“ALEXANDRU IOAN CUZA” UNIVERSITY OF IAŞI
FACULTY OF INFORMATICS

Bachelor Thesis

Blockchain usage in secured voting

Session: July, 2018

Scientific coordinator,
Conf. Dr. Iftene Adrian

“ALEXANDRU IOAN CUZA” UNIVERSITY OF IAŞI
FACULTY OF INFORMATICS

Blockchain usage in secured voting

Hardon
Elena-Andreea

Session: July, 2018

Scientific coordinator,
Conf. Dr. Iftene Adrian

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele Conf. Dr. Iftene Adrian

Data _____ Semnătura_____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnata Hardon Elena-Andreea cu domiciliul în Iași născut(ă) la data de 03.05.1996, identificata prin 2960503226749 absolvent(a) al(a) Universității „Alexandru Ioan Cuza" din Iași, Facultatea de Informatică specializarea Informatică, promoția 2017-2018 declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul: "Blockchain usage in secured voting" elaborată sub îndrumarea dl. Conf. Dr. Iftene Adrian, pe care urmează să o susțină în față comisiei este originală, îmi aparține și îmi asum conținutul sau în întregime. De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop. Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice in vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Dată …… Semnătură student ……

DECLARATIE DE CONSIMTAMANT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „Blockchain usage in secured voting”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” Iași să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, data

Absolvent
Elena-Andreea Hardon

_____

(semnătura în original)

Contents

Abstract

Nowadays it is hard to live without technology since it is all around us and sometimes, even though is not is the only way of doing usual things such as paying bills, buying clothes or even food or medicines, we pick the technology side where we can solve our daily issues in just a click away.

The real issues we are facing is the way our personal data, from shared thoughts, photos, conversations from social media, to bank accounts and documents that are private, is managed and stored. Another concern is related to who can access it and if we are safe sharing it in the online environment.

From all the needs that have been eased to be solved in a blink of an eye from home, a real help when it came about voting is still in discussion. There have been maybe decades since people had doubts regarding a real and non spurious counting of votes in elections.

With the help of nowadays technologies, this can now be possible. Blockchain, a cryptographic technology, allows voting even from home while eliminating the central authority that is counting the votes in the final stage of the elections. No paper will be needed and no time will be wasted since this electronic vote will be counted live and every user that will use this voting system, will have a copy of the votes in the network. No illegal voting will be allowed since every vote will be verified to be unique through a secured method.

The obvious truth is that the personal data of a voter could be in danger if is not protected as it should be and that its personal information is in a real danger if the data sent with the vote is not protected. This can be solved using hashes, blockchain, sidechains and smart contracts. The voter's data will be hidden, but safely kept and his vote counted if the data passes the verifying stage.

Marci McCarthy about the threatening coming from unprotected data of an user (Cole, 2016):

"*Modern information threats come in many forms: from hackers looking to steal intellectual property to oblivious employees who don't know they are putting data at risk. Companies are beginning to realize, however, that solid data protection protocols can actually serve as business enabler.*"

This paper is trying to offer a nowadays solution for a voting system that is based on blockchain. This will allow the voters to remain anonymous, but at the same time, the verification of a candidate votes to be possible.

Another thing that concerns nowadays society is the fact that the votes in elections could be tampered. This will not be possible since a transaction represented by a vote will be verified in order to be legitimate. Also, another advantage of using the blockchain technology is that the counting is accomplished in a public way and that both the voter, the time and the candidate id are recorded at the same time.

Motivation

Blockchain represents a revolutionary concept that innovated and amazed nowadays transactions and not only. It is said that this represents "the backbone of a new type of Internet".

"The blockchain is an incorruptible digital ledger of economic transactions that can be programmed to record not just financial transactions, but virtually everything of value", say Don & Alex Tapscott, authors of Blockchain Revolution (Rosic, 2016).

As quoted above, Blockchain, must be considered as a distributed database that keeps different kind of information that is updated regularly having an innovative advantage of not being controlled by a single entity and having more than one single point of failure.

Since the beginning of politics, *a trustworthy voting system* was desired by people to be sure that they can rely on it without having their votes tampered. Using blockchain, the need of the central authority to store and count all the votes will vanish. Hence, using this technology in a voting system, all the transactions (containing votes) in a distributed chain will be visible to anyone for a public verification of a candidate or political party, while keeping the voters identities secret.

In this project I will exemplify by creating an application based on Blockchain technology and on a few of the Bitcoin's components that a secured voting system can be achieved is a blink of an eye.

This application's aim is to create a publicly verifiable voting system by allowing the maintenance of the anonymity of the voters and prevention of the fraudulent votes. This will be possible by using *blockchain*, the distributed ledger and the concept of cryptocurrency. Using this concept, the voting system will use the same elements of a transactional application using blockchain. A concept used from bitcoind is the wallet which in this application is virtual representation of a ballot for the voter and of a voting ballot for the candidate. The main personas using this application are candidates and voters and each of them has their own wallet. The main difference from the basic bitcoin system is that here, only the voter has the possibility of voting (such as transferring virtual coins) and the candidate keeps the wallet only to receive and store the votes, without the possibility of transferring its received votes.

It has been thought that having a voting system that could keep the voter's identity secret and the data immutable is almost impossible. Now it is about time to prove the opposite because with the information stored on the voters wallets, each vote can be verified and unvalidated if it's proven that the vote has been tampered.

In conclusion, a Blockchain based voting system can gain much more trust and appreciation rather than any human supervised process. Just as Blockchain has been used in financial domain and people started trusting it, in the same way people will demand this approach over the classic voting process.

Chapter I - State of the art

Notions

1. What is Blockchain?

"A blockchain is a digitized, decentralized, public ledger of all cryptocurrency transactions. Constantly growing as 'completed' blocks (the most recent transactions) are recorded and added to it in chronological order, it allows market participants to keep track of digital currency transactions without central recordkeeping. Each node (a computer connected to the network) gets a copy of the blockchain, which is downloaded automatically. Originally developed as the accounting method for the virtual currency Bitcoin, blockchains – which use what's known as distributed ledger technology (DLT) – are appearing in a variety of commercial applications today. Currently, the technology is primarily used to verify transactions, within digital currencies though it is possible to digitize, code and insert practically any document into the blockchain"[1].

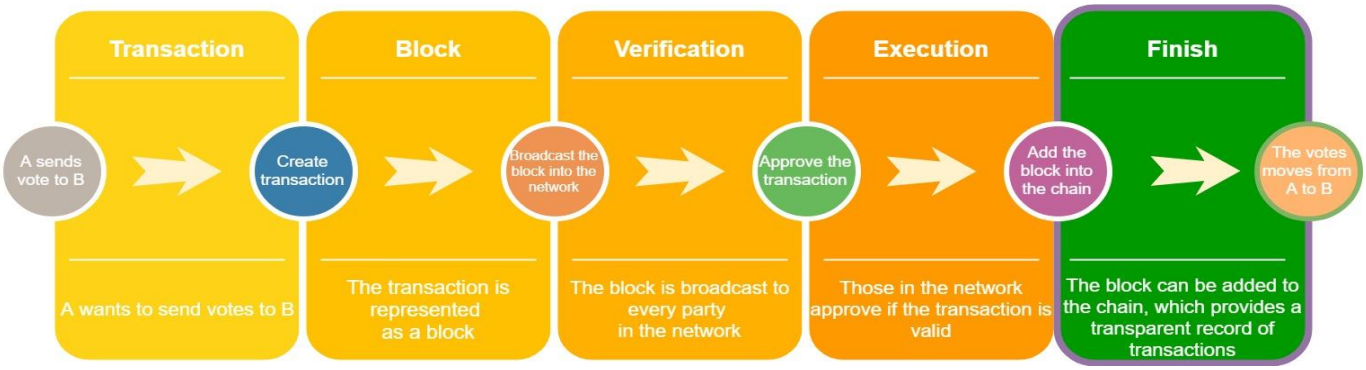


Figure 1: Blockchain usage in cryptocurrency

A blockchain is made of a list of blocks. It starts with a single block called the *Genesis Block.* Each block will store the following information: *index, timestamp, hash of the block, the previous block hash, data and nonce.*

The index represents the position of the current block into the chain, therefore the genesis block will have the index 0.

[1] "Blockchain" https://www.investopedia.com/terms/b/blockchain.asp

The timestamp represents a record of then the block has been created and it is really important because it helps to keep the blockchain in order.

The hash is an unique numeric value that identifies the data or the "digital signature". This number has a set of properties that must be applied in order to be a valid value such as: a fixed length, the same data results for the same hash, different data results for different hash, to be easy to compute, to be impossible to convert the hash into data and the smallest change into the data to be able to cause a big change into the hash. A valid hash is considered a hash to meet a certain requirement.

The leading zeros in from of a hash number represents the difficulty.

