**TRANSPARENT VOTING SYSTEM USING BLOCKCHAIN**

### A PROJECT REPORT

***Submitted by***

|  |  |
| --- | --- |
| **SUDHARSAN R** | **211418104271** |
| **VIMAL M** | **211418104311** |
| **YOGANANDAN S** | **211418104315** |

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

### COMPUTER SCIENCE AND ENGINEERING



**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

### MAY 2022

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

## BONAFIDE CERTIFICATE

Certified that this project report **“TRANSPARENT VOTING SYSTEM USING BLOCKCHAIN”** is the bonafide work of “**SUDHARSAN R (211418104271), VIMALM (211418104311) & YOGANANDAN S (211418104315)”** who carried out the

project work under my supervision.

### SIGNATURE SIGNATURE

**Dr.S.MURUGAVALLI,M.E.,Ph.D., Mrs.V.ANITHA MOSES, M.E(CSE) HEAD OF THE DEPARTMENT PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above-mentioned students were examined in End Semester project viva-voce held on .

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## DECLARATION BY THE STUDENT

### We SUDHARSAN R (211418104271), VIMAL M (211418104311) &

**YOGANANDAN S (211418104315)** hereby declare that this project report titled “**TRANSPARENT VOTING SYSTEM USING BLOCKCHAIN**”,

under the guidance of **Mrs.V.ANITHA MOSES** is the orginial work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI**

**KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI,M.E.,Ph.D.,**

for the support extended throughout the project.

We would like to thank our **Project Guide Mrs.V.ANITHA MOSES, M.E.,** and our **Project Coordinator Dr. N.PUGHAZENDHI, M.E., Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

### SUDHARSAN R [211418104271]

**VIMAL M [211418104311]**

**YOGANANDAN S [211418104315]**

## ABSTRACT

The aim of this project is to create a decentralized transparent voting and analysis system that can be implemented with blockchain to provide an efficient and highly secure justifiable method of implementing election systems in countries where traditional physical voting with gameable securities is used, increasing the chances of rigged elections. This system is designed to focus on a secure voting system, lower costs, faster wait times, no disparities due to various erroneous proxies, high scalability, and geographic independence. Overall, an effective election mechanism to strengthen the democratic process. Our dApp allows voters to vote from the comfort of their own homes, saving time and reducing the number of false votes registered.

## TABLE OF CONTENTS

**CHAPTER TITLE PAGE NO**

|  |  |
| --- | --- |
| **ABSTRACT** | **i** |
| **TABLE OF CONTENTS** | **ii** |
| **LIST OF FIGURES** | **iv** |
| **LIST OF TABLES** | **vi** |
| **1 INTRODUCTION** | **1** |
| 1.1 PROBLEM DEFINITION | 2 |
| **2 LITERATURE SURVEY** | **3** |
| **3 SYSTEM ANALYSIS** | **8** |
| 3.1 EXISTING SYSTEM | 9 |
| 3.2 PROPOSED SYSTEM | 9 |
| 3.3 FEASIBILITY STUDY | 10 |
| 3.4 REQUIREMENT SPECIFICATION | 10 |
| 3.4.1 HARDWARE REQUIREMENTS | 10 |
| 3.4.2 SOFTWARE REQUIREMENTS | 11 |
| 3.4.3 SOFTWARE SPECIFICATIONS | 11 |
| **4 SYSTEM DESIGN** | **15** |
| 4.1 ER DIAGRAM | 16 |
| 4.2 DATA DICTIONARY | 17 |
| 4.3 LEVEL 0 CONTEXT LEVEL DIAGRAM | 18 |
| 4.4 LEVEL 1 DATA FLOW DIAGRAM | 18 |
| 4.5 USE CASE DIAGRAM | 19 |
| 4.6 ACTIVITY DIAGRAM | 20 |
| 4.7 SEQUENCE DIAGRAM | 22 |
| 4.8 COLLABORATION DIAGRAM | 23 |

