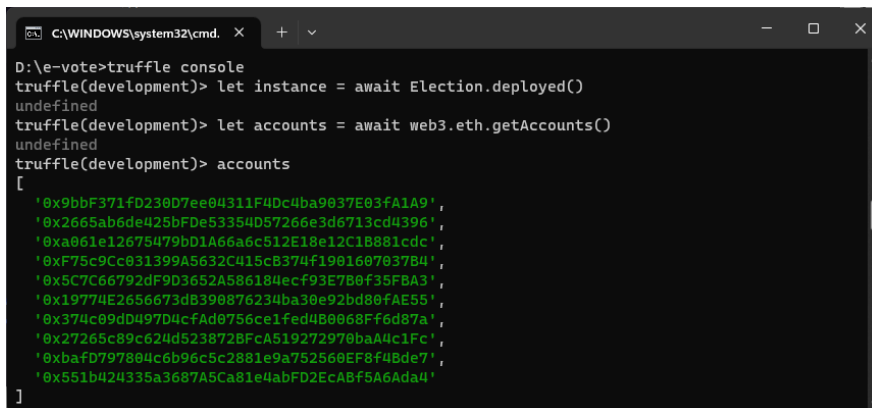
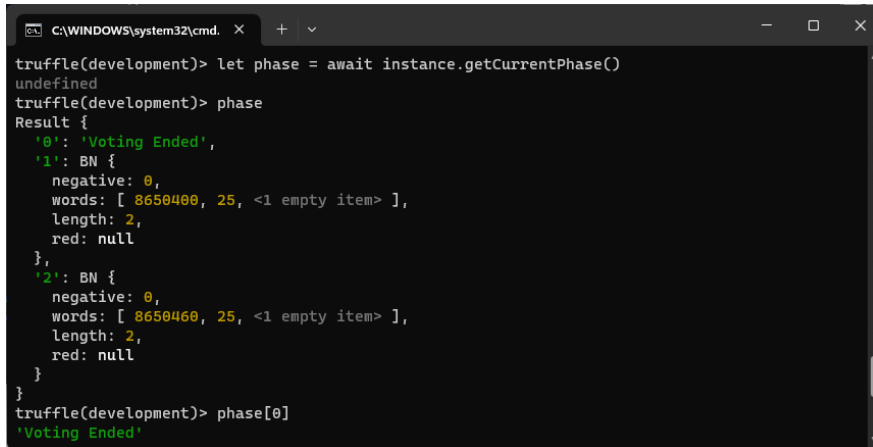


Figure 42: Getting accounts

- We can get the actual phase of the election process:



```
C:\WINDOWS\system32\cmd. x + v
truffle(development)> let phase = await instance.getCurrentPhase()
undefined
truffle(development)> phase
Result {
  '0': 'Voting Ended',
  '1': BN {
    negative: 0,
    words: [ 8650400, 25, <1 empty item> ],
    length: 2,
    red: null
  },
  '2': BN {
    negative: 0,
    words: [ 8650400, 25, <1 empty item> ],
    length: 2,
    red: null
  }
}
truffle(development)> phase[0]
'Voting Ended'
```

Figure 43: Interacting with smart contract

5.3 Smart Contracts deployment on Testnet

Now we will deploy our smart contracts on a testnet, “testnets are blockchains designed to mimic the operating environment of a ‘mainnet’ but exist on a separate ledger. These testnets help developers test their applications and smart contracts in a risk-free way before deploying their products to Ethereum’s mainnet environment.”[99]

There’s multiple testnet that we can find in the internet that developers use them to deploy there smart contract and test them. There’s a number of popular testnet like Goerli, Rinkbey, Sepolia, in our test we will use Sepolia testnet, with remix ide and Metamask. “Sepolia is a stable Ethereum testnet that merged from Proof of Work to Proof of Stake along with the Ethereum mainnet and serves developers with the infrastructure to deploy and test Solidity smart contracts. Sepolia has a faucet service to ensure developers only use Sepolia ETH for gas when staging smart contracts on the testnet.”[100]

a) Metamask configuration

- Switch the network to Sepolia test network :

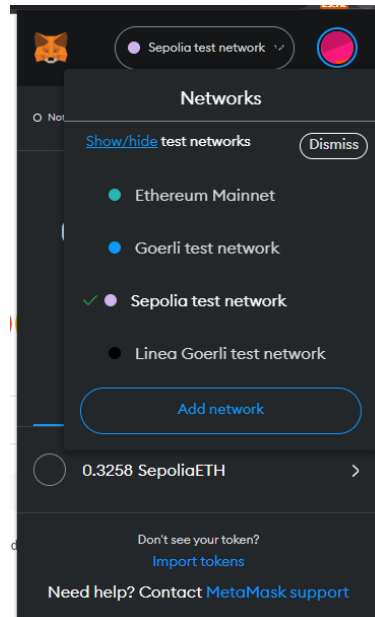


Figure 44: Switching to Sepolia test network

- After that we need some sepoliaETH Copy your account address and go to <https://sepoliafaucet.com/>, paste the address and click send Me ETH

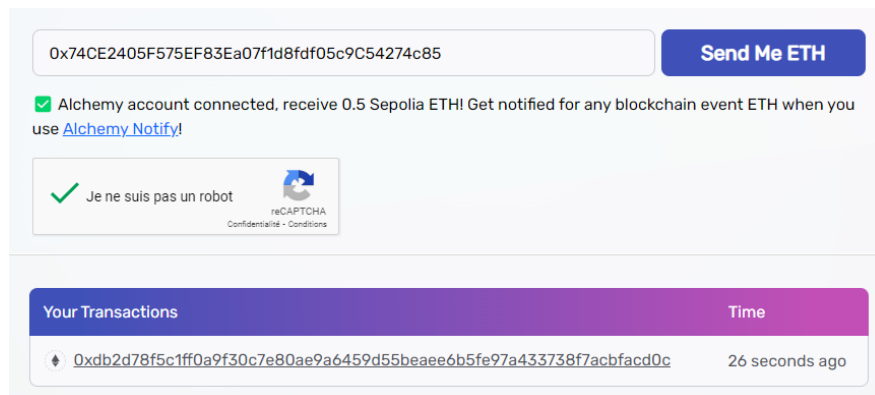


Figure 45: Getting some ETH

- We will get 0.5 ETH that we will need when making deployment and transaction with smart contracts.
- Go to remix IDE and added the accounts provided for the test and import them to metamask

