# DECENTRALIZED E-VOTING SYSTEM USING BLOCKCHAIN TECHNOLOGY

**By**

**ATORO TOLUWANI DANIEL**

**MATRIC NO: 18010301040**

**A PROJECT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE**
**AND MATHEMATICS, COLLEGE OF BASIC AND APPLIED SCIENCES,**
**IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS FOR THE**
**AWARD OF DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE**

**2022**

## DECLARATION

I hereby declare that this project has been written by me is a record of my own research work. It has not been presented in any previous application for a higher degree of this or any other University. All citations and sources of information are clearly acknowledged by means of reference.

**ATORO TOLUWANI DANIEL**

**Date**

# CERTIFICATION

This is to certify that the content of this project entitled **'DECENTRALIZED E-VOTING SYSTEM USING BLOCKCHAIN TECHNOLOGY'** was prepared and submitted by **ATORO TOLUWANI DANIEL** in partial fulfillment of the requirements for the degree of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE**. The original research work was carried out by him under by supervision and is hereby accepted.

---------------------------------------- (Signature and Date)

Dr. Akindele A. Onifade

B.Tech (Ogbomoso), MSc,Ph.D (Ibadan)

Supervisor

---------------------------------------- (Signature and Date)

Dr. M.O. Odim

Coordinator, Department of Computer Science and Mathematics

Accepted as partial fulfillment of the requirement for the degree of BACHELOR of SCIENCE (Computer Science)

# DEDICATION

This thesis is dedicated to the glory of Almighty God, **THE FATHER**, **THE SON** and **THE HOLY SPIRIT**, who has made this programme a success. I also dedicate this work to my father, my mother and my brothers for being a major source of support in every way.

# ACKNOWLEDGEMENT

Except the Lord builds the house, they labor in vain that build it..." Indeed, without the involvement of the Almighty God - Omnipotent, Omniscient, Omnipresent, this work would not have been a success. Great is thy faithfulness O Lord! I sincerely thank my supervisor, Dr. Akindele A. Onifade for his counsel, advice, instructions, free access and all the encouragement given to make this work a success. May the Lord guide, support, keep and promote him and his family.

# ABSTRACT

Many people's lives have been aided by digital technology in recent years. Unlike the election system, it employs a lot of traditional paper in its implementation. The issue of security and transparency is threatened by the extensive use of the traditional electoral method (offline). General elections are still run under a centralized system, which is overseen by a single institution. Some of the issues that can arise in traditional electoral systems include the ability for a corporation with complete control over the database and system to tamper with the database of significant opportunities. Blockchain technology is one of the solutions since it is based on a decentralized system in which multiple people own the complete database. The Bitcoin system, also known as the decentralized Bank system, has used blockchain. One of the main sources of database manipulation can be reduced by using blockchain in the dissemination of datasets on e-voting systems. This study examines the use of the blockchain algorithm to record voting results from every election location.

**Keywords:** *Blockchain technology, Decentralized environment, E-Voting, Privacy, Security, Trust.*

Table of Contents

# LIST OF FIGURES

Figures Page

# CHAPTER ONE

# INTRODUCTION

In today's society, e-voting is frequently used. However, when the decision is financially or politically significant, it is unclear how to assure that the verdict is honored. The most crucial characteristics are always correctness, security, and privacy. Secure e-voting is a type of multi-party computation that is done securely. During the voting process, a group of people make their decisions, which may or may not be kept private. To offer a consistent view to all voters, most e-voting techniques require a trustworthy public bulletin board. The election administrator, on the other hand, has yet to demonstrate that the public message board can be entirely trusted. Because the content is publicly trusted, some people understand blockchain may be utilized as a bulletin board.

As a decentralized database, blockchain offers new tools for developing trustless and decentralized systems. There is no centralized trustworthy coordinator in the blockchain system. Instead, the data block is stored locally by each node in the blockchain system. A decentralized and open-membership peer-to-peer network maintains the blockchain. Initially, this technology was created for the purpose of money transfer.

Researchers are attempting to repurpose Blockchain in various areas of research, such as coordinating the Internet of Things, carbon dating, and health-care. This spurred the creation of Ethereum, which is widely regarded as a watershed moment in the development of blockchain technology. It has a Turing complete programming language, and users can utilize the Ethereum network's smart contract to do the function.

The voting system might employ blockchain as a trusted public bulletin board. Furthermore, the blockchain smart contract functioned as a trusted computer whose output is publicly trusted. However, just substituting blockchain for the bulletin board is not a good idea. This might be seen in because there will be too many transactions for voters to detect and blockchain computation is extremely difficult. In this work, we propose a blockchain-based decentralized trustless e-voting system. The computation is based on a decentralized blockchain in a decentralized system. The trustless approach means that voters do not have to rely on the election administrator; instead, all voters share the same level of trust. The system's correctness is determined by the entire protocol. Furthermore, all voters will have cryptographic assurance that their privacy will be safeguarded.

The technique employs threshold encryption without the involvement of a trusted third party to ensure that no one can tally the election results before the end of the election. Furthermore, the tally result will not be changed even if the election administrator is hostile. Setting up a pair of public/secret keys is the encryption procedure. The public key is shared by all parties, but the secret key is kept separate and no one has access to the entire secret key before the key reconstruction stage. The secret key is reconstructed when at least n people contribute their secrets (Pathak., Suradkar., Kadam., Ghodeswar., & Parde., 2021).

A blockchain is a growing collection of blocks with cryptographic connections that began as a chain of blocks. Each block contains the preceding block's hash, timestamp, and transaction data. The blockchain was built with data security in mind. Voting is a transitional era of blockchain technology, and academics are attempting to capitalize on characteristics like transparency, secrecy, and non-repudiation, which are critical for voting applications. Efforts to use blockchain technology to protect and remedy elections have recently gained a lot of attention, thanks to the use of blockchain for electronic voting apps. (Jafar., Aziz., & Shukur., 2021).

## 1.1 Background to the Study

Due to the well-known initiatives in Bitcoin and Ethereum, the first things that come to mind when thinking about the blockchain are cryptocurrency and smart contracts. Bitcoin was the first digital currency to leverage the blockchain data structure. Ethereum introduced smart contracts, which take advantage of the immutability and distributed consensus of the blockchain while providing a crypto-currency solution that is equivalent to Bitcoin. Nick Szabo coined the term "smart contracts" in the 1990s, describing them as "a set of promises, specified in digital form, including protocols within which the parties fulfill on these promises" (Szabo, 1997). A smart contract is a piece of code that is deployed to the Ethereum network and accessible to everyone. The outcome of running this code is verified by a consensus method as well as by each individual member of the network (Wood, 2014).

A blockchain is now a collection of technologies that include the blockchain data format, a distributed consensus mechanism, public key cryptography, and smart contracts. We go over each of these technologies in greater depth below. Blockchain is a peer-to-peer network that builds a chain of blocks. As seen in Figure 1, each block in the blockchain has a cryptographic hash and a timestamp added to the previous block.

The Merkle tree block header and numerous transactions are contained within a block. Cryptography is a secure networking method that integrates computer science and mathematics to keep data and information hidden from others. It enables data to be securely sent in encrypted and decrypted versions across an unsecured network.

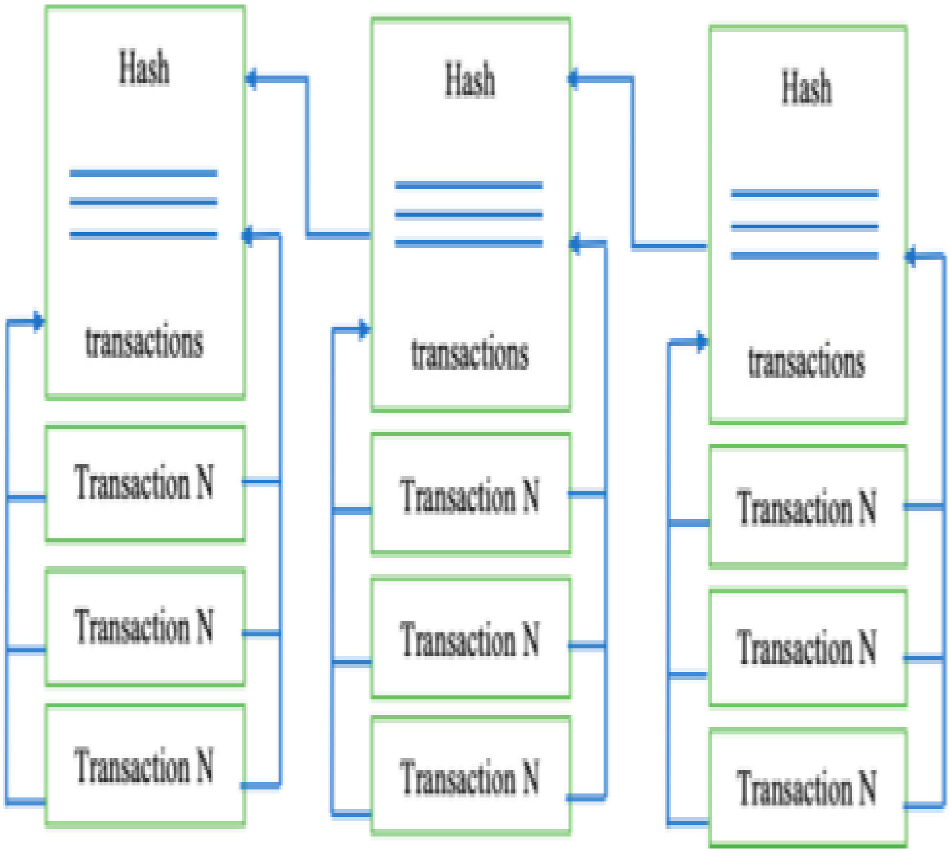


**Figure 1. 1 : The blockchain structure. (Jafar., Aziz., & Shukur., 2021)**

The goal of utilizing such a data structure is to establish immutability that can be shown. If a piece of data is modified, the hash of the block containing that piece of data must be computed, as must the hashes of all following blocks (Lejun., et al., 2021). To ensure that all data remains unaltered, just the hash of the most recent block must be used. Data contained in blocks in blockchain systems is produced during their creation from all validated transactions, which means no one can insert, delete, or edit transactions in a previously validated block without being noticed. The "genesis block," or the first zero-block, usually contains certain network parameters, such as the original set of validators (those who issue blocks).

### 1.1.1 Public key cryptography

The major applications of public key cryptography are: To determine the requester, all validators must possess their keypairs used to sign consensus messages, and all incoming transactions (requests to modify blockchain data) must be signed. In a blockchain setting, anonymity refers to the fact that anyone interested in using cryptocurrencies only needs to generate a random keypair and use it to control a wallet linked to a public key (Froomkin, 2016). The blockchain approach ensures that only the keypair owner has access to the wallet's funds, which is a provable property. When it comes to online voting, ballots must be accepted anonymously but only from people eligible to vote, therefore a blockchain cannot resolve the problem of voter confidentiality on its own.

### 1.1.2 Smart contracts

Smart contracts have given blockchain technologies a new lease on life. They promoted the use of blockchain technology to help improve a variety of areas. A smart contract is nothing more than a piece of code that expresses logic. Even so, when combined with the immutability of a blockchain data structure and widespread consensus, it can operate as an unconditionally trusted third party (Mohanta., Jena., Panda., & Sobhanayak., 2019).

It cannot be changed once it has been written, and all network members must check all stages. The beauty of smart contracts is that anyone with the ability to set up a blockchain node can verify its outcome.

## 1.2 Decentralized system

A decentralized system is an information network in which no single party has sole authority. Decentralized systems are often networked computers in the context of computing and information technology. The Internet, for example, is a decentralized system that has gotten more centralized over time. (ComputerHope, 2019)

## 1.3 Problem Statement

The current voting system used in the Nigerian electoral system has proven inefficient because the voter registration process is sluggish, the conventional gathering of results takes time and allows for result manipulation, and the difficult to access nature of election venues, which includes the vast distances that voters must travel to their registered location, has increased voters' apathy towards the electoral process.

The objective of voting in an election process as a formal procedure of expressing individual ideas for or against some motion is tainted by ballot box snatching and damage and other electoral violence, as well as concerns related with traditional ballot paper voting.