1. [SYSTEM ARCHITECTURE 24](#_TOC_250016)
   1. [MODULE DESIGN SPECIFICATION 26](#_TOC_250015)
      1. [LOGIN 26](#_TOC_250014)
      2. [INITIATING A NEW BALLOT 26](#_TOC_250013)
      3. [VOTING PROCESS 26](#_TOC_250012)
      4. [LIVE RESULTS 27](#_TOC_250011)
   2. [ALGORITHMS 27](#_TOC_250010)
2. [SYSTEM IMPLEMENTATION 29](#_TOC_250009)
   1. [CLIENT-SIDE CODING 30](#_TOC_250008)
   2. [SERVER-SIDE CODING 43](#_TOC_250007)
3. [TESTING 46](#_TOC_250006)
   1. [UNIT TESTING 47](#_TOC_250005)
   2. [INTEGRATION TESTING 49](#_TOC_250004)
   3. TEST CASES & REPORTS 50
4. [CONCLUSION 52](#_TOC_250003)
   1. [RESULTS AND DISCUSSION 53](#_TOC_250002)
   2. CONCLUSION AND FUTURE ENHANCEMENTS 54

[APPENDICES 55](#_TOC_250001)

* 1. SAMPLE SCREENSHOTS 56
  2. PLAGIARISM REPORT 63

[REFERENCES 65](#_TOC_250000)

## LIST OF FIGURES

**FIGURE NO TITLE PAGE NO**

|  |  |  |
| --- | --- | --- |
| **3.1** | VISUAL STUDIO CODE | 11 |
| **3.2** | REACT | 12 |
| **3.3** | NODE.JS | 13 |
| **3.4** | METAMASK | 13 |
| **3.5** | RINKBEY ETHEREUM TEST NETWOK | 14 |
| **4.1** | ER DIAGRAM | 17 |
| **4.2** | LEVEL 0 CONTEXT LEVEL DIAGRAM | 18 |
| **4.3** | LEVEL 1 DATA FLOW DIAGRAM | 18 |
| **4.4** | USE CASE DIAGRAM | 20 |
| **4.5** | ACTIVITY DIAGRAM | 21 |
| **4.6** | SEQUENCE DIAGRAM | 22 |
| **4.7** | COLLABORATION DIAGRAM | 23 |
| **5.1** | SYSTEM ARCHITECTURE | 25 |
| **A.1** | ADMIN LOGIN | 56 |
| **A.2** | CREATION OF A NEW BALLOT | 56 |
| **A.3** | BALLOT CREATION CONFIRMATION PAGE | 57 |
| **A.4** | VOTER LOGIN | 57 |
| **A.5** | INELIGIBLE VOTER’S PROMPT | 58 |
| **A.6** | ADD VOTERS | 58 |
| **A.7** | ELIGIBLE VOTER’S PAGE | 59 |
| **A.8** | CONFIRMATION OF VOTE | 59 |
| **A.9** | VOTE CONFIRMATION PAGE WITH IMMEDIATE | 60 |
|  | BALLOT UPDATION |  |
| **A.10** | ANOTHER VOTER’S PAGE WITH REAL-TIME RESULTS | 60 |
| **A.11** | SELECTION OF DIFFERENT PARTIES | 61 |

* 1. RESTRICTION OF MULTIPLE VOTES FROM SAME 61

VOTER ID

* 1. VOTER’S PAGE AT THE END OF POLLING 62
  2. ADMIN FINAL RESULTS PAGE 62

## LIST OF TABLES

**TABLE NO TITLE PAGE NO**

* 1. E-R DIAGRAM SYMBOL DESCRIPTION 16
  2. DATA DICTIONARY DESCRIPTION 17
  3. USE CASE DIAGRAM SYMBOL DESCRIPTION 19
  4. ACTIVITY DIAGRAM SYMBOL DESCRIPTION 21
  5. TEST CASES & REPORTS 51

# INTRODUCTION

## INTRODUCTION

* + - 1. **PROBLEM DEFINITION**

Many elections conducted on electronic voting machines are hand- counted, and many jurisdictions that use lever voting machines tally absentee ballots by hand. Because these devices are not meant to endure a long time, they must be serviced on a regular basis by only trained mechanics. Apart from equipment maintenance issues, there has also been a problem with only a small fraction of individuals voting.