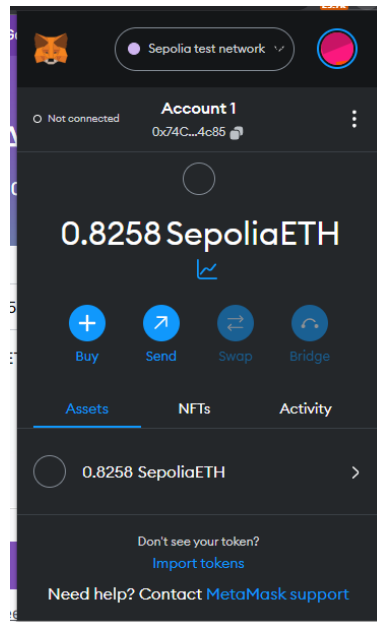


Figure 46: Account have eth

- Change environment to : Injected Provider -Metamask to link them together
- Create accounts to to use them for testing, send them an amount of ethereum from the main account. And now we are all set.

b) Deploy the contracts to Sepolia testnet

- Open the remix IDE and Compile and deploy the ERC20Token smart contracts first.

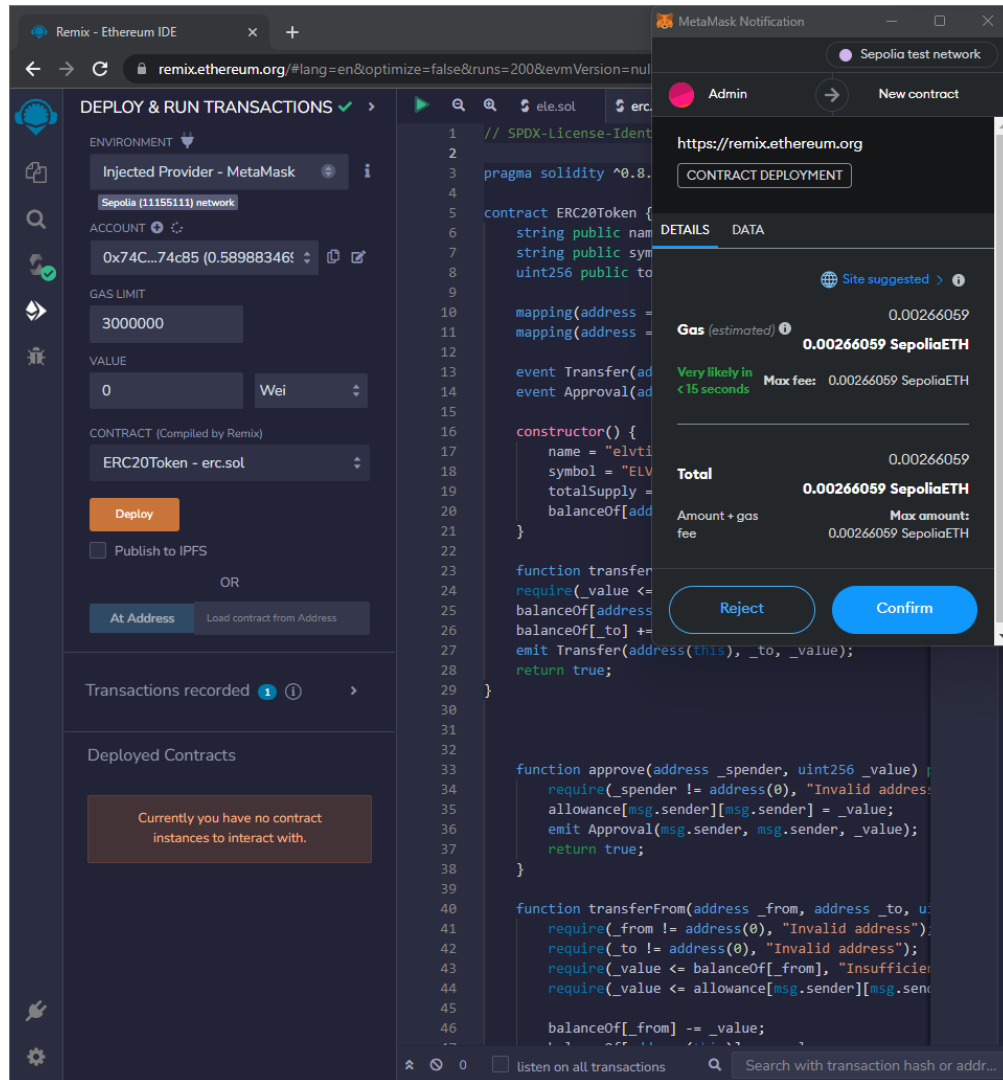


Figure 47: Deploying ERC20 Smart Contract

- We can check our ERCcontract in etherscan using the address
0xb1423c0c898f11f918eee8884583d2963cdc4b74