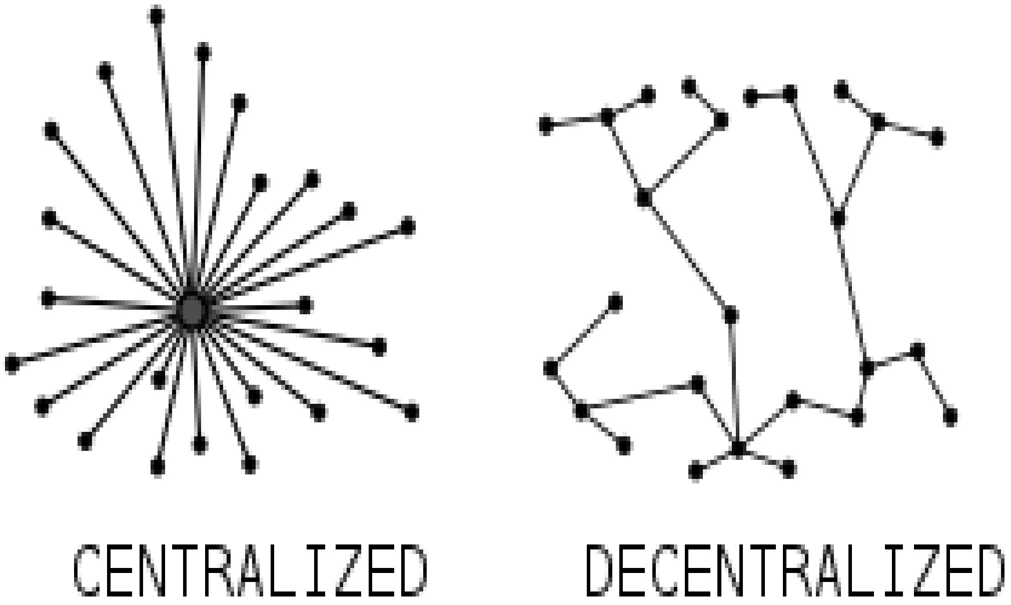


**Figure 1. 2: Graphical comparison of a centralized and a decentralized system.**

**(Wikipedia, 2022)**

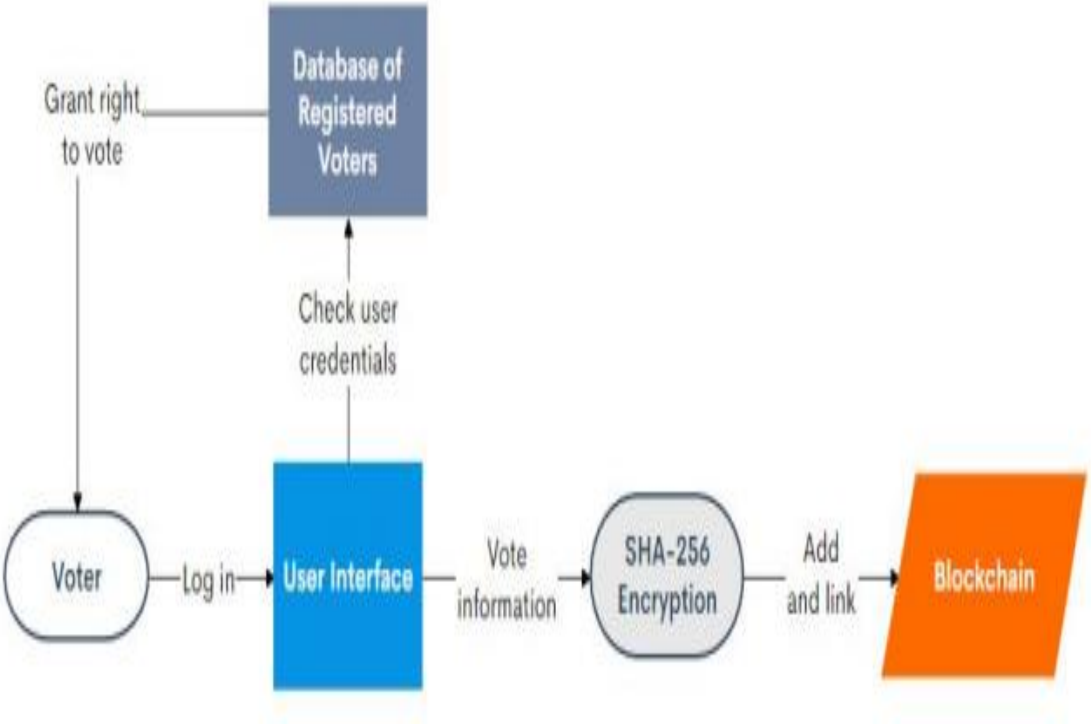


**Figure 1. 3: Simplified process of voting with a custom blockchain. (Dengo, 2020)**

## 1.4 Aim and Objectives

In the quest to design a successful system to tackle the issues stated in the problem statement, the aim and objectives of the project are outlined below.

### 1.4.1 Aim

The aim of this project is to develop an easy and efficient prototype of E-voting system based on digital application with enhanced security and protection of the system using blockchain.

### 1.4.2 Objectives

Project Objectives includes

1. A detailed study of the election processes as it pertains to voting.
2. Design and develop a software platform for voter registration, election voting, real-time election results collation and monitoring and mostly for voters remote access to elections
3. Implement a custom blockchain algorithm on the software platform that incorporates SHA-256 encryption
4. Design and develop an administration dashboard for the election administrators.
5. Run simulations and compare the results of the designed e-voting system and other voting systems

## 1.5 Significance of the project

In view of the rapid development of computer technology in virtually all fields of operation and its use in relation to information management, an e-voting system is beneficial to the university as It will provides a means conduct a more less stressful and fair elections at different levels (faculty, departments, school wide etc.) in the university.

To the society and mostly to Nigeria, it will provide INEC (Independent National Electoral Commission) with a means to conduct less costly and fair elections.

## 1.6 Definition of terms

i. Blockchain: A blockchain is a public ledger of transactions. The name comes from the database’s structure, which consists of individual records called blocks that are linked together in a single list called chain.

ii. Smart-contract: A smart contract is a technique for digitally validating contract agreements.

iii. Solidity: Solidity is a programming language for creating smart contracts that was created with n Ethereum’s Virtual Machine in mind.

iv. Framework: A software system is a physical or abstract platform that allows developers or users to specialize or avoid common code by using generic features.

v. Transcript: A transcript is proof of education.

vi. Exchange: that a partner institution accepts a student, but does not necessarily mean that the students have to find a counterpart from the other institution with whom to exchange.

vii. JavaScript: is a scripting language that enables you create dynamically updating content, control multimedia, animate images, often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS

viii. Database: is information that is set up for easy access, management and updating

# CHAPTER TWO

# LITERATURE REVIEW

In recent years, a number of publications have been published that have highlighted the security and privacy concerns with blockchain-based electronic voting systems. Reflects a comparison of a few blockchain-based electronic voting techniques.

The open vote network (OVN) was demonstrated by (McCorry, Shahandashti, & Hao, 2017), which is the first Ethereum-based deployment of an open and self-tallying internet voting protocol with complete user privacy. The voting size in OVN was limited by the framework to 50–60 electors. The OVN is powerless to prohibit rogue miners from destabilizing the system. By sending an invalid vote, a fraudulent voter can get around the voting procedure. The protocol makes no guarantees about violence resistance, and the electoral administrator wants to be trusted (Zhang, Wang, & Xiong, 2020) (Chaieb, Koscina, Yousfi, Lafourcade, & Robbana, 2020).

They also used an external library to perform the computation because Solidity does not support elliptic curve cryptography (Woda & Huzaini, 2021). The voting contract grew too large to be stored on the blockchain once the library was introduced. OVN is vulnerable to a denial-of-service attack because it has happened before in the Bitcoin network's history (Hjalmarsson, Hreioarsson, Hamdaqa, & Hjálmtýsson, 2018).

Lai et al. (Lai, Hsieh, Hsueh, & Wu, 2018) proposed a decentralized anonymous transparent electronic voting system (DATE) with a low level of participant confidence. They believe that the existing DATE voting mechanism is appropriate for large-scale electronic elections. Unfortunately, because there was no third-party authority on the scheme accountable for auditing the vote after the election process, their proposed solution is not robust enough to protect against DoS attacks. Because of the platform's limitations, this solution is only viable for modest sizes (Gao., Zheng., Guo., Jing., & Hu., 2019). Although Ring Signature protects individual voters' privacy, it is difficult to organize and coordinate multiple signer entities. They also employ PoW consensus, which has substantial limitations such as energy consumption: miners' "supercomputers" monitor a million computations each second, which occurs globally. This structure is costly and energy-intensive since it necessitates a lot of computer power.

The BSJC proof of completeness was proposed by Shahzad et al. as a trustworthy electronic voting technique. They utilized a system model to represent the overall structure of the system. It also aimed to address anonymity, privacy, and security issues in the election on a smaller scale. However, a slew of other issues has been raised. The proof of labor, for instance, is a mathematically complex and time-consuming task that necessitates a huge lot of effort. Another issue is involving a third party because there is a high chance of data alteration, spillage, and unfair tabulated results, all of which could affect end-to-end verification. The polling procedure may be delayed if the block is generated and sealed on a big scale (Shahzad & Crowcroft, 2019).

A blockchain-based anti-quantum electronic voting mechanism with an audit feature has been proposed by Gao et al. (Gao., Zheng., Guo., Jing., & Hu., 2019). They've also tweaked the code-based Niederreiter algorithm to make it more resilient to quantum attacks. The Key Generation Center (KGC) is a regulator and certificate-less cryptosystem. It not only protects the voter's confidentiality, but it also makes the audit process easier. However, a closer look at their approach indicates that even if the number of voters is tiny, the security and efficiency gains for a small-scale election are significant. If the number is large, some efficiency is sacrificed in order to improve security (Fernandez-Carames & Fraga-Lamas, 2020).

Yi (Yi, 2019) unveiled the Blockchain-based Electronic Voting Scheme (BES), which included approaches for enhancing electronic voting security in a peer-to-peer network utilizing blockchain technology. To prevent vote tampering, a BES based on the distributed ledger (DLT) could be used. On Linux systems in a peer-to-peer network, the system was tested and designed. Counter-measurement assaults are a key difficulty in this strategy. This strategy requires the engagement of responsible third parties and is not well suited to centralized use in a system with a large number of agents.

A distributed procedure, such as the use of secure multipart computers, may be able to solve the problem. However, if the computation function is sophisticated and there are too many participants, computing costs become increasingly substantial and possibly prohibitive. (Torra, 2019) (Alaya, Laouamer, & Msilini, 2020).

Khan, K.M. (Khan, Arshad, & Khan, 2020) developed the block-based e-voting architecture (BEA), which tested permissioned and permissionless blockchain designs under various scenarios involving voting population, block size, block production rate, and block transaction speed. Their studies also revealed surprising insights about how these characteristics influence the overall scalability and dependability of the electronic voting model, including interconnections between different parameters as well as internal security and performance measures. The electoral process, according to their plan, necessitates the generation of voter and candidate addresses. These addresses are then used to assign votes to candidates by voters. To keep track of votes cast and the progress of the vote, the mining group updates the record of the main blockchain. Until a miner updates the main record, the voting status is unverified. The vote is then cast at the polling location using a voting machine.

However, there are certain faults in this paradigm. There is no regulatory body to prevent invalid voters from voting, and the system is vulnerable to quantum attach. Their model is inaccurate, and they were unconcerned about voter integrity. Furthermore, because fewer people keep the network active, their strategy employing Distributed consensus allows testimony (data and facts) to be arranged into cartels, making a "51 percent" attack easier to organize.

This attack may be more concentrated, and it does not address the primary concerns about the blockchain voting system, namely scalability and delays in electronic voting. They employed the Multichain framework, which is a private blockchain developed from Bitcoin that is inappropriate for nationwide voting. As stated by the authors, this approach is only suitable for small and medium-sized voting contexts (Jafar., Aziz., & Shukur., 2021).

# CHAPTER THREE

# METHODOLOGY

The systems development approach is described in this section, along with the development phases the project underwent. A reuse-oriented software development process model was used to create the system. A system prototype was developed to discover flaws in a centralized voting system, which was then used to inform the development of a new requirement specification. The requirements were then validated using the test cases specified for the new system model.

## 3.1 System Development Process

The requirement in this system is the statement in domain-specific terms which specifies the verifiable constraint on the implementation that it should meet. Some of the requirements were gotten from developing a prototype system that uses a centralized database, with the flaws in this system highlighted and improvements made, a new system was developed using requirements gathered. The stages of development that the system would go through are:

1. Requirements Definition and Engineering.
2. Software and System Design.
3. Implementation and Unit testing.
4. Integration and System Testing.
5. Operation and Further Maintenance.

## 3.2 Requirements Definition and Engineering

The system was assessed during the requirement engineering stage of development, and the services that the suggested system needs, along with the numerous restrictions that determine the system's operation and development, were described in a precise manner.

The proposed system that was created in this project is known as e-LECT, and throughout the entirety of this paper, the project will be referred to by this name. Use case diagrams were used to explain and illustrate the User and System needs in this part, and both functional and non-functional requirements were mentioned. Use case diagrams would be employed as a requirement discovery tool to further clarify how the system's elements interact.

### 3.2.1 User Requirements

The functions the e-LECT system offers to system users and the limitations it is subject to are stated in the project's user requirements. The following are the user specifications for the e-LECT system:

1. Voting on multiple election occasions should be possible for users of e-LECT.
2. Voting on allowed electoral instances should be restricted to authenticated and authorized users.