For verification, most present E-voting applications rely on government-issued information, which is not the most effective means of authentication. Because when we have to store large amount of sensitive information in the local database, it is difficult to keep it secure. There is also the possibility of casting multiple ballots. To tackle these challenges, we've implemented a decentralised database and a smart contract in our dApp, which stops voters from casting duplicate votes.

# LITERATURE SURVEY

## LITERATURE SURVEY

**Azaria, Asaph, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman**. "MedRec: Using Blockchain for Medical Data Access and Permission Management." In Open and Big Data (OBD), International Conference on, pp. 25-30. IEEE, 2016.

This paper they have used the technique but for different use case which we have taken it as an inspiration for us all on the momentum for the development of the p2p based voting decentralized system. [1]

**E-Voting Systems using Blockchain**: An Exploratory Literature Survey, Second International conference on Inventive Research in Computing Applications (ICIRCA), 2020.

E-Voting or electronic voting is a means for the election process to be conducted without the use of traditional paper ballots. The e-voting process, to be implemented in a large-scale scenario, requires the addressing of concerns concerning the security and reliability of such a system. The Blockchain technology, introduced by Satoshi Nakamoto using the cryptographic currency Bitcoin in 2008, opens up possibilities of designing and developing a secure, transparent and decentralized system with the absence of a third party for access and control, in the election procedure of casting and counting of votes. [2]

**Guo, Ye, and Chen Liang**. “Blockchain application and outlook in the banking industry.” Financial Innovation 2, (no. 1: 24), 2016.

In this system election is represented by a set of smart contracts, which are instantiated on the blockchain by the election administrators. A smart contract for election is created and deployed on the blockchain network for every voting district. Freya Sheer Hardwick, Apostolos Gioulis, Raja Naeem Akram, and Konstantinos Markantonakis, “E-Voting with Blockchain” An E-Voting Protocol with Decentralization and Voter Privacy”.

In this system commercial protocol like Bit congress, Follow My Vote, TIVI are used as e-voting protocol. Haibo Yi, “Securing e-voting based on blockchain in P2P network”. [3]

**Ikhsan Darmawan** E-voting adoption in many countries: A literature review, First Published October 12, 2021.

Although the number of countries that have adopted e-voting has decreased lately, the number of academic publications on e-voting adoption has increased in the last two years. To date, there is no coherent narrative in the existing literature that explains the progress of the research on e-voting adoption. This article aims to answer the following research question: “How has research on the topic of e-voting adoption progressed over the last 15 years?” The article provides a semi-systematic review of 78 studies that were conducted from 2005 to 2020. [4]

**Nakamoto Satoshi**, inventing bitcoin, implementing the first blockchain, deploying the first decentralized digital currency “A Peer-to-Peer Electronic Cash System” original 20 March 2014.

Complete idea of what blockchain is and how it works as a whole, this legacy is developed by Satoshi Nakamoto Indonesia has held simultaneous general elections to elect the President/Vice President and legislative members on April 17, 2019. However, there were at least 4 (four) important problems raised in this election i.e., the problem of logistics distribution, the duration of the ballot counting that is too long, the inconsistent regulation of vote counting, and the error in votes recapitulation. Blockchain technology can be a solution to deal with those problems. [5]

**Nicole J. Goodman; Jon H. Pammett**, “Internet Voting in a Local Election in Canada”, in Internet and Democracy in Global Perspective, Studies in Public Choice 31, Eds. Bernard Grofman, Alex Trechsel, and Mark Franklin, Springer Verlag, 2014.

Internet voting developments in Canada are growing quickly, with activity focused in local elections, political party leadership votes and unions. In some instances, the federal structure of the Canadian state facilitates Internet voting use, while in others it inhibits it. The result of this system of divided jurisdiction is that Internet voting use in Canada resembles a patchwork, showing strong concentration in some areas and no penetration in other places. [6]

**Rafer Cooley; Shaya Wolf; Mike Borowczak** Conference: IEEE International Smart Cities Conference (ISC2), 2018.

Students at the University of Wyoming designed two blockchain-based voting systems during a class offered only once at the University. The first system (re- use) branched Ethereum to leverage its security and privacy benefits. The second system (re-invent) created a new blockchain voting system which used two separate chains, one for validating voters and another for securing votes. [7]

**R.S. Yashank** E-Voting System using HyperledgerSawtooth, Communication & Materials (ICACCM), International Conference on 2020.