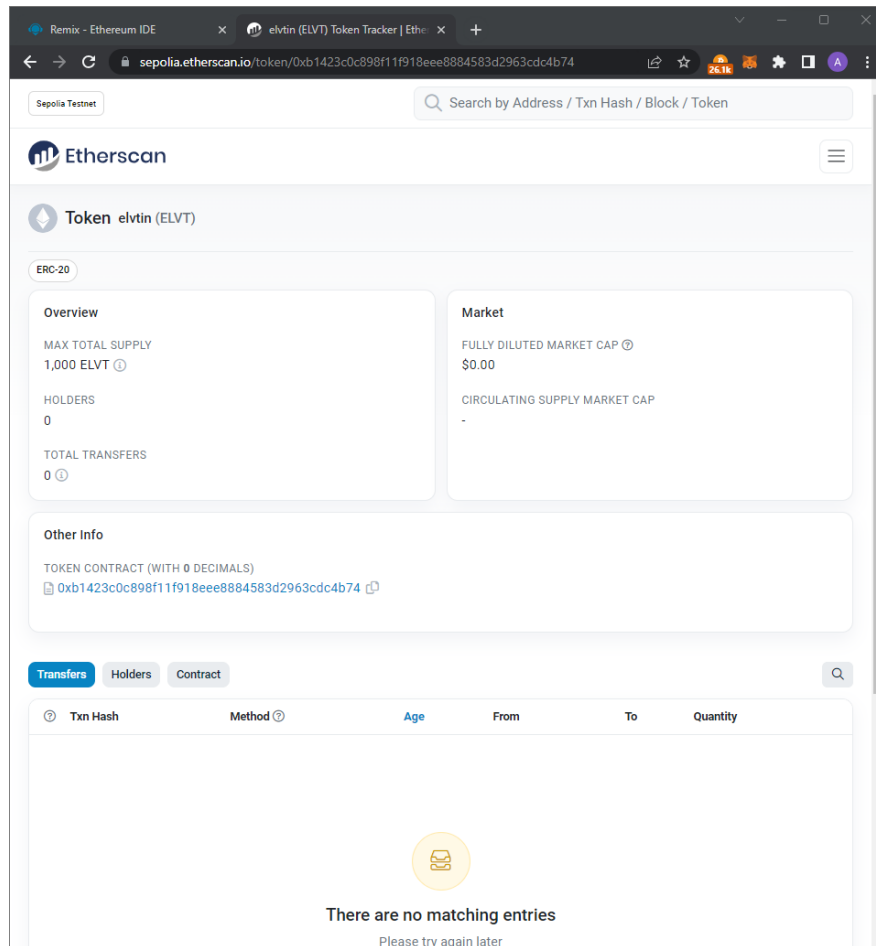


Figure 48: ERC20Contract Deployed

- Now, let's set votingStartTime, votingEndTime values and then use the ERC20 Address to deploy the voting contract

Unix Timestamp	1686411000
GMT	Sat Jun 10 2023 15:30:00 GMT+0000
Your Time Zone	Sat Jun 10 2023 16:30:00 GMT+0100 (heure normale d'Afrique de l'Ouest)
Relative	In 41 minutes

Figure 49: Timestamp Conversion

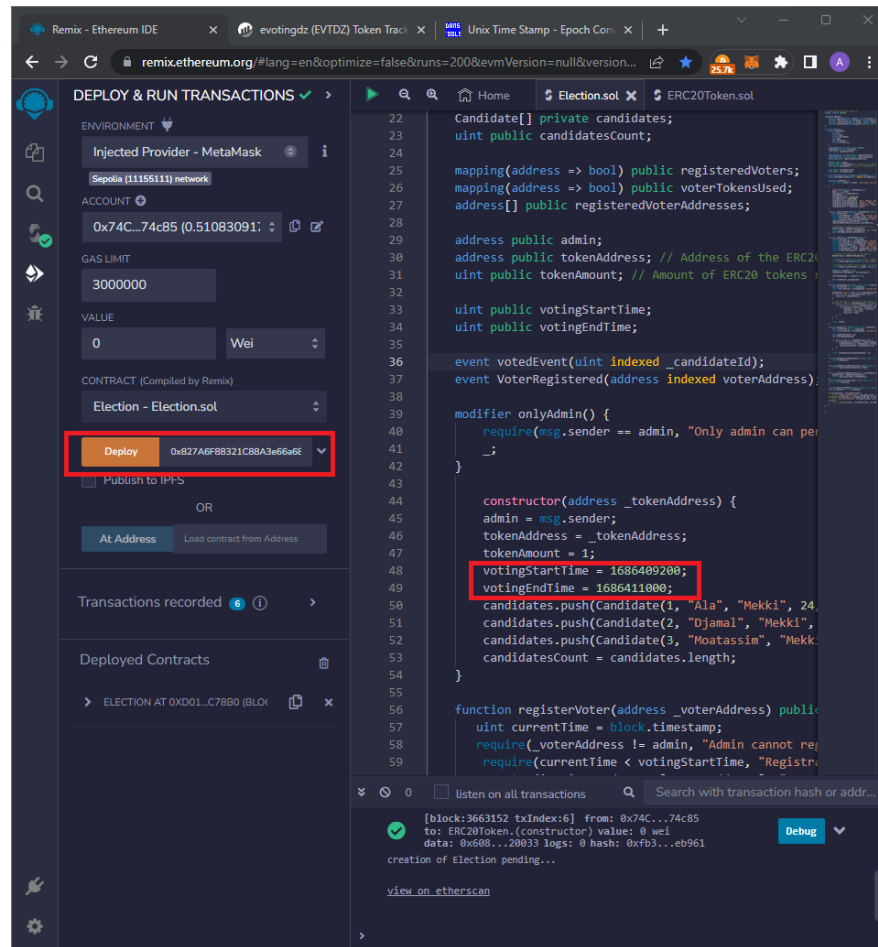


Figure 50: Deploying Voting SM With Setting Times and ERC20Token Address

- We can check our Voting contract in etherscan using the address
0xb1423c0c898f11f918eee8884583d2963cdc4b74

- We can see the different transaction done through the smart contract in transaction part below

The screenshot displays the Etherscan Sepolia Testnet interface for a specific contract address: 0xD9003683C37f44E1C20Dc14F4D18753Cf36804fa. The page includes a search bar, the Etherscan logo, and a 'Contract' label. Below this, there are sections for 'Overview' (showing an ETH balance of 0), 'More Info' (showing the creator address 0x74CE24...54274c85), and 'Multi Chain' (showing 1 address found via Blockscan). The 'Transactions' tab is selected, showing a list of transactions. Two transactions are visible, both originating from the creator address and interacting with the contract.