What the system must provide for its users is outlined in the aforementioned standards. Users should have the option to cast their votes for the candidates they favor in the requirements given above. The use of the system is then limited or affected by certain constraints. Only users who have been verified should be able to access the system in this project scenario, and even then, users must receive permission from election officials in order to cast a ballot in that election instance. This step is an exact reproduction of the manual procedures that were previously in use, where you might be handed a voter identification card but couldn't vote in a state's governor race if you weren't from that state.

### 3.2.2 System Requirement Specifications

The system's requirements were broken down into a more thorough explanation in this phase of the project, which also covers the system's features, services, and operating limitations. The e-LECT System's system requirements are as follows:

1. Factory Elections ought to produce several copies of other elections.
2. Voting for the candidates of your choice should be possible in election instances.
3. The system should allow users to register.
4. Authorized users should cast their votes if there are still election instances available.
5. Voting should not be permitted in election situations after they have ended or been closed.

As previously stated, the smart contract being developed should have a factory instance. The purpose of a factory contract is to create multiple instances of an electoral process, with the person who creates the instance of this contract becoming a chairman or official, with several responsibilities. In each case of a deployed election instance, at least two candidates must be initialized in order for users to choose from the options presented to them. Every user must complete a registration process in order to be authenticated to use the platform.

As previously stated, the smart contract being developed should have a factory instance. The purpose of a factory contract is to create multiple instances of an electoral process, with the person who creates the instance of this contract becoming a chairman or official, with several responsibilities. For every case of a deployed election instance, at least two contenders must be initialized in order for users to choose from the options presented to them. Every user must complete a registration process in order to be validated to use the platform. Election instances can be opened and closed by the creator of such instances based on the time limit specified. Basically, whoever creates the election instance is the only person who can open and close the election; additionally, the creator of an election instance is the only person who can authorize users to vote in that instance.

### 3.2.3 Functional Requirements

The e-LECT functional requirements are statements about the services that the system provides, how the system reacts to specific inputs, and how the system behaves in specific situations. The e-LECT system must meet the following functional requirements:

1. Voters can vote in any of the authorized election instances.
2. Voters can select from a list of available candidates to vote for.
3. Each election instance should have its own set of variables and functions that define the electoral flow.
4. After successfully registering, each user should receive a system-generated voter ID.

### 3.2.4 Non-Functional Requirements

Non-functional requirements on the e-LECT system are system constraints that affect how the system behaves and operates based on the input parameters; in this case, non-functional requirements specify system speed, size, ease of use, portability, and reliability; these are system functionality constraints. All of these are not directly related to the specific services provided by the system; however, several other non-functional requirements, such as availability and security, have an indirect impact on the system's behavior. The flow of processes in the system is greatly influenced by security. The list of security constraints on various aspects of the system (for users - Voters) is as follows:

1. Users have to sign up on the systems server.
2. Users must have a ID that is made by the system.
3. When registering, each user must have a unique Ethereum address.

List of security restrictions on different parts of the system (For admins - creators of Election instance):

1. Admins have to sign up as "admin."
2. Admins should also have an ID that is made by the system.
3. Administrators can't vote as administrators.

The system suggests using a server-side rendering client-side application to create the application in order to increase speed and usability. This would enable effective human-computer interaction. The system also suggests hosting it online for public access before deploying the smart contract on the main IOTA network.

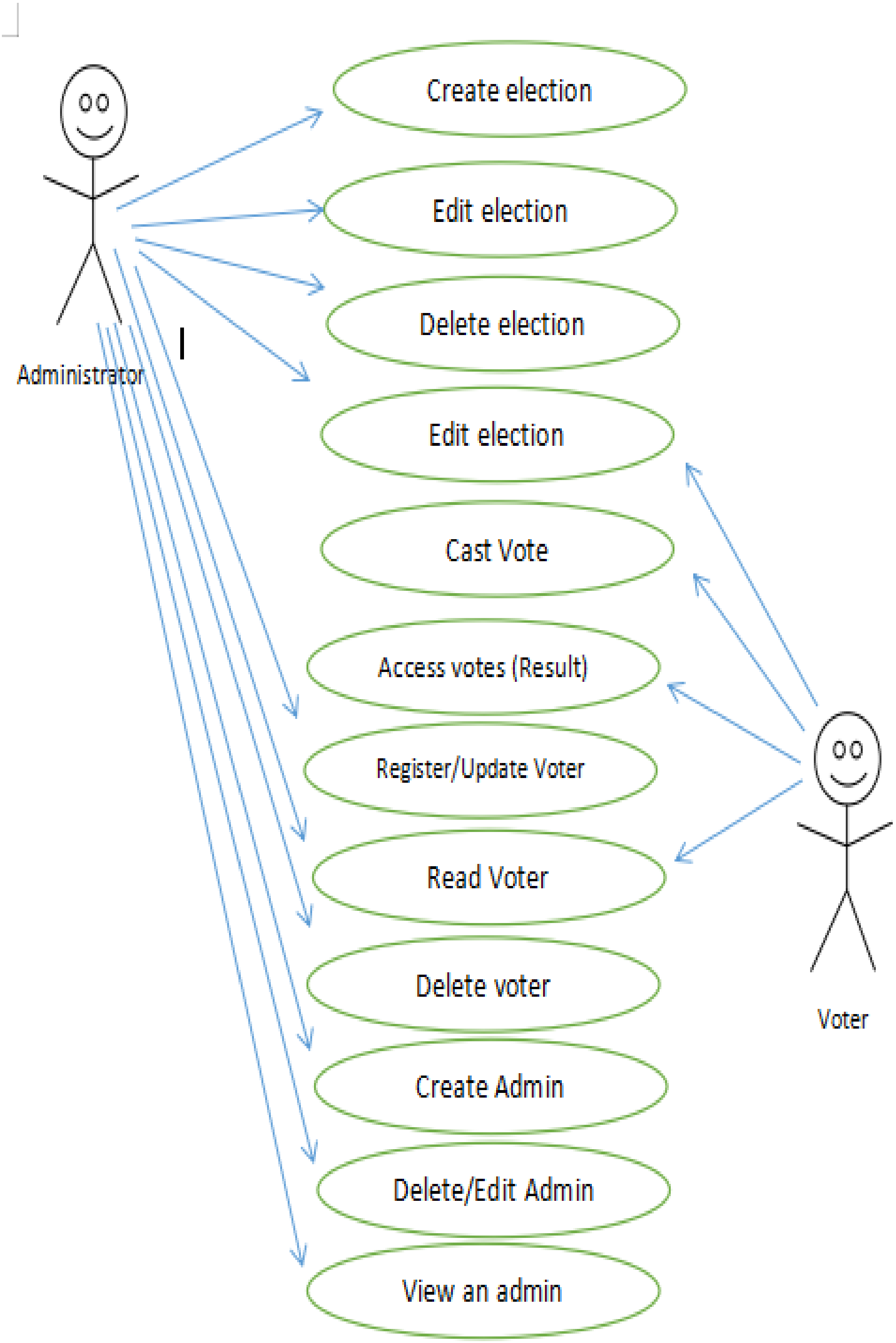


**Figure 3. 1: Use Case Diagram**

## 3.3 System and Software Design

Establishing an overarching system architecture and naming the key software system abstractions and their interactions is the first step in identifying the design process for the software. Several diagrams would be utilized to portray different parts of the program, and the standard system modeling phases would be adhered to. The requirements for the system would be used to generate designs for the system, and those designs would be consistent with the use case diagrams established during the requirement elicitation phase.

The design inputs, design activities, and design outputs make up the three stages of the general design model process. As parameters for developing the model of the system, we use the disclosed requirements (both functional and non-functional) as well as the needs of the user and the system itself. To proceed, we shall attempt to build the system's architectural design, the system's proposed interface, and the description of various and crucial data required at this level of the activity. The architecture of the smart contract is crucial to the planning of the infrastructure and the description of the data. The system architectural model would outline the many features of the smart contract, and the data descriptions would be shaped by the smart contract's architecture.

### 3.3.1 System Architecture

Figure 3.1 shows the overall design of the system. Each building block is discussed in detail below.

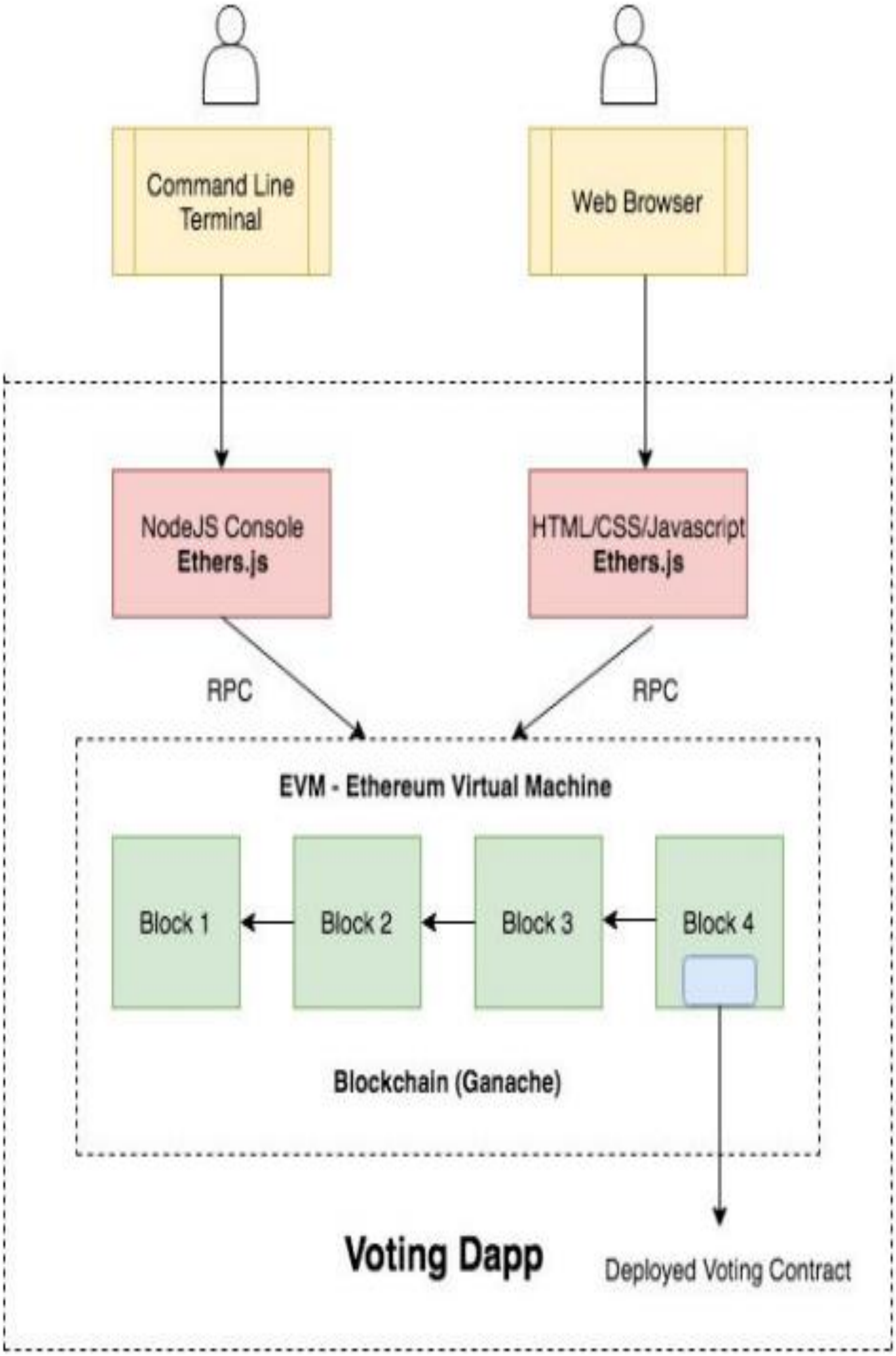


**Figure 3. 2: Proposed system architecture**

Registration is the starting point of our design; confirming the identity of a voter is crucial to ensuring the integrity of the electoral process. Every vote count, thus preventing identity theft is crucial. Our suggested service uses a user's legitimate identity card number to perform a cross-reference against the database, determining whether or not the user is qualified to register to vote. At that point, each voter receives a one-of-a-kind hash address that he can use to cast his ballot. Each hash receives enough Ethers for one vote.

### 3.3.2 Requirement Analysis

In this section, we'll go over the specifics of how this will be implemented. There are two main components: the registration system and the voting system.

**a. Registration System**

Users' already-collected personal information, such their name, address, and date of birth, is kept in a SQL database behind an HTML/CSS-based front end for the voting registration system. Users are authenticated and given a hash code/address to use as a voting credential if they are deemed to be legitimate users.