E-Voting or electronic voting is a modern alternative for the traditional voting system involving paper ballots. For the large-scale implementation of electronic voting, the design of the proposed system has to support reliability as well as security. A secure, transparent and decentralized e- voting system is proposed using the HyperledgerSawtoothblockchain framework. Restricted access of the system through election polling stations allows voters to cast their votes, which are recorded in the immutable blockchain state. [8]

**Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran**, Computer Networks: The International Journal of Computer and Telecommunications Networking, December 2017.

Recent years have witnessed tremendous growth in the number of smart devices, wireless technologies, and sensors. In the foreseeable future, it is expected that trillions of devices will be connected to the Internet. Thus, to accommodate such a voluminous number of devices, scalable, flexible, interoperable, energy-efficient, and secure network architectures are required. [9]

# SYSTEM ANALYSIS

## SYSTEM ANALYSIS

* + - 1. **EXISTING SYSTEM**

The current system lacks a secure blockchain architecture, and existing E- voting applications rely on government-issued data, which isn't the most reliable method of authentication. And also, for an election process to take place, the existing blockchain voting system comprises three phases: authentication, voting, and counting. Where several users' authentication and voting can be done in concurrently, and the counting phase begins only after the voting phase is completed.

## PROPOSED SYSTEM

The suggested system features a secure blockchain architecture, as well as well-protected security layers, ensuring that data is safe from all threats. In the proposed system, the authentication phase will be predefined, and the voting and result phases will be run simultaneously, allowing the proposed system to realize the main notion of openly showing live results to each and every voter. The web application in the proposed system is also built utilizing the React8 framework, which makes the system scalable. Our project is efficient in terms of loading speed and performance because we use React JS. In our dApp, we've also included a decentralized database and a smart contract to prevent people from voting twice.

## FEASIBILITY STUDY

### Operational Feasibility

There has been a lot of interest in remote voting in the last year as a way to make voting more convenient and, hopefully, boost election participation. Remote voting technologies are being studied as a possible replacement for regular voting systems. It is simple to maintain and use our product once it has been deployed in real time.

### Economic Feasibility

Remote voting may become the quickest, cheapest, and most efficient way to administer elections and tally votes because it just requires a few personnel and follows a basic process or procedure. Any entity that uses our product to conduct elections stands to benefit financially.

### Legal Feasibility

This contribution's major goal was to introduce the concept of transparent voting systems. In the creation of any system, security is crucial. The important elements of computer security are availability, integrity, secrecy, non- repudiation, and authentication; by combining these areas of security, they build a cohesive connection that helps ensure voter trust in our Transparent Voting System.

## REQUIREMENT SPECIFICATION

* + - * 1. **HARDWARE REQUIREMENTS**

Processor : Pentium Dual Core 2.00GHZ or above

Hard disk : 10 GB or above

RAM : 2GB (minimum)

Keyboard, Mouse : Required

* + - * 1. **SOFTWARE REQUIREMENTS** Operating system : Windows7 or above IDE : Visual Studio Code