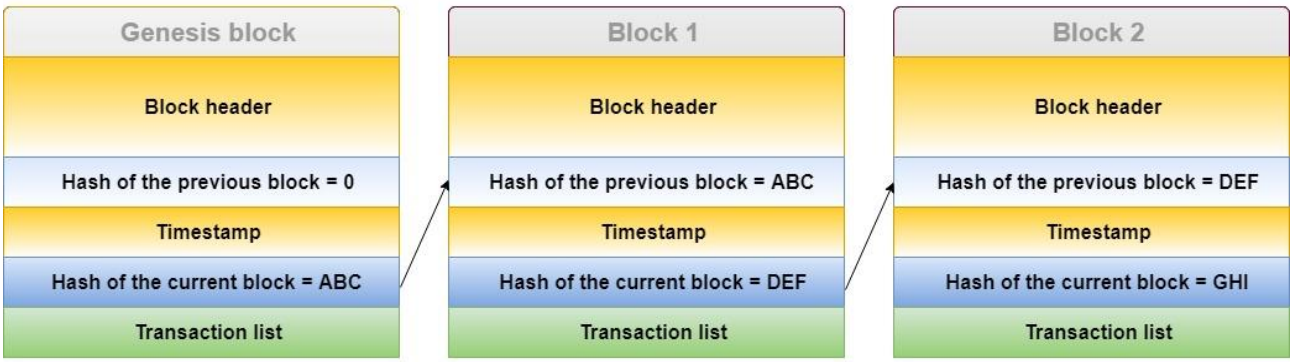


Figure 2: Blocks structure in the chain

As mentioned earlier, the hash function in blockchain is applied on each block, more precisely, it is taking the block data as input and returns an unique hash. Hence the hash is an unique representation of the entire block, the data mentioned earlier is a combination of the timestamp, index, block data, previous hash and the nonce. The data stored in the block in this application would be represented by the vote transactions such as the voter information and the actual vote.

Data mutation is an effect that affects the hash generation. Since the hash function takes the data as input, modifying its content will cause the modification of the hash. A change made into the hash function will cause a mutation into all the previous computed blocks because all the hashes are computed based on the previous hash, therefore subsequent hashed will change leading to an avalanche invalidation of blocks. Mining represents the process of finding a valid hash.

The nonce is a number used to find a valid hash. A lot of processing power is used to find a valid hash since the nonce iterates from 0 until the valid hash is found. An increased

difficulty level will lead to a decreased number of valid hashes, hence having less valid hashes it takes more power to find a valid one.

To add a new block into the chain a few vital conditions must be met: the added block index must be one greater than the last block index, the previous hash must be equal to the previous block hash, the block hash meets the difficulty requirement and that the block hash is correctly calculated. They are also called *the consensus protocol* and they are important because in order for ledgers to be useful, they must create an unchangeable timeline of events, which must be agreed by all nodes in the network so they can agree on the current state of the database. Since other peers on the internet will be adding blocks into the ledger, they must be also validated.

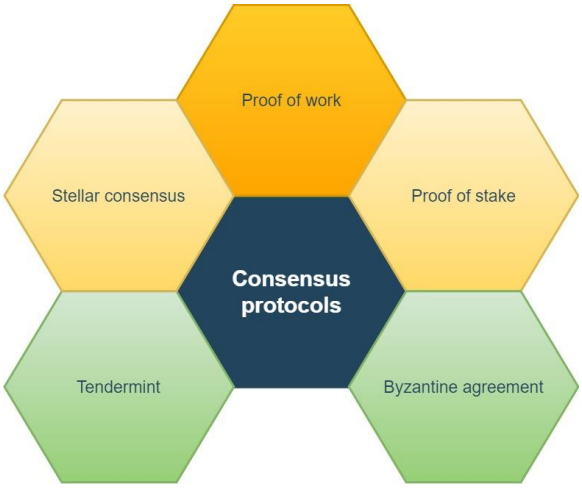


Figure 3: Consensus protocols types

Since the computation of hashes and the validation for newly added blocks is heavily power consuming, an entire global network of computers are needed to work together to keep the blockchain secured and consistent.

Each peer from the network can have one of the three states available at a moment in time: currently active, connected and disconnected. Peers communication is important since they ask one another for the status of each blockchain to determine who has the latest version of blockchain.

Peer communication works as it follows: first, add a peer and connect with it. The peer connected to is asked for the latest block in the chain. If it has a newer added block which is missing from the current peer, the latest block is sent.

When received, the block is verified to have the latest hash equal to the previous block hash. If the verification fails then the index of the peers block index is compared to the latest block. If not, nothing will happen, otherwise the peer is asked for the entire chain.

If the received chain from the peer is longer and is valid, than the current chain will be replaced and the latest block will be broadcasted to the connected peers. This is called *the longest chain rule*, because the longer valid chain takes precedence over a valid shorter chain.

If the equality of the previous hash with the latest block hash is met and the peer's block is valid than the peer's block is appended to the ledger and the latest block is broadcasted.

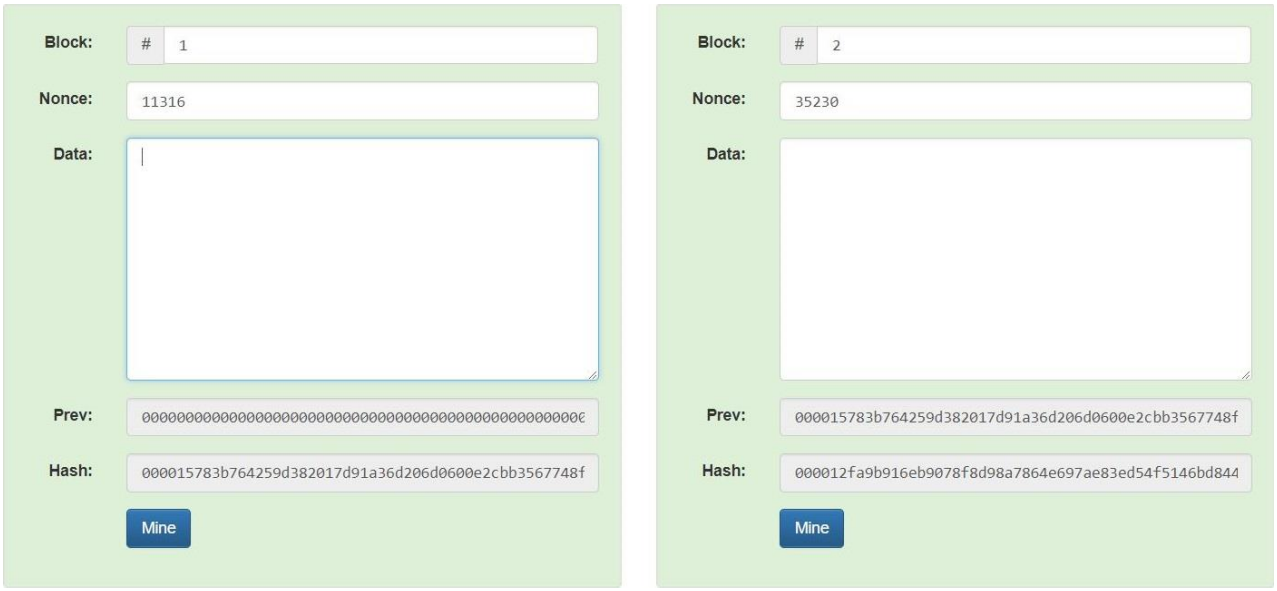


Figure 4: Blocks stored on blockchain

A property of the blockchain is ***immutability***. This means that is a block is moved, the block and the subsequent blocks will become invalid, therefore they will be rejected by the peers on the network. Earlier blocks are harder to hack because there are more subsequent invalid blocks to remine *(see figures 4 and 5)*.

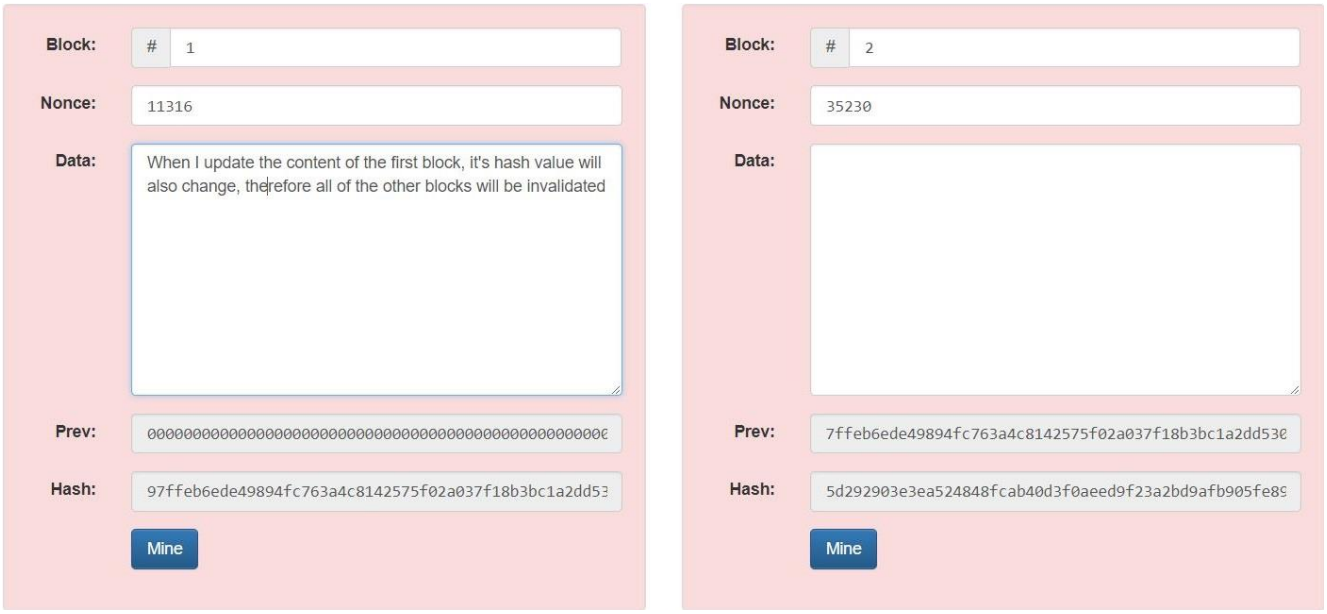


Figure 5: Immutable Blocks behaviour

An important attack to be mentioned is the 51% attack. This can be described as it follows: if a participant has more than 51% of the network's mining hashrate or computing power, he could outmine the network and hack the blockchain by preventing transactions from gaining confirmation and by that allowing to stop payments between the users. When there's more miners in the network, the processing power becomes more distributed and no one has majority power. This leads to a more secure blockchain.