**b. Voting System:**

The voting system is a decentralized app that uses Bootstrap or HTML for the frontend and a Blockchain for the back end. The smart contract is written in a language called "solidity." In the smart-contract, the name of the candidate and his or her symbol are written. A smart contract is the part of the voting system that is the actual logic. A Transaction is a change that is made to a blockchain.

The way the outside world talks to the Ethereum network is through transactions. When we want to change or update the state stored in the Ethereum network, we use a "transaction." A transaction fee or service charge must be paid for each transaction. Within an Ethereum network, a currency called ether moves around. Ether is mostly used as a service charge or transaction fee, which is also called a "gas fee." For this prototype, we are using the IOTA testnet on EVM, sothere is no need for gas fee since it is free. Ganache-CLI is being used for this project. This makes setting up a private network faster, and transactions are mined almost as soon as they happen. MetaMask is a bridge that lets you visit the decentralized web of the future in your browser right now. It lets you run Ethereum decentralized applications (dApps) right in your browser without having to run a full Ethereum node.

### 3.3.3 Voting with Smart Contracts

The voting procedure in question is conducted on IOTA Testnet because it is expensive to deploy smart contracts on the official blockchain. Figure 7 presents the general procedure. Following is a description of the remaining steps:

a. First, a smart contract is deployed on the blockchain, with the contract's owner saved as the "chairman," and configurations for voters and candidates, as well as functions for voting, granting the ability to vote, and counting votes, are defined. Code snippets for each of them are depicted in figures 7.1, 7.2, 7.3, and 7.4.

b. To vote, an individual must have a code for their Ethereum wallet, which the Chairman will distribute to them after the voting process has been initiated.

c. To cast their ballots, voters interact with the smart contract by making a transaction in their Ether wallet. If the user hasn't previously cast a vote, the smart contract will do so and give it to the candidate they choose. Every vote resets to the current victor. After the election is over, you can also call the function associated with the victorious candidate.

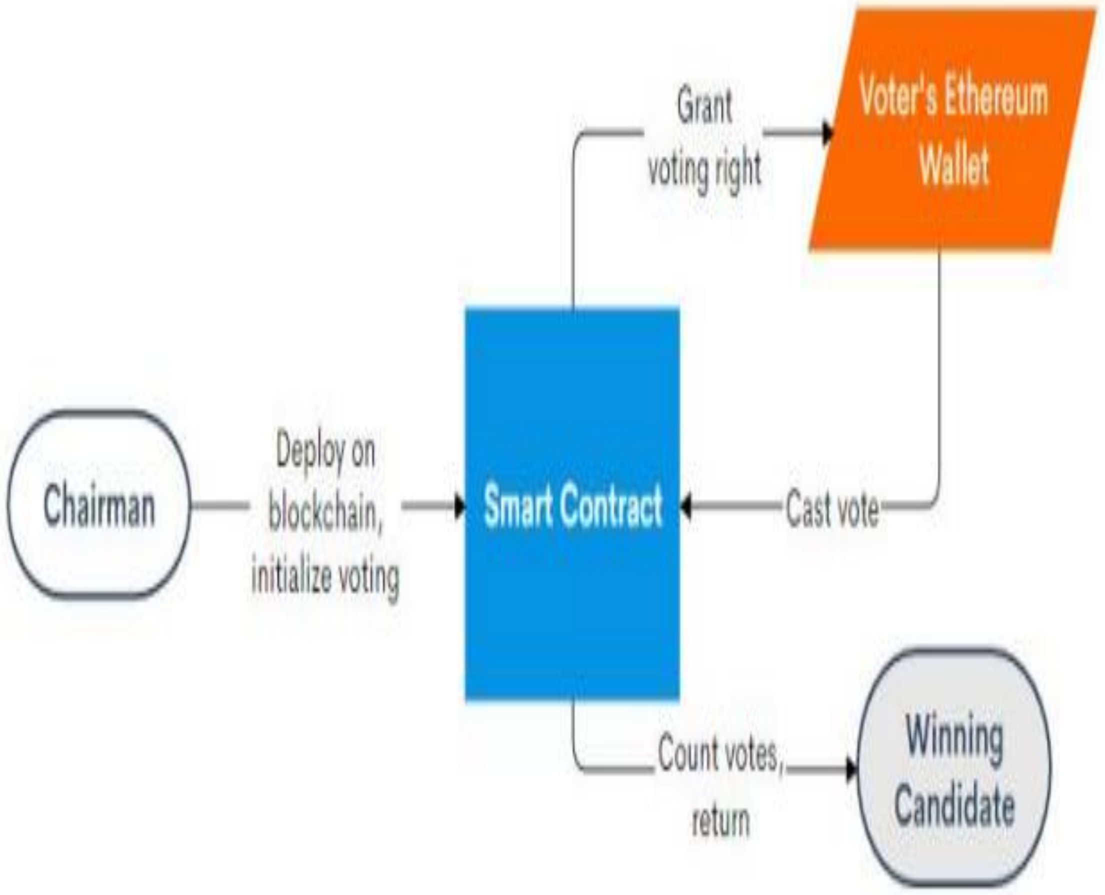


**Figure 3. 3: Process of voting with a smart contract**

### 3.3.4 The Smart Contract structure

The smart contract for this architecture is also split into two parts: the factory contract, which makes multiple copies of the election contract, and the election contract itself. The way the factory contract works with the election contract model is set by the variables and functions it has. An array of type ethereum address stores the addresses of all the contracts deployed by the factory contract, as well as the addresses of two special functions: the create election function, which creates a new election instance, and the getDeployedElections function, which returns the deployedElections variable.

Once the produced bytecode has been uploaded to the Ethereum network, the contract's abi creates several election instances

### 3.4.5 Flow Chart Diagram

The flowchart shows the system process sequentially utilizing the two actors in the system. Both voters and administrators have a role to play in the system, as their contribution represents both possible next steps and the completion of the process.

