Arab American University

Faculty of Graduate Studies

Palestinian E-Voting System Based on Blockchain Technology

By

Yurub Ishaq Ibrahim Awwad

Supervisor

Dr. Amjad Rattrout

This Thesis was Submitted in Partial Fulfillment of the Requirements for the Master's degree in Computer Science.

June / 2022

Thesis Approval

Palestinian E-Voting System Based on Blockchain Technology

By

Yurub Ishaq Ibrahim Awwad

This thesis was defended successfully on **25/June/2022** and approved by:

Committee Members	Signature
1. Supervisor: **Dr. Amjad Rattrout**	Dr Amjad RATTROUT .....
2. Internal Examiner: **Dr. Ahmad Ewais**	
3. External Examiner: **Dr. Rashid Jayousi**	

Declaration

I declare that this thesis entitled “Palestinian E-Voting System Based on Blockchain Technology” is my work and has been composed solely by myself and does not contain work from others researcher and has not been submitted for any other degree or scientific except the reference is made.

Name: Yurub Awwad

Signature: 

Date: The 10th of October 2022

Dedication

I dedicate this thesis to my Mother, my Wife, my Daughters, and my Son.

To my Brothers and Sisters.

To the soul of my Father who left sooner.

Acknowledgments

I would like to take this opportunity to express my deep regards to Dr. Amjad Rattrout for his advice, support, and time that he has spent reviewing my work. He provided me valuable suggestions that have had a significant impact and helped in overcoming many obstacles in writing this thesis in the best way.

I would also like to thank everyone who has supported me throughout my career, even if with a kind word.

Abstract

Integrity and transparency are two key metrics that must be achieved in any democratic voting. Without doubt, manual voting is a complicated, error-prone and costly process. The massive advances in computing and communication have the potential to have a significant positive impact on the voting process. It will take time for the voting process to be completely digitalized. Nevertheless, improvements in voting technology are being made gradually in many countries promising that a fully electronic voting system is applicable, With the emergence of Blockchain technology, the e-voting system has the potential to be constructed as a distributed system. This would make it possible to guarantee transparency and integrity, among other things.

The literature review was employed in this thesis technique to investigate the problems of developing a Blockchain-based E-Voting System. As a result, we constructed some Cryptography Algorithms-based functions to avoid any conflict between e-voting system features as part of our contribution. This conflict is mainly between the verifiability and the receipt-free features.

The thesis ultimate goals are achieved by implementing an e-voting system that has the basic key features as a result.

.

Table of Contents

Declaration ..... **II**

Dedication ..... **III**

Acknowledgments ..... **IV**

Abstract ..... **V**

List of Figures ..... **VIII**

List of Tables: ..... **IX**

List of Flowcharts ..... **X**

List of Algorithms ..... **XI**

List of Abbreviations ..... **XIII**

Introduction **14**

Literature Review **20**

Non-Blockchain Based E-Voting System 20
E-Voting Surveys 20
E-Voting Based on Blockchain 22
E-Voting in Arab Region 25

System Architecture and Methodology **28**

System Architecture 28
System Components 29
Methodology 31
System Layers 31
Public Key Infrastructure 33
The Blockchain Technology 34
Cryptography and System Functionality 35

Implementation **42**

System Actors 42
System Classes 44
Execution Stages 45
Registration Authority 49
The Election Authority 50

Voting Page 51
Results Page 52
Algorithms 54
The Smart Contract 60

Experiment Results and Discussion 63

Result Setup 63
Results 63
Blockchain Performance 67
Security Concerns 68

Conclusion and Future Works 71

Conclusion 71
Future Work 71

Bibliography 72

Appendices 76

الملخص 92

List of Figures

FIGURE 1:THE ARCHITECTURE OF PROPOSED SYSTEM. 28
FIGURE 2: IMPLEMENTATION LAYERS..... 32
FIGURE 3: SIMPLE PKI SYSTEM DIAGRAM 33
FIGURE 4: BLOCKCHAIN STRUCTURE (NAKAMOTO) 34
FIGURE 5: RING SIGNATURE (RIVEST AND TAUMAN)..... 38
FIGURE 6: VOTER USE CASES..... 42
FIGURE 7: REGISTRATION AUTHORITY USE CASES..... 43
FIGURE 8: ELECTION AUTHORITY USE CASES 43
FIGURE 9: VOTER CLASS 44
FIGURE 10: ELECTION AUTHORITY USER CLASS 45
FIGURE 11: REGISTRATION AUTHORITY CLASS 45
FIGURE 12: VOTING SEQUENCE DIAGRAM..... 47
FIGURE 13: REGISTRATION AUTHORITY PAGE 50
FIGURE 14: ELECTION AUTHORITY PAGE..... 50
FIGURE 15: LOGIN PAGE 51
FIGURE 16: VOTING PAGE 52
FIGURE 17: SELECTING PIN..... 52
FIGURE 18: RESULTS PAGE 53
FIGURE 19: VOTE VERIFY PAGE 54
FIGURE 20: LOADING PUBLIC KEYS FOR RING SIGNATURE..... 58
FIGURE 21: DEBUGGING VOTING PROCESS..... 64
FIGURE 22: COUNTING RESULTS..... 65
FIGURE 23: VERIFY PAGE 66
FIGURE 24: COMPARING TWO TRANSACTIONS 67
FIGURE 25: ACCOUNTS GENERATION AND TIME CONSUMPTION 68

List of Tables:

TABLE 1:PIN CODE PROBABILITY 39

List of Flowcharts

FLOWCHART 1: RSA FLOWCHART 36
FLOWCHART 2: VOTING PROCESS FLOWCHART 56
FLOWCHART 3: PREPARE BALLOT 57
FLOWCHART 4: SMART CONTRACT VOTE PROCEDURE 59
FLOWCHART 5: VOTE TALLY PROCESS 60

List of Algorithms

ALGORITHM 1: REGISTRATION AUTHORITY LOGIN..... 54
ALGORITHM 2: ELECTION AUTHORITY USER LOGIN..... 55
ALGORITHM 3: VOTER LOGIN ALGORITHM..... 55
ALGORITHM 4:SMART CONTRACT ALGORITHM..... 61

List of Appendices

APPENDIX A SMART CONTRACT USING SOLIDITY 76
APPENDIX B RING SIGNATURE USING C# 77
APPENDIX C JAVASCRIPT FUNCTIONS USING WEB3 86

List of Abbreviations

BC	Blockchain
ZNP	Zero Knowledge Proof
PoW	Proof of Work
PoS	Proof of Stake
PKI	Public Key Infrastructure
CPU	Central Processing Unit
EHR	Electronic Health Record
IoT	Internet of Things
CIA	Confidentiality, Integrity, Availability
MSISDN	Mobile Station Integrated Services Digital Network
DRE	Direct Recording E-Voting
ISO	International Standard Organization
SHA	Secure Hash Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
RAM	Random Access Memory
HTML	Hyper-Text Markup Language
CA	Certificate Authority
API	Advanced Programming Interface
JSON	JavaScript Object Notation
PIN	Private Identification Number
RSA	Rivest, Shamir, Adleman

Chapter One

Introduction

Introduction

We are witnessing massive advancements in computer and communication technologies. Shifting from manual to computerized is everywhere. The term for this shift is "digitalization of the business model" [1]. Cloud computing is also exploding in popularity [2]. The concept of converting a voting system to an online system, often known as the E-Voting System, which is an online system that allows voters to submit ballots (votes) in a democratic manner. The e-voting method is a step forward from remote electronic voting, which is likewise a step forward from postal voting. The history of postal voting dates back to the Roman Empire. [3].

Even the manual voting mechanism is a complicated system in general. In nature, it is also a distributed system. The necessity to computerize such a system arises from the requirement to accomplish a number of objectives. As an example, make the voting system more efficient, accurate, manageable, flexible, and accessible [4]. Also, to reduce paperwork. And contribute to a greener environment.

The e-voting system could be built on a comprehensive system that is centralized, or on a distributed system which is decentralized. As mentioned before the voting system is distributed by nature, we chose to build it on a distributed system. Distributed system also gives the e-voting system some properties to make the whole system more flexible. Which makes it transparent, scalable, and it avoids the single point of failure [5].

Because the distributed systems inherit their characteristics from the properties of complex systems, we can assume that our proposed model will be scalable, transparent, and decentralized. We can argue that our proposed system will also be self-organized with scale invariant distribution [6].

To build this distributed e-voting system and preserve the integrity of the voting process, we chose to build it using Blockchain technology. Blockchain Technology is the technology behind the Bitcoin, the famous Cryptocurrency system [7].

Blockchain has several attractive characteristics to be used in distributed systems such as it achieves and maintains integrity in these systems [8], that's why Blockchain is used in cryptocurrency, and this is the main reason to adapt Blockchain technology in this proposed e-voting system.

However, cryptocurrency is the first use of the Blockchain technology, but it is not the only use. Blockchain has been seen in many other fields like Internet of Things (IoT), decentralized messaging, and cloud storage systems, Electronic Health Records EHR systems [9].

There is no doubt, any E-Voting System has to meet at least the same requirements of the normal manual Voting System, and it should give its own added value. the Requirements of any E-Voting System which mentioned in [10]:

- Completeness: all the valid votes should be counted correctly.
- Soundness: no one can disrupt the voting process.
- Privacy: the voting is in secret, the voter but nobody else should know the ballot.
- Un-Reusable: the voter can only vote once.
- Eligibility: the authorized voters are the only allowed to vote.
- Fairness: Nobody can affect the voting process.
- Verifiable: no one can falsify the result of the voting process.

- Robustness: even if a cheat is happened the election result must reflect the actual result.
- Incoercibility: no one has to be forced to vote.
- Receipt-freeness: no one can obtain a receipt of the ballot.
- Mobility: Voting can be done from anywhere.
- Convenience: the voting system should allow the voters to submit their votes in one session with no need to special skills or equipment.

Another two points can be added which are inherited from the nature of the complex systems:

- Transparency: Voters should be able to verify that their votes were counted correctly.
- Availability: the voting system should be accessible anytime anywhere during the voting session.

The main objective of this study is to deliver an E-voting System key features based on Blockchain. But from the list of these features, we can see two features are conflicting with each other, the ability to verify the vote was counted correctly, and the receipt-freeness. Therefore, finding a solution for this conflict is the secondary objective of this study.

This thesis is divided into six chapters: an introduction that briefly describes the thesis problem question, a Literature review chapter that examines some related works in order to gain a better understanding of what an E-Voting System based on Blockchain is, a Methodology chapter that explains how the thesis will implement the system and its architecture which illustrates the structure of our implementation using the Unified Modeling Language., the Implementation chapter shows how to design the system using the tools available and provides some screen shots

of the interfaces, Experiment Results and discussion chapter that discusses the results, and a Conclusion chapter that conclude the thesis.

Chapter Two
Literature Review

Literature Review

The related works to E-Voting is reviewed in this chapter, some of this related works is directly related to the E-Voting system based on Blockchain, while others are related to Online Voting System that are not built on the Blockchain.

The literatures which are following are ordered by time of released, and grouped into four groups, the first group includes the articles about E-voting System which are not related to Blockchain, the second group includes articles summaries the e-voting system and surveys, while the third one is grouping the E-Voting Systems that are based on Blockchain. The last one is a review to related works about e-voting in the region.

Non-Blockchain Based E-Voting System

Many related works have been mentioned in [10] as it describes the requirements and properties of any e-voting system. Beside the requirements which were discussed in a previous section, they introduced their end to end verifiable e-voting system. The proposed system is a web application that uses steganography cryptography to secure the communication.

E-Voting Surveys

The authors in [11] showed that building a trusted e-voting system is facing a major risk. That risk is in general due to human element in the election. Based on their search many threats are rising from the programmers, operators, developers, vendors, and even the workers in elections campaigns. Most of these threats were discovered after admission or after source code reviews. Their study showed that the cost of such risk may rise up to $100 million at that time. Such number gives us how important is the risk assessment in e-voting systems.

In [12] which is a statistical research based on five elections done by different groups of volunteers. the statistics showed that how the institution that perform the election should be trustable, also the e-voting systems should also be trusted, which is not easy without many successful attempts and trials. The study used an e-voting system based on Public Key Infrastructure (PKI) provided by third party on cloud. The statistics data conducted using questioners before and after the election process for each voter.

Authors in [13] showed in their comprehensive review of e-voting systems which included 87 references, they showed that the lack of implementing e-voting systems is due to security issues or voters are seeing it as an insecure system. their study illustrated some basic requirements and should be requirements; Correctness, Privacy, Un-reusability, Eligibility, Robustness, Verifiable, Usability, beside some additional requirements like Fairness, Uncoerablity, Efficiency, Mobility, End to End Verifiable. The study confirmed that some requirements are making conflicts from technical perspective like Validation and receipt-free. The study also focused on the Usability requirement which is defined by the ISO-9241-11: “The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”. The roles of any E-voting system was illustrated in their study, roles like Voting Registration, Voting Process, mentioned the cryptographic tools and algorithms used in some present e-voting systems like; Mix-net, Homomorphic Encryption, Secret Sharing, Blind Signature, Anonymous Submission, Zero-Knowledge Proof, and Deniable Signature. Also they discussed some of current e-voting system found in their literature like DRE, Sensus, PunchScan, Scantegrity, Civitas. They concluded their study with that building usable, verifiable and transparent e-voting system is a challenge which needs more efforts in future researches.

This artwork in [14] is a survey that compares some of the existing e-voting system, the comparison is done by answering a few questions like whether the system uses hardware or software, does it provide mobility, is anonymity preserved? The results were shown in a table and most systems claimed to provide anonymity and voter verification. Therefore, we will discuss two systems [15] and [16] in later literatures to see whether this study is correct or not.

The study [17] discusses the e-voting system applications based on Blockchain, what should an e-voting system based on Blockchain cover, which type of Blockchain should be used. They also concluded that an e-voting system based on Blockchain will face some problems like privacy and speed.

E-Voting Based on Blockchain

In [18] the authors chose to build their e-voting system based on the Blockchain technology. They implemented permissioned Blockchain using Geth which is an open source Blockchain inherited from the Ethereum Blockchain. Their choice came after a compression of three different platforms Geth, Exonum, and Quorum. They preferred Geth because it is more friendly than others. It uses multiple development languages. Not like Exonum which uses its own Language named Rust. While Quorum doesn't provide that much of transactions per second. Their main objective is to build a distributed E-voting System on private network to avoid Single Point of Failure and minimize threats. Their main contribution was by the suggestion to use the Zero-Knowledge-Proof to separate the voter identity from the ballot. But that was only a suggestion which was not implemented in their work.

The work in [19] has discussed the disadvantages of Client Server topology in E-Voting systems, which should rely on a trusted central authority, beside that this Client Server topology

has some shorten in the CIA triangle of security. Beside this central authority is forming a single point of failure. Therefore, they proposed a Blockchain based e-voting system to get benefits from the inherited properties of Blockchain technology which are; data integrity, non-centralized, transparency, and the availability. Their system was built on Mobile Station International Subscriber Directory Number MSISDN. They concluded that their proposed system achieved the main objective which is to restrict the voter to vote only once based on the MSISDN. But it needs more work to be adapted in governmental voting system

In [20] a proposal to implement an e-voting system that is based on Blockchain. Their proposed system is using the Shamir's secret sharing to gain anonymity, while the Blockchain technology is to integrate management through the voting process. Their idea is to separate two Blockchains using the Shamir's secret sharing, one Blockchain is permissioned Blockchain to maintain the process of the elections, while the sidechain is to maintain the counting and result of the voting process. The proposed method does not provide a way to gain verifiability.

The study in [15] aims to design a database architecture to store the e-voting system data using Blockchain technology. They have introduced the Blockchain advantages pointing out the benefits from using such advantage in the e-voting recording system. they also finalized their study with charts showing up the relation between the number of nodes and the size of the storage needed. In addition to a graph that is showing the relation between the number of nodes and the calculation time for the whole ledger. Their study did not include other aspects like the anonymity and verifiability. But they have secured the data integrity using the Secure Hashing Algorithm SAH-256 and the digital signature of the ECDSA (Elliptic Curve Digital Signature Algorithm).

The work in literature [16] is meant to produce BroncoVote, which is a voting system that based on Ethereum blockchain, they used Pailler Homorphic encryption for privacy concern. They managed the system using smart contract. Their work contribution is the use of Pailler Homorphic encryption to preserve voting privacy, the idea of this encryption is that you can perform calculation on encrypted data and have same results as if this data was not encrypted. But the individual voter verifiability is mentioned in the future work of their study.

The authors of [21] have proposed a voting system architecture based on waves Blockchain, their main goal was to coast down the computational power used in other Blockchain technology. Which will make it possible to use this technology as a lite version that capable to be used on smartphones. Waves Blockchain main feature is to replace the proof of work with a proof of stake, the main difference between them is that, proof of work needs a lot a computational power by the miner to validate the transaction, while proof of stake does not take that much of power because the probability of choosing miner is based on the stake they have in the block. The architecture they proposed describes the voting process, also the zero knowledge proof and mix net algorithms are used for anonymity purpose.