Backend : Node Js, Solidity

Frontend : React

## SOFTWARE SPECIFICATIONS VS CODE



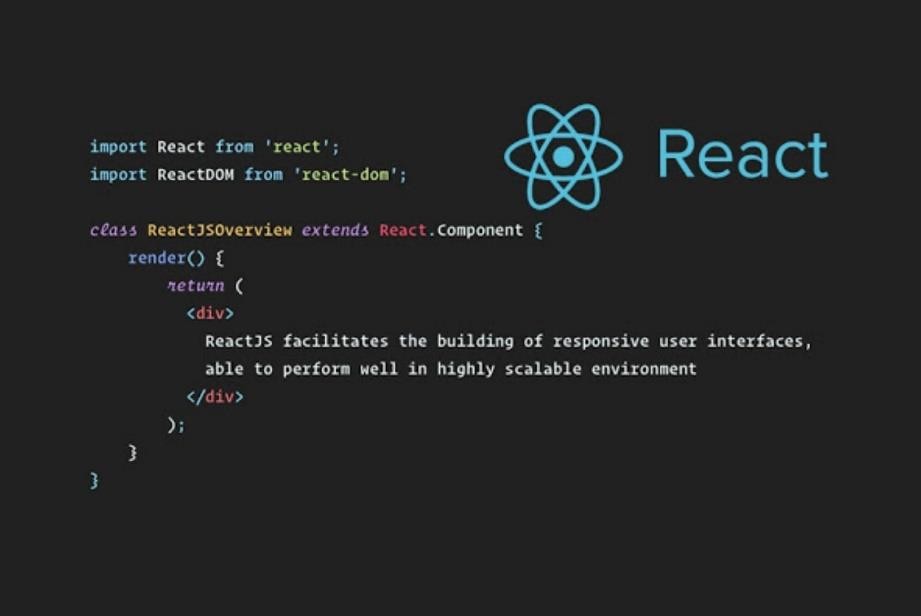
### Figure 3.1 – Visual Studio Code

V8 is the JavaScript execution engine which was initially built for Google Chrome. It was then open-sourced by Google in 2008. Written in C++, V8 compiles JavaScript source code to native machine code at runtime. As of 2016, it also includes Ignition, a bytecode interpreter.

### Package management

npm is the pre-installed package manager for the Node.js server platform. It installs Node.js programs from the npm registry, organizing the installation and management of third-party Node.js programs. Packages in the npm registry can range from simple helper libraries such as Lodash to task runners such as Grunt.

## FRONT END SPECIFICATIONS REACT



### Figure 3.2 – React

React.js was designed by Jordan Walke, a Facebook programmer, who tried to solve the problems that this social network encountered when incorporating the ads. Although Walke's work began in 2010, it was not until May 2013 that Mark Zuckerberg's company launched React as an open-source solution. Since then, there have been many enthusiasts of this library that continue to contribute to its improvement.

Thanks to the work of Walke, Facebook, and all the programmers who have contributed to its optimization, we now have a tool that allows us to develop web applications in which the front-end views are directly associated with the back-end data they receive.

Other alternatives to manipulate the elements of the DOM such as JQuery or even pure Javascript result in confusing and difficult to maintain code. React avoids these issues by proposing an architecture based on components, which are pieces of code that use HTML, CSS, and Javascript so that they contain both logic and presentation.

## BACKEND SPECIFICATION NODE JS



### Figure 3.3 – Node.js

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.

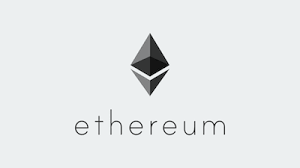
## METAMASK



### Figure 3.4 – MetaMask

MetaMask is a software cryptocurrency wallet used to interact with the Ethereum blockchain. Developers achieve a connection between Metamask and their decentralized applications by using a JavaScript plugin such as Web3js or Ethers to define interactions between Metamask and Smart Contracts.

### SOLIDITY (BLOCKCHAIN TECHNOLOGY – ETHEREUM)



**Figure 3.5 – Rinkbey Ethereum test network**

Blockchain being a technology that uses distributed ledgers, can be ideally used for this process. The blockchain network can be either a permission-less network like Bitcoin or Ethereum where anyone is allowed to interact with the network or a permissioned network like Hyperledger Fabric, HyperledgerSawtooth, or Exonum where only known members are allowed to interact with the network. Another important issue to be addressed is the anonymity of the voter. Due to the increase in research and progress in the field of big data analytics, this data is susceptible to discovery and manipulation. This can be resolved by using techniques like one-time ring signatures and homomorphic encryption. Blockchain is so-called, as it consists of a chain of blocks, that is, interconnected nodes that have their copy of the distributed ledger that contains the history of all transactions. Data is processed and put in a block through a process called mining. Every block contains a hash of the previous block and hence it forms a chain of blocks, with the first block known a the genesis block. Hence, it forms a linked list kind of structure. Blockchain has a number of ledgers where data can only be appended but not deleted or tampered. Consequently, it is immutable. Blockchain can either be public, where anyone can read or write data onto the blockchain, or private (permissioned), in which case only a few restricted individuals can read or write data.