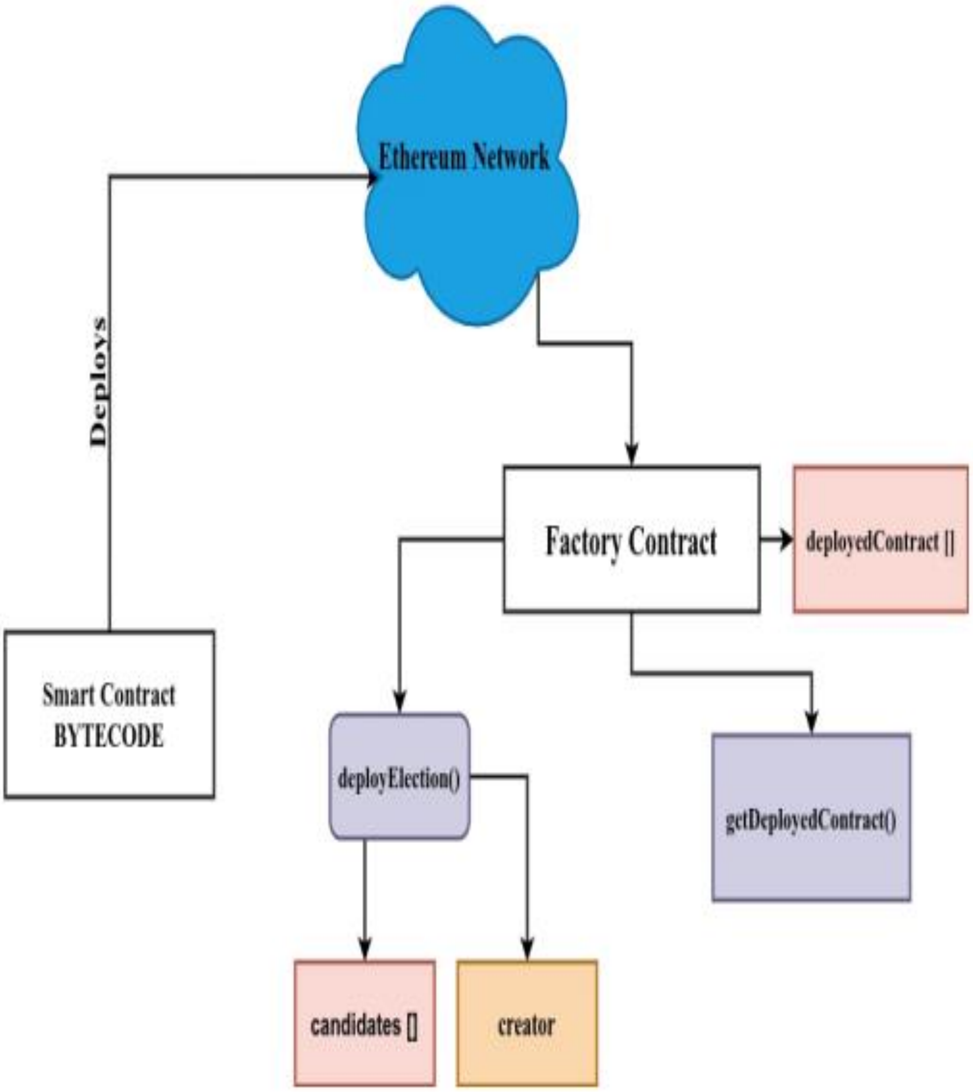


**Figure 3. 4: Factory Contract Architecture**

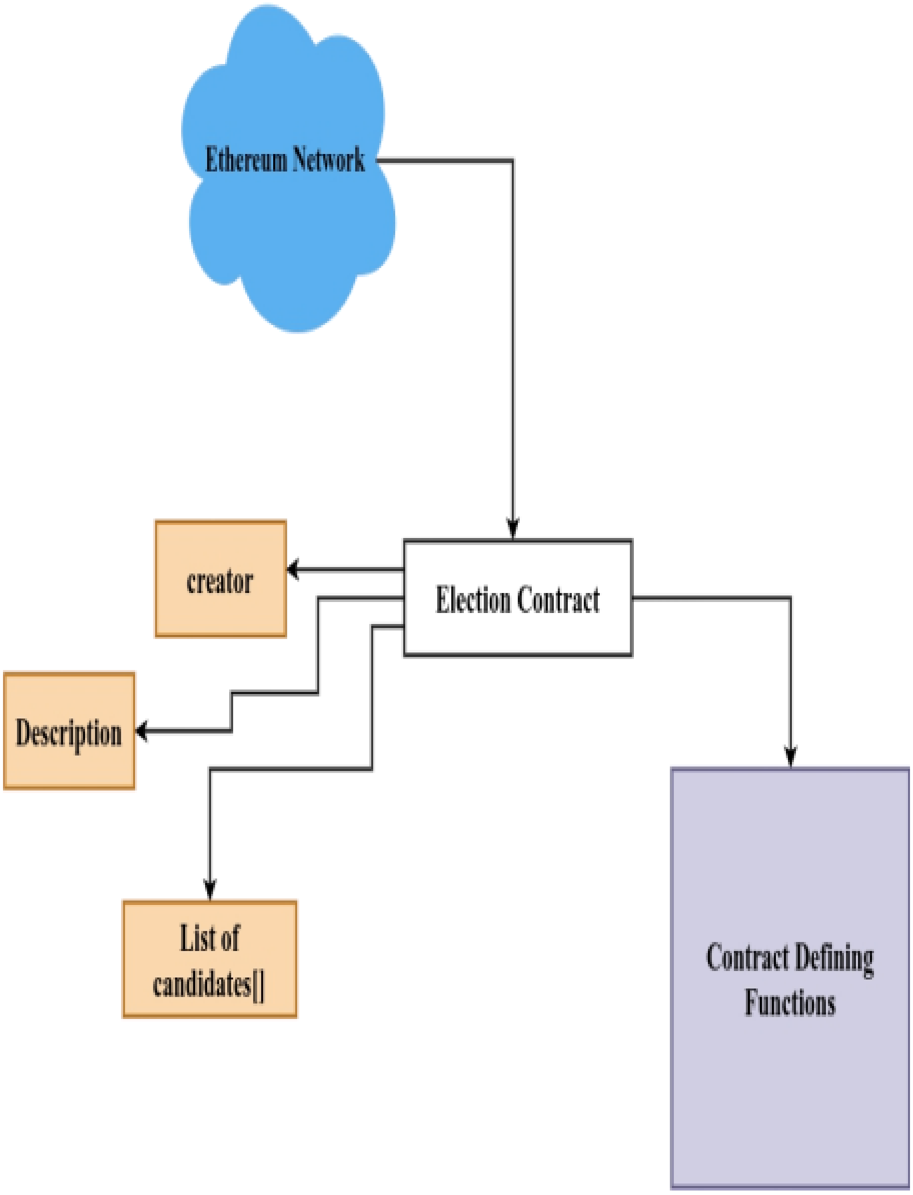


**Figure 3. 5: Election Contract Architecture**

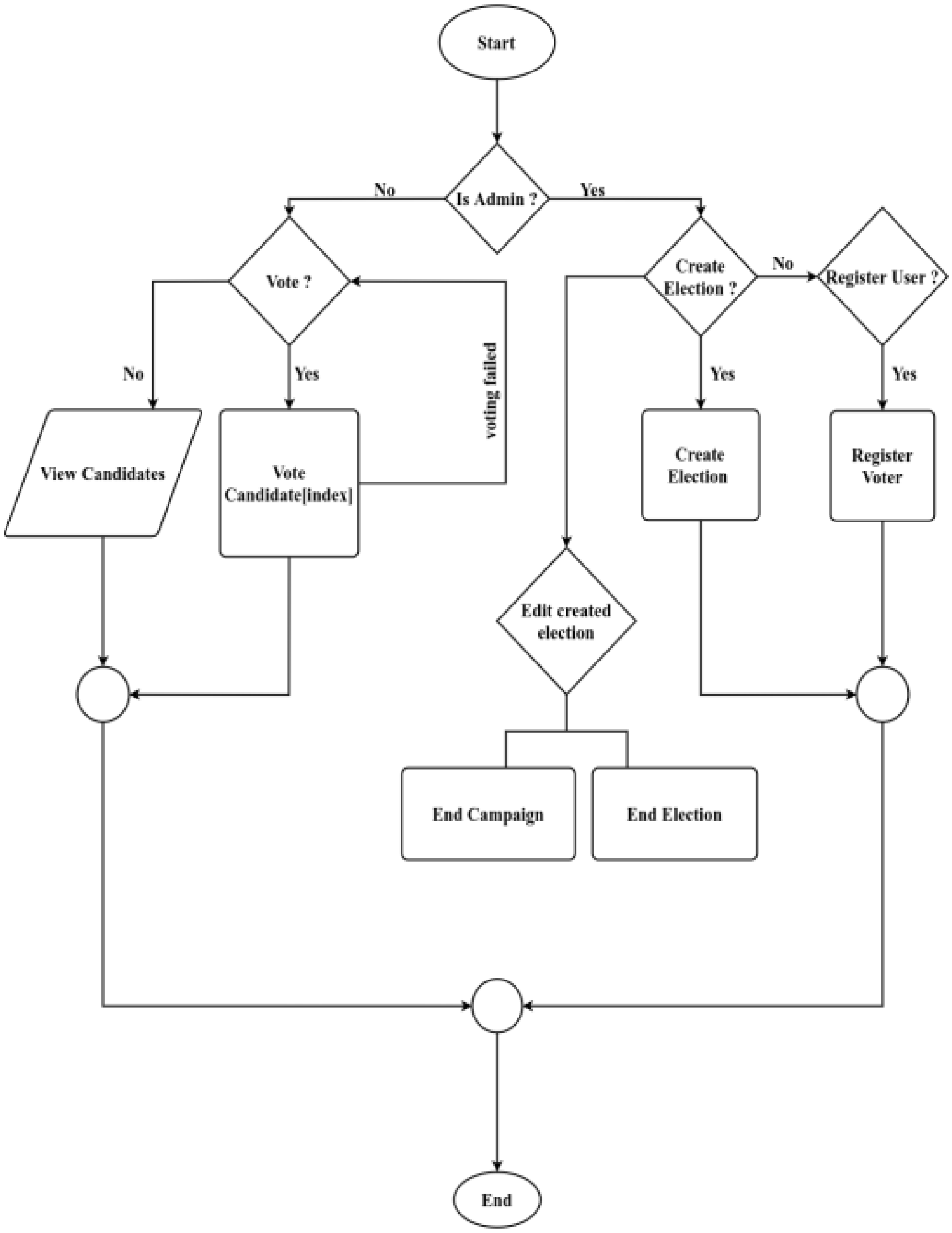


**Figure 3. 6: System flow chart diagram**

# CHAPTER FOUR

# IMPLEMENTATION AND RESULT

In this part of the project, the designs and clearly stated requirements were used to put the system into action and test it. When putting the project into action, a ganache local server and test accounts and private keys were used to connect the browser's metamask extension.

The system was built with the help of the programming languages solidity, javascript, HTML, CSS, and styling languages. The system was put through a series of test cases. The smart contract was built and tested using the remix online IDE, and the javascript testing library mocha was used to define more of the contract's functions and features.

## 4.1 Software and Hardware Requirements

For computers to be able to run this program, they would need to have the following apps and software installed:

- System operating system: Windows 10
- Web browser: Google chrome, Mozilla Firefox, Microsoft Edge
- Extensions (WEB): Metamask

## 4.2 System Development

The system was built using the Windows operating system and a development strategy focused on reusing existing code and components. The initial step in deploying a system is the compilation of the contract; the smart contract programming language solidity includes a compiler that does this, resulting in a build that contains the abi and bytecode.

After this bytecode is generated, it is published to an ethereum network, and the contract's address is used in conjunction with the abi to carry out network interactions with the contract. To facilitate interaction between the user interface, the backend, and the Ethereum network, the web3 package library is utilized.

These systems development resources include:

1. Visual Studio Code: Text Editor
2. Remix: Smart contract testing
3. Xampp: Local database

## 4.3 Application Images

The pictures for the application, displaying the various software pages, for both system actors.

### 4.3.1 System Home Screen

The voter and system administrator will both first see this page. It is the system's home page.

### 4.3.2 System Login Screen

Users must enter their email and password to log in on this page if they already have an account with the system.

### 4.3.3 System Register Screen

The system's register screen collects the user's email and other information and checks to see whether it already exists. System administrators have already been registered as the admin from the beginning when the system was formed, thus the admin only needs to log in.

### 4.3.4 System Dashboard

Each user type (administrator and voter) on the system has their own unique dashboard.

#### a) Client Dashboard/Vote screen

The user's meta-data is displayed on the client dashboard, along with any elections that are currently open.

### b) Admin Dashboard

On the admin dashboard, the users/voter’s component is rendered first. This component lists system users. Generate election allows the admin to create election instances using the factory contract and compiled contract abi. The election data component shows all system elections to the admin. The factory contract opens another application page. In this re-routed page, the admin adds users to an election, terminates the campaign, and then ends the election.

## 4.4 System Testing

The system needed configuration throughout development in order to perform even fundamental tests. By configuring the system to behave as an Ethereum network and linking metamask to the localhost server, we can simulate the blockchain's decentralized nature.

### 4.4.1 Smart Contract testing with Remix

Remix was used for testing the smart contract and simulating the application's whole cycle, from contract deployment to user interaction