2. What are ledgers?

Ledgers are a representation of transactions that exist as old as the hills since there has always been a need of keeping the track of distributed information, only that there was only a representation on paper or even on stones.

Digitalised transactions became the best way of storing data from the end of the 20th century, but there was not such an innovation since they were only of copy of what people were doing in the past.

3. What is a Distributed Ledger Technology (DLT)

Distributed Ledger Technology represents a system capable of keeping the track of all the transactions and their metadata recorded from multiple places at the same time since the beginning. Unlike the regular databases, this distributed ledgers are independent and do not need a central authority (data storage).

This concept is based on nodes that maintain the veracity of the data and that bring the innovative concept over the basic record-keeping that was used before and that is the cryptography.

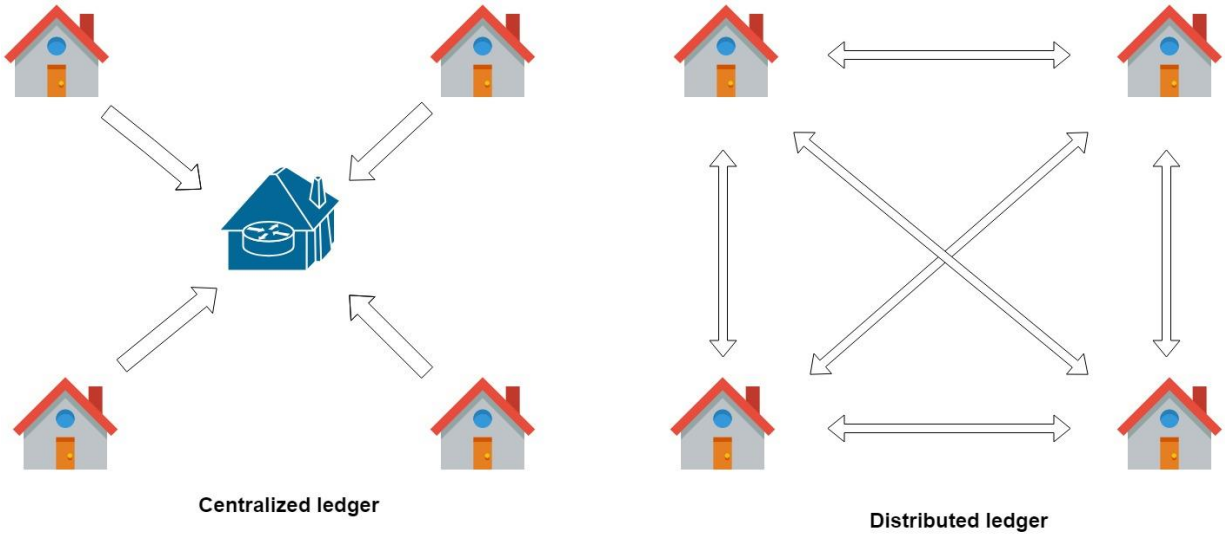


Figure 6: Centralized vs Distributed Ledgers

Blockchain is an instance of a Distributed Ledger Technology that chain blocks of information together, ensuring the security of the data and its veracity by broadcasting data from one block to another.

These distributed ledger systems are of a real importance since they cut out the existence of the ancient central authority that kept track of the transactions, passing now the security role to the blocks which significantly reduces the costs and more important, add a safer layer of security.

Moreover, the security is at it's high because each node in the network holds the history of the transactions, thereby it is harder to be attacked.

4. What are Smart Contracts and how can they be used in voter privacy?

A *Smart Contract* is a software that stores the rules for negotiating the terms of a contract and automatically verifies the contract and then executes the agreed terms. Blockchain coupled with smart contracts technologies removes the reliance on central systems between transacting parties. Untrusted parties can transact directly with each other using smart contracts.

The word "Contract" has a lot of luggage that might send the reader to another conception about it. A better understanding of this work in this paper might be "a thing that is executed automatically".

An analogy for the contracts used with blockchain would be the ticket machine in a station that says: "Pay the required price, follow these steps and you will get the ticket". Therefore, a contract is actually a piece of code both human and machine readable that makes machines accomplish the human needs.

"Keep in mind, we've had both computation and execution before. But never one that was finalized in a neutral, provable, trustable way on (digital) stone."[2]

When it comes about using bigger amounts of money through this kind of machines it becomes harder because our trust reduced. Here it comes the trust issues that smart contracts are trying to solve, since in blockchain there is no third party involved anymore.

"*Setting something on a blockchain is like setting something in stone. It makes trust easier*."

Smart contracts are stored on the blockchain which all parties have a copy of. They can execute agreed stored process when are triggered by an agreed event. For example a smart contract for a voting system may be consisted from: when the voter identity is confirmed, its vote may be counted.

All the contract transactions are stored in chronological order on the chain for the future access along with the complete audit of events.

[2] "Smart contracts for dummies…" https://medium.freecodecamp.org/smart-contracts-for-dummies-a1ba1e0b9575

If any party tries to change a contract(a transaction) the others notice that and prevent it and if it fails, no data or integrity loss will occur and the system will continue to work in its normal range.

5. What does mining stands for?

"Bitcoin mining is the process by which transactions are verified and added to the public ledger, known as the blockchain and also the means through which new bitcoin are released.

Anyone with access to the internet and suitable hardware can participate in mining. The mining process involves compiling recent transactions into blocks and trying to solve a computationally difficult puzzle. The participant who first solves the puzzle gets to place the next block on the block chain and claim the rewards.

The rewards, which incentivize mining, are both the transaction fees associated with the transactions compiled in the block as well as newly released bitcoin."[3]

The mining process is a really powerful operation which involves a great amount of computational resources. Even if a lot of computational power is involved in mining, the block discovery won't become easier since once in a while, the difficulty of resolving the puzzle grows, maintaining the mining hard. If the opposite happens, again, the difficulty of the mining will downgrade in order to make mining easier.

[3] "Bitcoin mining" https://www.investopedia.com/terms/b/bitcoin-mining.asp

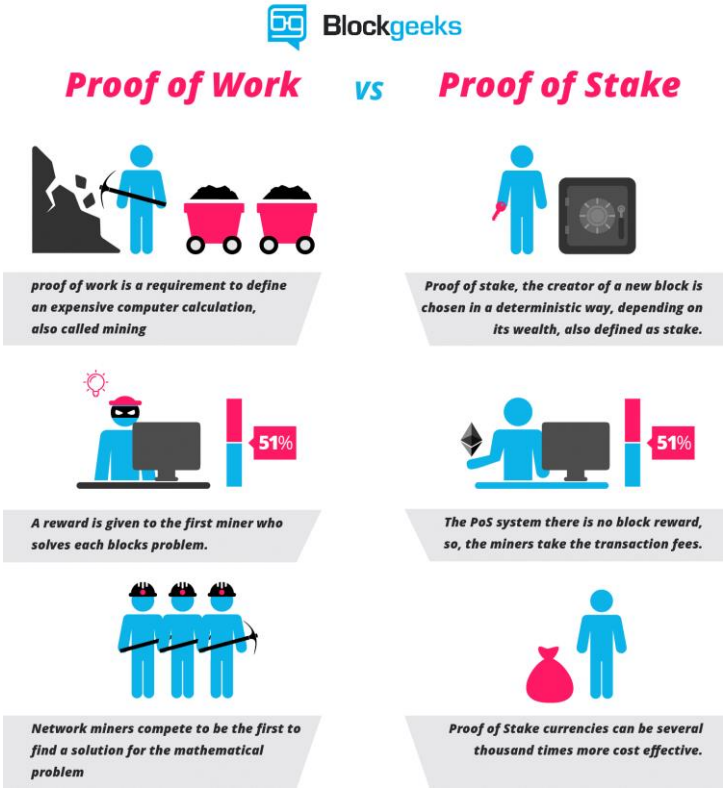


Figure 7: Centralized vs Distributed Ledgers[4]

Even if it's hard to believe, in the early days of Bitcoin, the mining was made with the help of the CPU's from normal computers. Once the Bitcoin became famous, the Graphic cards gained popularity too since they were even better at mining than the CPU's. Of course it didn't take much longer until some special hardware has appeared in order to make mining a lot more easier. The hardware is called Application-Specific Integrated Circuit and it was specially made for Bitcoin mining.

Solving a puzzle consists in applying a mathematical formula over the data stored in a block from the ledger, turning it into a hash of the block. An interesting fact about the hashes is that they are not that hard to be created for a piece of data, but the process of turning back a hash into data is almost impossible without knowing how it was produced. Miners don't just use the data from the current block to compute a hash, but the previous block hash is also required. This is how miners solve a block puzzle.

[4] "Proof of Work vs Proof of Stake: Basic Mining Guide" https://blockgeeks.com/

6. What does a transaction do?

A transaction is an event which in this application will carry information such as: the public key of the voter, the public key of the candidate, the number of votes to be transferred, a list of inputs which refers to the previous transactions showing that the voter has "funds" to spend.