The work in [22] The study is basically dividing the process of voting to fit two separated Blockchains, one is for the voters that records the eligible voters and whether they voted or not, and the other Blockchain records the actual voting process to achieve anonymity. They also divided the voters into groups of constituencies to protect the final result.

The proposed model [23] is to enhance the scalability, efficiency and latency, by combining public Blockchain with sharding model. They introduced a Proof of Credibility instead of Proof of work to reduce the power consumption, but still requires the Proof of stack that Ethereum based

on. Their model is working as two layers, the low layer is for devices which uses the proof of Credibility to increase efficiency, then as groups each node in the upper layer will use the Proof of Stack to verify the shaded groups. They also applied an evaluation of the proposed model using Ethereum to test the cost during the elections which showed a leaner relation between the number of voters and the election cost.

E-Voting in Arab Region

The work in [24] is a case study of United Arab Emirates UAE e-voting in 2006. The interesting thing about this voting is the shifting from the appointment by rulers' court to an Election using E-Voting system. The decision was made as modern concept because UAE is keep tracking the ICT world novation. They achieved this jump after a pre-voting stages to enrich the society of the knowledge they need to trust this E-Voting Process. They have utilized Awareness, Registration Help Desk, and training staff to carry out the process. This e-voting campaign had 74.4% of participants, 93.5 of them were satisfied about this election.

The conducted research of [25] is to find solutions for building an E-Voting system in countries with political conflicts, financial and demographical issues. Therefore, Palestine was the case study for this research. The research used Interviews to collect facts and evidence about the barriers of implementing E-Voting System in Palestine. Four barriers were identified as a finding which are; political will, capability, trust and the acceptance of such systems. The Political will comes from the fear of loss, the capability is built of three actors, the human resources and infrastructure which are not issues but the third one which is a financial matter is really an issue. Trust is identified in this research as the main barrier because any human made systems is never secured hundred percent. Therefore, there is a high risk that this system is vulnerable. While the acceptance is only achieved if the other three mentioned items are met.

Authors of [26] discussed the trust in e-voting system in Kingdom of Bahrain. Their work is an analysis of a questioner covered 453 participants. The study performed just after the third e-voting campaign in the kingdom. The main objectives of this study were to highlight the peoples' concerns about e-voting. Analytic of this survey focused on two points; one is the privacy issues while the other is the fearing fraud.

The literature review shows many ways to achieve the objectives of an E-Voting System, but none of them solved the conflict between vote verification and receipt-free. Also for our knowledge none used the combination of ring-signature with smart-contract on Ethereum Blockchain to implement an E-Voting System which we used.

Chapter Three System Architecture and Methodology

System Architecture and Methodology

System Architecture

The system architecture of our proposed system, can be described using the following design which shows the different components of the that are interacting together [27], to perform the functional operations of the system.

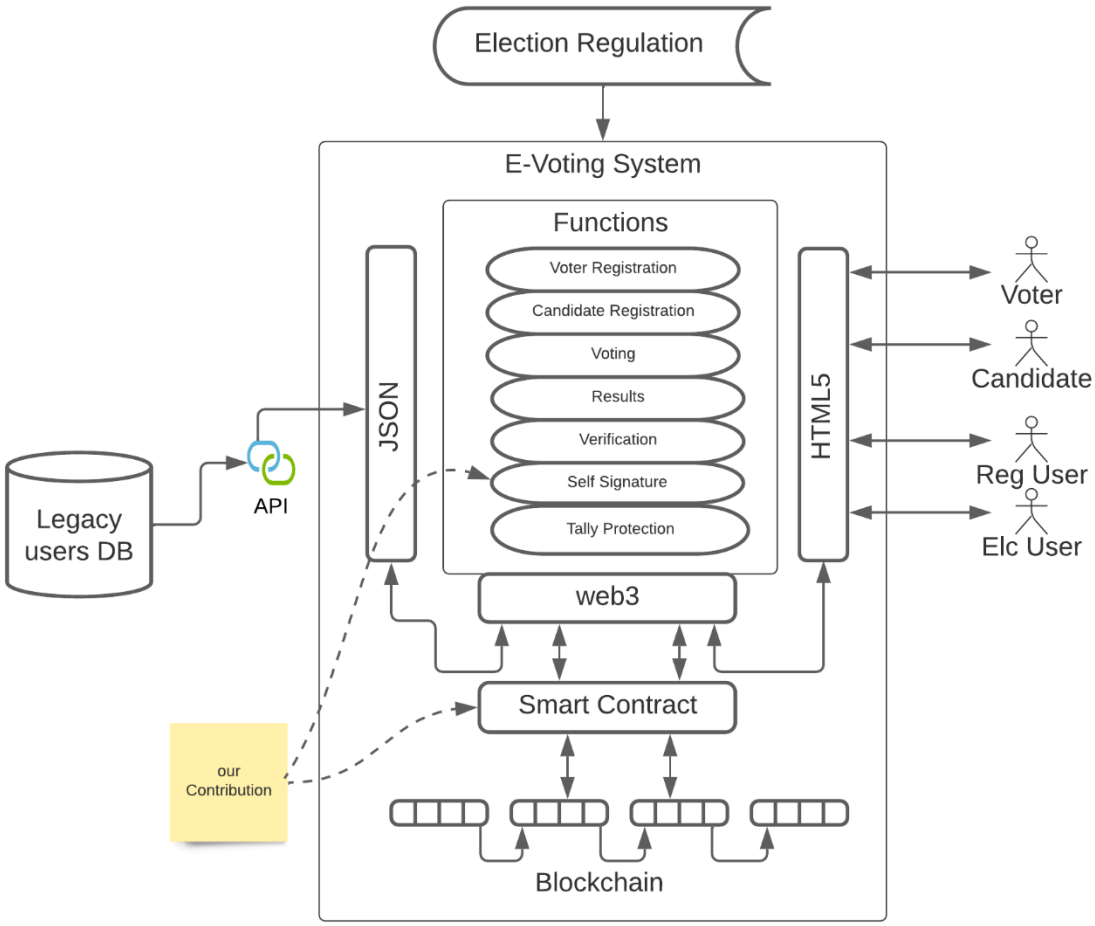


Figure 1:The architecture of proposed system.

Figure 1 illustrates the different components of our proposed system. the system is interacting with some actors like voters, candidates and authority users. The system has to be authorized by the Authorities and following the Regulations. The figure also shows the different

functions used by the system. the system uses the Blockchain as a backend database to store every transaction using smart-contract. The interaction between the Blockchain and the user interface is using web3 library. The choice of web3 library is based on three advantages. Its compatibility with different browsers, the many online documentations, and to make the interface customized and friendlier.

System Components

The components of the system are detailed by their functionality which are:

Election Regulations: which represent the Regulations and Lows that control the Election Process.

Legacy Users DB: is a Database that contains the legitimate voters and it feeds the system with the needed user data. This data is fed using an API that communicate the system and the Database.

Voter: which is the Legitimate citizen who has the right to vote by the Regulations related to election Process. This voter can perform the tasks: Vote, Register, and Verify.

Reg User: or Registration Authority, which performs the registration operations for the voters and candidates. The Tasks are: Register, Assign Keys, Modify Voters, Modify Candidates.

Elc User: or Election Authority, the committee which is responsible for the Election session, its tasks are: Define Election, Set Election parameters like start and end time, Perform the final count and publish the results.

Candidate: which is a voter who is registered as candidate by the Registration Authority.

Blockchain: which is used as back-end Database to store the election process.

JSON Object: used to perform a Like Database for users' and candidates' data.

HTML5: is used as front end interface for all actors.

Web3: A JavaScript library which is used to communicate the system with the Blockchain.

Self-Signature: this is our mechanism to apply verification on Ballot without the ability to poof for others that this Ballot content belongs to a which voter.

Tally Protection: a method to protect the results from being exposed before the tallying time.

Solidity Smart Contracts: which is a programming language used by the Ethereum Blockchain to change the state of the Blockchain. These smart contracts will have the following E-Voting system functions:

a- Voter Registration: which is to give the legacy user a Public and Private Keys or what is call Blockchain account on the Blockchain network.

b- Candidate registration: to mark a voter is being a legitimate Candidate.

c- Voting: which is performing a ballot like operation on the Blockchain.

d- Results: Perform the final count for ballots and view the result.

Methodology

The implementation of this system is based on the methods and tools found in the literature review. A system built of Blockchain, using the open platforms available for this technology like Ethereum, while searching for cryptography algorithms to be used in some functions like the anonymity requirement using the ring signature algorithm and so on.

This research will use the literature reviews and existing systems examinations to make an overall study that gives this research the tools, which will help in finding solutions to solve the research problem and achieve the features and functionality of the E-Voting System.

The implementation will be tested using case studies by performing a simulation of voting process from the start of declaring election to finally tallying results. Also Monitoring system performance could be done using existing Operating Systems performance tools like for example to monitor performance in Linux OS. For instance, “htop” can be used to monitor the CPU and RAM usage during the voting process. Additionally, it is possible to use smart contracts in the Blockchain to provide statistics including concurrent users and votes per minutes and so on [28].

System Layers

Programing this such system will be set in three layers as shown in Figure 2:

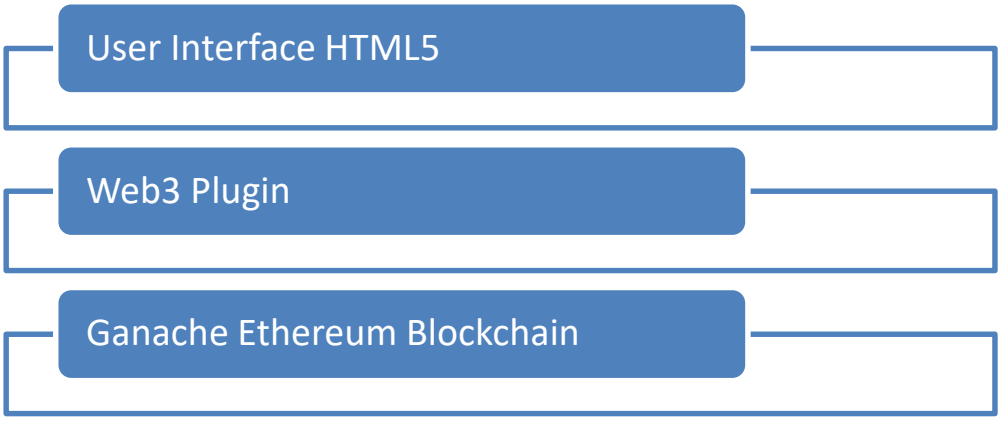


Figure 2: Implementation Layers

Layer 1 [Top]: The user interface, using HTML5 to provide the user with the ability to use the system. and to give the system the functionality and properties of the e-voting system.

Layer 2 [Intermediate]: a java platform to interact between the user interface and the Blockchain. For this layer we will use web3 java library, all functions use web3 library are in Appendix C.

Layer 3 [Bottom]: The decentralized database. The Blockchain that e-voting system will use to store every transaction related to the system. in this layer we will use the Ganache network as a Testnet network of the Ethereum Blockchain network. This Ethereum Blockchain network is public Blockchain network just like Bitcoin, but it is based on the Proof- of-Stack (PoS) instead of Proof-of-Work (PoW) as consensus algorithm to give the network scalability and to overcome the power consumption issue related to PoW [29].

The programing of the system on the Blockchain will be using the deployment of smart contract. Using Solidity programming language to make a small program that make transactions

on the Blockchain. When this program is deployed on the Blockchain it will be used by the Web3 Library to store and read the voting components and variables on the Blockchain. The system is using the Blockchain network as a database to store its variables. Simply the smart contract will perform the functions of the system on the Blockchain.

Public Key Infrastructure

Public Key Infrastructure PKI, is a global mechanism to establish a trusted and secure communications between two parties relaying on a third one which performs the trusted authority which called Certificate Authority CA [30]. **Error! Reference source not found.** Shows a simple architecture of PKI.

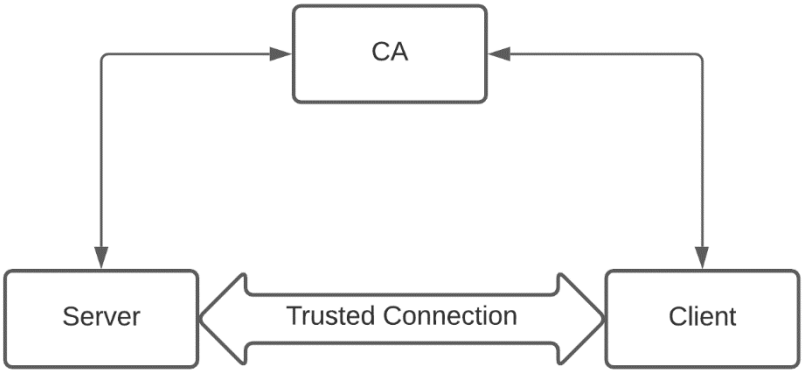


Figure 3: Simple PKI system diagram

According to the National Strategy of Palestine 2017-2022 provided by the Ministry of Telecommunications and IT, the Palestinian Government had started a project to make an e-government in many fields, while we were making the proposal of this thesis, there were three running projects: e-payment, Public Key Infrastructure PKI, and digital signature. The PKI and

digital signature projects are promising projects that makes the ability to implement an e-voting system more realistic. Because PKI and digital signature will give solutions for e-voting system authentications and authorization issues appeared in many E-voting systems yet, like in [31] [19] [32].

Because of the lack of PKI implementation in Palestine, a list of trusted and authorized voters, trusted authorized candidates, and trusted authorities will be built on a rational database to simulate that they are parts of a public key infrastructure.

The Blockchain Technology

Blockchain data structure is built by blocks as seen in Figure 4, based on [31] each block is linked to the previous block using a hash function. The hash function is a mathematical way to build a fixed length code of any arbitrary data, this code is very hard to be generated with any other data. The chain is ordered in a chronological order, that any new block can only be added to the end of the chain, the first block is the genesis block which starts the chain [7].

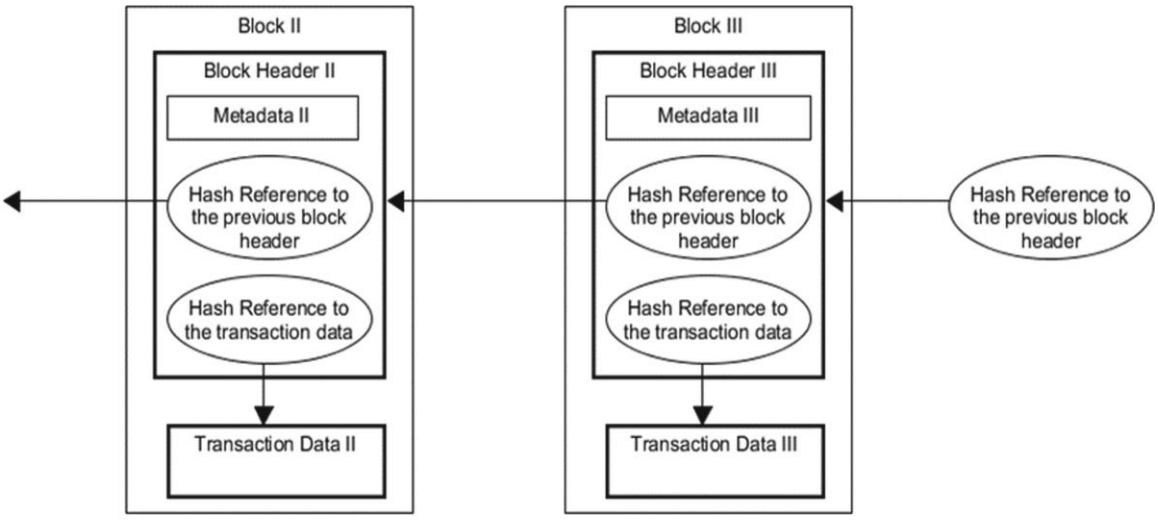


Figure 4: Blockchain structure [7]

The only way to insert a block between any two existing blocks is by rebuilding the reset of the chain, which is hard because of the implementation of proof-of-work or proof-of-stake, this proof concept is similar to mining for gold, miners make effort to find the gold, and so in Blockchain technology you make effort by doing a hard calculation to find a nonce, this hard calculation means CPU usage, which means power consumption [33].

The Blockchain technology provided a way to stamp the data shared by participants in a way to prevent any change to any existing block in the chain, leaving adding blocks to chain is the only way to add data which called transactions. This stamp is achieved by including the hash of previous block in current block data, then the next block will also include the current block hash in its data.

The Blockchain technology can be used as non-permissioned or permissioned Blockchain, non-permissioned Blockchain also known as public Blockchain, where no limitation on block access, that any participant in the Blockchain can see and verify the whole system. while permissioned Blockchain which is known as private Blockchain where the system relies on some external party to provide the authority and limitation on access to Blockchain [34].