Transaction Hash	Method	Block	Age	From	To
0x4c4c9f2332913a7ac...	Register Voter	3680631	31 secs ago	0x74CE24...54274c85	0xD90036...f36804fa
0xf5e2da1fc16990450...	0x60806040	3680619	3 mins ago	0x74CE24...54274c85	Contract Creation

Powered by Ethereum

Figure 51: Account hve eth

- Interacting with Smart Contract
 - Voter Registring :

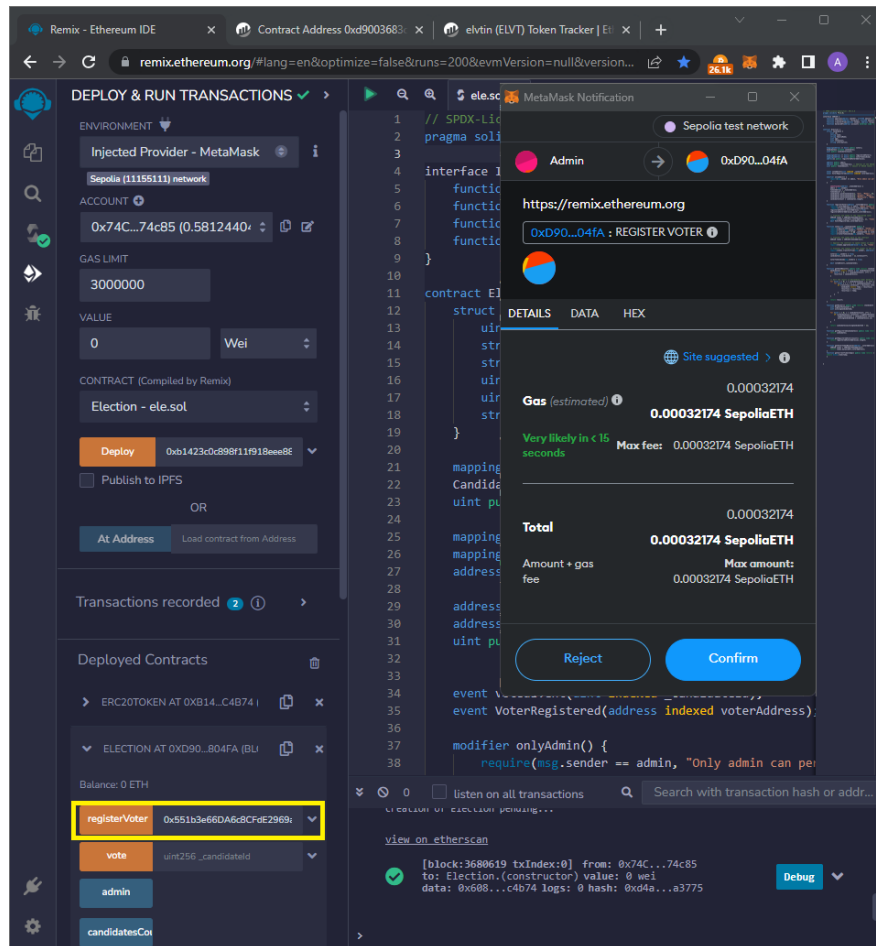


Figure 52: Admin Registring Voter

– Voter Casting vote :

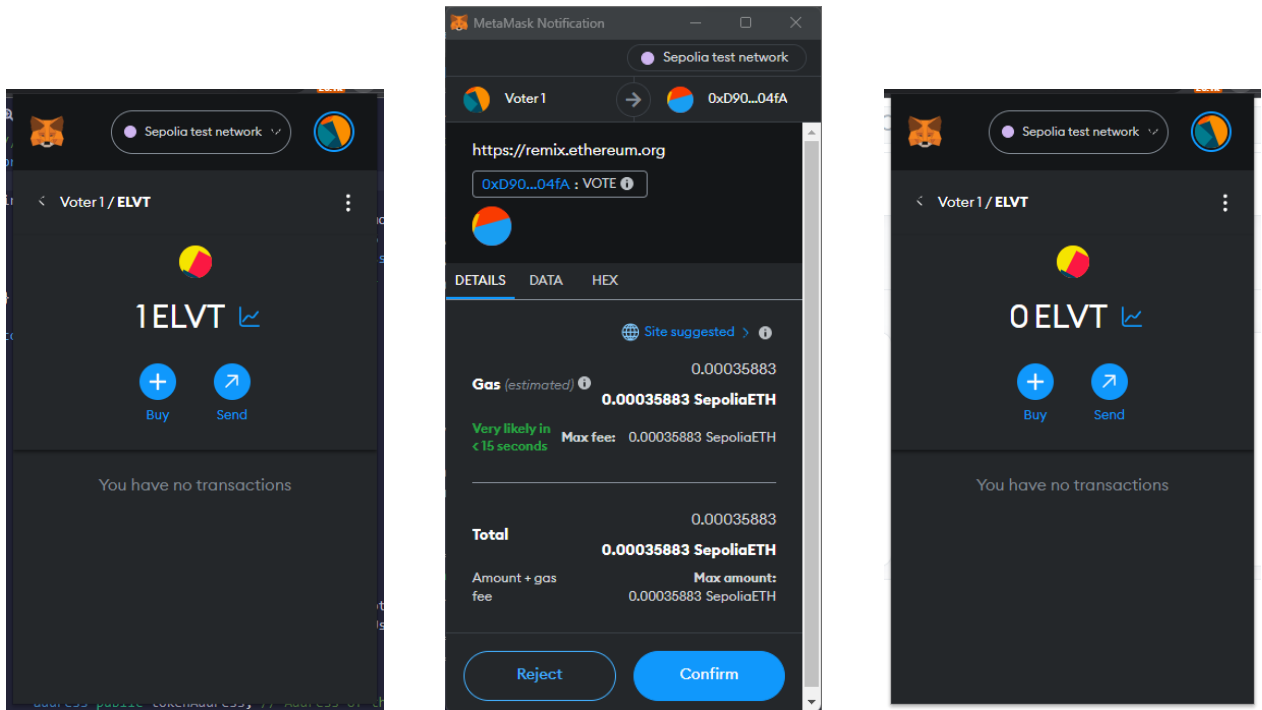


Figure 53: Voting Process

- Retrieving information from contract :