Also, a list of outputs that show the amount received in the transaction will be used as input in future transactions and the last but not the least important is the cryptographic signature, which proves that the owner of the funds has created the transaction and that the data has not been changed (eg: preventing from changing the amount of votes sent).

Signatures in transactions are really important as they provide two important features in the process such as: allowing the voter to use its votes and the second one, preventing from changing the transaction data before another block is added on the ledger.

Figure 8: Bitcoin transaction process

An example from Bitcoin of a basic transaction could be like this: A wants to send 10 bitcoin to B. In order for A to send the coins to B, he needs to know the address of B. In order to create the address, the public key is hashed into a *PubKeyHash* and then is converted into a *base58check* format.

Once you have a private key, it is easy to generate the public key and then the PubKeyHash, but it will not work the other way around, but there's no problem in switching between the Bitcoin address and PubKeyHash.

Another step in creating the transaction with the 10 bitcoin, A has to create a *transaction output* using B's address. The *transaction output represents an index* to help the network find this transaction when the owner of the provided address, that is B, wants to spend the bitcoin he owns.

Once the transaction has been validated and mined by the miners, B's wallet will "contain" 10 bitcoin. It will not actually contain the amount he has, but instead, *B's wallet will track the outputs that correspond to the address*. These outputs are also called *UTOX*'s (unspent transaction outputs) and they are idle until someone will unlock and use them as inputs for another transactions.

Bitcoin is not an account base system but is more a network in which users hold wallets with private keys to unlock the UTOX's.

Now C, another member of the network has joined and B wants to send its 10 bitcoin from the UTOX to C. In order for B to send the amount of bitcoin, he must create a transaction that contains an input and an output.

The input is represented by the UTOX's that belong to B corresponding to the PubKey. Next, a signature must be provided to sign the transaction along with B's public key. The output consists of a new index to locate the transaction, the amount of bitcoins and a new PubKey script with the address provided by C.

7. Virtual wallet vs. crypto wallet

The *virtual wallet* concept was introduced before the concept of cryptocurrency and it was and is still being used to *store the credit card information* so that any user that is using it is free of carrying his wallet full of cards.

With the introduction of the worldwide known Bitcoin, the virtual wallet was replaced here by the *crypto wallet* that upgraded both the purpose and functionality of the old one by offering an entity in which the user can use *virtual coins*.

There is a big misconception about the way the coins are used in this technology, because *a crypto wallet doesn't actually store coins*. There is actually *no currency stored in the Bitcoin*.

Everything that exists and on which this system is based are the *transactions stored on blockchain.* There is still something that crypto wallets store, but that is *the public and the private key of the user* which have a big significance in signing the transactions. In order to spend the coins, the private key stored in the wallet should be the same as the public key as the currency is assigned to.

This types of wallets have different security levels depending on which model is used from: desktop, mobile, online, paper and hardware and the service provider.

It is proven that an *offline version is incomparably less riskier* than a web server which is vulnerable and prone to hacking attacks.

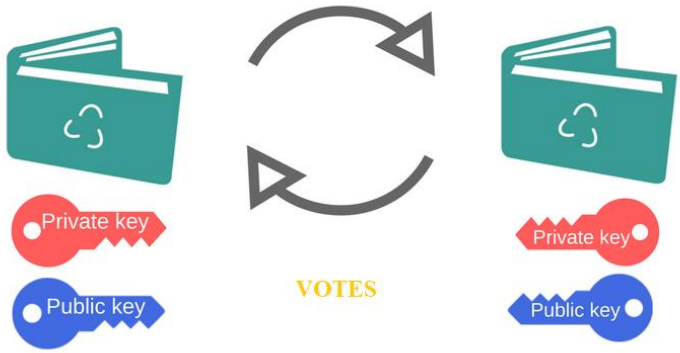


Figure 9: Crypto wallets process

"If public and private keys match, the balance in your digital wallet will increase and the senders will decrease accordingly... A cryptocurrency wallet is a software program that stores private and public keys and interacts with various blockchain to enable users to send and receive digital currency and monitor their balance. If you want to use Bitcoin or any other cryptocurrency, you will need to have a digital wallet."[5]

The same approach has been used for this application that is using blockchain for storing transactions, in which a user, such as a voter, uses its votes by sending them(voting) to a candidate.

[5] "Cryptocurrency Wallet Guide" https://blockgeeks.com/guides/cryptocurrency-wallet-guide/

8. Elliptic Curve Digital Signature Algorithm

The Elliptic Curve Digital Signature Algorithm is the cryptography behind the Bitcoin wallets.

*"It consists of combining the math behind finite fields and elliptic curves to create **one way equations**, meaning you can choose your private key (some number) and easily calculate your public key (some other number). However, I can't take my public key (or anybody else's for that matter) and easily calculate their private key. In fact, for Bitcoin it would take trillions of computers trillions of years of continuous guessing of different private keys to figure out which one creates a given public key."*[6]

Finite fields represents a finite set numbers while the elliptic curves are groups. Groups are defined in the mathematics field if they have some certain properties such as: closure, associativity, an identity element and an inverse for each element.

When finite fields and elliptic curves are combined there starts the cryptography. In Bitcoin, we use some specific parameters to secure the transactions such as:

- ***prime modulo***, which is a really big number represented in hexadecimal as *FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F* that specifies the size of the finite field;
- ***the elliptic curve:*** with a=0 and b = 7 (where a and b represent the coefficients of the elliptic curve equation);
- ***base point P in hexadecimal;***
- ***order in hexadecimal:*** FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141. This numbers are chosen by a cryptography standard run by the organisation Standards for Efficient Cryptography Group (SEC).

Therefore, the Bitcoin's private key can be computed as it follows using scalar multiplication: ***public key = private key * base point P.***

"ECDSA works on the hash of the message, rather than on the message itself. The choice of the hash function is up to us, but it should be obvious that a cryptographically-secure hash function should be chosen. The hash of the message ought to be truncated so that

[6] "How does ECDSA work in Bitcoin" https://medium.com/@blairlmarshall/how-does-ecdsa-work-in-bitcoin-7819d201a3ec

the bit length of the hash is the same as the bit length of nn (the order of the subgroup). The truncated hash is an integer and will be denoted as z."[7]

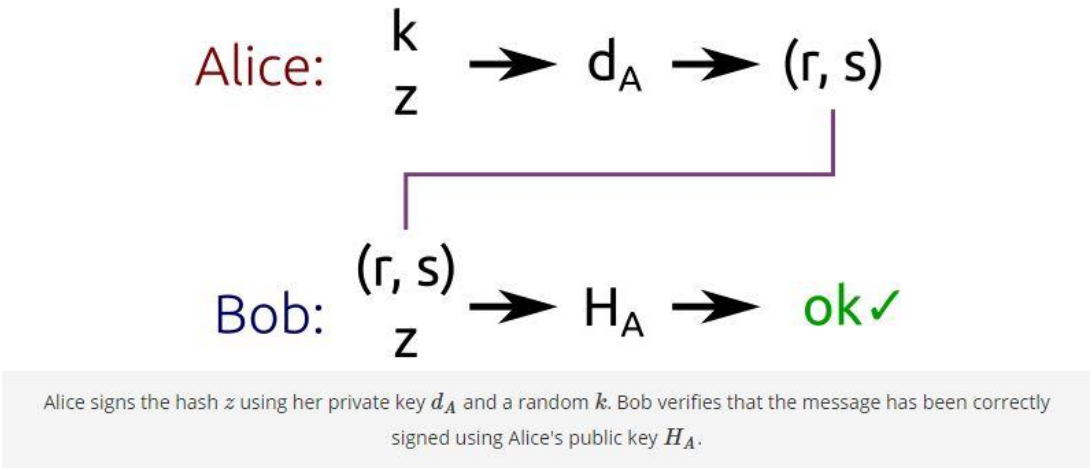


Figure 10: Signing and checking transactions

In Bitcoin the *private key* is a random secret generated number known only by the one who generated.The only person who can spend the funds from the ledger is the one that has the same private key as the funds.

The *public key* is a key that corresponds to the private one and is not secret. It is used mainly to determine if a signature is valid without the need of the private key.

Both the public and the private keys provide the input needed to sign a transaction and to verify it. The signature is a number that proves that the signing operation was applied over the content. *The signature is generated from a hash and a private key*.

[7] "Elliptic Curve Cryptography: ECDH and ECDSA" http://andrea.corbellini.name/2015/05/30/elliptic-curve-cryptography-ecdh-and-ecdsa/

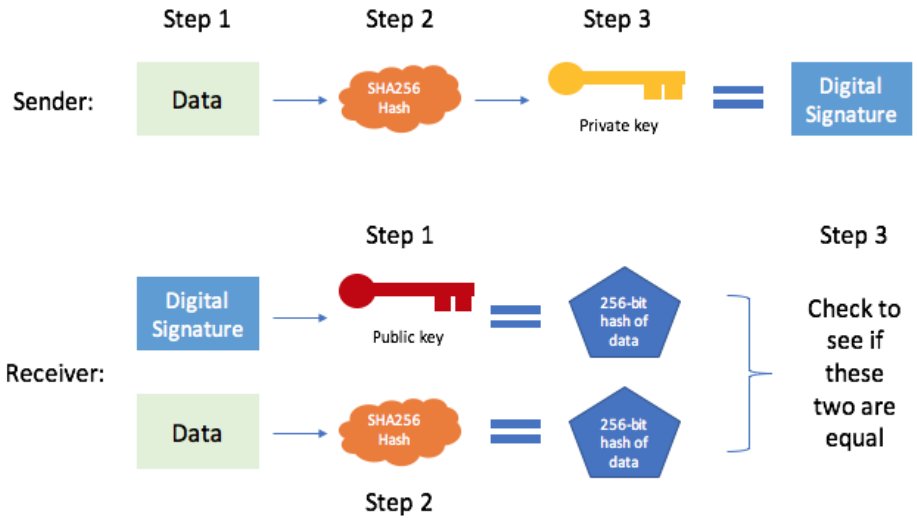


Figure 11: Digital signatures[8]

[8] "How does a bitcoin transaction actually work?" https://medium.com/@blairlmarshall/how-does-a-bitcoin-transaction-actually-work-1c44818c3996

9. Angular 5 and its usage in the application

"Angular is an all-encompassing JavaScript framework that is frequently used by developers all over the world for building web, desktop and mobile applications."[9]

This application is using the Angular framework to create a web interface for the Voting System using Blockchain concept.