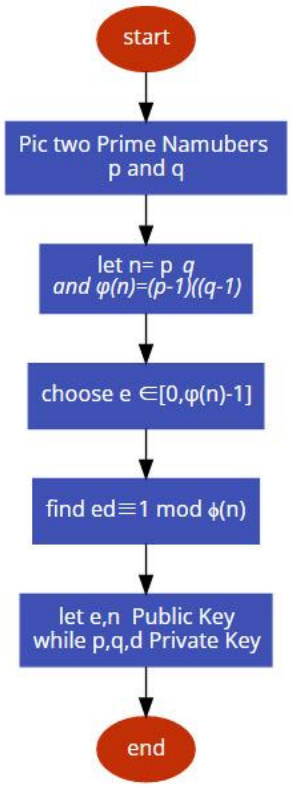
Cryptography and System Functionality

The E-Voting system should meet the basic requirements mentioned before, some of these requirements need some technical solutions with some Cryptographic Algorithms. For instance, to apply validity on voting we will use an Algorithm named Ring Signature which will be explained later. The Ring Signature gives the system the ability to validate that the voter is one of the eligible voters without revealing this voter identity, while to allow the user to verify his vote, we will use the random PIN codes and keep preventing the ability to prove that for others. Finally, to avoid

any disclosure of results before the tally we will make Asymmetric encryption of each Vote using the keys owned by the Election Authority, those algorithms are described next.

Asymmetric Encryption Algorithms (RSA)

The famous Asymmetric Encryption Algorithm RSA which is named for the first letters of its founders Rivest–Shamir–Adleman. This type of encryption uses two keys, one for encryption and the other for decryption. The parties involved in the process should keep one key as a secret, this secret key is named Private Key. While the other one is to be shared with the other parties which is named Public. The next Flowchart 1 is showing steps of generating the key pair [35].



Flowchart 1: RSA flowchart

To generate cypher text C from Message M using public key e,n. we use $C= m^e \bmod n$. While to decrypt the cypher text C and return the original message M using private key p,q,d we

use $M=C^d \bmod n$. note that n = (p q). the idea is to keep p and q private and never share any of them [36]. For our implementation we will use a Java Library to perform the encryption and decryption using the RSA Algorithm.

Ring Signature

How to leak a secret? That is the title of paper which formalized the Ring Signature Algorithm [37]. The idea is how would some member of a group leak a secret about this group without expose himself. But in the same time he shall proof that the secret came from a member of this group.

This Algorithm assumes that:

- There is a message has to be signed.
- All members share their public key.
- The signer should have both Private and Public Keys in hand.

To simplify this algorithm, we hash the message, pick a random number, assign dummy values to each public key expect the signer key, we calculate a one-way encryption of each dummy value using their assigned public key, we use the message hash with the random number and the result of each calculation in an equation (symmetric encryption) to produce a value equals to the chosen random number. For this equation to be satisfied we use the private key of the signer to generate the dummy value that should be used with the public key of the signer in verification process.

The output of the Ring Signature has 3 tuples, the Public Keys, with corresponding generated sign values, and the test value. To verify this signature, we do:

- We compute a one-way encryption of each pair.
- We hash the message.
- And finally apply the equation and check whether the test value satisfies the equation.

Figure 5 is an illustration of the ring signature from the original paper of how to leak a secret.

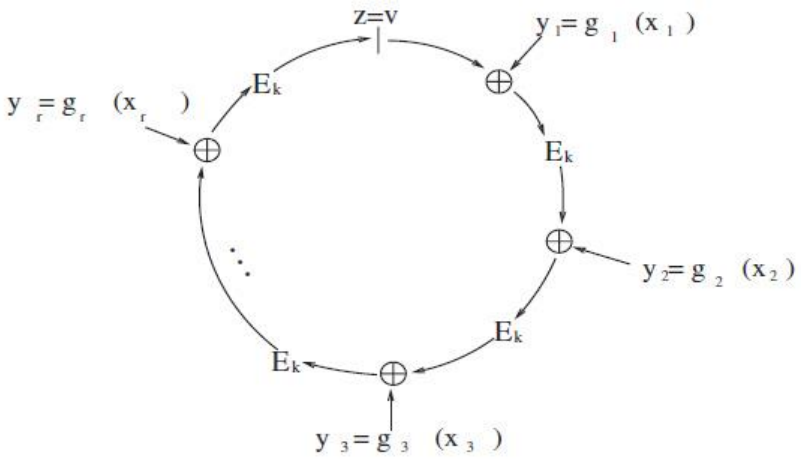


Figure 5: Ring Signature [37]

Based on the art of work in **[37]** the ring in this figure shows that it does not matter from which signer we start the verify. And a chosen random number v used in the equation should also be the result of the equation z=v.

Smart Contracts

The smart contract is the way we have chosen to interact with the Blockchain. A smart contract on Ethereum is built using Solidity language, this type of software runs on Blockchain, and usually used in the distributed applications DAPP [38].

When this piece of software is compiled and submitted to the Blockchain network there is no way to modify, delete or update it [13], this gives the smart contract the immutability.

Self-Signature Mechanism

We named this mechanism Self-Signature. This mechanism is used to achieve the verifiability and prevent receipt-free conflict. That's because only the voter her/himself can make sure that this signed ballot is for her/him.

Let X be a Ballot with two parts Candidate P1 and PIN code P2, let P1's address space =3, let P2 a number of 4 digits freely chosen, Assuming A and B are two parties, B wants to make sure that A votes for specific Candidate, for that, B asks A to specify a PIN code P2 in the Ballot as a Mark that this ballot is their Ballot which voted for specific P1.

Then the address space of this Ballot p(X) = 3 * 10000= 30000.

If P1's address space is still 3. But P2 is formed of two parts with two digits chosen from four pairs, for example: Assuming ABCDEFGH are numeric digits, any digit can be from 0 to 9, and randomly generated four pairs, then the possible formations of four digits from selecting two pairs are limited to the items shown in Table 1:

AB CD EF GH:

Table 1:PIN Code Probability

1	AB	CD	ABCD
2	AB	EF	ABEF
3	AB	GH	ABGH
4	CD	AB	CDAB
5	CD	EF	CDEF

6	CD	GH	CDGH
7	EF	AB	EFAB
8	EF	CD	EFCD
9	EF	GH	EFGH
10	GH	AB	GHAB
11	GH	CD	GHCD
12	GH	EF	GHEF

A list of all available PIN codes are shown in **Error! Reference source not found.**.

Then the address space of B is 12, with probability of free choice is p(X)=3 * 12 / 30000= 0.0012;

Assuming B wants A to specify ABCD PIN code as a proof then its chance is less than 1% to success. In counteract for A who wants to verify his vote, it is less than 1% that this vote is not his;

Chapter Four
Design and Implementation

Implementation

To build the three layers described in the architecture chapter using the tools and functions specified in the methodology, we use the Unified Modeling Language which provides a standard way to illustrate the design in a graphical view.

Modeling the System Parts

System Actors

The system design is based on the functions of each object involved in the voting system. Those functions can be generated from the use case of each actor in the system:

Voter:

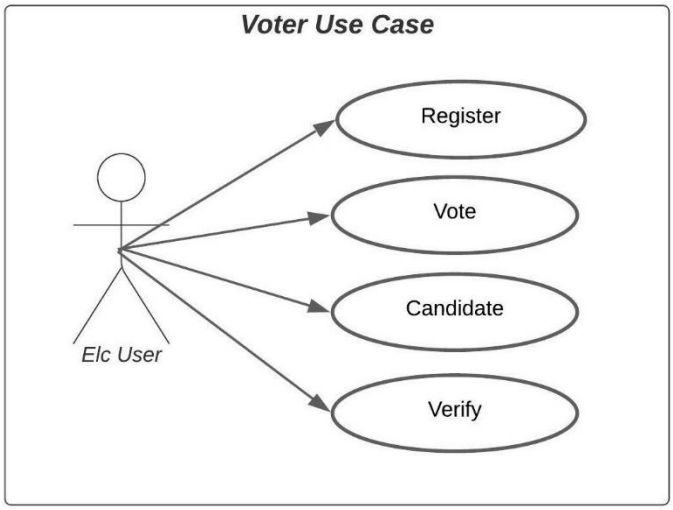


Figure 6: Voter Use Cases

Figure 6 shows that the voter can perform Register, Vote , Candidate himself, and verify that his ballot was counted correctly

Registration Authority:

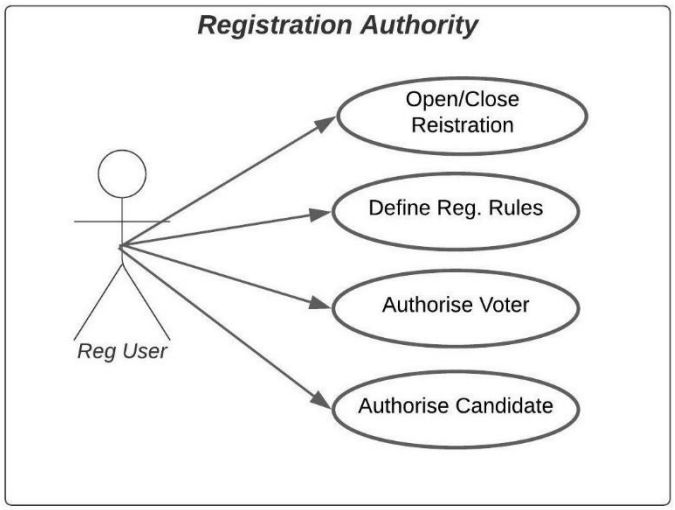


Figure 7: Registration Authority Use Cases

The Registration Authority as shown in Figure 7, can Open Close Registration, define rules, Authorize Voter and Candidates.

Election Authority:

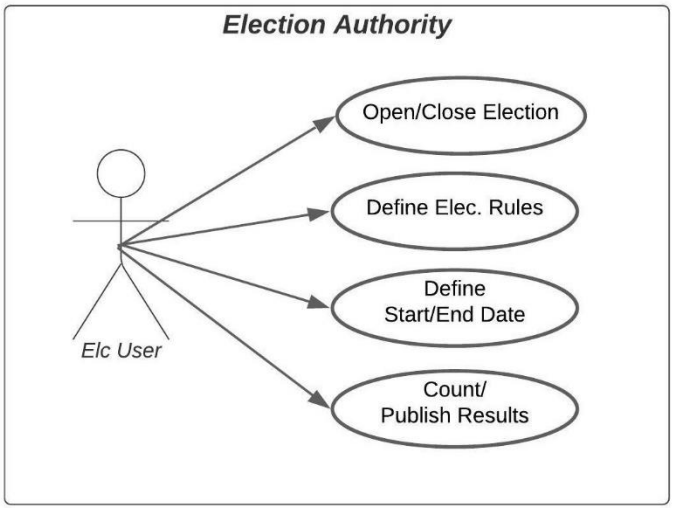


Figure 8: Election Authority Use Cases

Election Authority Actor in Figure 8, should have the ability to open and close Election, Define related rules, set the start and end dates for the election. And finally they are the responsible for tallying the final result.

System Classes

To illustrate what type of data could be used from the legacy Database and fed into the system, a voter class can be used for this purpose:

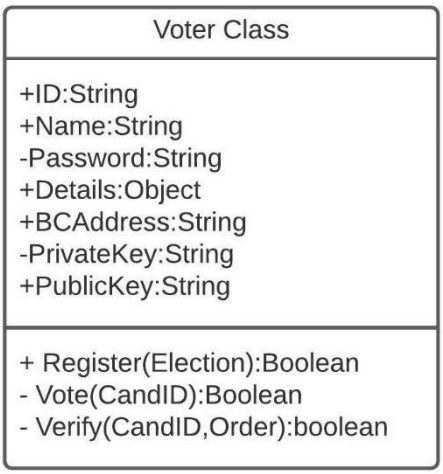


Figure 9: Voter Class

Figure 9 identifies the basic data types for this class, note that password and private key are have private access.

The Election Authority Users should have a class with a minimum properties and functions as shown in Figure 10:

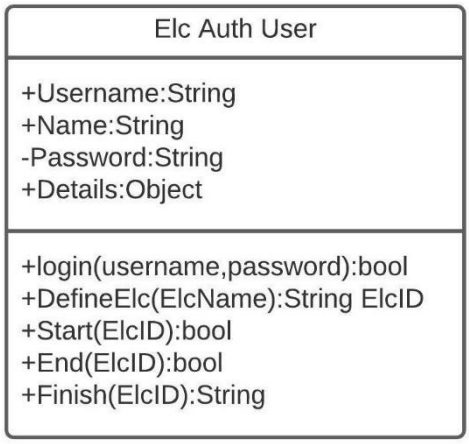


Figure 10: Election Authority User Class

The same is for Registration Authority, which also should have the minimum properties and functions as shown in Figure 11:

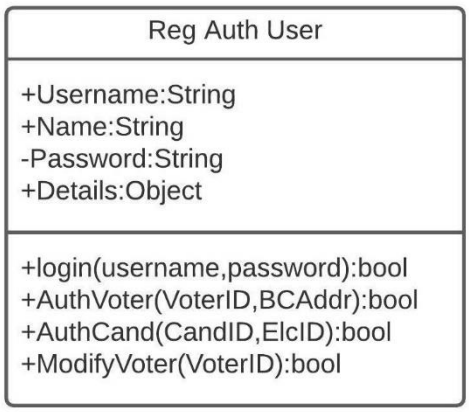
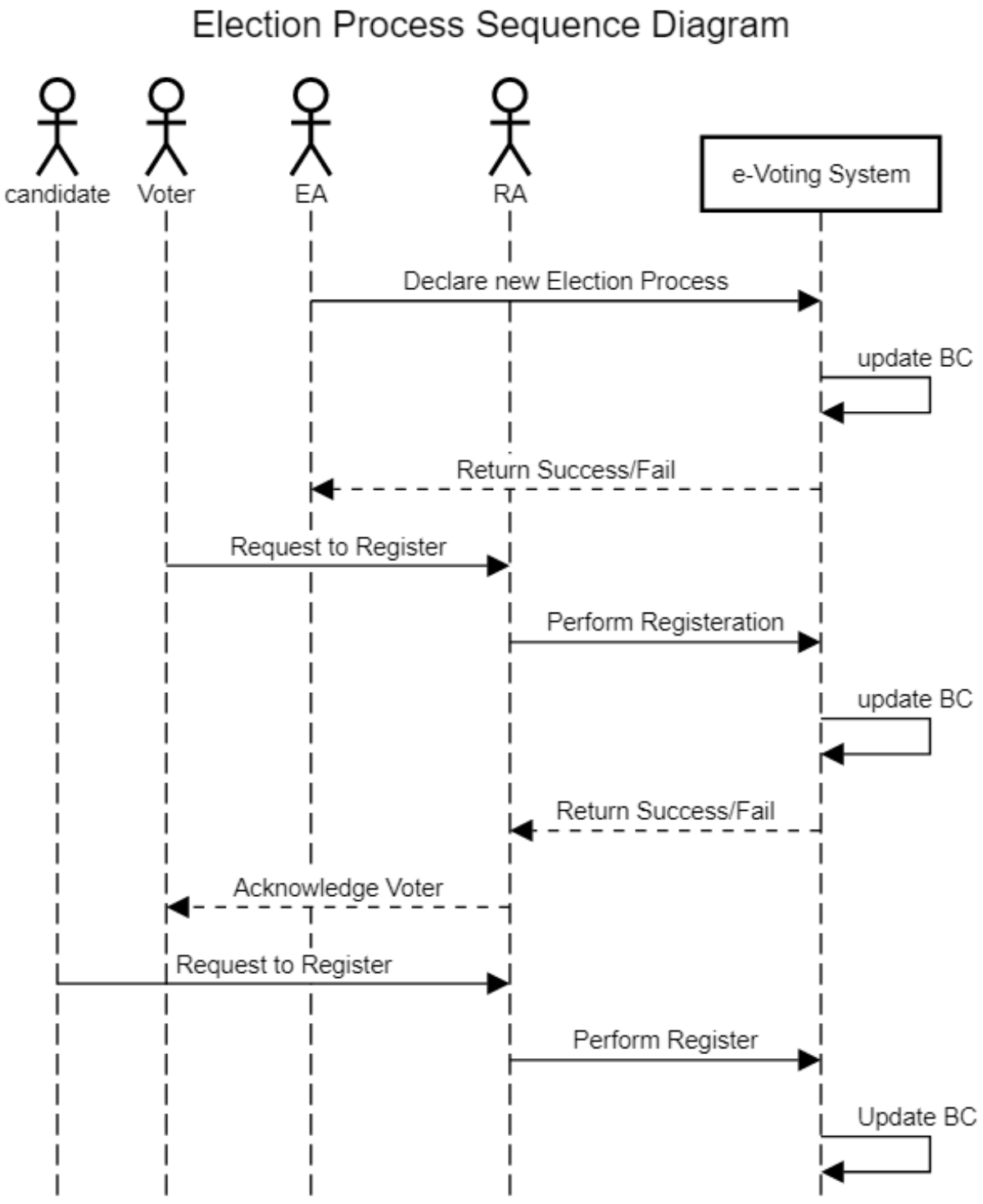


Figure 11: Registration Authority Class

Execution Stages

The whole process is taking place by executing the election in stages, the execution of the stages is sequential:

The sequence diagram in Figure 12 is an illustration of e-Voting process


Election Process Sequence Diagram
candidate
Voter
EA
RA
e-Voting System
Declare new Election Process
update BC
Return Success/Fail
Request to Register
Perform Registeration
update BC
Return Success/Fail
Acknowledge Voter
Request to Register
Perform Register
Update BC