# SYSTEM DESIGN

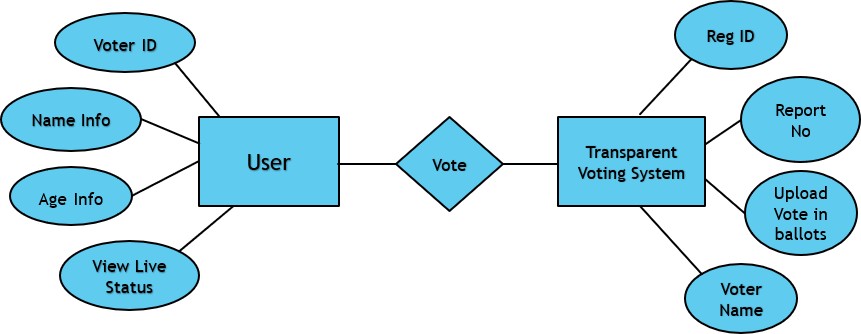
## SYSTEM DESIGN

* + - 1. **ER DIAGRAM**

ER stands for Entity Relationship. These diagrams display the relationship of entities that are used and stored in the database. They explain the structure of the whole process. these diagrams can be made using three basic concepts, attributed, relationships, and entities.

|  |  |  |
| --- | --- | --- |
| **Symbol Name** | **Symbol** | **Description** |
| Entity |  | An entity is represented by a rectangle which contains the entity’s name. |
| Attribute |  | In the Chen notation, each attribute is represented by an oval containing attribute’s name |
| Strong Relationship |  | A relationship where entity is existence-independent of other entities, and PK of Child doesn’t contain PK component of Parent Entity. A strong relationship is represented by a single rhombus |
| One or More |  | It represents One or More |
| Many - to - Many |  | It represents a one through many on both sides of a relationship |

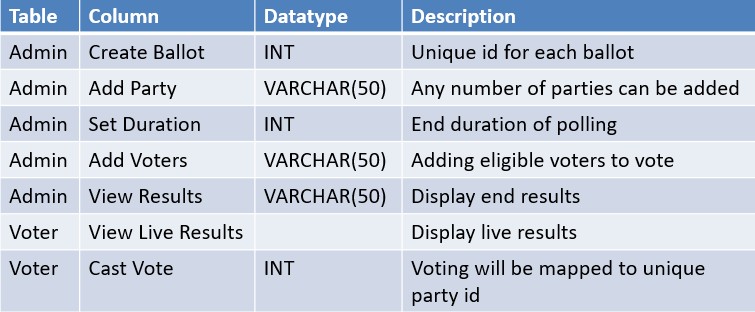
### Table 4.1 – ER Diagram Symbol Description



**Figure 4.1 – ER Diagram**

## DATA DICTIONARY

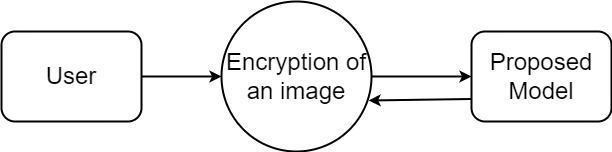
The data dictionary is an organized listing of all data elements that are pertinent to the system, with precise, rigorous definitions so that both user and system analyst will have a common understanding of inputs, outputs, components of stores and intermediate calculations.