Angular framework is a framework working on *MVVM pattern* and it contains elements such as *components, models and services*. Combined all together they form a single page web application. Every new page that seems to be created is just a new layer on top of a main component.

The application has 4 main components: *login page, admin page, candidate page* and *voter page*. In each of those there are presented a set of information and actions depending on the role of the user logged.

The login page requires the user to login using different information such as full name, its national personal number, password and the type of user.

The candidate page contains a profile tab in which the candidate can see its details and the number of votes received and another tab containing the live blockchain.

The voter page contains a profile tab, the tab with the love blockchain, a tab with the candidates from the campaign and their details and a tab in which he can vote.

[9] "Angular5 release" https://dzone.com/articles/angular-5-release-whats-new

Figure 12: Login page

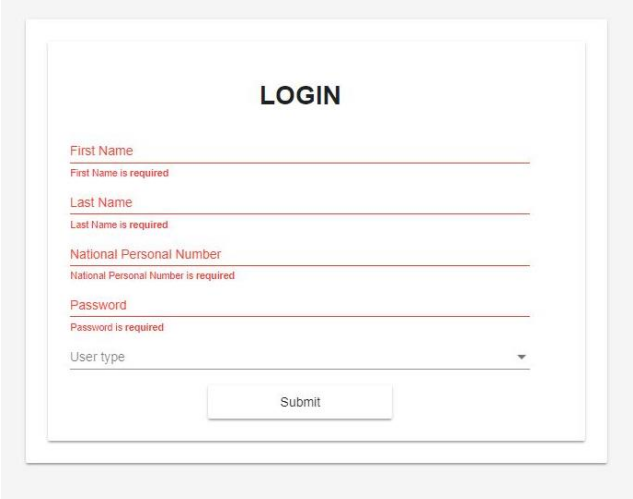
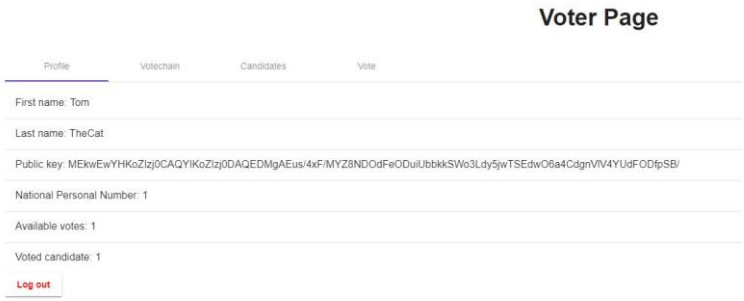


Figure 13: Voting page



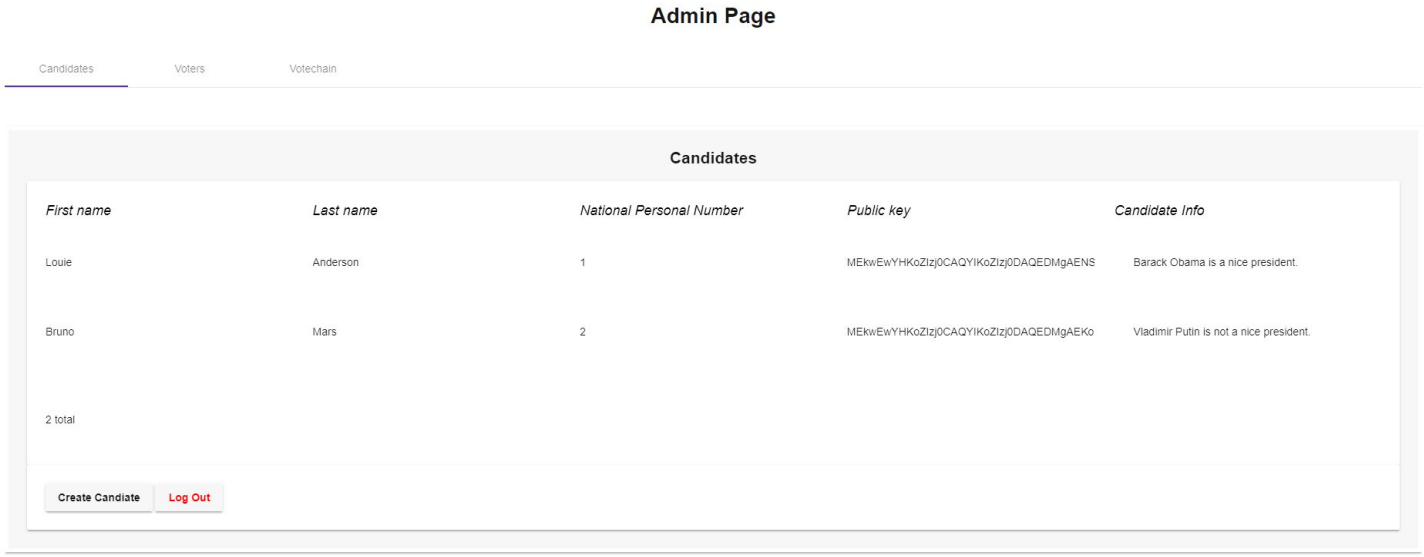


Figure 14: Admin page

Figure 15: Candidate page

Chapter II - Similar applications

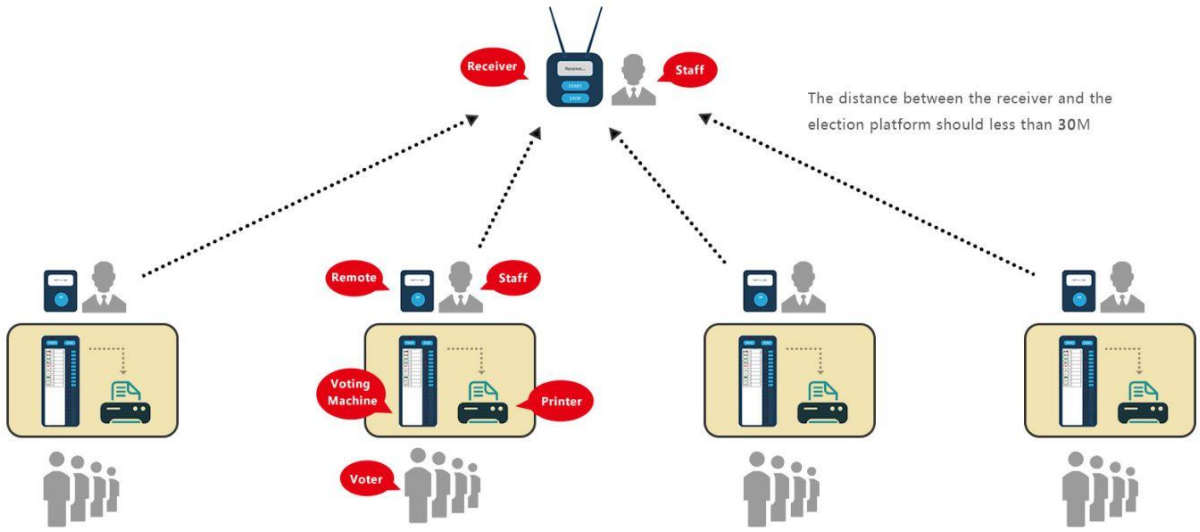
There is a large amount of applications that are trying to bring the best from the voting procedure and which promise to offer a secured solution. In this chapter a few of them are enumerated and presented.

1. SunVote

SunVote is a voting system that covers multiple cases from Interactive Voting System to Election Voting System. It brings with it hardware features such as a keypad, a remote, a receiver, a printer and EVM software.

The technology offers a simple operation solution that combines the old paper operation mode which is read via the provided software.The vote-board and the remote control are reusable while the receiver is collecting all the data into the polling station.

Some advantages of this system are represented by the encrypted data storage used in the receiver and by the high efficency of it, which says that one vote machine is able to help 120 voters complete the election in one hour.The electronic voting machine provides key



functions such as: single vote authorize, results display and results storage.

Figure 16: SunVote voting system

2. nVotes

This voting system is more interactive than the previous one and it brings more features on customizing the campaigns and ballots by adding images and other type of media in order to offer a personalized solution for candidates and a safe solution for the users.

It also offers a 24/7 day experience time in which voters can cast their votes just with a need of Internet connection. As a stage of the authentication it provides a system of personalized email or SMS to electorate with a different one time password.

As for the security of the voting system, the application provides ballots encrypted in the web browser using a key generated by multiple independent election authorities which ensure the privacy of the votes.