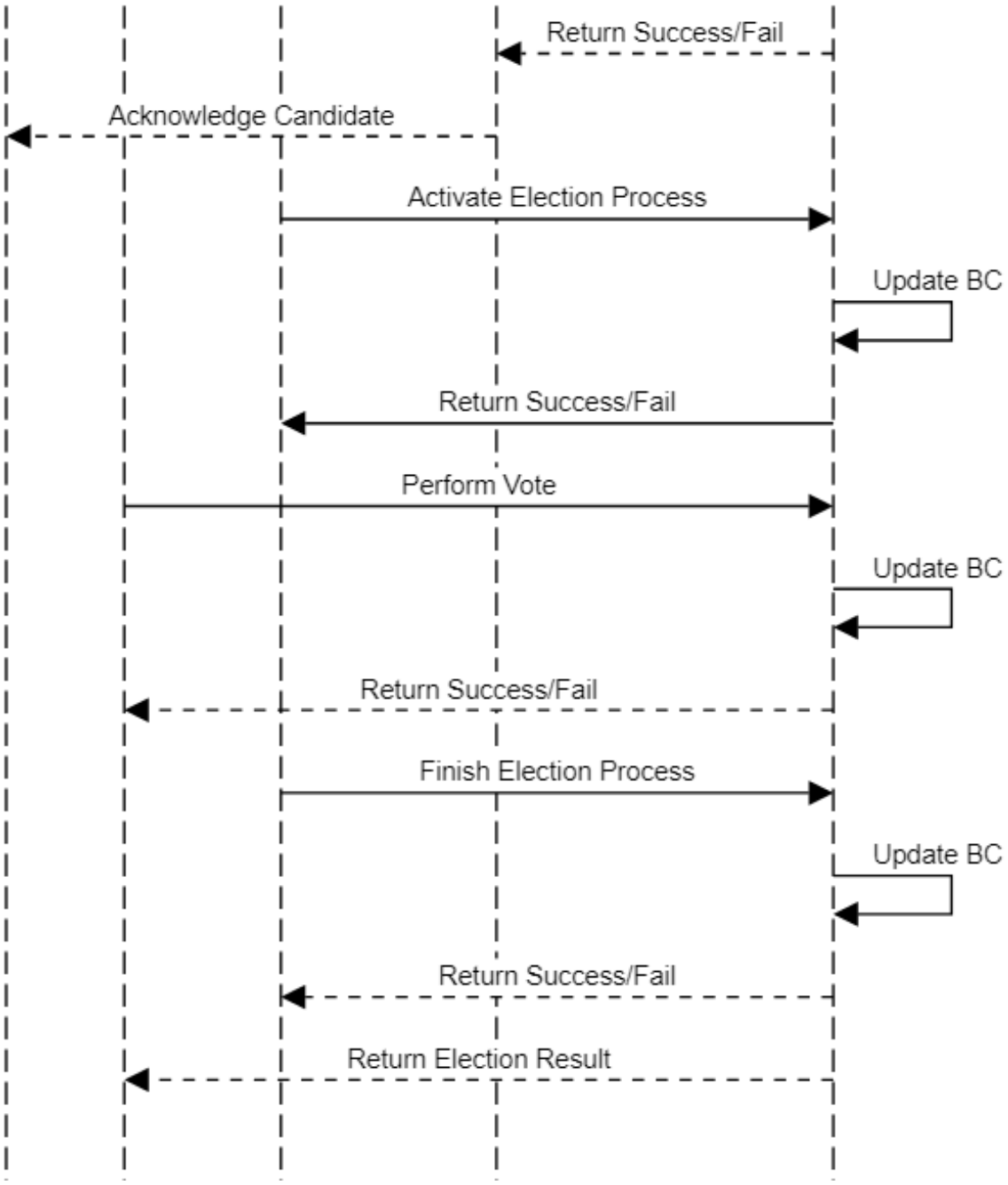


Figure 12: Voting Sequence Diagram

1- The Election Setup

The Election Authority Login into their system and define an election campaign. They set a name, id, start date, end date. They can finish a running campaign, but should not be able to edit a finished one.

2- Candidate Registration

Any voter who has the right to be a candidate by the regulations should communicate with the Registration Authority. The Registration Authority should mark or add this voter to the candidate list.

3- Voters Registration

Normally by laws, a voter list is created by the Registration Authority, but the voter should update his status and confirm that he is listed before the campaign begins. This could be done in this system by testing to login the system before the election due date. Also the Registration authority should combine each voter with an individual Blockchain account to be used in our system.

4- Voting

Here is our main contribution in this thesis. On the election due date, the voters can login into the system and vote for their candidate. The system will check if the voter on the login page has the right to vote or not. If the voter has the right to vote. It will redirect him to the voting page. Once the voter chose his candidate. The smart contract will update the state of the Blockchain to mark the Blockchain account as voted. The smart contract also will generate a ring signature for this voter and using the strategy of how to leak a secret [37] pin a ballot for his candidate and sign it using this signature. Now the page will show up to the voter five random numbers. Each one is consist of two digits', and he should pick up two numbers and combine them as a pin for his own reference. The smart contract should sign this numbers with his ring signature. This is how only the voter can verify that this vote is his vote.

5- Counting Votes

When the election end date time comes. The smart contract for the voting should be inactive. The Election Authority should trigger the Finish function to count all ballots for each candidate and show the results in the results page.

6- Overall verification

The Blockchain structure can validate the overall verification of the process. The end user or voter can return to his pin code to verify that his vote was counted and counted correctly.

Layer 1, HTML5 Pages.

The user interfaces are an HTML pages and are subjected to be used by three types of user, the Registration Authority, The Voters, The Election Authority

Registration Authority

The responsibility of this authority is to validate voters, and candidates, generate and assign each voter a Blockchain account which contains a public and private key.

Also, this page should allow the authority to manage the profile of each voter.

This page is built using HTML5 and loads the voters' data from JSON file instead of Database for simplicity, the authentication to login into the system could be built on Blockchain using smart contracts [39]. But also, this is built using simple username and password for this implementation to make it easier to debug and control.

The next Figure 13 is a snapshot of web page for Registration Authority:

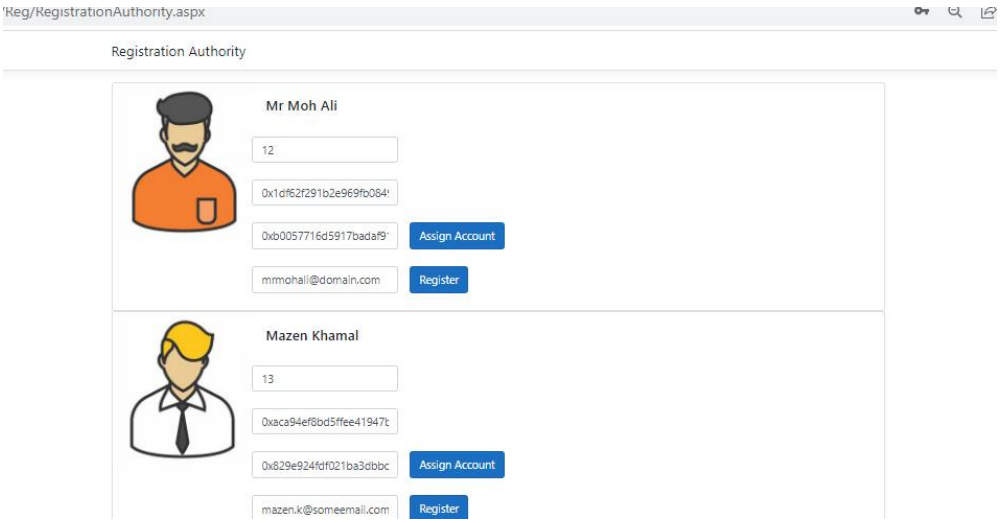


Figure 13: Registration Authority Page

The Election Authority

The responsibility of this authority is to define the rules that control the election process, the election Authority Page provides its users the ability to Define new election, start the Election, end the election, count the ballots and tallying the results. The next figure is a snapshot of the Election Authority Page.

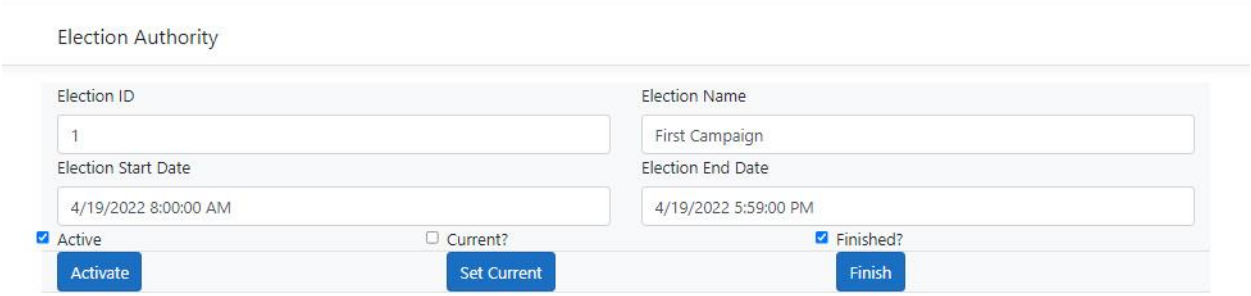


Figure 14: Election Authority Page

Both Registration Authority and Election Authority would use tow identical login pages for their tasks, also the Voter Login page will look like the same but with a little change in the tasks that does, the login page shown in Figure 15 is the Voter Login:

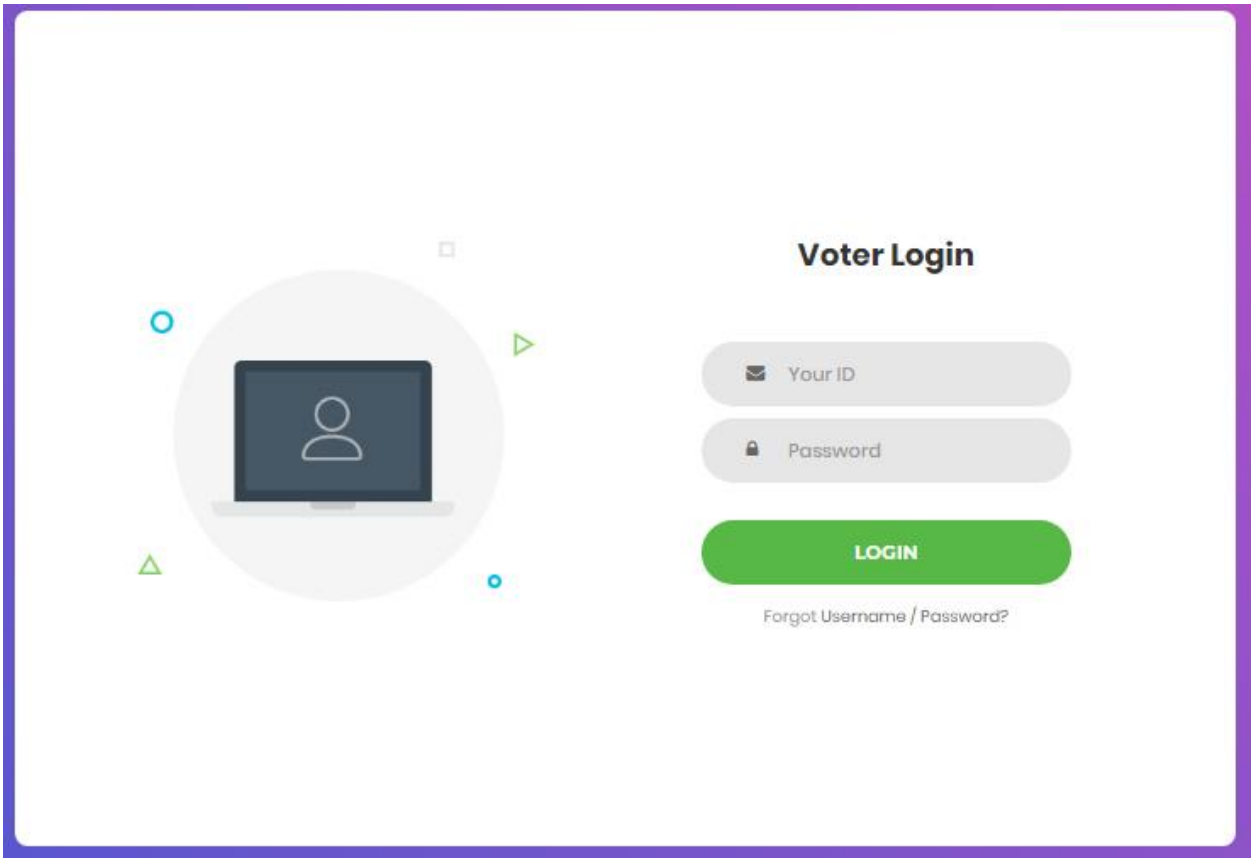


Figure 15: Login Page

Voting Page

Voters need a user interface to choose their candidate, the next image Figure 16 is a snapshot of this page:

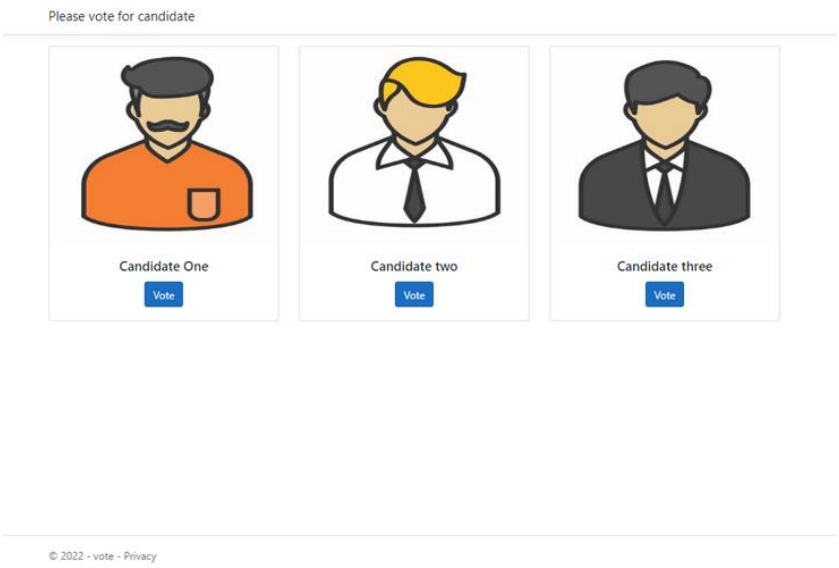


Figure 16: Voting Page

The next page after choosing the candidate, the web will display the PIN code page which is showing in Figure 17: Selecting PIN.