**Table 4.2 – Data Dictionary Description**

## LEVEL 0 CONTEXT LEVEL DIAGRAM

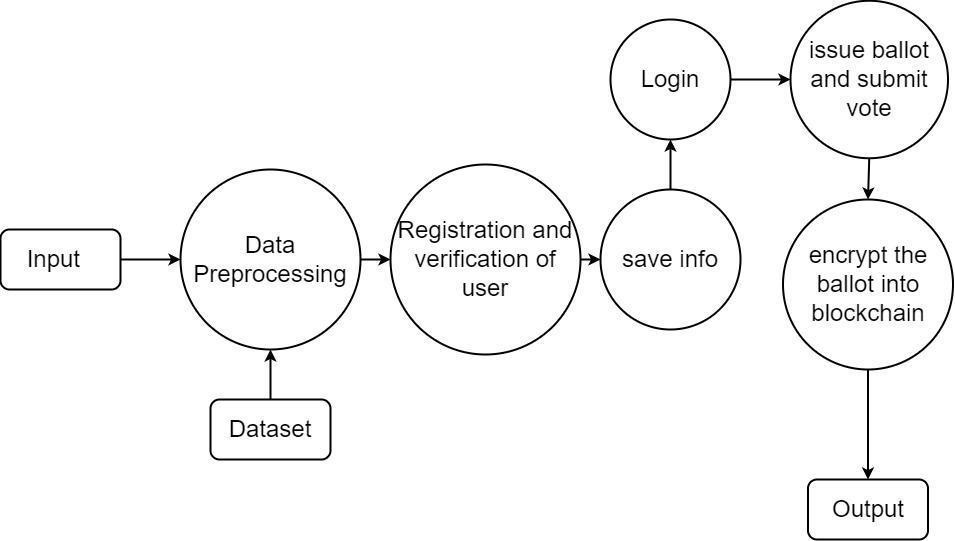
This is basically a contextual diagram, also referred to as a “context diagram”. It only represents the top level or the 0 Level in the whole process.it gives an abstraction kind of view and shows the whole system as a single process and its relationship to externalities.



**Figure 4.2 – Level 0 Context Level Diagram**

## LEVEL 1 DATA FLOW DIAGRAM

Level DFDs represent the complete system as a single process. it notates every process and sub-process that comes together in a sequence to form the complete system. This along with and 2-level data flow diagrams comprise the “fundamental system model”.



**Figure 4.3 –Level 1 Data Flow Diagram**

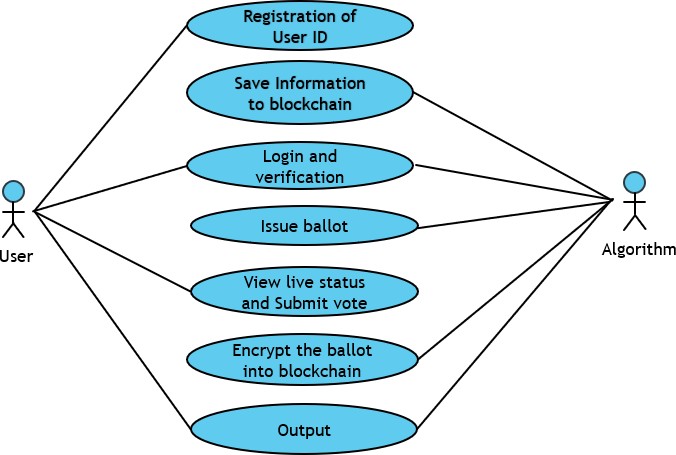
## USE-CASE DIAGRAM

A use case diagram is a standard diagram that shows all interactions between the user, dataset, and algorithm used. It is developed in the early stages of the process.

Below are the Symbols used in Use case diagram with Description:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Symbol Name** | **Symbol** | | | | | **Description** |
| Actor |  | | | | | Actors are the users of a system. |
| Usecase |  | | | | | Label the ovals with verbs that represent the system's functions. |
| Data Source |  | | | | | A Non-Human Actor is represented by this symbol. |
|  |  |  |  |  |
|  | | | | |