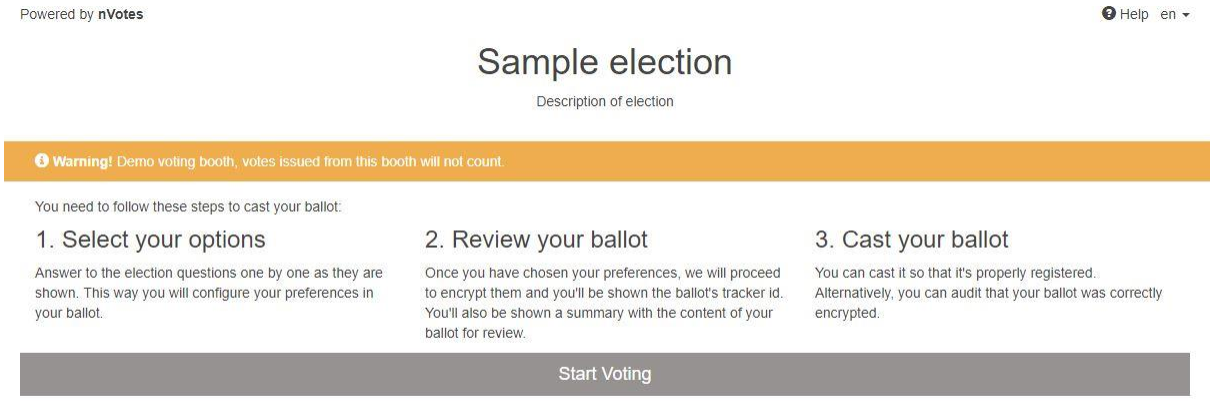


Figure 17: nVotes voting system

3. Follow My Vote

Follow My Vote is a secure and transparent online voting solution that uses Blockchain technology to keep the ballots safe.

Unlike the other systems presented, Follow My Vote cover both the lack of transparency in elections and the lack of security of election systems.Using Blockchain technology, they are gaining transparency.

Another key feature is that “at the voter’s request, there would even be a way to allow a voter to cast their vote online in an election and follow their vote into the ballot box to

ensure that their vote was safely and securely stored without being changed or altered in any way."[10]

Figure 18: Follow My Vote voting system

4. Conclusion

In conclusion, there are multiple voting solutions which offer different ways of securing both the process and the outcome of the elections but from all of them, the ones based on blockchain seem to be more trustworthy than the others by not having a third party involved in the process and also by keeping the data immutable.

[10] https://followmyvote.com/

Chapter III - Application Architecture

The architecture of an application is the foundation of what is built and it requires a certain amount of care because following it, the application's agility, scalability and reliability will come out. It comes to this when the application will need a sudden change and it will offer an enormous help if built correctly.

1. MVC Architecture

The application follows a MVC pattern that consists of three layers: Model, View and Controller. This pattern is mostly known for isolating the application's logic from the user interface and has support for the separation of concerns.

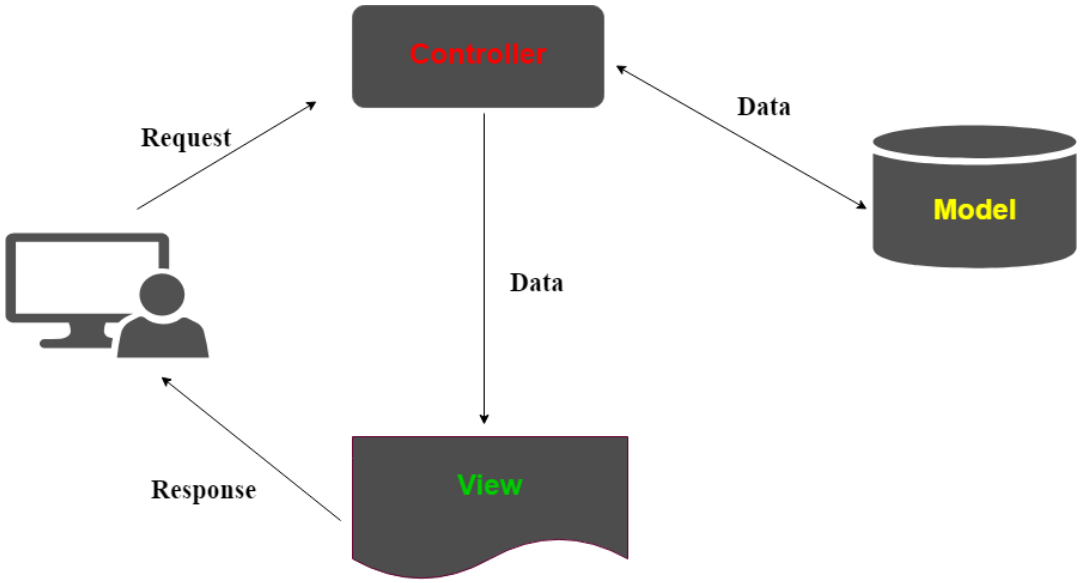


Figure 19: MVC

A summary of what it does is that the controller receives a request from a user interface and it uses the Model to prepare the data needed by the View. The view uses the data prepared by the controller to create a final response.

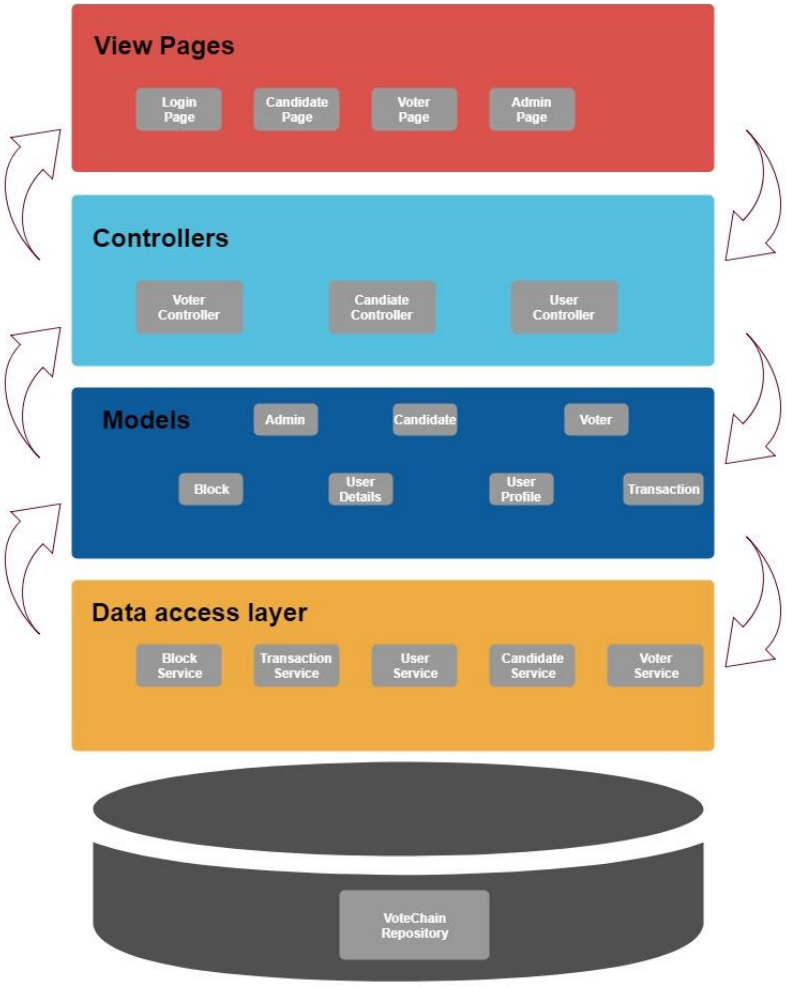


Figure 20: The application's MVC diagram

1. Model Layer

The model represents an object that has the role of defining and carrying the data structure. "If the state of this data changes, then the model will usually notify the view (so the display can change as needed) and sometimes the controller (if different logic is needed to control the updated view)"[11].An important model from the application is the *Block model* used to update the Votechain. It contains elements such as: current hash, previous hash, the current date-time and a list of the block's current transactions.

[11] "MVC Architecture" https://developer.mozilla.org/en-US/Apps/Fundamentals/Modern_web_app_architecture/MVC_architecture

```
public class Block
{
   // Current's block signature
   private String currentHash;
   // The previous block signature
   private String previousHash;
   // The transactions
   public ArrayList<Transaction> transactions = new ArrayList<>();
   // Current date
   private LocalDate dateTime;
}
```

Figure 21: The block model

2. Controller Layer

The controller contains all the logic and it updates both the model or the view depending on the request in response to input from the users of the application using it.

An example from the application would be the *User Controller* which handles the login part of the application. When the request of logging in in sent from the users of the application, one of the Admin, Candidate or Voter models will be updated.