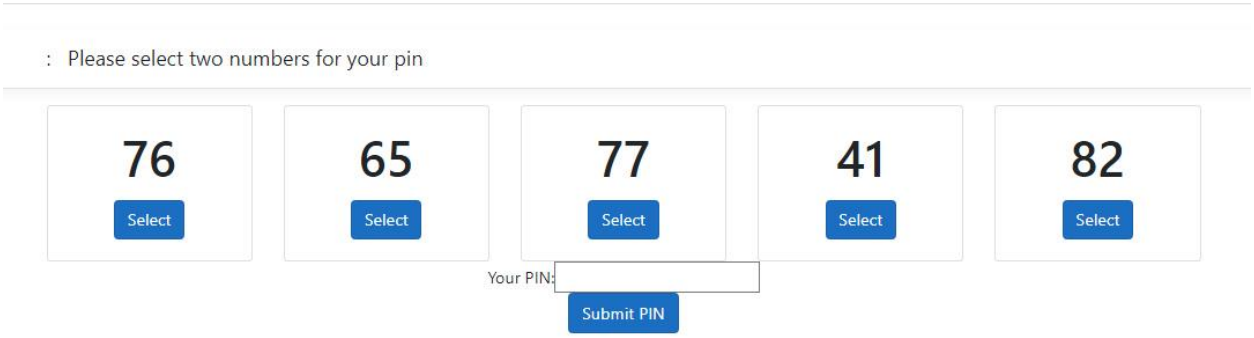


Figure 17: Selecting PIN

Results Page

The results page is a page contains a statistical information about the current number of voters which voted. And the final result will be shown only after the Election Authority Finalize and count the election process. This page is shown in Figure 18:

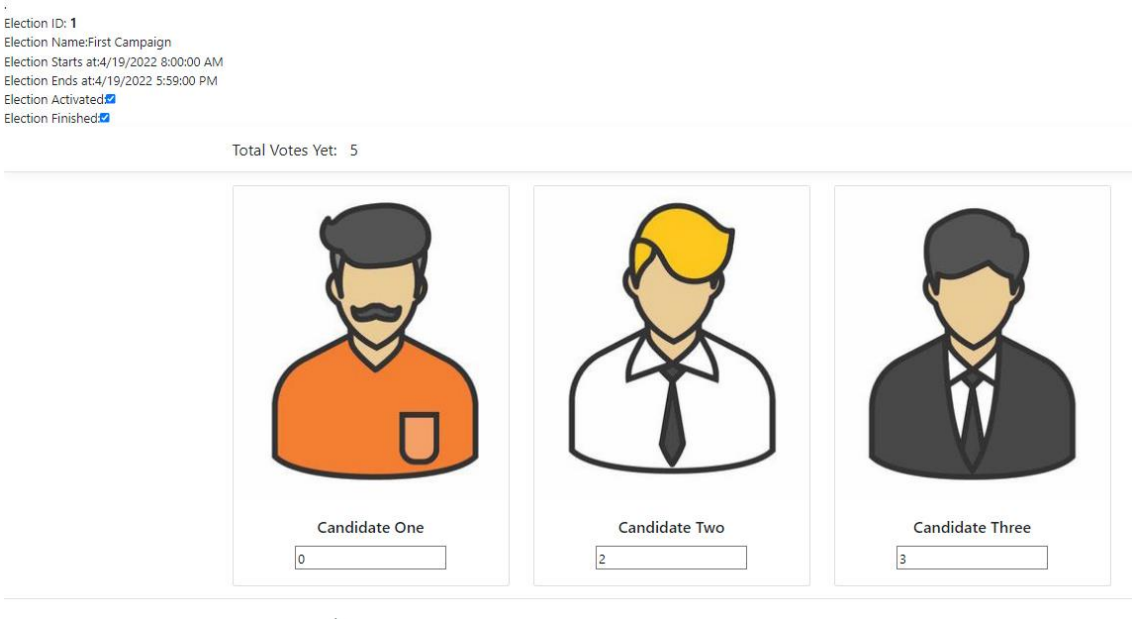


Figure 18: Results Page

The Verifying Page will have an interface as shown in the next image Figure 19: Vote Verify Page:

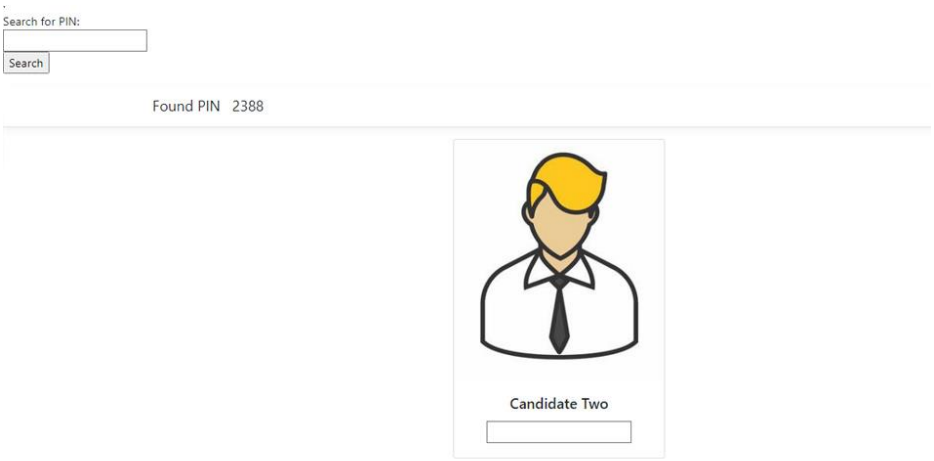


Figure 19: Vote Verify Page

Algorithms

Each of Election Authority, Registration Authority and Voter pages are in a separated folders or containers. while The code behind these pages can be illustrated in the pseudo algorithms as shown in next algorithms:

The Registration Authority Login Page

Algorithm 1: Registration Authority User Login

```
Input: USER(String),PASS (String)
Output: PAGE(String)
begin
    user ← USER
    pass ← PASS
    PAGE ← Login.html //default case is to login failed
    if userisAuth then
        PAGE ← Registration.html
    end
    return PAGE
end
```

Algorithm 1: Registration Authority Login

The Election Authority Function:

Algorithm 2: Election Authority User Login

Input: $USER(String)$,PASS (String)
Output: $PAGE(String)$
begin
  $user \leftarrow USER$
  $pass \leftarrow PASS$
  $PAGE \leftarrow Login.html$ //default case is to login failed
  if $userisAuth$ **then**
    $PAGE \leftarrow ElectionAuth.html$
  end
  return $PAGE$
end

Algorithm 2: Election Authority User Login

The Voter Login Function:

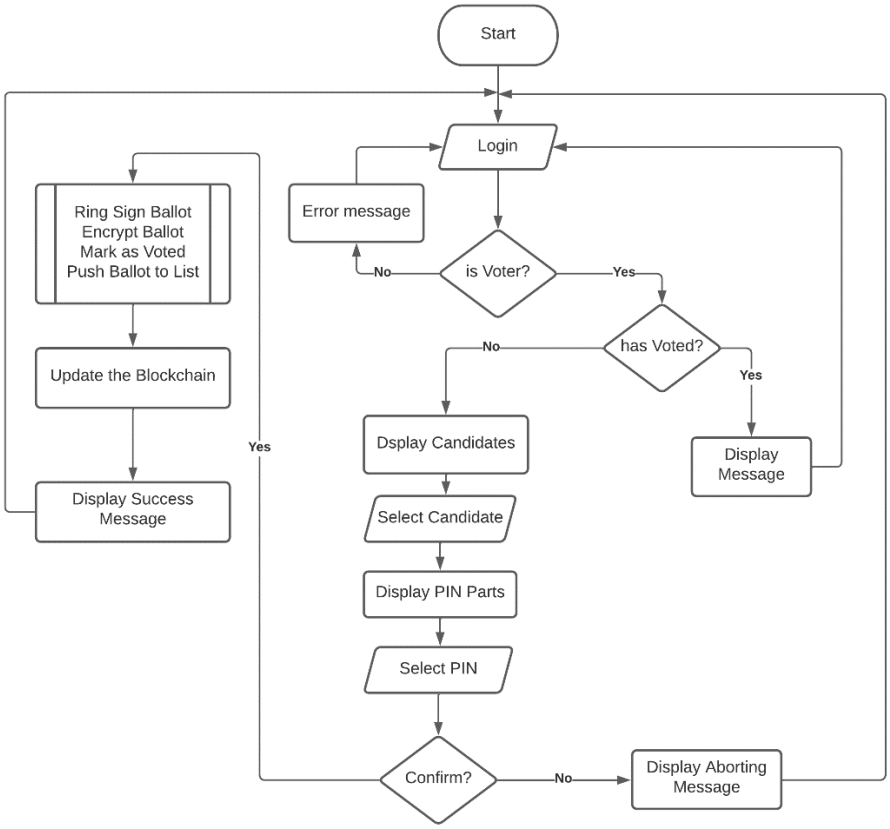
Algorithm 3: Voter Login

Input: $USER(String)$,PASS (String)
Output: $PAGE(String)$,BCA (Blockchain Account)
begin
  $user \leftarrow USER$
  $pass \leftarrow PASS$
  $PAGE \leftarrow Login.html$ //default case is to login failed
  if $userisAuth$ **then**
    $PAGE \leftarrow Voting.html$
    $BCA \leftarrow user.keys$
  end
  return $PAGE$
end

Algorithm 3: Voter Login Algorithm

For both Registration and Election Authorities they have a very similar mechanism. While the Voter login has additional data to be loaded and used. Which is the Blockchain Account information that will be used in the voting process. For that purpose, the Blockchain account information will be passed to the voting page.

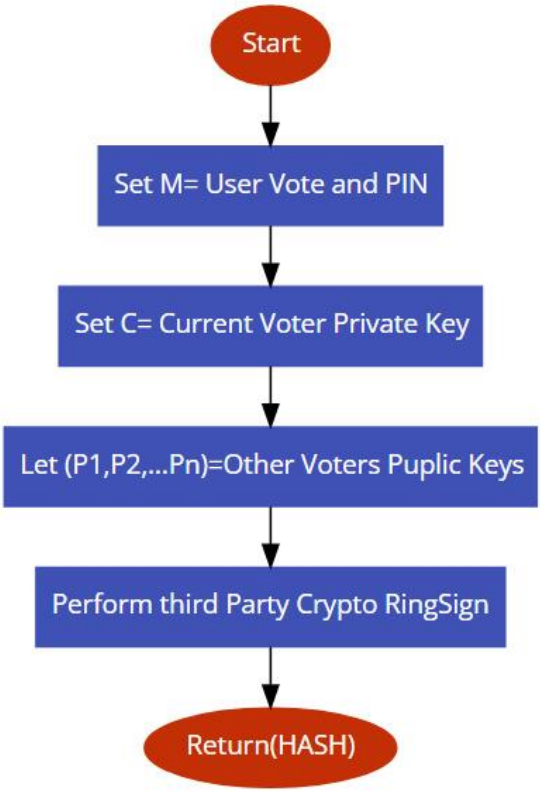
The Voting flowchart is illustrated in Flowchart 2.



Flowchart 2: Voting Process Flowchart

The process of preparing a ballot is shown in Flowchart 3. This process is to prepare the ballot as a string ready to be signed and encrypting, the implementation will use Ring Signature JavaScript Library “ring-crypto” to sign the ballot using all voters’ public keys. Then the ballot will be encrypted with its signature using one-way trap-door function RSA. Another JavaScript

Library named JSEncrypt The Used Public key for encryption is selected by the Election Authority. The Authority reserves the Private key to be used in Tally stage.



Flowchart 3: Prepare Ballot

Next is flowchart of Loading Ring Signature Key:

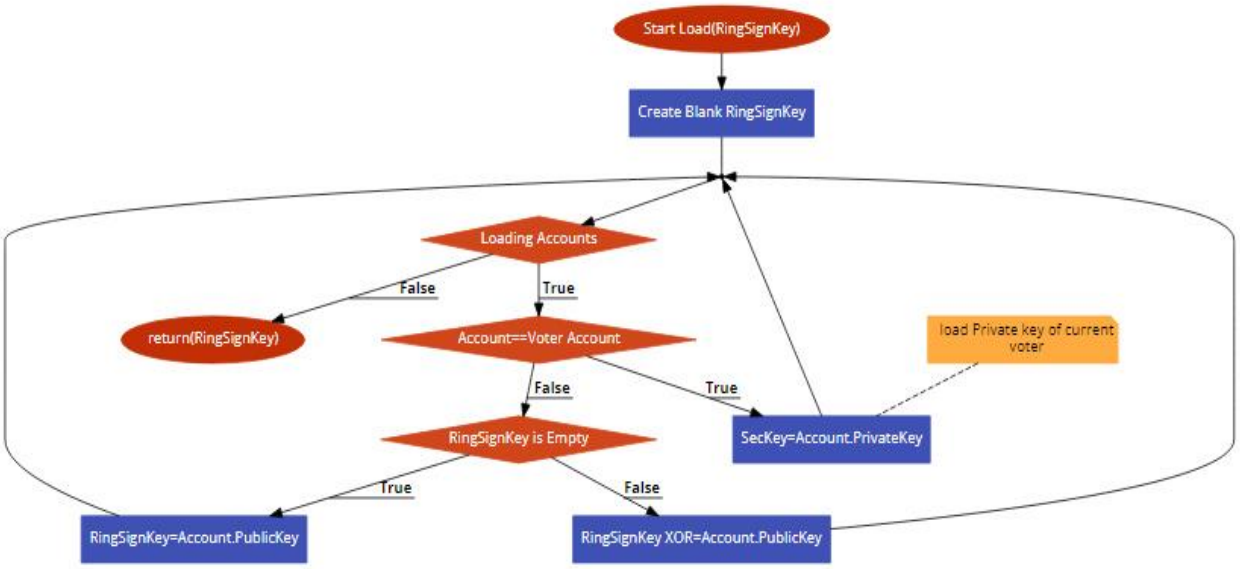
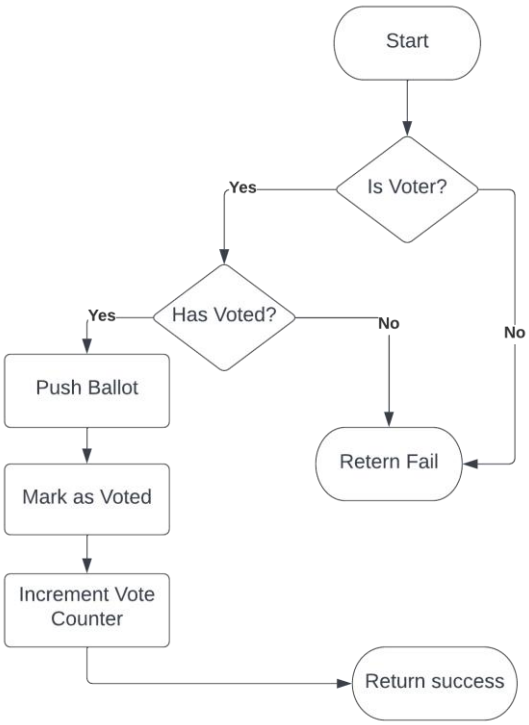


Figure 20: loading Public Keys for Ring Signature

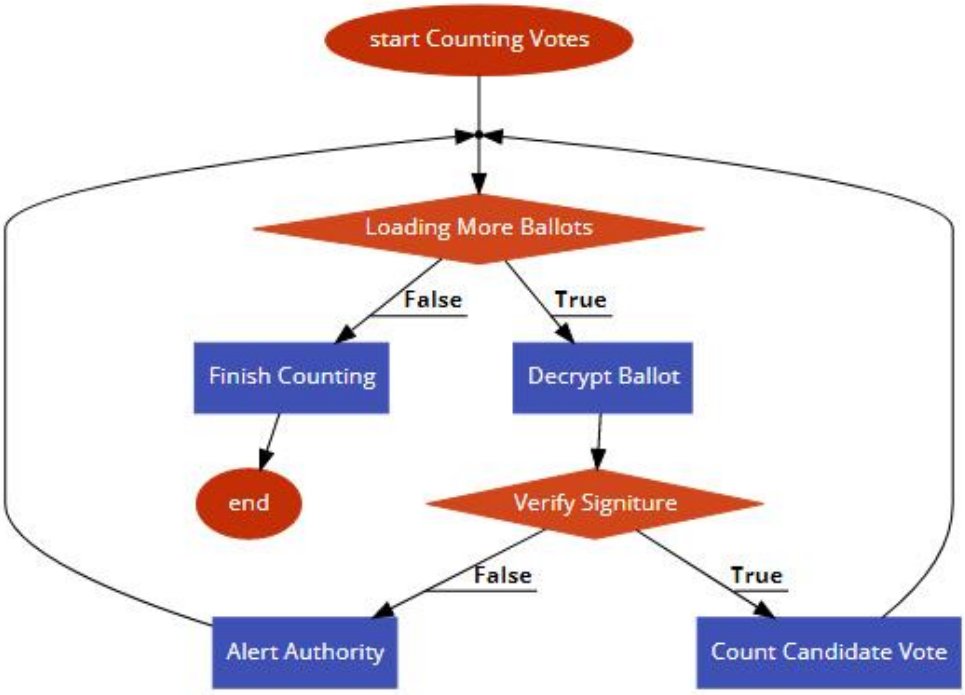
While implementing Ring Signature using java script, the library failed to hand the BigNumber data type, and therefore, the implementation of Ring Signature was built from scratch using C#, the code is found in Appendix B. and because of the nature of C# is server side, the call was through Application Programming Interface API and using JavaScript Library called Axios.

While the vote function which is used by the smart contract is shown in the next flow chart:



Flowchart 4: Smart Contract Vote Procedure

The tallying process is illustrated in the next flowchart:



Flowchart 5: Vote Tally Process

The Flowchart 5 shows that process for count each candidate votes. All ballots are being loaded one by one, a decryption of each ballot, verified and then counted to its chosen candidate. There is no reference to the voter here, all votes are submitted chronically and being request by iteration on index until reaching total voters count.

The Smart Contract

The code behind the smart contract is showing in Algorithm 4, the source code which is written in Solidity is found in Appendix A :

Algorithm 4: Elections SmartContract

Input: Voter Address,Voter Status, Ballot Content, election Status

Output: TotalVoted, Success/Fail,Ballot Content

```
def totalVoters,Finished,Active;
def ListOfBallot[], ListOfVoters[],ListOfVoted[];
def constructor():{
    TotalVoters=0;
    Finished=false;
    Active=false;}
def Activate():{
    Active=true;}
def Finish():{
    Finished=true;}
def isVoter(_Address):{
    return listOfVoters[_Address];}
def isVoted(_Address):{
    return ListofVoted[_Address];}
def AddRemoveVoter(_Address):{
    if (isVoter(_Address){
        ListofVoters[_Address]=false;
    }else {
        ListofVoters[_Address]=true;
    }
    return true;}
def gettotalVotes():{
    return TotalVotes;}
def getBallotByKey(_key):{
    return ListofBallots;}
def doVote(_Ballot,_Address):{
    if (!isVoter(_Address) || isVoted(_Address)
    || !isActive() || isFinished()){
        return false;
    }else {
        ListofBallots[TotalVoters]= _Ballot;
        ListofVoted[_Address]=true;
        TootalVoters++;
    }
    return true;
    }
```

Algorithm 4:Smart Contract Algorithm

The algorithm above is the core of the election process, and some functions like finish and activate can only set the value to true, and once it is true it cannot be set back to false. Which gives us the immunity that we need.