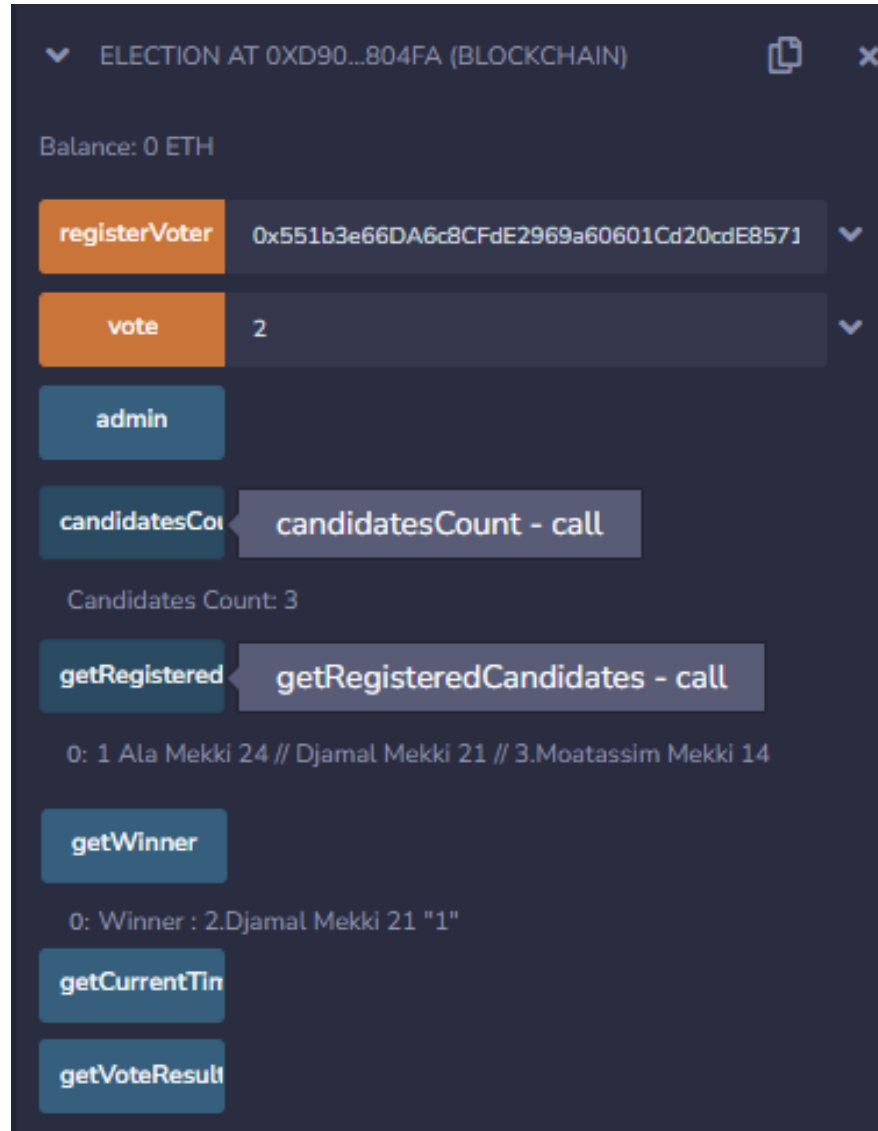


Figure 54: Functions And Calls to interact with Voting Smart Contract

5.4 Screenshots from the user interface of the application

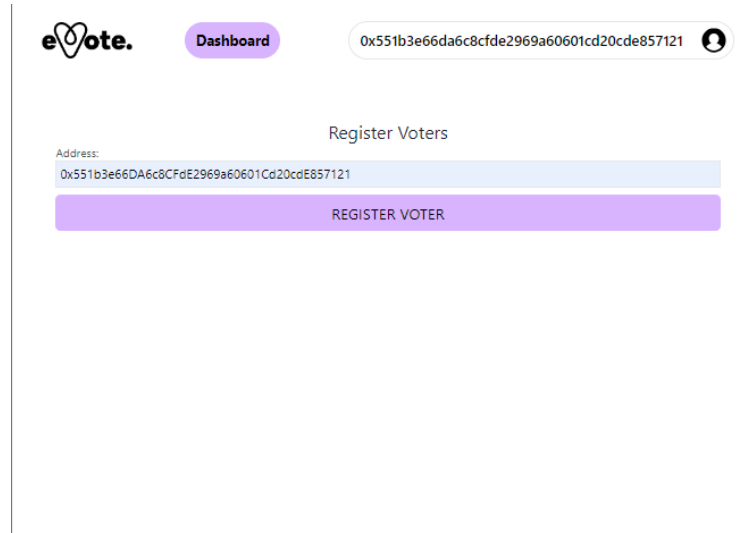


Figure 55: Admin Dashboard



Figure 56: Voting Page

 Voting **Results** 0x551b3e66da6c8cfde2969a60601cd20cde857121 

Results Table

id	name	party	age	voteCount	information
1 (Draw)	Candidate 1	Party A	24	1	Lorem ipsum dolor sit amet.
2 (Draw)	Candidate 2	Party B	21	1	Consectetur adipiscing elit.
3	Candidate 3	Party C	14	0	Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 57: Results Page

GENERAL CONCLUSION

In conclusion, blockchain technology has provided real innovation in the field of decentralized, secure transactions, especially starting with its first version, Bitcoin. This technology solves major problems such as double spending, by guaranteeing integrity and consensus within a decentralized network. In fact, several components provide a global view of the data transaction, hashing and security process. Consensus mechanisms such as proof-of-work (PoW) and proof-of-stake (PoS) are essential to reach agreement and preserve network integrity.

The technological aspects of blockchain technology, such as centralized, decentralized, and distributed networks, distributed registers, peer-to-peer networks, and cryptography, help ensure blockchain systems' security and immutability. Blockchains can be applied in lots of ways thanks to the idea of smart contracts, which are autonomous programmable solutions.

Electronic voting systems currently come across challenges in terms of security, efficiency and trust. By integrating blockchain technology into these systems, these challenges can be overcome, improving transparency, immutability and auditability. The design and implementation of a blockchain-based voting system requires an understanding of the Ethereum blockchain, smart contracts, the ERC20 token standard and wallet integration. Employing ERC20 tokens within a voting system offers the possibility to conduct secure and traceable voting operations, assuring the voting process's integrity.

In the end, blockchain-based electronic voting systems optimize transparency, eliminate corruption, expand easy access, and increase confidence in the electoral system. Electronic voting could develop into an even more safe, effective, and reliable process for democratic choice-making by using the benefits of blockchain technology.