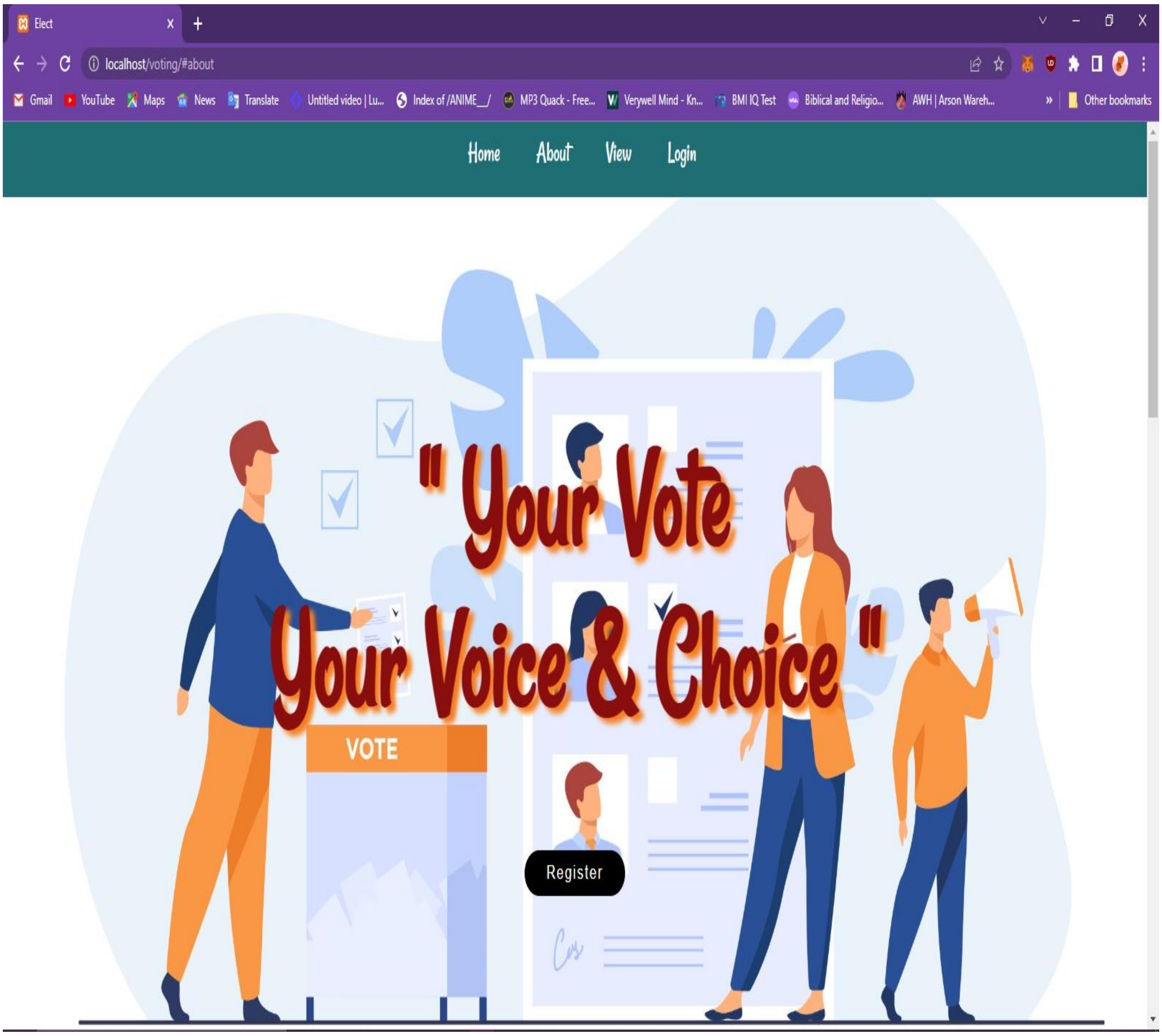


**Figure 4. 1: System Home Screen**

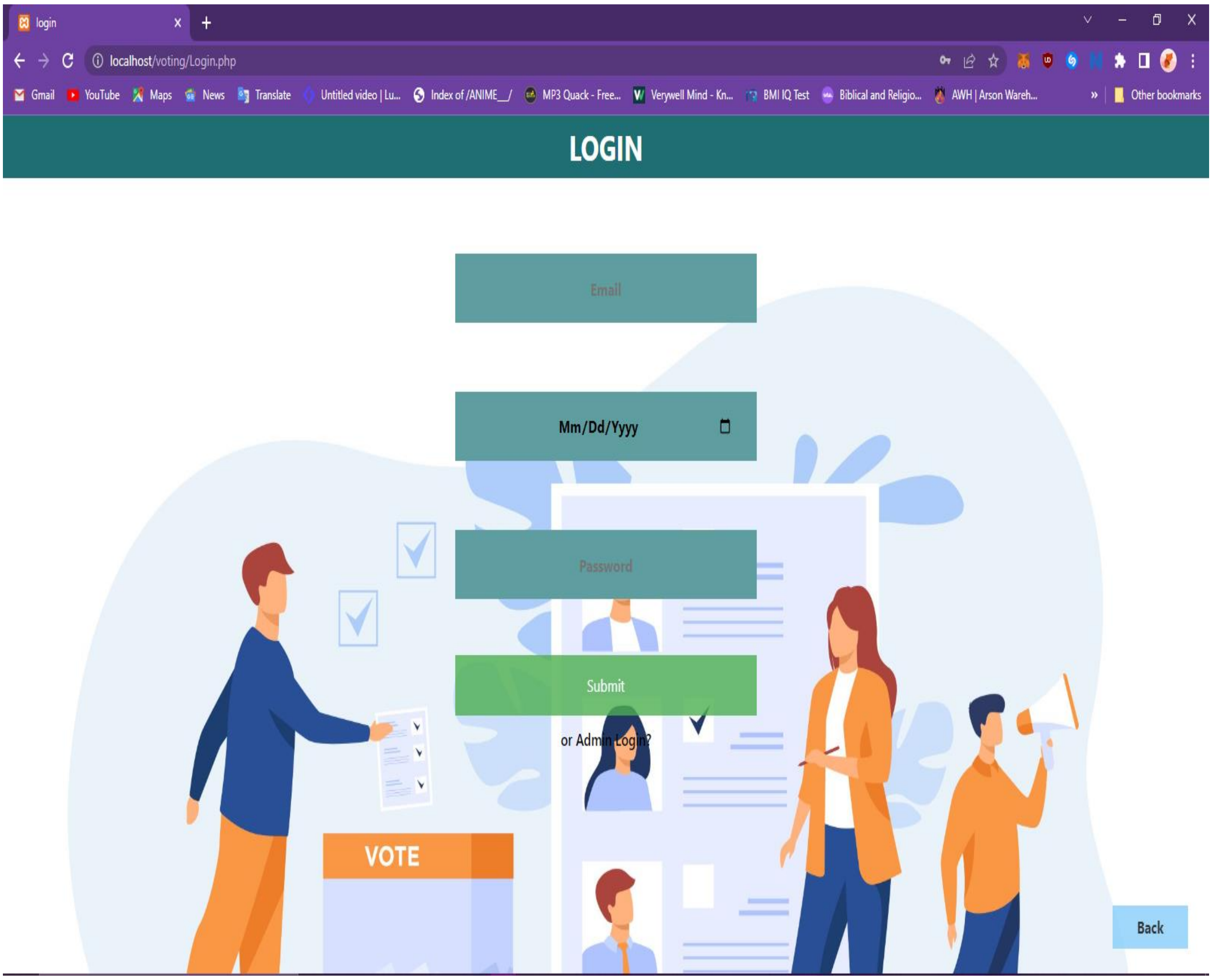


Figure 4. 2: Login Screen 1

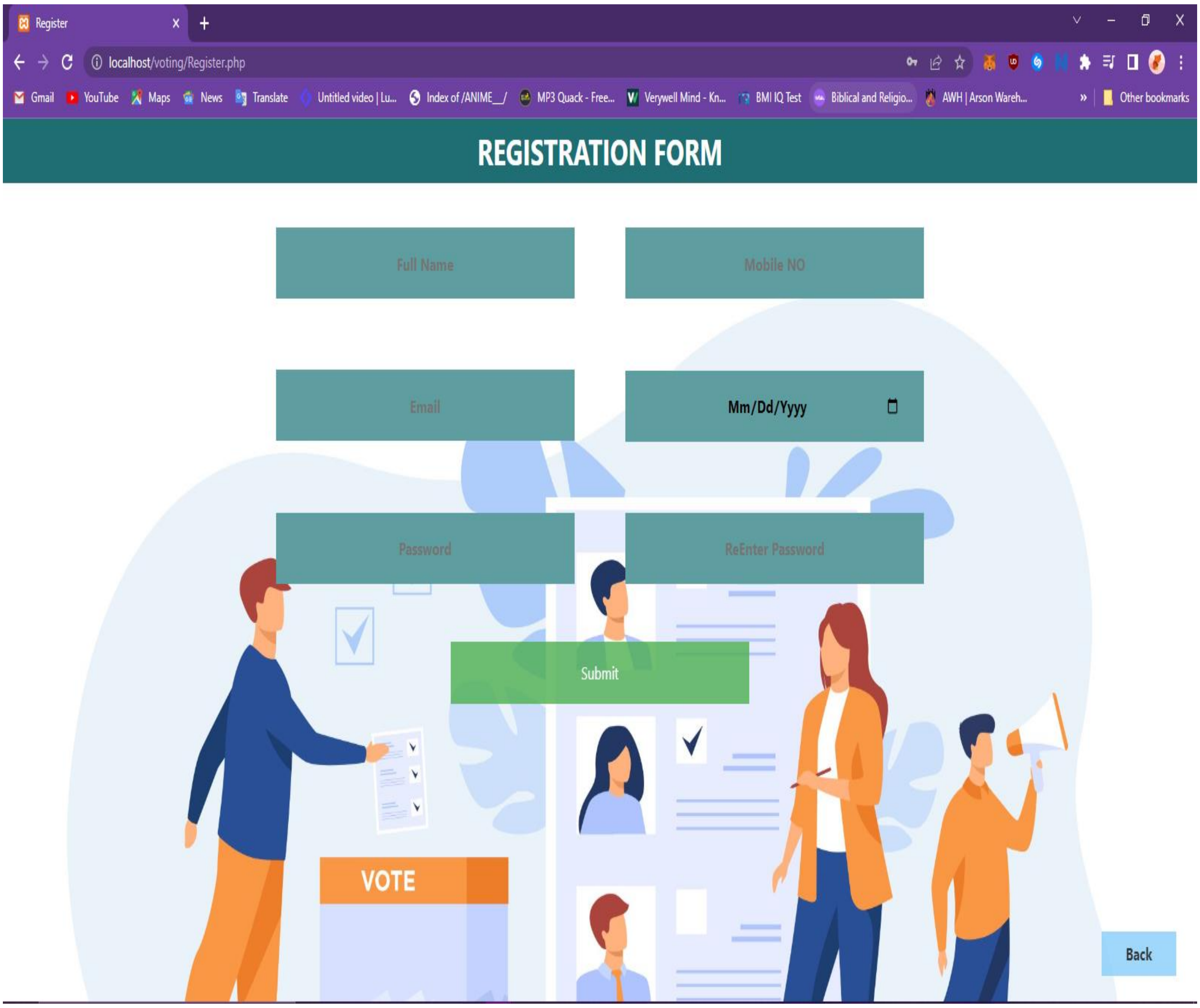
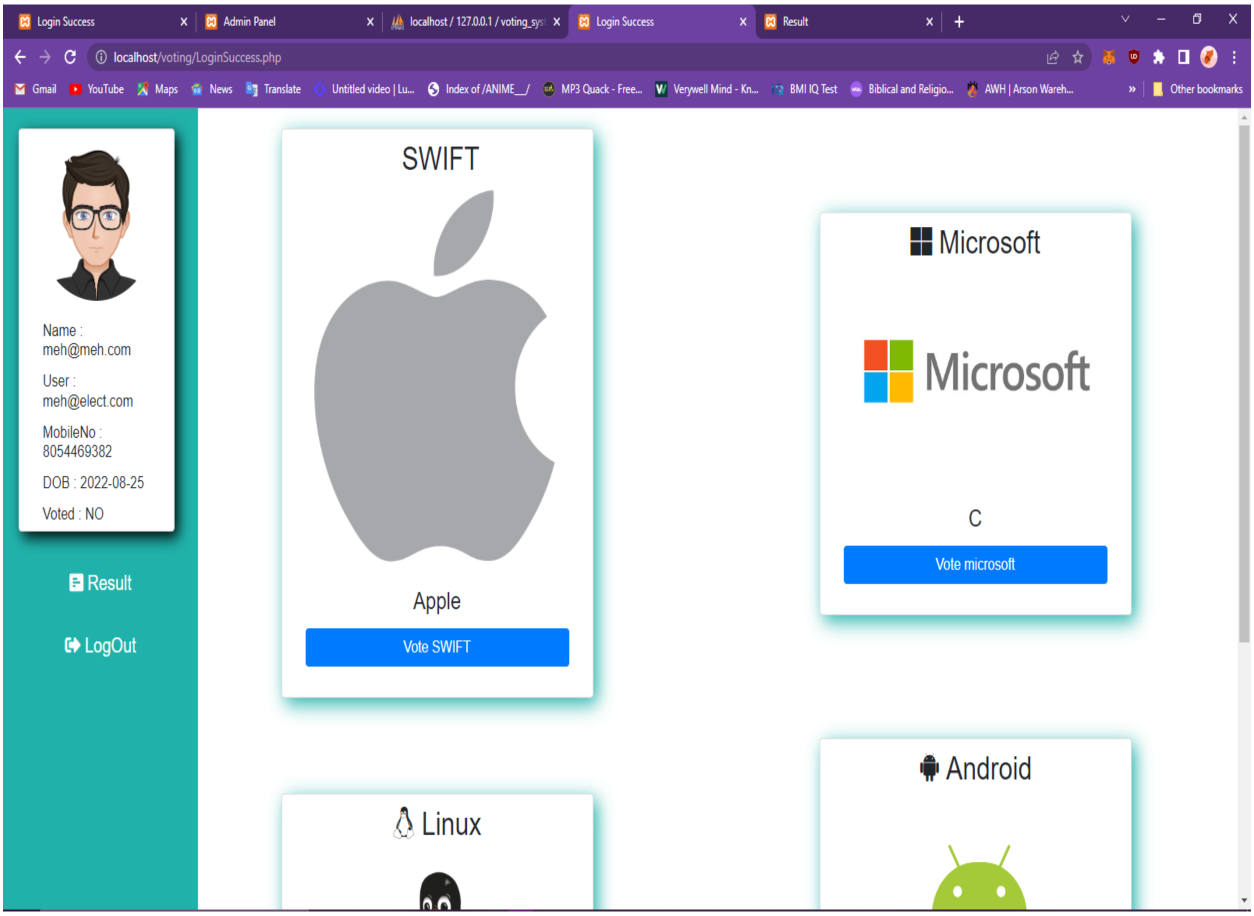