Chapter Five
Experiment Results and Discussion

Experiment Results and Discussion

Result Setup

The implementation is tested by making an election process demo, which is by following these steps:

1- Deploying smart contract on the Blockchain and getting its address.
2- Election Authority updates the rational Database with the new smart contract address to be used in all transactions.
3- All involved voters should register to the registration authority.
4- Election Authority should now activate the current Election.
5- Voters log into the system using their username and password.
6- Voters use their private key to prepare their ballots. This private key is used for Ring Signature Generation, voter legitimate, and to check whether the voter has voted or not.
7- Authority Private Key is used to encrypt the ballot.
8- Voters submit their votes using the Authority private key.
9- Voters list is updated to mark the voter as voted using their address and signed using their private key.
10- At the end the Election Authority mark the Election process as finished.
11- The result page counts the ballots for each candidate.

Results

The system architecture previously discussed is implemented using open source tools. The system is consisting of three layers; the user interface, the Blockchain and the java libraries web3 and cryptography. Are all orchestrated to work together to handle all the process of voting.

The user interface which is HTML5 pages are making the navigation simple and easy for users to pick their candidates.

A test election was executed for three candidates; ten voters participates in the demo. The result in the Figure 21 shows a debugging of submitting Vote to the Blockchain.

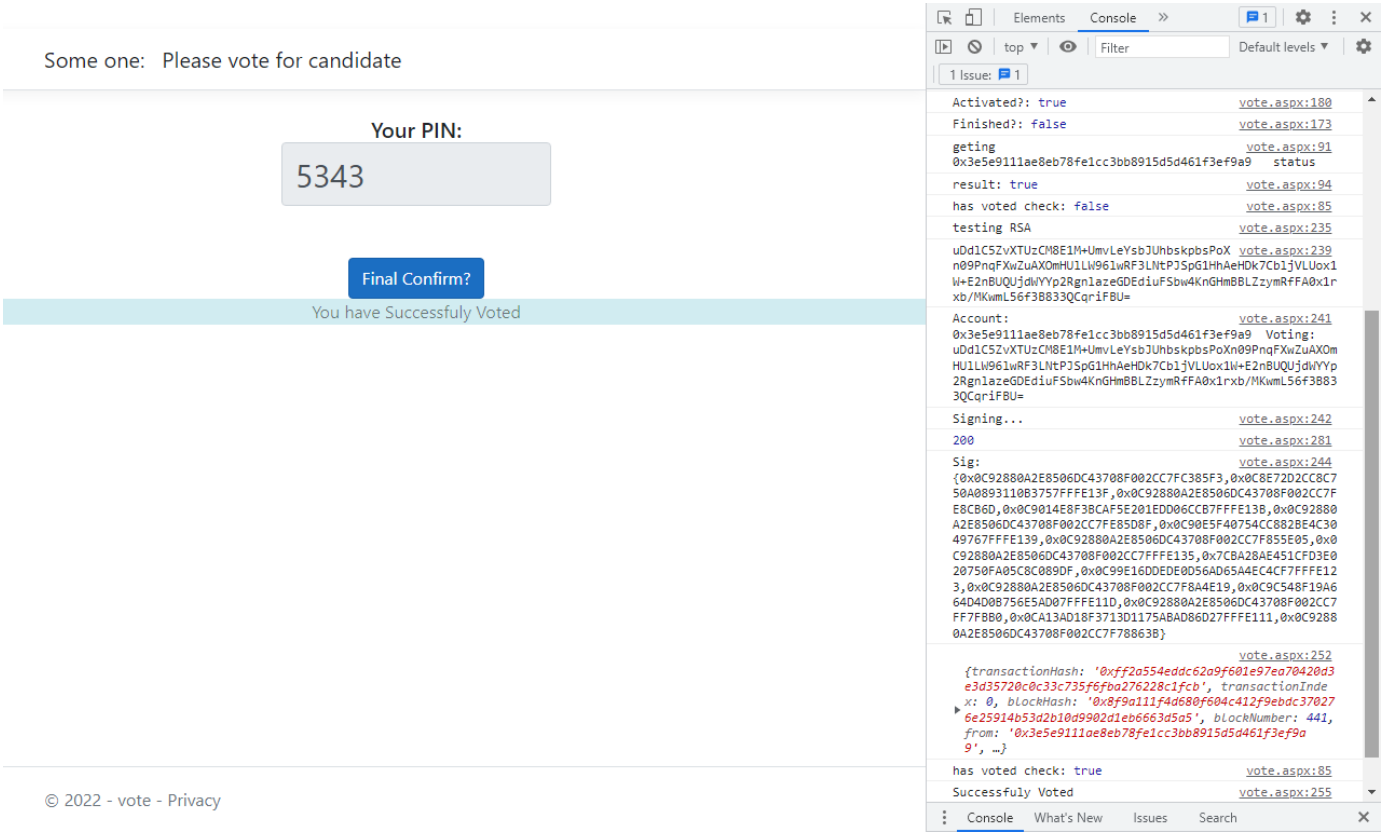


Figure 21: Debugging Voting Process

In Figure 21 the debugging is first sows a check on election status is it Active or not, is it finished or not, then checking the status of the voter is registered or not, then checks to see whether the voter has vote or not yet. If all satisfy then the system will concatenate the vote with the pin and encrypt them using the authority Public key, the encrypted data is signed with a Ring Signature. next the system will push that Ballot with its signature into the list of Ballots and increment the

total votes and marking the voter as voted in the List of Voted. And checks once again if the voter has voted, and response with success if the check result is “voted”. Otherwise it will rise an error because the transaction was not completed successfully.

Once the election Authority Finishes the Election, the result page should count the ballots for each candidate after decrypting the ballot using the Election Authority Private Key.

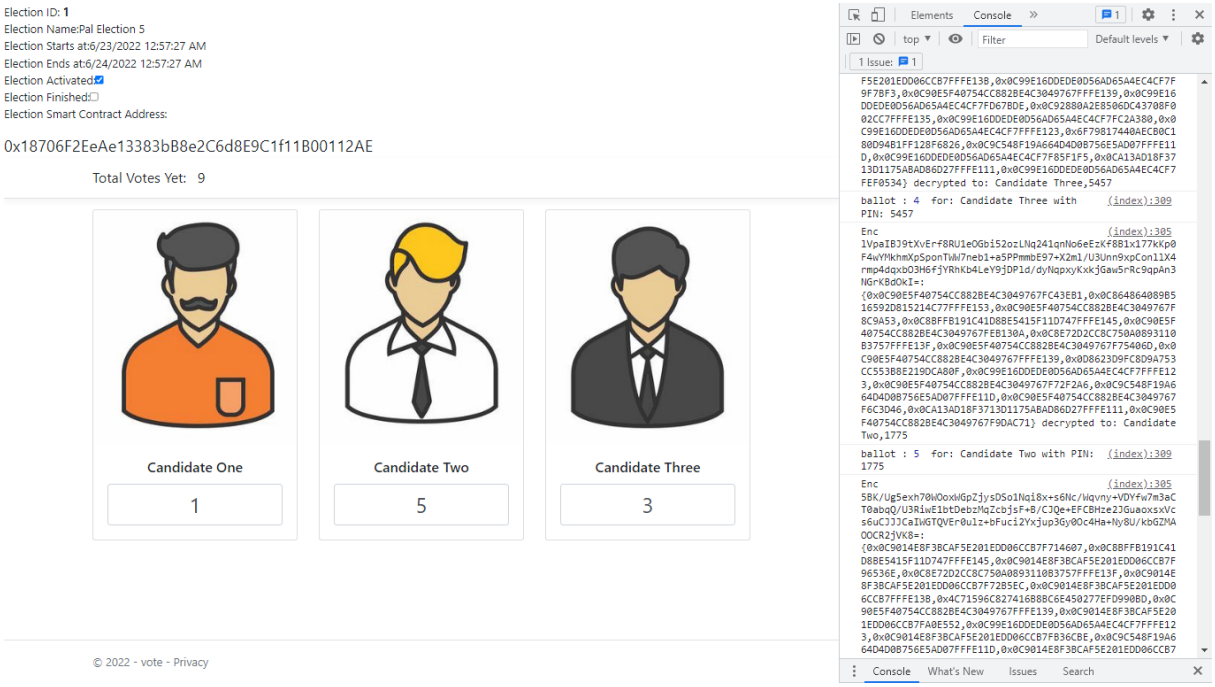


Figure 22: Counting Results

The next Figure 22 is a debugging of the counting process, and it shows the encrypted data being decrypted and counted, PIN code and Ring Signature are shown too.

Any voter can search for his/her PIN code using verify page. Next Figure 23 is a snapshot of this page.

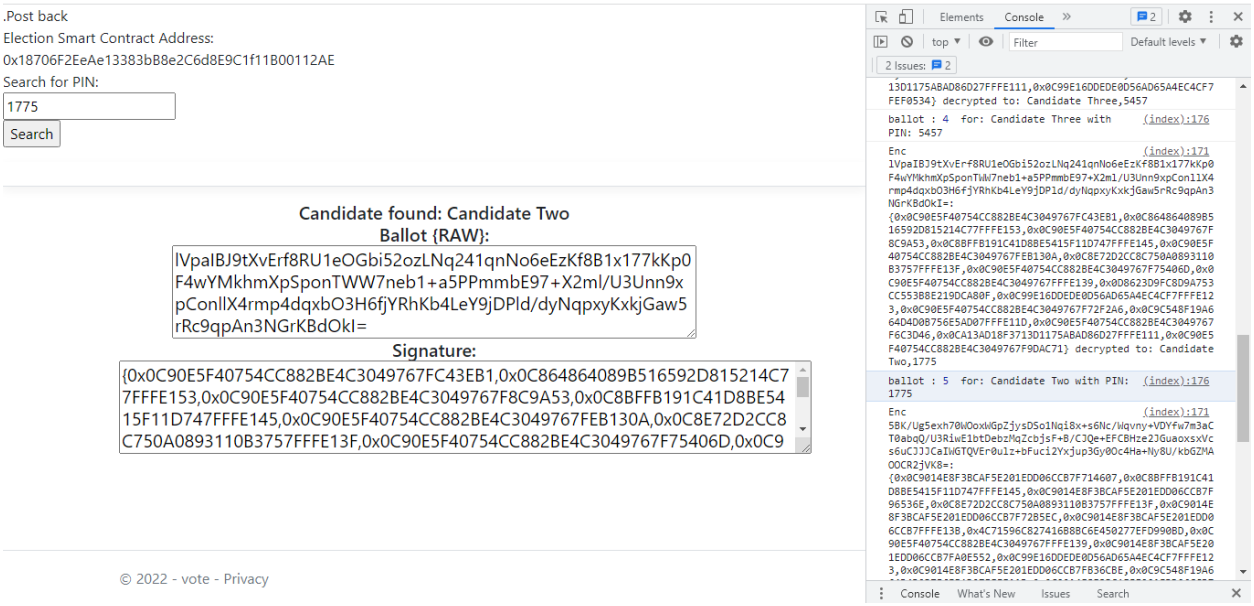


Figure 23: Verify Page

The debugging is showing that a loop through all Ballots and decrypting the data to search for the PIN is needed, as there is no direct access to this information without this loop and decryption.

The anonymity is achieved by submitting the transaction using the authority address and signed by the Ring Signature, that leads to make all votes to come from the same transaction source address. Therefore, no one can tell who is the owner of this ballot. Figure 24 is showing two transactions having the data of two different ballots.

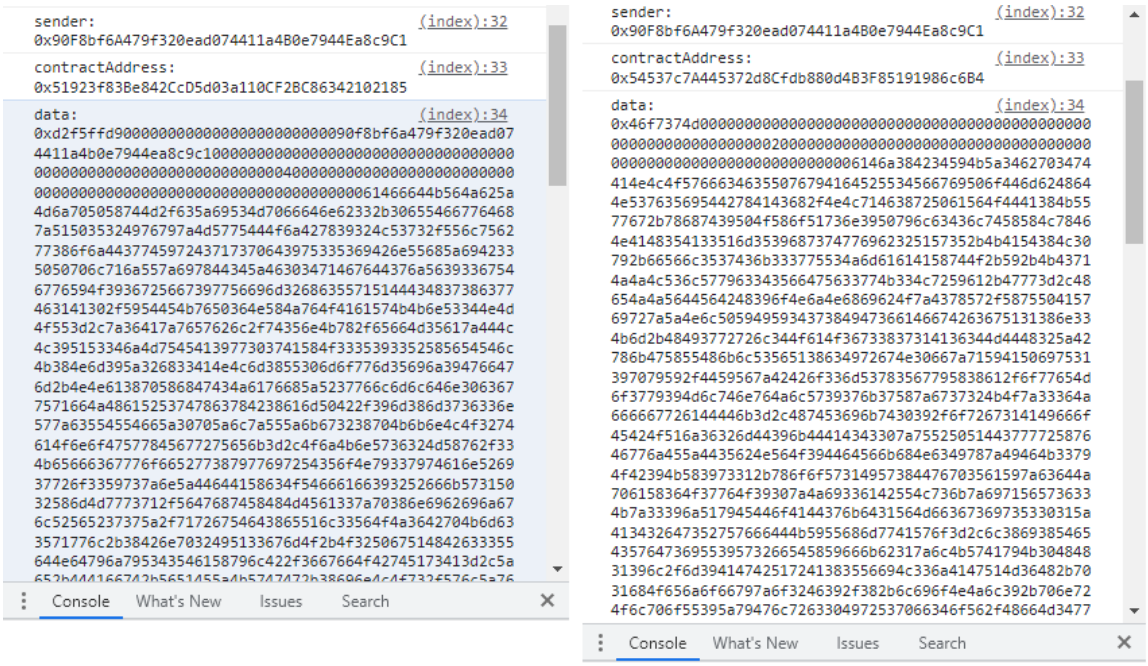


Figure 24: Comparing two transactions

The results above show that the objectives of this thesis are achieved and can be summarized in the findings listed next:

- Blockchain can be used to store election data for immunity and data protection.
- Smart Contract can be used to manage the Blockchain and store data. assuring there is no alteration after a condition is met.
- One-way trap door is a good way to achieve privacy and secrecy even with access to the Blockchain raw data.
- Ring signature is excellent when it comes to privacy assurance with no repudiation.

Blockchain Performance

The creation of key pairs does not consume much time. A linear relation between the increase of generated key pairs number and the generation time using ganache. This relation is shown in Figure 25. Note this generation procedure is done using normal PC with no special specification.

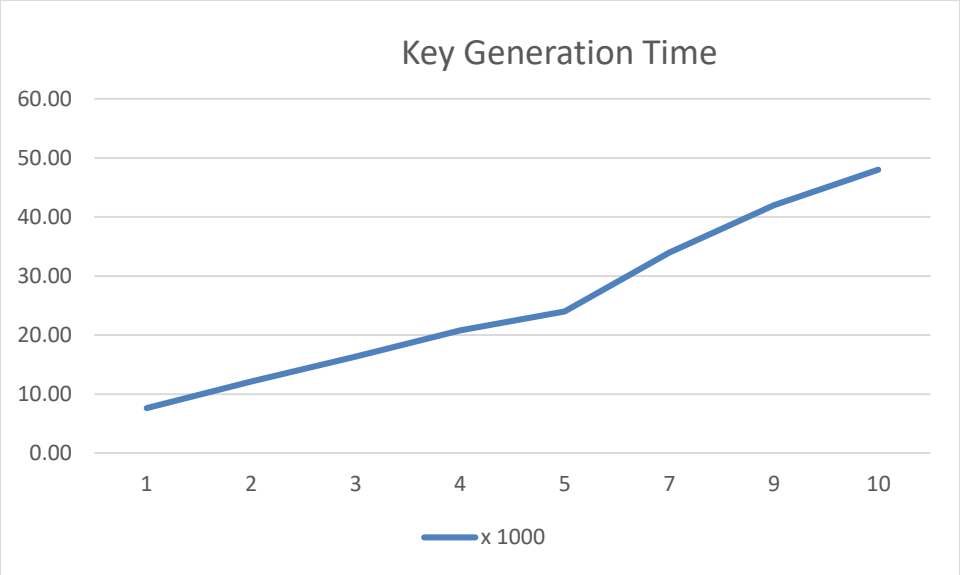


Figure 25: Accounts generation and time consumption

Also the generation of large number of key pairs which depends on the number of voters is a onetime process. Therefore, it is not an issue if it consumed more time than the experimental records.

Unfortunately, the ganache network which is used as a test net for the Blockchain in this implementation does not support clustering, also transactions are mined instantly with no cost, which will make it useless to test Blockchain performance because signing any transaction does not consume any calculation resources like the signing in production network.

Security Concerns

The only secured data is the non-existence one. In other words, there is no %100 secure system [40]. Therefore, like any other system there some security concerns about the data, the flow, and the behavior of the people involved in the E-Voting system process.

The concerns are list in three main points:

- The voters key pairs should be secured in all data states, which are storage, transmission and processing.
- The flow of the process should be monitored to avoid any manipulation or cheating. The privileges should also be restricted and apply Zero Trust theory in all election Stages.
- The Zero Trust should be applied in the environment [41], the administrators, employees, and even the high Authorities should be monitored.