REFERENCES

- [1] Imran Bashir. *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained*. 2nd. Packt, 2018. ISBN: 978-1-78883-904-4.
- [2] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. <https://bitcoin.org/bitcoin.pdf>. Accessed on June 11, 2023. 2008.
- [3] Andreas M. Antonopoulos. *Mastering Bitcoin: Programming the Open Blockchain*. 2nd. City: O'Reilly, 2017. ISBN: 978-1-491-95438-6.
- [4] <https://www.zenledger.io/blog/who-controls-bitcoin>. Accessed on March 11, 2023.
- [5] <https://medium.com/@deependrasg/what-is-double-spending-problem-in-blockchain-how-bitcoin-solve-this-problem-9f9f1c237db0>. Accessed on April 10, 2023.
- [6] <https://auth0.com/blog/an-introduction-to-ethereum-and-smart-contracts>. Accessed on April 10, 2023.
- [7] <https://medium.com/@deependrasg/what-is-double-spending-problem-in-blockchain-how-bitcoin-solve-this-problem-9f9f1c237db0>. Accessed on April 10, 2023.
- [8] <https://www.opennode.com/blog/bitcoin-proof-of-work/>. Accessed on April 11, 2023.
- [9] <https://www.thewolfofallstreets.io/bitcoin-and-the-byzantine-generals-problem>. Accessed on April 12, 2023.
- [10] <https://witscad.com/course/blockchain-fundamentals/chapter/blockchain-characteristics>. Accessed on April 13, 2023.
- [11] <https://www.getsecureworld.com/blog/blockchain-security/what-are-the-six-layers-of-blockchain-technology>. Accessed on April 13, 2023.

- [12] Pinchen Cui and et al. “Blockchain in IoT: Current Trends, Challenges, and Future Roadmap”. In: *Journal of Hardware and Systems Security* 3.4 (Dec. 2019), pp. 338–364. DOI: 10 . 1007/s41635-019-00079-5. URL: <https://doi.org/10.1007/s41635-019-00079-5>.
- [13] Imran Bashir. *Mastering Blockchain: A Technical Reference Guide to the Inner Workings of Blockchain, from Cryptography to DeFi and NFTs*. Fourth. Packt Publishing Ltd., 2023.
- [14] <https://www.shiksha.com/online-courses/articles/structure-of-a-block-in-blockchain>. Accessed on April 13, 2023.
- [15] Shubhani Aggarwal and Neeraj Kumar. “Architecture of Blockchain”. In: *Advances in Computers* 121 (2021), pp. 171–192. DOI: 10 . 1016/bs . adcom . 2020 . 08 . 009. URL: <https://doi.org/10.1016/bs.adcom.2020.08.009>.
- [16] Shubhani Aggarwal and Neeraj Kumar. “Phases of Operation”. In: *Advances in Computers* 121 (2021), pp. 285–299. DOI: 10 . 1016/bs . adcom . 2020 . 08 . 014. URL: <https://doi.org/10.1016/bs.adcom.2020.08.014>.
- [17] <https://assets.kpmg.com/content/dam/kpmg/pdf/2016/06/kpmg-blockchain-consensus-mechanism.pdf>. Accessed on May 01, 2023.
- [18] Dylan Yaga and et al. *Blockchain Technology Overview*. NIST IR 8202. National Institute of Standards and Technology, 2018. URL: <https://doi.org/10.6028/NIST.IR.8202>.
- [19] <https://www.ledger.com/academy/blockchain/what-is-proof-of-work>. Accessed: May 2, 2023.
- [20] <https://www.horizen.io/academy/consensus-mechanisms/>. Accessed: May 2, 2023.
- [21] <https://www.gemini.com/cryptopedia/blockchain-network-decentralized-distributed-centralized#section-blockchain-networks-a-reconfiguration>. Accessed: May 5, 2023.
- [22] <https://berty.tech/blog/decentralized-distributed-centralized/>. Accessed: May 5, 2023.

- [23] <https://community.neontools.io/web3-guide/23>. Accessed: May 5, 2023.
- [24] <https://101blockchains.com/what-is-dlt/>. Accessed: May 5, 2023.
- [25] <https://cointelegraph.com/learn/what-are-peer-to-peer-p2p-blockchain-networks-and-how-do-they-work>. Accessed: May 5, 2023.
- [26] Rashid Khan and et al. *Learning IoT with Particle Photon and Electron: Develop Applications on One of the Most Popular Platforms for IoT Using Particle Photon and Electron with This Fast-Paced Guide*. Packt Publishing, 2016.
- [27] <https://www.analyticsvidhya.com/blog/2022/09/concept-of-cryptography-in-blockchain/>. Accessed: May 6, 2023.
- [28] <https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences>. Accessed: May 6, 2023.
- [29] <https://www.cisco.com/c/en/us/products/security/encryption-explained.html#~encryption-algorithms>. Accessed: May 6, 2023.
- [30] <https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences>. Accessed: May 6, 2023.
- [31] Pedro Franco. *Understanding Bitcoin: Cryptography, Engineering, and Economics*. Wiley, 2015.
- [32] <https://medium.com/techskill-brew/cryptography-in-blockchain-part-6-blockchain-basics-129ec058c574>. Accessed: May 7, 2023.
- [33] <https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between>. Accessed: May 8, 2023.
- [34] <https://techbullion.com/four-types-of-blockchain-technology/>. Accessed: May 8, 2023.
- [35] Vivek Acharya and et al. *Oracle Blockchain Quick Start Guide: A Practical Approach to Implementing Blockchain in Your Enterprise*. Packt Publishing, Limited, 2019.