Figure 4. 3: Register Screen



Name : meh@meh.com

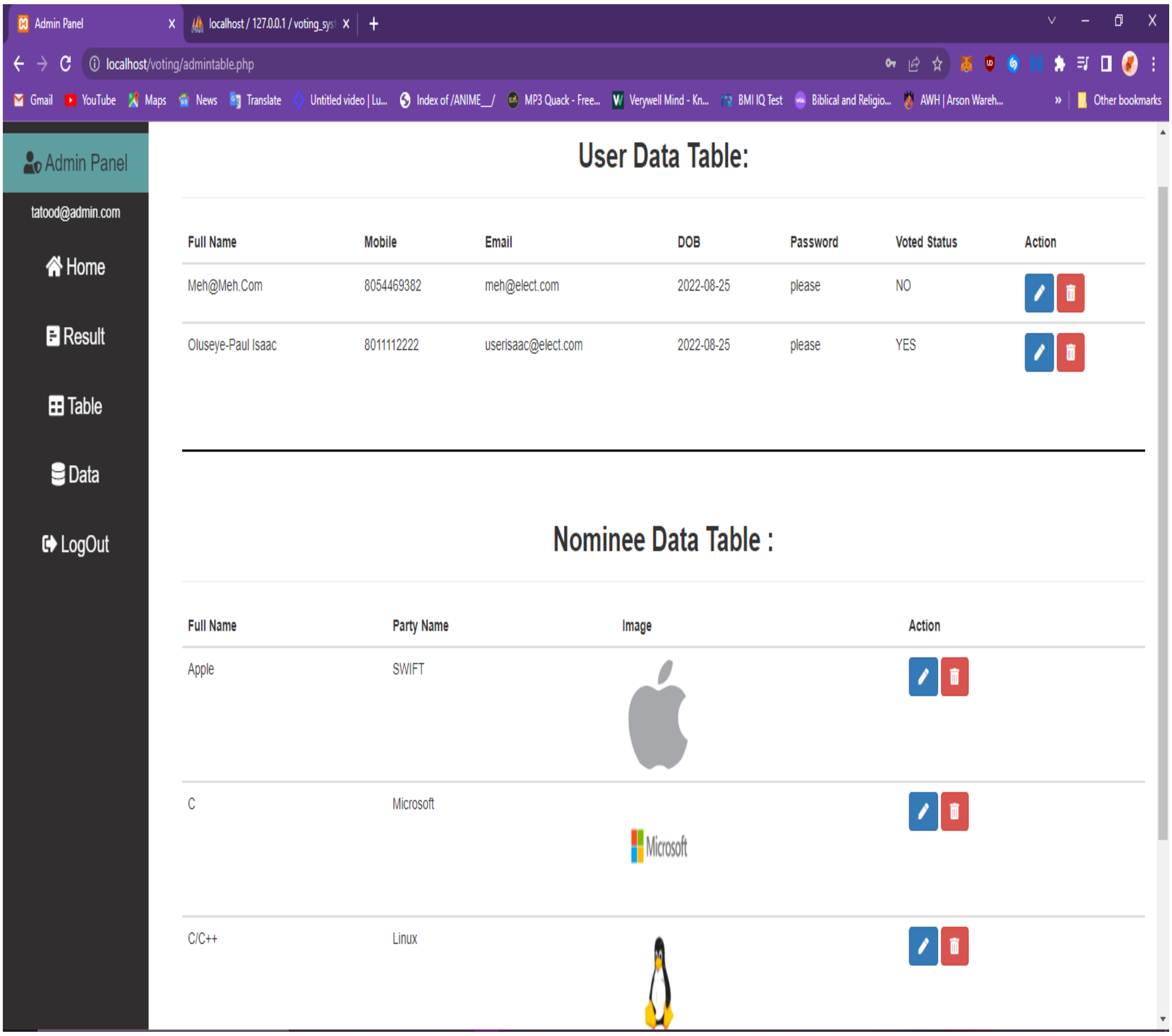
User : meh@elect.com

MobileNo : 8054469382

DOB : 2022-08-25

Voted : NO

**Figure 4. 4: Client Dashboard**



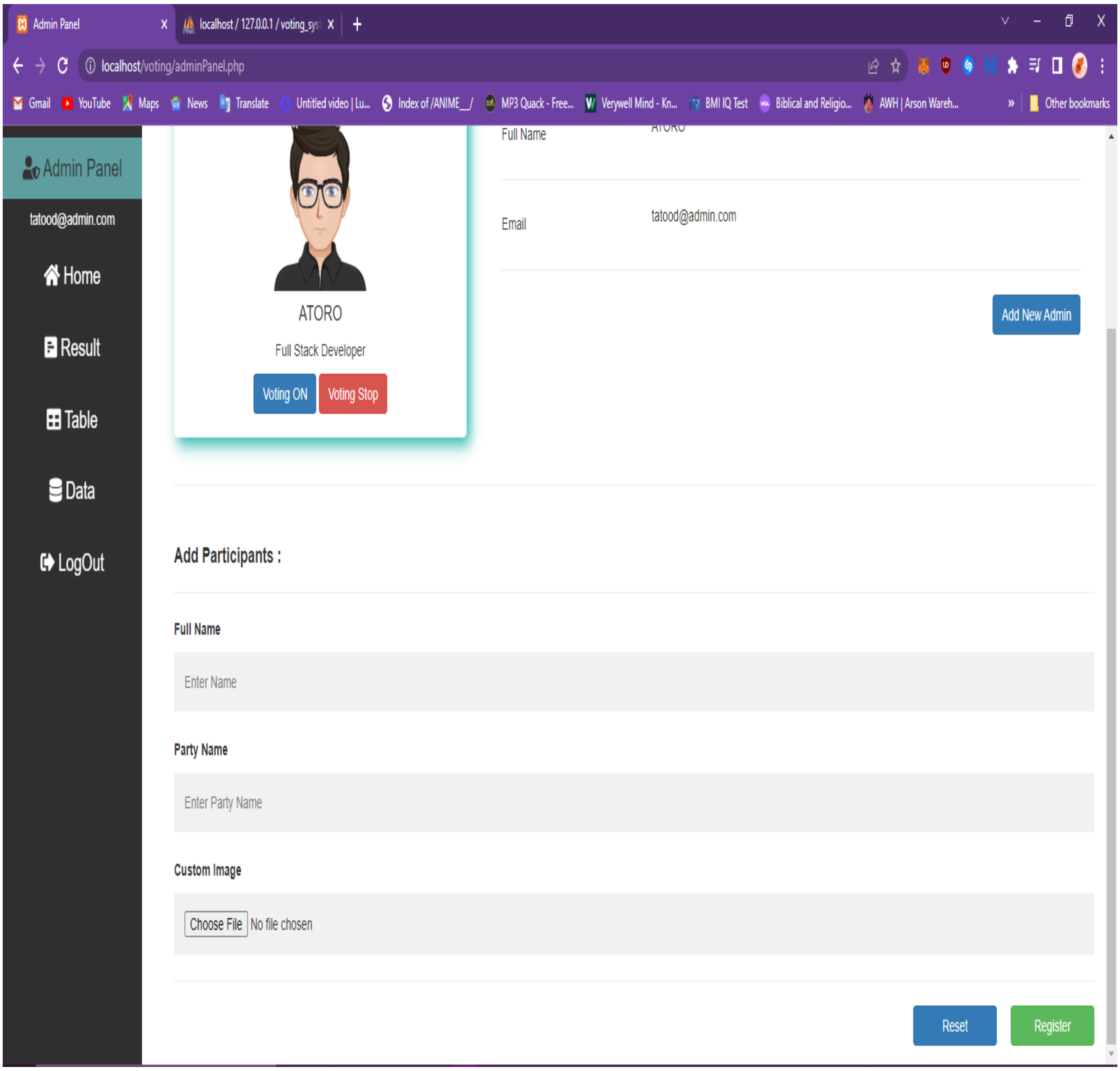
## User Data Table:

| Full Name | Mobile | Email | DOB | Password | Voted Status | Action |
|---|---|---|---|---|---|---|
| Meh@Meh.Com | 8054469382 | meh@elect.com | 2022-08-25 | please | NO | ✎ 🗑 |
| Oluseye-Paul Isaac | 8011112222 | userisaac@elect.com | 2022-08-25 | please | YES | ✎ 🗑 |

## Nominee Data Table :

| Full Name | Party Name | Image | Action |
|---|---|---|---|
| Apple | SWIFT | | ✎ 🗑 |
| C | Microsoft | Microsoft | ✎ 🗑 |
| C/C++ | Linux | | ✎ 🗑 |

Figure 4. 5: Admin First Component



Full Name ATORO

Email tatood@admin.com

Add Participants :

Full Name

Enter Name

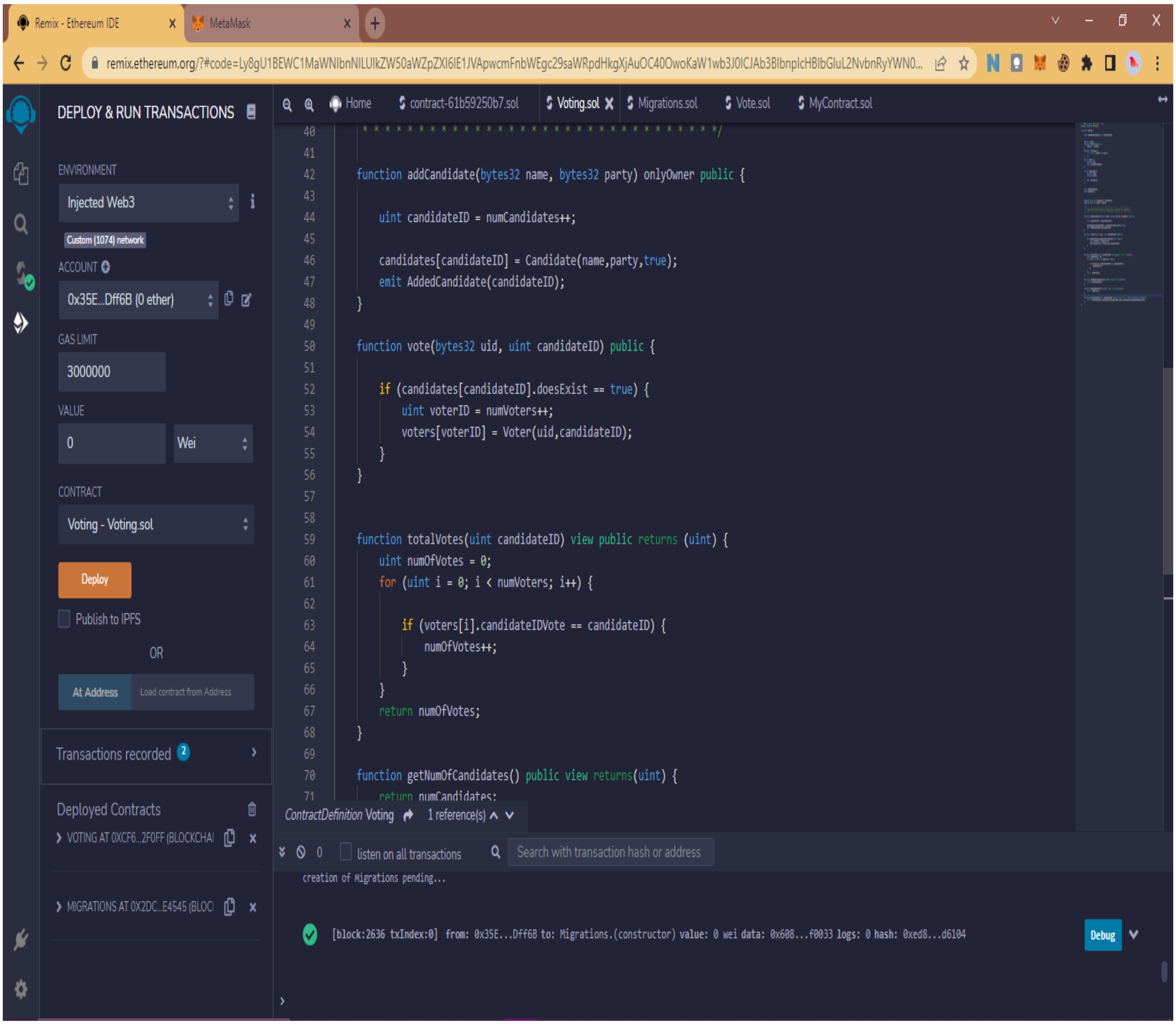
Party Name

Enter Party Name

Custom Image

No file chosen

**Figure 4. 6: Admin Second Component**



remix.ethereum.org/?#code=Ly8gU1BEWC1MaWNlbnNlLUlkZW50aWZpZXI6IE1JVApwcmFnbWEgc29saWRpdHkgXjAuOC40OwoKaW1wb3J0ICJAb3BlbnplcHBlbGluL2NvbnRyYWN0...

ENVIRONMENT: Injected Web3

Custom (1074) network

ACCOUNT: 0x35E...Dff6B (0 ether)

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Voting - Voting.sol

Deployed Contracts

› VOTING AT 0XCF6...2F0FF (BLOCKCHAI

› MIGRATIONS AT 0X2DC...E4545 (BLOC

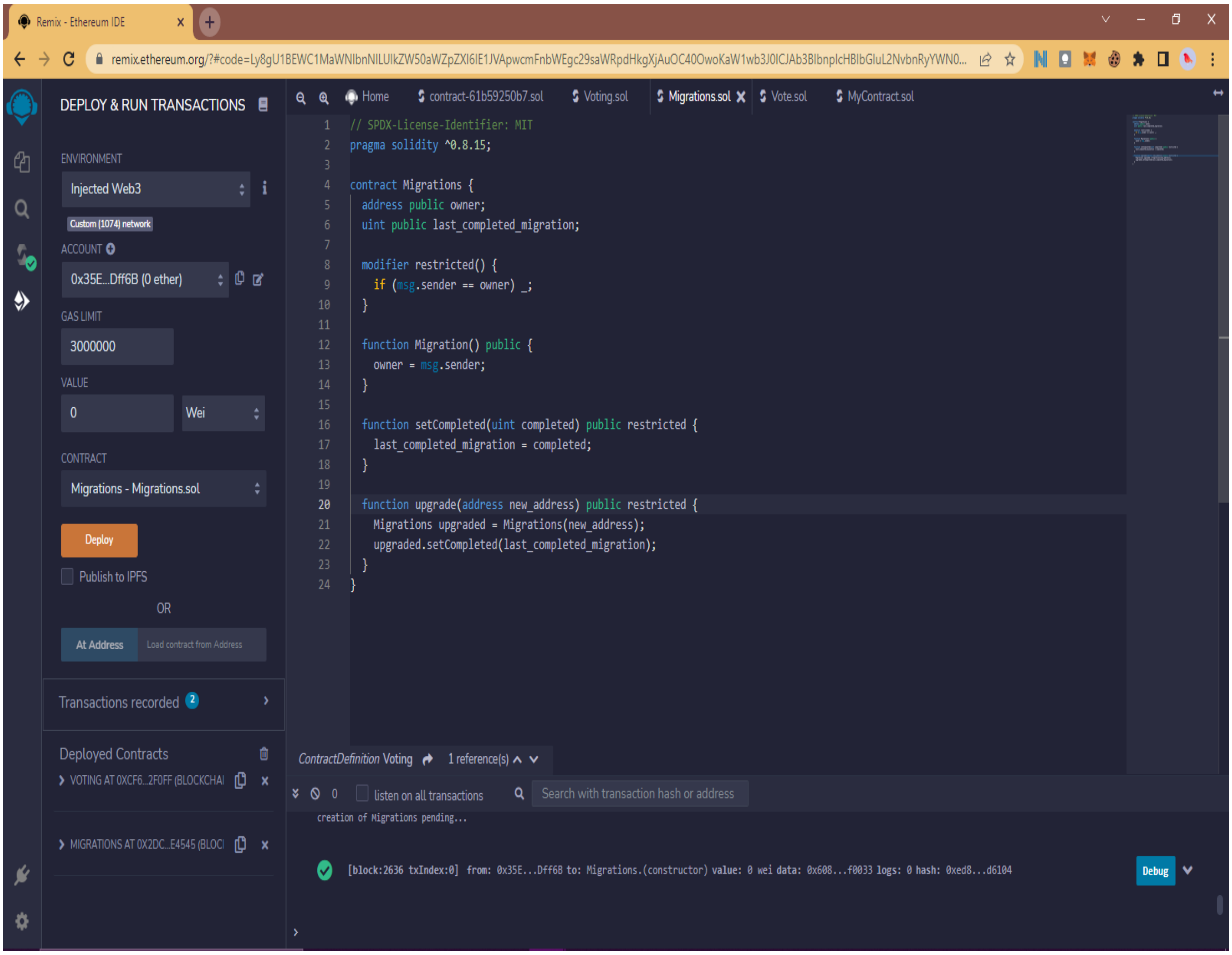
```
 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */

function addCandidate(bytes32 name, bytes32 party) onlyOwner public {

    uint candidateID = numCandidates++;

    candidates[candidateID] = Candidate(name,party,true);
    emit AddedCandidate(candidateID);
}

function vote(bytes32 uid, uint candidateID) public {

    if (candidates[candidateID].doesExist == true) {
        uint voterID = numVoters++;
        voters[voterID] = Voter(uid,candidateID);
    }
}


function totalVotes(uint candidateID) view public returns (uint) {
    uint numOfVotes = 0;
    for (uint i = 0; i < numVoters; i++) {

        if (voters[i].candidateIDVote == candidateID) {
            numOfVotes++;
        }
    }
    return numOfVotes;
}

function getNumOfCandidates() public view returns(uint) {
    return numCandidates;
```