Another example would be the *Voter Controlle*r which contains an endpoint that is taking over with the requests of voting and which uses the Voting Model POJO to prepare the actions and the requirements response needed.

```
@PostMapping ("/vote")
public ResponseEntity<Map<String, String>> voteCandidate(@RequestBody VotingModel
votingModel)
{
   boolean canVote = voterService.vote(votingModel.getVoterId(), votingModel.getCandidateId());
   if (canVote)
   {
      return new ResponseEntity<>(Collections.singletonMap("response", "OK"), HttpStatus.OK);
   }
   return new ResponseEntity<>(Collections.singletonMap("response", "BAD_REQUEST"),
BAD_REQUEST);}
```

Figure 21: Voter Controller -> vote request

3. View Layer

The view layer defines how the app's data should be displayed. The application has some POJO's defined in order to prepare what to be sent to the user interface in order to be displayed. One of the main views of the application is the login form which prepares the data to be sent to the controller in order to be verified.

```
<h1 class="example-h1" align="center">LOGIN</h1>
<mat-form-field class="example-full-width">
  <input matInput placeholder="First Name" [formControl]="formControl" [(ngModel)]="firstName"
         type="text">
  <mat-error *ngIf="formControl.hasError('required')">
    First Name is <strong>required</strong>
  </mat-error>
</mat-form-field>

<mat-form-field class="example-full-width">
  <input matInput placeholder="Last Name" [formControl]="formControl" [(ngModel)]="lastName"
         type="text">
  <mat-error *ngIf="formControl.hasError('required')">
    Last Name is <strong>required</strong>
  </mat-error>
</mat-form-field>

<mat-form-field class="example-full-width">
  <input matInput placeholder="National Personal Number" [formControl]="formControl"
         [(ngModel)]="nationalPersonalNumber"
         type="text">
  <mat-error *ngIf="formControl.hasError('required')">
    National Personal Number is <strong>required</strong>
  </mat-error>
</mat-form-field>

<mat-form-field class="example-full-width">
  <input matInput placeholder="Password" [formControl]="formControl" [(ngModel)]="password"
         type="password">
  <mat-error *ngIf="formControl.hasError('required')">
    Password is <strong>required</strong>
  </mat-error>
</mat-form-field>

<mat-form-field class="example-full-width">
  <mat-select name="userType" placeholder="User type" [(ngModel)]="type">
    <mat-option *ngFor="let userType of userTypes" [value]="userType.value">
      {{ userType.viewValue }}
    </mat-option>
  </mat-select>
</mat-form-field>
```

Figure 22: Login view

4. Conclusion

The MVC architecture pattern keeps a *separation between levels* of the application and it makes the *flow easier to read and understand* for the developer. It also has a great contribution to an easier integration with the frontend component which follows a MVVM (model view view model) pattern.

Chapter IV - Application overview

The application is a representation of the *Blockchain usage in secured voting*, by using the main concepts of this technology such as *immutable blocks, unchangeable timeline of the events* and *secured signed transactions.* Its purpose is to offer a *secured and reliable solution* for the electronic systems that still use a third party to keep the evidence of the voting ballots.

1. Application flow

This application is a multi-user one and it provides different functionalities for each type of logged user. The three types of users are: admin, candidate and voter.

The admin user has the capacity of viewing and adding both candidates and voters is the system and also can check out the live Votechain.

- It's worth to mention that this application will mainly *rely on a provided database with the voters* of a state/company/group and that the admin account should be used only for special occasions or at most for accepting requests of registration.

At this stage of the application, a small database of users is provided to exemplify the functionality of the application and the admin will have the power to add both candidates and voters to show that a hybrid can be build on top of what is existent.

The candidate user has smaller privileges and therefore he can only check his actual profile and the live votechain.

A few more functionalities that could bring an *improvement* would be to edit its presentation details and also to see a list of the other candidates participating and their status while the campaign is on the go. He is not able to vote through this account but he is allowed to have a voter account too, to vote any other candidate.

The voter user is the main persona using the application and it has several options when accessing the platform. He can check his profile that contains personal details, *the number of available votes and the voted candidate* if it's the case. He might also verify the available candidates and some details about them and the live votechain.

Some *future functionalities* to be added for the voter would be the possibility of *tracking the vote* to see when it has been added on the blockchain and the alternative of changing his mind and *vote for another candidate* before the election is over.

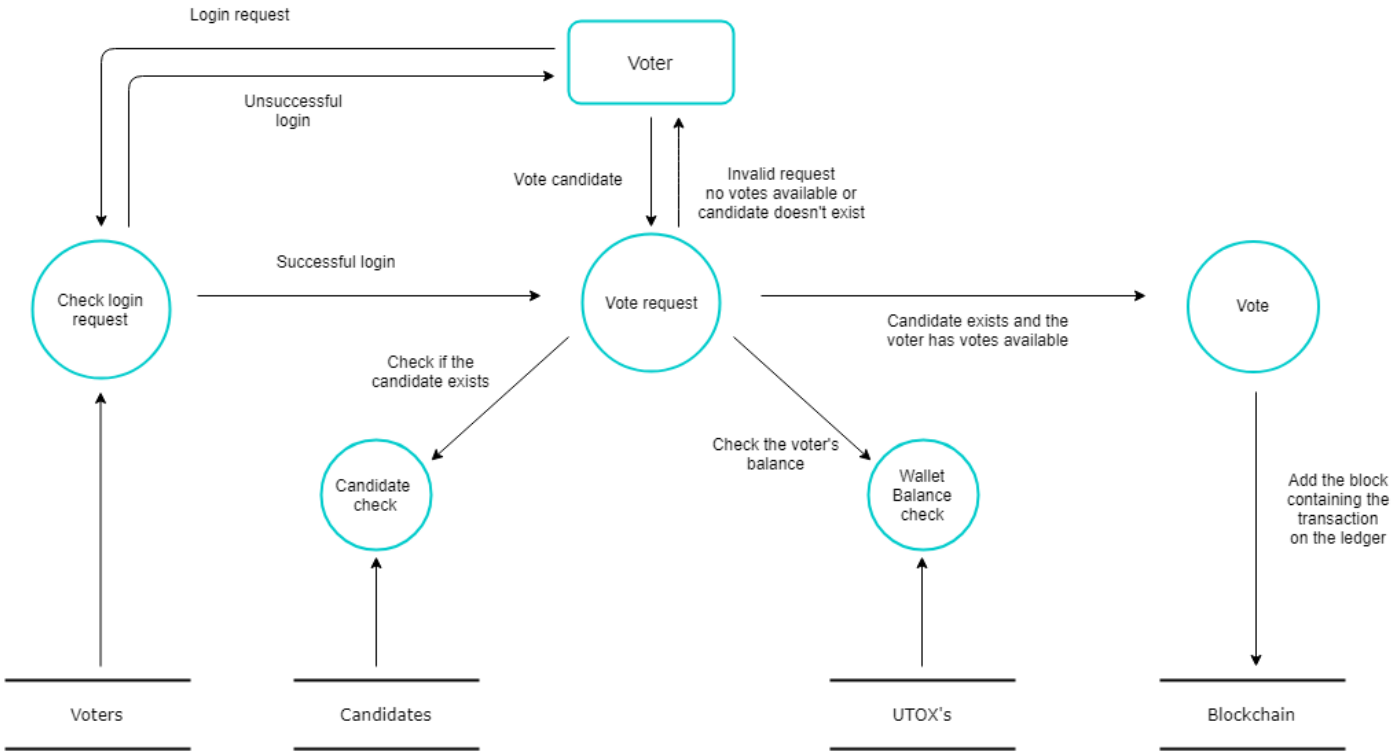


Figure 23: Application flow diagram for the voter user

The most important functionality of the application is the voting one but before explaining the process behind it a short review of the candidates and voters models is required.

As mentioned above, the application contains models which carry the data. In order for the voting process to take place there are five entities worth mentioning of.

The both the Voter and the Candidate Model are represented through an entity which encapsulates the vital elements to perform the actions such as:

- Personal information
    - *name*
    - *national personal number*
    - *password*
    - *id*
- A wallet.

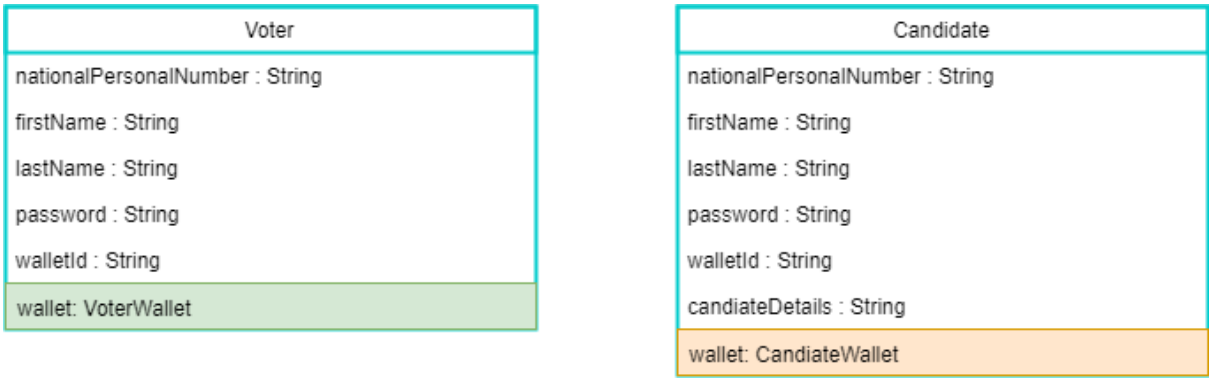


Figure 24: Voter and Candidate Models

The wallet is like a second identifier for a voter or a candidate and represents the address on which the transactions will point.