- [36] Shubhani Aggarwal and Neeraj Kumar. “Blockchain Components and Concepts”. In: *Advances in Computers* 121 (2021), pp. 387–398. DOI: 10.1016/bs.adcom.2020.08.019. URL: <https://doi.org/10.1016/bs.adcom.2020.08.019>.
- [37] <https://www.insiderintelligence.com/insights/blockchain-technology-applications-use-cases/>. Accessed: May 9, 2023.
- [38] <https://dataconomy.com/2022/05/17/blockchain-use-cases-2022/>. Accessed: May 9, 2023.
- [39] https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html. Accessed: June 12, 2023.
- [40] <https://www.ibm.com/topics/smart-contracts>. Accessed: May 14, 2023.
- [41] Zibin Zheng and et al. “An Overview on Smart Contracts: Challenges, Advances and Platforms”. In: *Future Generation Computer Systems* 105 (Apr. 2020), pp. 475–491. DOI: 10.1016/j.future.2019.12.019. URL: <https://doi.org/10.1016/j.future.2019.12.019>.
- [42] <https://chainwitcher.com/smart-contracts/>. Accessed: May 14, 2023.
- [43] <https://metaschool.so/articles/what-is-a-smart-contract/>. Accessed: May 15, 2023.
- [44] <https://www.halborn.com/blog/post/blockchain-consensus-mechanisms-compared>. Accessed: May 15, 2023.
- [45] <https://www2.deloitte.com/us/en/insights/focus/signals-for-strategists/using-blockchain-for-smart-contracts.html>. Accessed: May 22, 2023.
- [46] Shafaq Naheed Khan and et al. “Blockchain Smart Contracts: Applications, Challenges, and Future Trends”. In: *Peer-to-Peer Networking and Applications* 14.5 (Sept. 2021), pp. 2901–2925. DOI: 10.1007/s12083-021-01127-0. URL: <https://doi.org/10.1007/s12083-021-01127-0>.

- [47] <https://content.enkronos.com/what-is-a-smart-contract-advantages-and-disadvantages/>. Accessed: May 22, 2023.
- [48] <https://www.socialistsanddemocrats.eu/what-we-stand-for/why-voting-matters>. Accessed: May 22, 2023.
- [49] <https://www.worldblaze.in/advantages-disadvantages-of-paper-ballot-voting/>. Accessed: May 23, 2023.
- [50] Ivan Pankevych and et al. “Voting Methods: Challenges, Modern Tendencies and Experience Globally”. In: *Journal of Legal, Ethical and Regulatory Issues* 24.1S (May 2021), pp. 1–9. URL: <https://www.abacademies.org/abstract/voting-methods-challenges-modern-tendencies-and-experience-globally-11051.html>.
- [51] <https://blogs.lse.ac.uk/usappblog/2020/09/25/long-read-how-blockchain-can-make-electronic-voting-more-secure/>. Accessed: May 23, 2023.
- [52] <https://www.britannica.com/topic/electronic-voting>. Accessed: May 23, 2023.
- [53] Muhammad Budiana. “The Use of E Voting in the General Election System in Indonesia”. In: *Legal Brief* 11.5 (2022). Published by IHSA Institute (Institut Hukum Sumberdaya Alam), pp. 2656–2662. ISSN: 1979-522X. DOI: 10.35335/legal.
- [54] <https://aceproject.org/ace-en/focus/e-voting/historical-overview>. Accessed: May 26, 2023.
- [55] <https://www.comparitech.com/blog/information-security/electronic-voting-risks/>. Accessed: June 12, 2023.
- [56] <https://www.nist.gov/document/evempauiditingpdf>. Accessed: June 12, 2023.
- [57] Robert Krimmer and et al., eds. *Electronic Participation: 14th IFIP WG 8.5 International Conference, ePart 2022, Linköping, Sweden, September 6–8, 2022, Proceedings*. Springer Nature Switzerland, 2022. DOI: 10.1007/978-3-031-23213-8.

- [58] <https://aceproject.org/ace-en/focus/e-voting/default>. Accessed: June 1, 2023.
- [59] Piret Ehin et al. “Internet Voting in Estonia 2005–2019: Evidence from Eleven Elections”. In: *Government Information Quarterly* 39.4 (Oct. 2022), p. 101718. DOI: 10.1016/j.giq.2022.101718. URL: <https://doi.org/10.1016/j.giq.2022.101718>.
- [60] https://www.elexion.fr/blog-elexion/34_vote-electronique-avantages-et-inconvnients.html. Accessed: June 12, 2023.
- [61] <https://www.keyvote.fr/vote-electronique-avantages-et-inconvenients>. Accessed: June 12, 2023.
- [62] <https://www.moyens.net/securite/fonctionnement-du-vote-electronique-avantages-et-inconvenients-vs-vote-papier/>. Accessed: June 12, 2023.
- [63] Sheer Hardwick and et al. “E-Voting With Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy”. In: *IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)* (2018), pp. 1561–1567. DOI: 10.1109/Cybermatics_2018.2018.00262. URL: https://doi.org/10.1109/Cybermatics_2018.2018.00262.
- [64] IEEE. “Proceedings of the 2nd International Conference on Inventive Research in Computing Applications (ICIRCA 2020)”. In: *2nd International Conference on Inventive Research in Computing Applications (ICIRCA 2020)*. IEEE. July 2020.
- [65] Muhammad Shoaib Farooq and et al. “A Framework to Make Voting System Transparent Using Blockchain Technology”. In: *IEEE Access* 10 (2022), pp. 59959–59969. DOI: 10.1109/ACCESS.2022.3180168.
- [66] <https://builtin.com/blockchain/blockchain-voting-future-elections>. Accessed on June 1, 2023.

- [67] <https://www.publish0x.com/informative-crypto-related-topics/the-pros-and-cons-of-blockchain-voting-xpngvoe>. Accessed on June 12, 2023.
- [68] <https://beincrypto.com/blockchain-voting-securing-democracy/>. Accessed on June 3, 2023.
- [69] Maria-Victoria Vladucu et al. “E-Voting Meets Blockchain: A Survey”. In: *IEEE Access* 11 (2023), pp. 23293–308. DOI: 10.1109/ACCESS.2023.3253682. URL: <https://doi.org/10.1109/ACCESS.2023.3253682>.
- [70] <https://www.synopsys.com/glossary/what-is-blockchain.html>. Accessed on June 3, 2023.
- [71] <https://web3fits.com/understanding-blockchain/>. Accessed on June 3, 2023.
- [72] <https://t2conline.com/how-public-ledger-provides-accountability-and-trust/>. Accessed on June 3, 2023.
- [73] Alisa Sydow et al. “Leveraging Blockchain’s Potential – The Paradox of Centrally Legitimate, Decentralized Solutions to Institutional Challenges in Kenya”. In: *Journal of Business Venturing Insights* 14 (Nov. 2020), e00170. DOI: 10.1016/j.jbvi.2020.e00170. URL: <https://doi.org/10.1016/j.jbvi.2020.e00170>.
- [74] Uzma Jafar et al. “A Systematic Literature Review and Meta-Analysis on Scalable Blockchain-Based Electronic Voting Systems”. In: *Sensors* 22.19 (Oct. 2022), p. 7585. DOI: 10.3390/s22197585. URL: <https://doi.org/10.3390/s22197585>.
- [75] <https://chain.link/education-hub/smart-contract-automation>. Accessed on June 3, 2023.
- [76] <https://ethereum.org/en/developers/docs/gas/>. Accessed on June 9, 2023.
- [77] <https://ethereum.org/en/developers/docs/intro-to-ethereum/>. Accessed on June 9, 2023.