Chapter Six Conclusion and Future Work

Conclusion and Future Works

Conclusion

Our implementation of E-Voting system has been completed. This system has fulfilled the minimum requirement of any Voting System, which was discussed in the results section mentioned before. which are; Completeness, Soundness, Privacy, Un-Reusable, Eligibility, Fairness, Robustness, Incoercibility, Convenience. Beside the contribution of our work solved the conflict between Receipt-freeness and Verifiable. While Robustness, Incoercibility, Convenience are met by applying the overall architecture.

The completeness and soundness are related to the environment more than the system. while Un-Reusable is by Blockchain by nature. Fairness is justified by registration before the election start.

We met the Verifiability by adding PIN codes, and Receipt-freeness by making them random and uncontrolled by the Voter.

Future Work

The is no such project “perfect”, but at least an E-Voting System built on Blockchain that provides integrity and verifiability is a success, our work could be a step in any future related E-Voting System in Palestine. Especially if a PKI project is done.

Performance measurement, integration with PKI, and overall verification as a function should be taken in concern in future work.

Also a review to the regulation and local law regarding elections should be considered in any future work.

Bibliography

[1] V. Parida, D. Sjödin and W. Reim, "Reviewing literature on digitalization, business model innovation, and sustainable industry: Past achievements and future promises.," in *Sustainability 11.2*, 2019.

[2] D. Zissis and D. Lekkas, "Securing e-Government and e-Voting with an open cloud computing architecture.," in *Government Information Quarterly 28.2*, 2011.

[3] J. P. Gibson, R. Krimmer, V. Teague and J. Pomares, "A review of e-voting: the past, present and future.," *Annals of Telecommunications*, pp. 279-286, 2016.

[4] J. P. Nogueira and F. d. Sá-Soares, "Trust in e-voting systems: a case study.," in *Mediterranean Conference on Information Systems*, Berlin, 2012.

[5] X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso and P. Rimba, "A taxonomy of blockchain-based systems for architecture design.," in *IEEE international conference on software architecture (ICSA)*, 2017.

[6] R. Albert and A.-L. Barabási, "Emergence of scaling in random networks.," *science 286.5439*, pp. 509-512, 1999.

[7] S. Nakamoto, "A peer-to-peer electronic cash system.," 2008. [Online].

[8] N. Kube, "Daniel Drescher: Blockchain basics: a non-technical introduction in 25 steps.," pp. 329-331, 2018.

[9] M. H. Miraz and M. Ali, "Applications of blockchain technology beyond cryptocurrency.," *arXiv preprint arXiv*, 2018.

[10] L. Rura, B. Issac and M. Kumar Haldar, "Implementation and Evaluation of Steganography Based Online Voting System.," *International Journal of Electronic Government Research (IJEGR)*, pp. 71-93, 2016.

[11] T. W. Lauer, "The risk of e-voting.," *Electronic Journal of E-government 2.3*, pp. 177-186, 2004.

[12] P. Van den Besselaar, A.-M. Oostveen, F. De Cindio and D. Ferrazzi, "Experiments with e-voting technology: experiences and lessons.," 2003.

[13] K.-H. Wang, S. K. Mondal, K. Chan and X. Xie, "A review of contemporary e-voting: Requirements, technology, systems and usability.," *Data Science and Pattern Recognition 1.1*, pp. 31-47, 2017.

[14] K. Garg, P. Saraswat, S. Bisht, S. K. Aggarwal, S. K. Kothuri and S. Gupta, "A comparitive analysis on e-voting system using blockchain.," in *4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, 2019.

[15] R. Hanifatunnisa and B. Rahardjo, "Blockchain based e-voting recording system design.," in *11th International Conference on Telecommunication Systems Services and Applications (TSSA).*, 2017.

[16] G. G. Dagher, P. B. Marella, M. Milojkovic and J. Mohler, "Broncovote: Secure voting system using Ethereum ’s blockchain.," 2018.

[17] Curran and Kevin, "E-Voting on the Blockchain.," *The Journal of the British Blockchain Association 1.2*, 2018.

[18] F. Þ. Hjálmarsson, G. K. Hreiðarsson, M. Hamdaqa and G. Hjálmtýsson, "Blockchain-based e-voting system.," in *IEEE 11th International Conference on Cloud Computing (CLOUD).*, 2018.

[19] D. Khoury, E. F. Kfoury, A. Kassem and H. Harb, "Decentralized voting platform based on Ethereum blockchain.," *IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*, 2018.

[20] F. Fusco, M. Ilaria Lunesu, F. E. Pani and A. Pinna, "Crypto-voting, a Blockchain based e-Voting System.," *KMIS*, 2018.

[21] N. Faour, "Transparent voting platform based on permissioned blockchain.," *arXiv preprint arXiv*, 2018.

[22] A. Barnes, C. Brake and T. Perry, "Digital Voting with the use of Blockchain Technology.," *Team Plymouth Pioneers-Plymouth University*, 2016.

[23] Y. Abuidris, R. Kumar, T. Yang and J. Onginjo, "Secure large-scale E-voting system based on blockchain contract using a hybrid consensus model combined with sharding.," *Etri Journal 43.2*, pp. 357-370, 2021.

[24] F. Salem, "Enhancing Trust in e-Voting through Knowledge Management: The case of the UAE.," in *MANAGING KNOWLEDGE TO BUILD TRUST IN GOVERNMENT*, New York, 2007.

[25] F. J. Shat and P. Abbott, "The main factors affecting e-voting service implementation: the case of Palestine.," in *Complexity in Information Systems Development.*, 2017.

[26] H. Ali and H. A. Mubarak, "e-Voting: An investigation of factors that affect public trust in Kingdom of Bahrain.," in *International Journal of Electronic Government Research (IJEGR) 14.2*, 2018.

[27] R. T. Monroe, A. ,. M. Kompanek and D. Garlan, "Architectural styles, design patterns, and objects.," in *software 14.1*, 1997.

[28] Y. Meng and J. Li., "Data sharing mechanism of sensors and actuators of industrial IoT based on blockchain-assisted identity-based cryptography.," in *Sensors 21.18*, 2021.

[29] F. Saleh, "Blockchain without waste: Proof-of-stake.," in *The Review of financial studies 34.3* , 2021.

[30] U. Maurer, "Modelling a public-key infrastructure.," in *European Symposium on Research in Computer Security*, Berlin, 1996.

[31] M. Pawlak, J. Guziur and A. Poniszewska-Marańda, "Voting process with blockchain technology: auditable blockchain voting system.," *International Conference on Intelligent Networking and Collaborative Systems. Springer,* 2018.

[32] K. M. Khan, J. Arshad and M. Mubashir Khan, "Secure digital voting system based on blockchain technology.," *International Journal of Electronic Government Research (IJEGR)*, pp. 53-62, 2018.

[33] Z. Yang, K. Yang, L. Lei, K. Zheng and V. CM Leung, "Blockchain-based decentralized trust management in vehicular networks.," *Internet of Things Journal 6.2,* pp. 1495-1505, 2018.

[34] S. Jairam, J. Gordijn, I. da Silva Torres, F. Kaya and M. Makkes, "A decentralized fair governance model for permissionless blockchain systems.," in *Proceedings of the International Workshop on Value Modelling and Business Ontologies*, 2021.

[35] R. L. Rivest, A. Shamir and L. Adleman, "On Digital Signatures and Public-Key Cryptosystems.," in *Massachusetts Inst of Tech Cambridge Lab for Computer Science,* 1977.

[36] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms.," in *IEEE transactions on information theory 31.4*, 1985.

[37] R. L. S. Rivest and Y. Tauman, ""How to leak a secret."," in *International Conference on the Theory and Application of Cryptology and Information Security.*, Berlin, 2001.

[38] M. Wohrer and U. Zdun, "Smart contracts: security patterns in the ethereum ecosystem and solidity.," in *International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, 2018.

[39] E. Borgsten and O. Jiang, "Authentication using Smart Contracts in a Blockchain. MS thesis.," *MS Thesis*, 2018.

[40] S. Williams, Cryptography and network security: Principles and practices., Pearson Education 17, 2006.

[41] L. Carr, A. J. Newtson and J. Joshi, "Towards modernizing the future of American voting.," in *IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, 2018.

Appendices

Appendix A Smart Contract Using Solidity

```
pragma solidity ^0.5.10;
contract voting {
  uint TotalVotes;
  bool private Finished;
  bool private Active;
  mapping(uint => string) public votes;
  mapping(address => bool) public votedlist;
  mapping(address => bool) public voterList;
  constructor()  public{
      TotalVotes=0;
      Finished=false;
      Active=false;
  }
  function doVote(string memory _vote)  public returns(bool){
    if ( (isFinished()==true) || (isActive()==false)) {
        return false;
    }else {
        TotalVotes+=1;
        votes[TotalVotes-1]=_vote;
        //votedlist[_addr]=true;
        return true;
    }
  }
  function getTotalVotes() public view returns(uint) {
    return TotalVotes;
  }
  function getVoteByKey(uint _key) public view returns( string memory){
    return votes[_key];
  }
  function setVoted(address _addr)public {
    votedlist[_addr]=true;

  }
  function isVoted(address _addr) public view returns(bool){
    return votedlist[_addr];
  }
  function isVoter(address _addr) public view returns(bool){
    return voterList[_addr];
  }
  function addRemoveVoter(address _addr,bool _status) public {
    voterList[_addr]=_status;
  }
```

```
  function Finish()public{
    Finished=true;
  }
  function isFinished()public view returns (bool){
    return Finished;
  }
  function Activate() public{
    Active=true;
  }
  function isActive() public view returns(bool){
    return Active;
  }
}
```