ContractDefinition Voting 1 reference(s)

0 □ listen on all transactions  Search with transaction hash or address

```
creation of Migrations pending...

✓ [block:2636 txIndex:0] from: 0x35E...Dff6B to: Migrations.(constructor) value: 0 wei data: 0x608...f0033 logs: 0 hash: 0xed8...d6104
```

**Figure 4. 7: Remix Ide Compiler**



Remix - Ethereum IDE
remix.ethereum.org/?#code=Ly8gU1BEWC1MaWNlbnNlLUlkZW50aWZpZXI6IE1JVApwcmFnbWEgc29saWRpdHkgXjAuOC40OwoKaW1wb3J0ICJAb3BlbnplcHBlbGluL2NvbnRyYWN0...
DEPLOY & RUN TRANSACTIONS
Home
contract-61b59250b7.sol
Voting.sol
Migrations.sol
Vote.sol
MyContract.sol
Deploy
At Address
Load contract from Address
OR
Transactions recorded 2
Deployed Contracts
VOTING AT 0XCF6...2F0FF (BLOCKCHAI
MIGRATIONS AT 0X2DC...E4545 (BLOC
ContractDefinition Voting
1 reference(s)
Search with transaction hash or address
Debug

ENVIRONMENT
Injected Web3
Custom (1074) network

ACCOUNT
0x35E...Dff6B (0 ether)

GAS LIMIT
3000000

VALUE
0 Wei

CONTRACT
Migrations - Migrations.sol

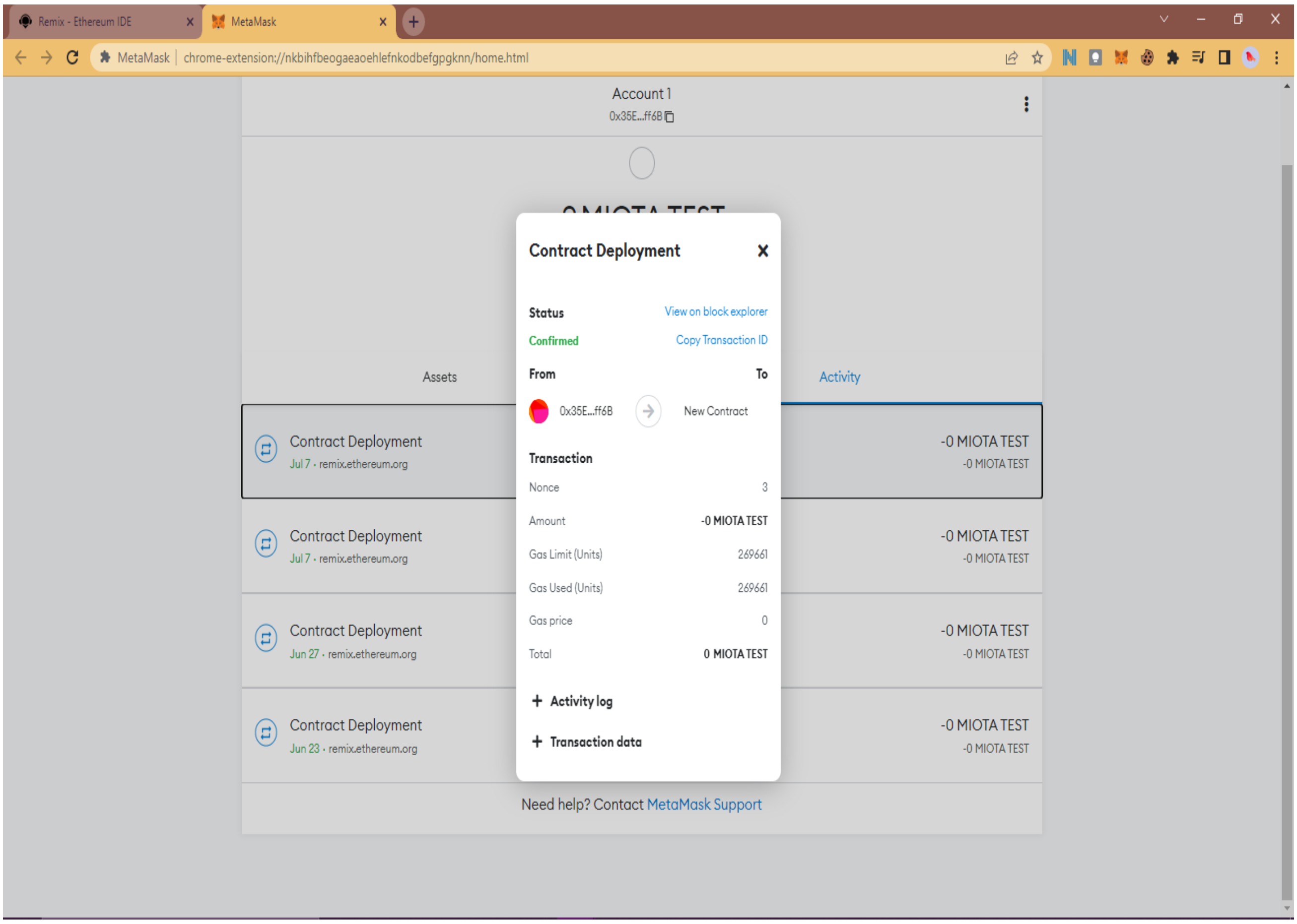
□ Publish to IPFS

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.15;

contract Migrations {
  address public owner;
  uint public last_completed_migration;

  modifier restricted() {
    if (msg.sender == owner) _;
  }

  function Migration() public {
    owner = msg.sender;
  }

  function setCompleted(uint completed) public restricted {
    last_completed_migration = completed;
  }

  function upgrade(address new_address) public restricted {
    Migrations upgraded = Migrations(new_address);
    upgraded.setCompleted(last_completed_migration);
  }
}
```

0 □ listen on all transactions

```
creation of Migrations pending...

✓ [block:2636 txIndex:0] from: 0x35E...Dff6B to: Migrations.(constructor) value: 0 wei data: 0x608...f0033 logs: 0 hash: 0xed8...d6104
```

**Figure 4. 8: Remix Ide Compiler**


Remix - Ethereum IDE
MetaMask
MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html

Account 1
0x35E...ff6B

Contract Deployment
Jul 7 · remix.ethereum.org
-0 MIOTA TEST
-0 MIOTA TEST

Contract Deployment
Jul 7 · remix.ethereum.org
-0 MIOTA TEST
-0 MIOTA TEST

Contract Deployment
Jun 27 · remix.ethereum.org
-0 MIOTA TEST
-0 MIOTA TEST

Contract Deployment
Jun 23 · remix.ethereum.org
-0 MIOTA TEST
-0 MIOTA TEST

Contract Deployment

Status: Confirmed

From: 0x35E...ff6B
To: New Contract

Transaction

| | |
|---|---|
| Nonce | 3 |
| Amount | -0 MIOTA TEST |
| Gas Limit (Units) | 269661 |
| Gas Used (Units) | 269661 |
| Gas price | 0 |
| Total | 0 MIOTA TEST |

Need help? Contact MetaMask Support

**Figure 4. 9: Metamask Extension transaction**

# CHAPTER FIVE

# SUMMARY, CONCLUSION AND RECOMMENDATION

## 5.1 Summary

This system was made to make electronic voting better. The main goal of introducing blockchain to conventional E-voting system is to build confidence in the system and make it easier to keep the application secure.

## 5.2 Conclusion

In conclusion, the System shows the problems with paper-based voting and the limitations of a centralized voting system. It also shows how blockchain can be used in other parts of daily life and how it can be used in voting. The system itself has some limits on how it can be used. For example, registering voters for certain elections has to be done by hand. If the number of users in the system goes up, it will be hard for system administrators to sign up voters for the validated voting instances.

## 5.3 Recommendation for Further Study

It is possible to improve this system's identification process because it registers users with system-generated end users who may become invalid if something happens to them. Face recognition may be integrated into the design so that each user who joins not only has their address attached to the profile but also their biometric property in the circumstance that the user dies and someone else applies using the user's id number.

# REFERENCES

Alaya, B., Laouamer, L., & Msilini, N. (2020). Homomorphic encryption systems statement: Trends and challenges. *Computer Science Review, 36*, 100235.

Chaieb, M., Koscina, M., Yousfi, S., Lafourcade, P., & Robbana, R. (2020, July 28). *DABSTERS: Distributed Authorities using Blind Signature To Effect Robust Security in e-voting.* Retrieved from HAL: https://hal.archives-ouvertes.fr/hal-02145809

ComputerHope. (2019, July 6). *Decentralized System*. Retrieved from ComputerHope: https://www.computerhope.com/jargon/d/decentral.htm

Dengo, M. (2020). Blockchain Voting: A Systematic Literature. *UNIVERSITY OF TARTU, Institute of Computer Science, Computer Science Curriculum*.

Fernandez-Carames, T. M., & Fraga-Lamas, P. (2020). Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing Attacks. *IEEE Acess, 8*, 21091–21116.

Froomkin, A. M. (2016, January 16). *Anonymity and its Enmities.* Retrieved from SSRN: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2715621

Gao., S., Zheng., D., Guo., R., Jing., C., & Hu., C. (2019). An Anti-Quantum E-Voting Protocol in. *IEEE Access, 7*, 115304–115316.

Hjalmarsson, F. P., Hreioarsson, G. K., Hamdaqa, M., & Hjálmtýsson, G. (2018). Blockchain-Based E-Voting System. *IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2-7.

Jafar., U., Aziz., M. J., & Shukur., Z. (2021). Blockchain for Electronic Voting System—Review and Open. *Faculty of Information Science and Technology, The National University of Malaysia, Bangi 43600, Malaysia;*.

Khan, K. M., Arshad, J., & Khan, M. M. (2020). Resistant Blockchain Cryptography to Quantum Computing Attacks. *Future Generation Computer Systems, 105*, 13-26.

Lai, W., Hsieh, Y.-C., Hsueh, C.-w., & Wu, J.-L. (2018). DATE: A Decentralized, Anonymous, and Transparent E-voting System. *1st IEEE International Conference on Hot Information-Centric Networking (HotICN),* 15-17.

Lejun., Z., Minghui., P., Weizheng., W., Yansen., S., Shuna., C., & Seokhoon, K. (2021). Secure and Efficient Data Storage and Sharing Scheme Based on Double Blockchain. *Transactions on Emerging Telecommunications Technologies, 32*(10).

McCorry, P., Shahandashti, S. F., & Hao, F. (2017). A smart contract for boardroom voting with maximum voter privacy. *Financial Cryptography and Data Security*, 357-375.

Mohanta., B. K., Jena., D., Panda., S. S., & Sobhanayak., S. (2019). Blockchain technology: A survey on applications and security privacy Challenges. *Internet Things*.

Pathak., P. M., Suradkar., A., Kadam., A., Ghodeswar., A., & Parde., P. (2021). International Journal of Scientific Research in Science and Technology. *Blockchain Based E-Voting System*, 134-140.

Shahzad, B., & Crowcroft, J. (2019). Trustworthy Electronic Voting Using Adjusted Blockchain Technology. *IEEE Access*, 24477–24488.

Szabo, N. (1997). Formalizing and securing relationships on public networks. *First Monday, 2*(9).

Torra, V. (2019). Random dictatorship for privacy-preserving social choice. *International Journal of Information Security, 19*, 537–545.

Wikipedia. (2022, January 4). *Decentralised system*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Decentralised_system

Woda, M. S., & Huzaini, Z. (2021). A Proposal to Use Elliptical Curves to Secure the Block in E-voting System Based on Blockchain Mechanism. *Theory and Engineering of Dependable Computer Systems and Networks*, 466-476.

Wood, D. G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 1-32.

Yi, H. (2019). Securing e-voting based on blockchain in P2P network. *EURASIP Journal on Wireless Communications and Networking*, 137.

Zhang, S., Wang, L., & Xiong, H. (2020). Chaintegrity: blockchain-enabled large-scale e-voting system with robustness and universal verifiability. *International Journal of Information Security*, 323-341.