- [78] <https://www.edureka.co/blog/ethereum-tutorial-with-smart-contracts/>. Accessed on June 9, 2023.
- [79] <https://dev.to/yusuferdogan/ethereum-transactions-2b4m>. Accessed: June 12, 2023.
- [80] <https://ethereum.org/en/developers/docs/smart-contracts/>. Accessed on June 9, 2023.
- [81] <https://docs.soliditylang.org/en/v0.8.20/>. Accessed on June 9, 2023.
- [82] Andreas M. Antonopoulos and Gavin Wood. *Mastering Ethereum: Building Smart Contracts and DApps*. First. O'Reilly, 2019.
- [83] <https://docs.openzeppelin.com/contracts/3.x/tokens#ERC20>. Accessed on June 9, 2023.
- [84] <https://learn.metamask.io/lessons/what-is-a-crypto-wallet>. Accessed on June 9, 2023.
- [85] <https://blog.logrocket.com/5-blockchain-wallets-smart-contract-dapp/>. Accessed on June 9, 2023.
- [86] <https://ethereum.org/en/developers/docs/standards/tokens/erc-20/>. Accessed on June 9, 2023.
- [87] <https://docs.openzeppelin.com/contracts/3.x/tokens#ERC20>. Accessed on June 9, 2023.
- [88] <https://docs.soliditylang.org/en/v0.8.20/types.html#mapping-types>. Accessed on June 9, 2023.
- [89] <https://solidity-by-example.org/events/>. Accessed on June 9, 2023.
- [90] <https://solidity-by-example.org/constructor/>. Accessed on June 9, 2023.
- [91] Nils-Christian Bormann and Matt Golder. “Democratic Electoral Systems around the World, 1946–2020”. In: *Electoral Studies* 78 (Aug. 2022), p. 102487. DOI: 10.1016/j.electstud.2022.102487. URL: <https://doi.org/10.1016/j.electstud.2022.102487>.

- [92] <https://docs.soliditylang.org/en/v0.8.14/structure-of-a-contract.html>. Accessed on June 9, 2023.
- [93] <https://docs.soliditylang.org/en/v0.8.20/contracts.html#function-modifiers>. Accessed on June 9, 2023.
- [94] <https://github.com/ethereum/remix-ide>. Accessed on June 9, 2023.
- [95] <https://code.visualstudio.com/docs>. Accessed on June 9, 2023.
- [96] <https://trufflesuite.com/docs/truffle/>. Accessed on June 9, 2023.
- [97] <https://trufflesuite.com/docs/ganache/>. Accessed: June 12, 2023.
- [98] <https://metamask.io/>. Accessed: June 12, 2023.
- [99] <https://www.alchemy.com/overviews/sepolia-testnet>. Accessed: June 12, 2023.
- [100] <https://www.alchemy.com/dapps/sepolia>. Accessed: June 12, 2023.

ملخص

تدرس هذه المذكرة وتستكشف كيف يمكن استخدام تقنية سلاسل الكتل للمساعدة في جعل العملية الانتخابية تجري بشكل أكثر شفافية وأكثر أماناً وكفاءة. من خلال التطرق للمشاكل المعروفة لأساليب التصويت التقليدية، تقترح المذكرة نظام تصويت جديد قائم على تقنية سلاسل الكتل باستخدام معيار ERC20 الخاص بإيثريوم، مما يوفر انتخابات آمنة وشفافة. مما وجدنا فيه من نتائج البحث، أنه يمكن أن يساهم تنفيذ نظام انتخابي قائم على تقنية سلاسل الكتل بشكل كبير في تحسين إجراءات التصويت، مما يقلل من مخاطر الاحتيال والتلاعب. يساعد الجانب اللامركزي في تقنية سلاسل الكتل على جعل عملية التصويت أكثر ديمقراطية. بشكل عام، يهدف البحث الى تطوير الانتخابات من خلال الإشارة إلى قوة تقنية سلاسل الكتل للمساعدة في بناء عملية تصويت أكثر كفاءة وأماناً وشفافية.

Abstract

This thesis studies and explores how blockchain can be used to help make the electoral process run more transparently, more secure, and more efficiently. By addressing the known limitations of conventional voting methods, the thesis suggests a promising new blockchain-based voting system. Using Ethereum's ERC20 token standard, providing secure and transparent elections. According to the research results, the implementation of a blockchain-based electoral system can make a big contribution to better voting procedures, reducing the risk of fraud and manipulation. The decentralized aspect of blockchain helps to make the voting process more democratic. Overall, this research serves to inform this in election by pointing to the power of blockchain technology to help build a voting process that is at once more efficient, more secure, and more transparent.

Résumé

Cette mémoire étudie et explore comment la blockchain peut être utilisée pour rendre le processus électoral plus transparent, plus sûr et plus efficace. En abordant les limites connues des méthodes de vote conventionnelles, la mémoire suggère un nouveau système de vote prometteur basé sur la blockchain. L'utilisation de la norme de jeton ERC20 d'Ethereum permet d'organiser des élections sécurisées et transparentes. Selon les résultats de la recherche, la mise en œuvre d'un système électoral basé sur la blockchain peut contribuer grandement à l'amélioration des procédures de vote, en réduisant le risque de fraude et de manipulation. L'aspect décentralisé de la blockchain contribue à rendre le processus de vote plus démocratique. Dans l'ensemble, cette recherche sert à informer cette élection en soulignant le pouvoir de la technologie blockchain pour aider à construire un processus de vote qui est à la fois plus efficace, plus sûr et plus transparent.