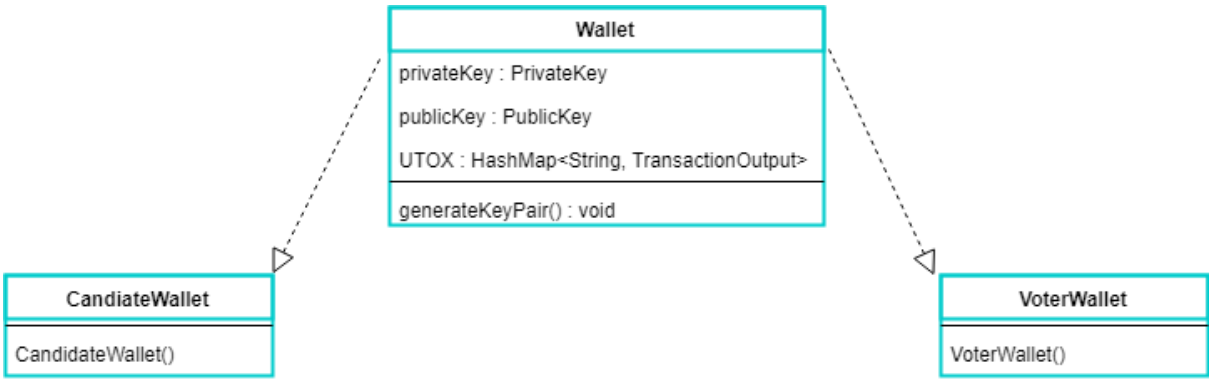


Figure 25: Wallet models

The private key is a random secret generated number and it allows the owner of the wallet to consume the votes contained by it. We will later see that the only person capable of spending the votes in this application is the voter.

The *public key* is a key that corresponds to the private one and is used mainly to determine if a signature is valid without the need of the private key.

Both the public and the private keys provide the input needed to sign a transaction and to verify it. The signature is a number that proves that the signing operation was applied over the content. *The signature is generated from a hash and a private key*.

The keys are generated using Bouncy Castle cryptography library and ECDSA described in the first chapter and the model is inspired from the blogpost mentioned in the footnotes section.

```
public void generateKeyPair()
{
  try {
     KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("ECDSA", "BC");

     // SHA1PRNG = SHA1 pseudo random number generator
     SecureRandom random = SecureRandom.getInstance("SHA1PRNG");

     // Type of elliptic curve used
     ECGenParameterSpec ecSpec = new ECGenParameterSpec("prime192v1");

     // Initialize the key generator and generate a KeyPair
     keyPairGenerator.initialize(ecSpec, random);
     KeyPair keyPair = keyPairGenerator.generateKeyPair();

     // Set the public and private keys from the keyPair
     privateKey = keyPair.getPrivate();
     publicKey = keyPair.getPublic();

  } catch (Exception e) {
     e.printStackTrace();
  }}
```

Figure 26: Key Pair Generator[12]

The *Block Model* is the one that is being stored and the one that also keeps inside it the list of transactions. In this application, each block will contain only a single transaction and that is the one which defines from which wallet has been transferred a vote and to which candidate wallet is sent (the block model is shown in Figure 21).

The *Transaction model* is the most complex class of all because it also contains some key functions for a transaction such as generating hashes, generating and verifying signatures and getting the inputs and outputs value.

[12] "Creating Your First Blockchain with Java" https://medium.com/programmers-blockchain/creating-your-first-blockchain-with-java-part-2-transactions-2cdac335e0ce

```
public class Transaction
{
  // The hash of the transaction
  public String transactionId;
  // Sender's public key(the address) -> verifies the integrity
  @JsonSerialize (using = KeySerializator.class)
  public transient PublicKey sender;
  // The receivers public key (the address) -> verifies the integrity
  @JsonSerialize(using = KeySerializator.class)
  public transient PublicKey recipient;
  // The transacted value
  public int value;

   /**The signature meant to prevent frauds in our wallet
   The signature performs 2 tasks:
   1. They allow the owner to spend the votes
   2. Prevent others from tampering with their submitted transaction before a new block
   is mined
    */
  public byte[] signature;

  @JsonIgnore
  public transient ArrayList<TransactionInput> inputs;
  @JsonIgnore
  public transient ArrayList<TransactionOutput> outputs = new ArrayList<>();

  // How many transactions have been generated.
  private int sequence;
 }
```

Figure 27: Transaction Model

Now as there is an overview of the components, the workflow of the voting process will be detailed. Since this is a model view controller application, most of the logic of the operation is stored in a service used then in the controller.

The voting process takes two main personas, the voter and the candidate. Each of them has a wallet in which the votes will be stored. As mentioned in the beginning of the thesis the votes, just as bitcoins are not actually stored.

The application repository contains besides a list of voters, candidates and the actual blockchain(being referred as votechain), a list of *Unspent Transaction Outputs*.

```
public class TransactionOutput{
  public String id;
  public PublicKey owner;
  public int value;
  public String parentTransactionId;
}
```

A transaction output is nothing more than an entity which holds information about previous transactions outputs like: the public key of the owner of the votes, the id of the parent transaction from which this was created, the amount of votes owned and the id of the transaction output. It also has boolean method which checks if the transaction belongs to a certain wallet (using its public key).

This is how the votes are actually stored, a list of transactions resume that contains the previous incomes received from other wallets.The vote request is started by a voter user and in order to transfer votes, he must send them to a certain address. That address is represented by a candidate's public key.

The first check is then made to see if the voter has available votes in his wallet. To do this the getBalance() method will iterate through the list of unspent transaction outputs and will return the sum of values of the transaction outputs that belong to the public key of the voter's wallet.

If the candidate has no transaction output entry in the list or if the sum of the incomes is smaller than the amount of votes he wants to transfer, the vote process in interrupted. Otherwise the next step is to see if the candidate actually exists. From the user interface, the voter will only pick the name of the candidate he wants to vote and in the background, a search is made to see if that candidate still exists and gets its public key for the next step.

The Voter Service class will then call a method that will create a new transaction with the given information such as the voter public key, the public key of the candidate and the number of votes to send which is by default 1.

The sendFunds() method from the VoterService will then generate a signature to sign the transaction and then the voter's unspent transaction outputs will be updated.

This is the main flow of the voting process in this application which might be improved by using a consensus protocol such as proof of work or proof of stake.

2. Conclusion

Blockchain is a powerful concept that revolutionized nowadays way of securing information stored by eliminating the third party factor and by relying on a structured, distributed and accessible database while not having just a single point of failure.

Bitcoin, the most known cryptocurrency system and the first one that has appeared on the market has gained humans trust in electronic transactions and moreover it offered a great ramp-up on the Blockchain concept.

The application which makes the subject of this thesis is following both the Blockchain and the Bitcoin concepts to ensure a secured experience when it comes to something so important as voting. The voters and candidates own a crypto wallet with which they can send and receive votes. This feature simulates the wallets from the Bitcoin systems. The main components of the wallet are represented by public and private key which have the purpose of

My personal contribution over this application consists in combining of both Blockchain and Bitcoin concepts and components in order to create a secured voting system. I consider that the application is a great starting point for any election type that can exist.

At the time when the thesis has been started, there were only a few Blockchain databases that were available on market but they were not free for usage and that created a limitation. The current version used a simple way of storing the data in lists that then have the possibility to be transferred into files and reused.

The basis provided by this application can be easily extended in order to provide a larger amount of functionalities like creating campaigns, adding multiple protocols to be used in campaigns as the administrator or the creator of the campaign wishes.

Also on the UI part there are many possibilities of improving the user experience by adding more interactive features such as the one described above, plus the option of editing a user's profile.

To sum up, the thesis *Blockchain usage in secured voting* represents a secure solution for a voting system that explores the *hash chains* used by Blockchain and the *crypto wallets* used by Bitcoin that also offers the possibility of expansion by using the latest databases and any of the current consensus protocols.

Bibliography

Blair, M. (2018) *How does ECDSA work in Bitcoin.* Medium https://medium.com/ @blairlmarshall/how-does-ecdsa-work-in-bitcoin-7819d201a3ec

Blair, M. (2018) *How does a bitcoin transaction actually work?* Medium https://medium.com/@blairlmarshall/how-does-a-bitcoin-transaction-actually-work-1c44818c3996

Bitcoin (2018) *Block Chain.* Bitcoin https://bitcoin.org/en/developer-guide#blocks-first

Bitcoin (2018) *Public Key Formats.* Bitcoin https://bitcoin.org/en/developer-guide#public- key-formats

Cole, B. (2016) *End-user security awareness first line of data protection defense*. TechTarget Network. 25 January 2016. https://searchcompliance.techtarget.com/video/End-user-security-awareness-first-line-of-data-protection-defense

Dmitry, B. (2017) *Blockchain Address 101: What Are Addresses on Blockchains?* Blockgeeks https://blockgeeks.com/guides/blockchain-address-101/

Dmitry, B. (2017) *Smart Contracts: The Blockchain Technology That Will Replace Lawyers* Blockgeeks https://blockgeeks.com/guides/smart-contracts/

Kass, (2018) *Creating Your First Blockchain with Java. Part 1.* Medium. https://medium.com/programmers-blockchain/create-simple-blockchain-java-tutorial-from-scratch-6eeed3cb03fa

Kass, (2018) *Creating Your First Blockchain with Java. Part 2—Transactions.* Medium. https://medium.com/programmers-blockchain/creating-your-first-blockchain-with-java- part-2-transactions-2cdac335e0ce

Rosic, A. (2016) *What is Blockchain Technology*? *A Step-by-Step Guide For Beginners*. Blockgeeks. https://blockgeeks.com/guides/what-is-blockchain-technology/

Rosic, A. (2016) *What is Bitcoin? A Step-By-Step Guide For Beginners*. Blockgeeks. https://blockgeeks.com/guides/what-is-blockchain-technology/

Rosic, A. (2016) *Blockchain Glossary: From A-Z.* Blockgeeks. https://blockgeeks.com/guides/blockchain-glossary-from-a-z/

Rosic, A. (2017) *Proof of Work vs Proof of Stake: Basic Mining Guide.* Blockgeeks https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/