Appendix B Ring Signature Using C#

```
using System;
using System.Text;
using System.Security.Cryptography;
using System.Numerics;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Globalization;
using System.Data.SqlClient;

    public partial class _Default : Page
    {
        public bool dbg = false;
        public class myPoint
        {
            public myPoint(BigInteger _x, BigInteger _y)
            {
                X = _x;
                Y = _y;
            }
            public BigInteger X;
            public BigInteger Y;
        }
        public class Key
        {
            public BigInteger[] Pub;
            public BigInteger Priv;
```

```
        public bool isSigner;
    }
    public BigInteger ParseIt(String x)
    {
        if (x.StartsWith("0x"))
        {
            x = x.Replace("0x", "0");
            return BigInteger.Parse(x, NumberStyles.AllowHexSpecifier);
        }
        else
        {
            return BigInteger.Parse(x);
        }
    }
    protected void Page_Load(object sender, EventArgs e)
    {

        List<Key> keys = new List<Key>();

        int _s = 0;
        String connectionString = "Data Source=DESKTOP-VPJP6SI\\SQLEXPRESS;Initial
Catalog=bcUsers;User ID=user;Password=Password";
        SqlConnection conn = new SqlConnection(connectionString);
        conn.Open();
        if (Request.Form.AllKeys.Length == 0)
        {
            return;
        }

        string btnVerify = Request.Form.Get("Verify");
        string btnSign = Request.Form.Get("Sign");
        string strSignature = Request.Form.Get("txtSignature");
        string strSignedData = Request.Form.Get("txtSignedData");
        string strDataToSign = Request.Form.Get("txtmsg");
        string strSigner = Request.Form.Get("txtSigner");
        String strAPI = Request.Form.Get("API");
         if (btnVerify == "Verify")
        {
            String md5 = "0x" + getMD5(strSignedData);
             doVerify(strSignature);
            return;
        }
        if (btnSign == "Sign")
        {
```

```
            SqlCommand cmd = new SqlCommand("select * from (select PrivateKey,PublicKey,1
as _s,'A' as Ord from voters where id=" + strSigner +
                " union select top 3 '0x0' as PrivateKey,PublicKey,0 as _s, 'B' as Ord from Voters
where id <>" + strSigner + ") as t order by ord ", conn);
            SqlDataReader dr = cmd.ExecuteReader();

            int cnt = 0;
            while (dr.Read())
            {
               Key mk = new Key();
               mk.Priv = ParseIt(dr.GetValue(0).ToString());
               String testLen = dr.GetValue(1).ToString();

               String testStr = "0x0" + dr.GetValue(1).ToString().Substring(2, 64);
               String testStr2 = "0x0" + dr.GetValue(1).ToString().Substring(66, 64);
               mk.Pub = new[] { ParseIt(testStr), ParseIt(testStr2) };

               mk.isSigner = false;
               if (dr["_s"].ToString() == "1")
               {
                  _s = cnt;
                  mk.isSigner = true;
               }
               cnt = cnt + 1;
               keys.Add(mk);

            }
            dr.Close();

            conn.Close();
         }


         String myMessage = strDataToSign;



         List<BigInteger> ix = new List<BigInteger>();
         List<BigInteger> sx = new List<BigInteger>();
         BigInteger Arb = Arbitary(786854332);
         BigInteger[] OPrimZ = ConstructOPrim(Arb * Arb, keys[0].Priv, Arb);
         BigInteger[] OPrim = { OPrimZ[0], OPrimZ[1] };
         String bitsOPrim = OPrim[0].ToString("X") + OPrim[1].ToString("X");
         BigInteger bnOPrim = BigInteger.Parse("0" + bitsOPrim,
NumberStyles.AllowHexSpecifier);
         ix.Add(GetH(bnOPrim, myMessage));
```

```
        sx.Add(0);
        for (int i = 1; i < keys.Count; i++)
        {
            BigInteger arb = Arbitary(789789789);
            sx.Add(arb);
            BigInteger[] tmpOa = getOkis(keys[i].Pub, ix[i - 1], arb);
            String tmpo1 = tmpOa[0].ToString("X");
            String tmpo2 = tmpOa[1].ToString("X");
            BigInteger tmpO = BigInteger.Parse("0" + tmpo1 + tmpo2,
NumberStyles.AllowHexSpecifier);
            ix.Add(GetH(tmpO, myMessage));

        }
        BigInteger last_s = getSforI(keys[0].Priv, ix[ix.Count - 1], OPrimZ[2]);
        BigInteger[] test = getOkis(keys[0].Pub, ix[ix.Count - 1], last_s);
        String lastTest = "0" + test[0].ToString("X") + test[1].ToString("X");
        BigInteger testBn = BigInteger.Parse(lastTest, NumberStyles.AllowHexSpecifier);
        sx[0] = last_s;
        String FinSig = "{" + myMessage;
        for (int i = 0; i < ix.Count; i++)
        {
            FinSig += "," + keys[i].Pub[0].ToString("X64") + "," +
keys[i].Pub[1].ToString("X64") + "," + ix[i].ToString("X32") + "," + sx[i].ToString("X2");
            BigInteger[] nO = getOkis(keys[i].Pub, ix[i], sx[i]);
            BigInteger bitO = BigInteger.Parse("0" + nO[0].ToString("X2") +
nO[1].ToString("X2"), NumberStyles.AllowHexSpecifier);
            BigInteger w = GetH(bitO, myMessage);

        }
        FinSig += "}";

        if (strAPI == "API")
        {
            Response.Write(FinSig);
            return;
        }
        print("final Signiture:<p>" + FinSig);
        int start = 3;
        BigInteger t = Verify(keys[(start + 1) % ix.Count].Pub, ix[(start) % ix.Count], sx[(start +
1) % ix.Count], myMessage);
        BigInteger StartBn = ix[start];
        BigInteger _last_i = 0, _last_s = 0;

        Key _last_Key = new Key();
        for (int i = (start + 1) % ix.Count; i != start; i = (i + 1) % ix.Count)
        {
```

```
            t = Verify(keys[(i + 1) % sx.Count].Pub, t, sx[(i + 1) % sx.Count], myMessage);
            _last_i = t;
            _last_s = sx[(i + 2) % sx.Count];
            _last_Key = keys[(i + 2) % sx.Count];
        }

        t = Verify(_last_Key.Pub, _last_i, _last_s, myMessage);
        if (dbg) print("Finish result:" + t.ToString("X"));
    }
    //-------------------------------------------------------
    public void doVerify(String strSign)
    {
        String noBrackets;
        noBrackets = strSign.Replace("{", "");
        noBrackets = noBrackets.Replace("}", "");
        String[] sign = noBrackets.Split(',');
        String myMessage = sign[0];
        List<Key> keys = new List<Key>();
        List<BigInteger> ix = new List<BigInteger>();
        List<BigInteger> sx = new List<BigInteger>();
        List<BigInteger> LBN = new List<BigInteger>();

        for (int i = 1; i < sign.Length; i++)
        {
            LBN.Add(ParseIt("0x0" + sign[i]));

        }
        for (int i = 0; i < LBN.Count; i += 4)
        {
            Key k = new Key();
            // String strLBN = LBN[i].ToString("X");
            //strLBN = "0x" + strLBN;
            // String Part1 = "0x0" + .Substring(2, 64);
            //String Part2 = "0x0" + strLBN.Substring(66);
            //print("len Key:" + strLBN.Length);
            k.Pub = new[] { (LBN[i]), (LBN[i + 1]) };
            keys.Add(k);
            ix.Add(LBN[i + 2]);
            sx.Add(LBN[i + 3]);
        }

        int start = 3;
        print("Keyx:" + keys[start].Pub[0].ToString("X") + " y:" +
keys[start].Pub[1].ToString("X"));
        BigInteger t = Verify(keys[(start + 1) % ix.Count].Pub, ix[(start) % ix.Count], sx[(start +
1) % ix.Count], myMessage);
```

```
    print("start point:" + ix[start].ToString("X"));
    print("test result:[" + start + "]" + t.ToString("X"));
    BigInteger StartBn = ix[start];
    BigInteger _last_i = 0, _last_s = 0;
    Key _last_Key = new Key();
    for (int i = (start + 1) % ix.Count; i != start; i = (i + 1) % ix.Count)
    {
        print("Keyx:" + keys[i].Pub[0].ToString("X") + " y:" + keys[i].Pub[1].ToString("X"));
        t = Verify(keys[(i + 1) % sx.Count].Pub, t, sx[(i + 1) % sx.Count], myMessage);
        print("test result:[" + i + "]" + t.ToString("X"));
        _last_i = t;
        _last_s = sx[(i + 2) % sx.Count];
        _last_Key = keys[(i + 2) % sx.Count];
    }
    if (_last_i == StartBn)
    {
        print("<B><div ><font color=green>Success</font></div>");
        // print("Sig:" + FinSig);

    }
    else
    {
        print("<B><div ><font color=red>Failed</font></div>");
    }

}
//--------------------------------------------------------------------------------
public BigInteger Verify(BigInteger[] _pub, BigInteger _i, BigInteger _s, String _m)
{
    BigInteger[] sg = ECMultiplication(_s, g);
    BigInteger[] ik = ECMultiplication(_i, _pub);
    BigInteger[] ret = ECaddition(ik, sg);
    String bitStr = "0" + ret[0].ToString("X") + ret[1].ToString("X");
    BigInteger bits = BigInteger.Parse(bitStr, NumberStyles.AllowHexSpecifier);
    BigInteger h = GetH(bits, _m);
    return h;
}
public BigInteger[] getOkis(BigInteger[] _pub, BigInteger i, BigInteger s)
{

    BigInteger[] sg = ECMultiplication(s, g);
    BigInteger[] ik = ECMultiplication(i, _pub);
    BigInteger[] ret = ECaddition(ik, sg);

    return ret;
```

```
}
public BigInteger getOkisBn(BigInteger[] _pub, BigInteger i, BigInteger s)
{

   BigInteger[] sg = ECMultiplication(s, g);
   BigInteger[] ik = ECMultiplication(i, _pub);
   BigInteger[] ret = ECaddition(ik, sg);

   String bitStr = "0" + ret[0].ToString("X") + ret[1].ToString("X");
   BigInteger _ret = BigInteger.Parse(bitStr, NumberStyles.AllowHexSpecifier);

   return _ret;
}
public BigInteger[] ConstructOPrim(BigInteger i, BigInteger _Priv, BigInteger _s)
{
   BigInteger Z = (i * _Priv) + _s;
   BigInteger[] oPrim = ECMultiplication(Z, g);
   BigInteger[] RoPrim = { oPrim[0], oPrim[1], Z };
   return RoPrim;
}
public BigInteger getSforI(BigInteger Priv, BigInteger _i, BigInteger Z)
{
   BigInteger t = 0;
   t = Z - (_i * Priv);
   return t;
}
public BigInteger GetH(BigInteger O, String m)
{
   string str = getMD5(m);
   BigInteger mh = BigInteger.Parse("0" + str, NumberStyles.AllowHexSpecifier);
   string oStr = O.ToString("X");
   string H = oStr + mh.ToString("X");
   string stri = getMD5(H);
   BigInteger i = BigInteger.Parse("0" + stri, NumberStyles.AllowHexSpecifier);

   return i;
}
private static Random random;
private static object syncObj = new object();
public BigInteger Arbitary(int n)
{

   lock (syncObj)
   {
      if (random == null)
         random = new Random();
```

```
            BigInteger b = n ^ new BigInteger(random.Next(n));
            String mdstr = getMD5(b.ToString("X"));
            BigInteger _n = BigInteger.Parse("0" + mdstr, NumberStyles.AllowHexSpecifier);
            return _n;
        }
        // return n;
    }
    protected void print(string s)
    {
        Response.Write("<p>Out:" + s);
    }
    public string getMD5(string input)
    {
        // Create a new instance of the MD5CryptoServiceProvider object.
        MD5 md5Hasher = MD5.Create();

        // Convert the input string to a byte array and compute the hash.
        byte[] data = md5Hasher.ComputeHash(Encoding.Default.GetBytes(input));

        // Create a new Stringbuilder to collect the bytes
        // and create a string.
        StringBuilder sBuilder = new StringBuilder();

        // Loop through each byte of the hashed data
        // and format each one as a hexadecimal string.
        for (int i = 0; i < data.Length; i++)
        {
            sBuilder.Append(data[i].ToString("x2"));
        }

        // Return the hexadecimal string.
        return sBuilder.ToString();
    }




    static BigInteger p =
BigInteger.Parse("11579208923731619542357098500868790785326998466564056403945758400790883467166 3");
    static BigInteger n =
BigInteger.Parse("11579208923731619542357098500868790785283756427907490438260516314151816149433 7");
    static BigInteger[] zero = { 0, 0 };
    static BigInteger[] g = {
```

```
BigInteger.Parse("55066263022277343669578718895168534326250603453777594175500187360389116729240"),

BigInteger.Parse("32670510020758816978083085130507043184471273380659243275938904335757337482424")};

        public static void Main()
        {
            BigInteger[] g0 = { 0, 0 }, g2 = ECdouble(g), g4 = ECdouble(g2);
            BigInteger[] z = ECMultiplication(7, g);
            Console.WriteLine(z[0].ToString());
            Console.WriteLine(z[1].ToString());
            Console.WriteLine(IsOnCurve(z));
            Console.WriteLine(IsOnCurve(zero));
        }

        public static BigInteger inverse(BigInteger a, BigInteger m) { return BigInteger.ModPow(a, m - 2, m); }
        public static BigInteger[] ECdouble(BigInteger[] point)
        {
            if (point[1] == 0) return zero;
            BigInteger slope = (3 * BigInteger.ModPow(point[0], 2, p) * inverse((2 * point[1]), p)) % p;
            BigInteger x = (BigInteger.ModPow(slope, 2, p) - (2 * point[0])) % p;
            BigInteger y = (slope * (point[0] - x) - point[1]) % p;
            if (x < 0) x += p;
            if (y < 0) y += p;
            BigInteger[] coord = { x, y };
            return coord;

        }
        public static BigInteger[] ECaddition(BigInteger[] point1, BigInteger[] point2)
        {
            if (point1[1] == 0) return point2;
            if (point2[1] == 0) return point1;
            if (point1[0] == point2[0])
            {
                if (point1[1] == point2[1]) return ECdouble(point1);
                return zero;
            }
            BigInteger slope = ((point2[1] - point1[1]) * inverse(point2[0] - point1[0], p)) % p;
            BigInteger x = (BigInteger.ModPow(slope, 2, p) - point1[0] - point2[0]) % p;
            BigInteger y = ((slope * (point1[0] - x)) - point1[1]) % p;
            if (x < 0) x += p;
            if (y < 0) y += p;
```

```
        BigInteger[] coord = { x, y };
        return coord;
    }

    public static BigInteger[] ECMultiplication(BigInteger k, BigInteger[] Gpoint)
    {
        BigInteger[] powerOfTwo = Gpoint;
        BigInteger[] result = zero;
        k %= n; if (k < 0) k += n;
        while (k > 0)
        {
            if ((k & 1) == 1) result = ECaddition(result, powerOfTwo);
            k >>= 1;
            powerOfTwo = ECdouble(powerOfTwo);
        }
        return result;
    }
    public static bool IsOnCurve(BigInteger[] point)
    {
        BigInteger x = point[0] % p; if (x < 0) x += p;
        BigInteger y = point[1] % p; if (y < 0) y += p;
        return BigInteger.ModPow(y, 2, p) == (BigInteger.ModPow(x, 3, p) + 7) % p;
    }
}
```

Appendix C Javascript Functions Using Web3

```
<script language="javascript">
var publicKey = `
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDlOJu6TyygqxfWT7eLtGDwajtN
FOb9I5XRb6khyfD1Yt3YiCgQWMNW649887VGJiGr/L5i2osbl8C9+WJTeucF+S76
xFxdU6jE0NQ+Z+zEdhUTooNRaY5nZiu5PgDB0ED/ZKBUSLKL7eibMxZtMlUDHjm4
gwQco1KRMDSmXSMkDwIDAQAB
-----END PUBLIC KEY-----`;
bcServer="http://localhost:7545";//this is ganache server host and port
const web3 = new Web3(Web3.givenProvider || bcServer);
var acc;// acc stands for account
var TV=0;// TV Stands for TotalVoters
const getAccounts = async () => {// changed to wait activation
console.log("private key is:",voterPrivKey); //voterPrivKey is loaded from the server side after
authentitication
```

```
  let account = await web3.eth.accounts.privateKeyToAccount(voterPrivKey)
var balance=await (web3.eth.getBalance(account.address));
acc=account;
  console.log(acc.address," balance:",balance);
}
async function LoadAcc(){
  console.log("Loading private key is:",voterPrivKey);
    let account = await web3.eth.accounts.privateKeyToAccount(voterPrivKey)
  var balance=await (web3.eth.getBalance(account.address));
  acc=account;
    console.log("Loading", acc.address," balance:",balance);
}
async function hasVoted(_addr){
    const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
    result=await voting.methods.isVoted(_addr).call();
    console.log("has voted check:",result);
    return result;
}

async function isVoter(_addr){
    const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
   console.log("geting",_addr,"  status");
     result =await  voting.methods.isVoter(_addr).call();
    console.log("result:",result);
     return result;
  }

 async function getTotalVotes(){
  const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
  TV= await voting.methods.getTotalVotes().call();
   console.log("vaue=",TV);
   return TV;
}

 function getVoteByKey(_key){
   const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
var vote=    voting.methods.getVoteByKey(_key).call();
    console.log("Vote=",vote);
}
function countForCand(v){
  const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
var tmpV=  getTotalVotes();
var i=0;
var x=0;
  for (i=0;i<tmpV;i++){
  console.log(voting.methods.getVoteByKey(i).call());
```

```
}
}

async function getTV(){
 const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
 console.log("calling total votes");
 var result=-1;
  result =await  voting.methods.getTotalVotes().call();
  console.log(result);
 console.log("Called");
return result;
}
async function addRemoveVoter(_addr,_addrem){
  const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
  console.log("setting",_addr," Voter to:",_addrem);
  var result=false;
   result =await  voting.methods.addRemoveVoter(_addr,_addrem).send();
   console.log("result:",result);
   return result;
}
async function amIActive(_item,_addr){
tmp=await isVoter(_addr);
if (tmp.status){
  console.log("yes",_item);

}else {
  console.log("no:",_item);
}
alert("test:",tmp);
return tmp;
}
async function isFinished(){
    const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
    result=await voting.methods.isFinished().call();
    console.log("Finished?:",result);
    return result;
}
async function isActivated(){
    const voting=new web3.eth.Contract(ABI,smartAddr,{from:acc.address});
    result=await voting.methods.isActive().call();
    console.log("Activated?:",result);
    return result;
}
async function submitMyVote(_addr){
  _ballot=document.getElementById('chosenCand').value + ":" +
document.getElementById('myPin').value ;
```

```
 var balance=await (web3.eth.getBalance(smartAddr));
  console.log("Contract addr:",smartAddr," balance:",balance);
 console.log("submiting: balot",_ballot);
 result=await isActivated();
 if (!result){
    console.log("Election not started yet");
    document.getElementById("FinishStatusFail").innerHTML="Failed to Complete Voting";
  return false;
 }
 result=await isFinished();
 if (result){
    console.log("Election Finished Man!");
    document.getElementById("FinishStatusFail").innerHTML="Failed to Complete Voting";
    return false;
 }
 result=await isVoter(_addr);
 if (!result){
  console.log("failed not a Voter");
  return false;
 }else {//voter
    result=await hasVoted(_addr);
    if (result){
      console.log("Voter Already Voted ");
      return false;
    }else {//not voted
      let vaccount = await web3.eth.accounts.privateKeyToAccount('0x4f3edf983ac636a65a842ce7c78d9aa706d3b113bce9c46f30d7d21715b23b1d')
      const voting=new web3.eth.Contract(ABI,smartAddr,{from:vaccount.address,gasLimit: "1200000"});
      console.log("testing RSA");
      var encrypt = new JSEncrypt();
       encrypt.setPublicKey(publicKey);
  console.log("Account:",_addr," Voting: ",_ballot)
       console.log("Signing...");
    var strsig= await GetSignature(_ballot,VID);
       console.log("Sig:",strsig);
       var sigSlics= strsig;
       var encrypted="";
       for (i=0;i<sigSlics.length;i+=110){
         var slc=sigSlics.slice(i,110+i);
         console.log("Slice=",i," :",slc );
         if (i==0){
           encrypted=encrypt.encrypt(slc);
         }else {
         encrypted+=","+encrypt.encrypt(slc);
```

```
        }
        }
         try {
          console.log("Submitting:",encrypted);
             result=await voting.methods.doVote(encrypted ).send();
         } catch (err){
            console.log("Error:",err);
         }finally {
          console.log("Marking as Voted.....");
          res=await voting.methods.setVoted(_addr);
          if (res) {
            console.log("Marked As Voted");
          }
         }
         if (result) {
            console.log(result);
            res=await hasVoted(_addr);
            if (res){
               console.log("Successfuly Voted");
               document.getElementById("FinishStatusSuccess").innerHTML="You have
Successfuly Voted";
            }
         else {
               console.log("Failed:",res);
               document.getElementById("FinishStatusFail").innerHTML="Failed to Complete
Voting";
         }
       return result;
         }else {
            document.getElementById("FinishStatusFail").innerHTML="Failed to Complete
Voting";
       console.log("General Failure");
       return result;
     }
   }
   }
  }

// Using Axios
async function GetSignature(data,id) {
   var res = await
axios.post('/cring/default.aspx',{txtmsg:data,txtSigner:id,Sign:'Sign',API:'API'},{
       headers: {
          'Content-Type': 'multipart/form-data'
       }});
   if(res.status == 200){
```

```
        console.log(res.status)
    }
    return res.data;
}
</script>
```

الملخص

تعد الانتخابات المقياس الحقيقي للواقع الديموقراطي في البلدان، ولكي تكون الانتخابات شفافة ونزيهة، يجب على مؤسسات الدولة التي تُعنى بالانتخابات، أن تعمد الى كل ما من شأنه ان ينجح العملية الانتخابية. والنجاح في مثل هذه العمليات قابل للقياس.

ان التوجه نحو اتمتة معظم الاعمال من كتابة نص الى قيادة سيارة، بدء في تسارع كبير نحو الاعتماد على تكنولوجيا المعلومات والانترنت والذكاء الاصطناعي وما الى ذلك من ادوات في العقدين الاخيرين. إلا ان الذهاب نحو انتخابات تعتمد على الاتمتة والانترنت والتقنيات الجديدة يكاد يكون معدوما.

ففي بعض الدول تمت تجربة حوسبة الانتخابات وتفاوتت هذه النتائج ما بين العدول عنها وما بين اعتمادها بشكل تام.

ومع ظهور تقنية سلاسل الكتل، جاءت فكرة هذه الرسالة لبناء نظام انتخابات الالكتروني مبني على سلاسل الكتل. وقد كان اختيار هذه السلاسل لأنها توفر حماية لسلامة البيانات التي تحدث عليها، وقبل البحث في هذا المجال لم اكن اتوقع الكم الهائل من الابحاث في هذه التقنية وفي هذا الموضوع.

وبعد القراءة في مجالات سلاسل الكتل والنظر بإمعان في القضايا العالقة من خصائص الانتخابات الاكترونية المقترحة في البحوث العلمية التي صادفتها، وجدت ان بعض الصفات لم تكن لتتحقق، أو كانت الحلول لها بأشكال غير عملية. وانا هنا اتحدث عن خاصيتين متناقضتين وهما كيف لناخب ان يتحقق من أن صوته قد احتسب بشكل صحيح وفي ذات الوقت أن لا يكون بإمكانه ان يثبت ذلك الصوت لأحد غيره.

وكانت هذه الفكرة هي المحفز وراء البحث عن طريقة لعمل ذلك وتكللت بالنجاح باعتماد طريقة يختار فيها الناخب بالاضافة الى مرشحه رقم مكون من ارقام عشوائية سهلة الحفظ لكنه مجبر للاختيار منها تثبت مع الصوت باحدى طرق التشفير والتوقيع الالكتروني. وعليه فهو يستطيع مقارنة الاصوات المحصى بالارقام المرفقة ولا يستطيع ان يثبت لأحد غيره ان هذه الارقام قد اختارها هو.

إلا أنه لازال هنالك الكثير من العمل ليخرج هذا المشروع من مرحلة النجاح الجزئي الى النجاح الذي يؤهله ليكون ركيزة في اي مشروع لانتخابات الكترونية تعتمد سلاسل الكتل.