### Table 4.3 – Use Case Diagram Symbol Description



**Figure 4.4 – Use-Case Diagram**

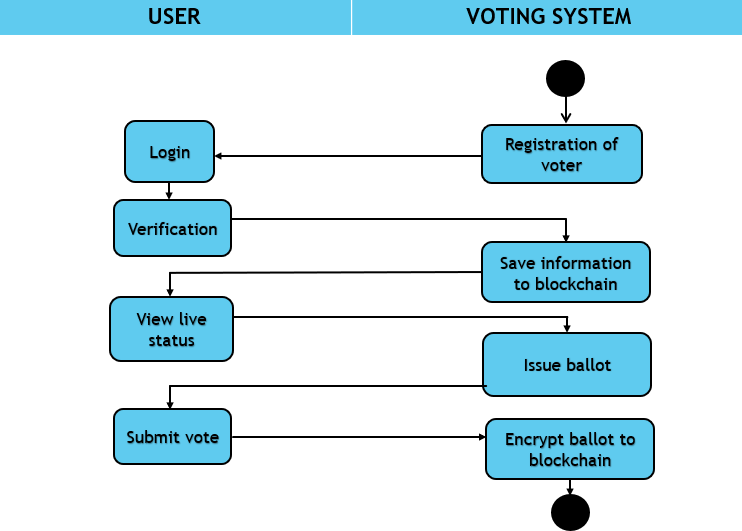
## ACTIVITY DIAGRAM

In simple terms, a diagram that represents the order of all activities is called the activity diagram. It shows the workflow between different activities that take place in the whole process. However, these are not exactly flowcharts but are similar.

|  |  |  |
| --- | --- | --- |
| **Symbol Name** | **Symbol** | **Description** |
| Start/Initial State |  | A small filled circle followed by an arrow represents start point for any activity diagram. |
| Activity State |  | An action state represents the non-interruptible action of objects. |

|  |  |  |
| --- | --- | --- |
| Decisions and Branching |  | A diamond represents a decision with alternate paths.The outgoing alternates should be labelled with a condition or guard expression. You can also  label one of the paths "else." |
| Final State |  | An arrow pointing to a filled circle nested inside another circle represents the final action state. |

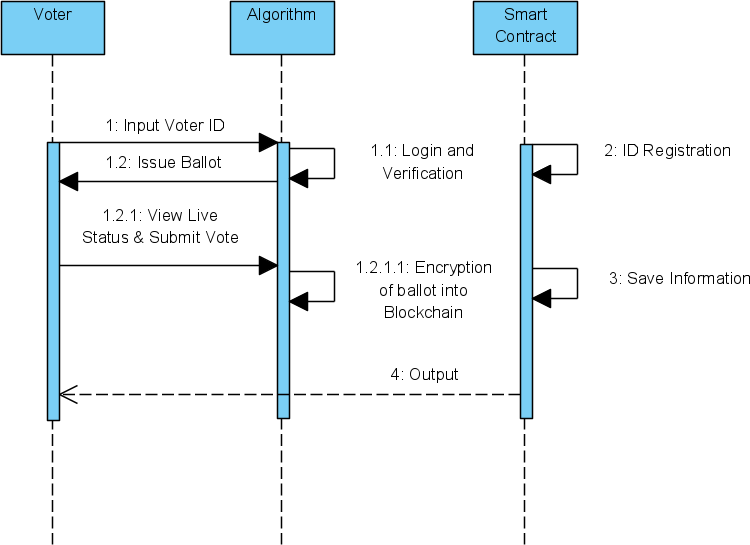
### Table 4.4 – Activity Diagram Symbol Description



**Figure 4.5 – Activity Diagram**

## SEQUENCE DIAGRAM

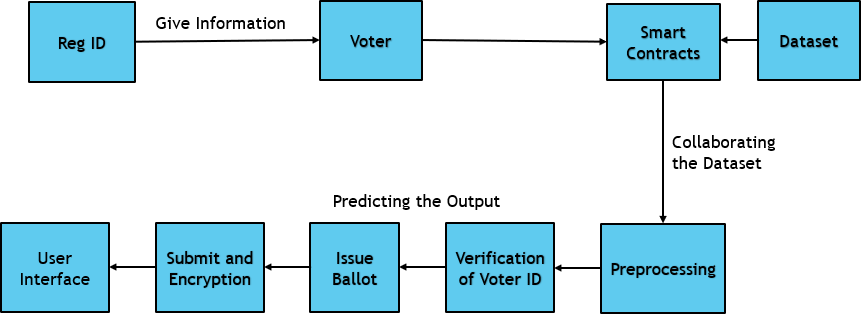
These are other kinds of interaction-based diagrams that show how all the operations are carried out. They capture the context of collaborations between objects and processes.



**Figure 4.6 – Sequence Diagram**

## COLLABORATION DIAGRAM

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

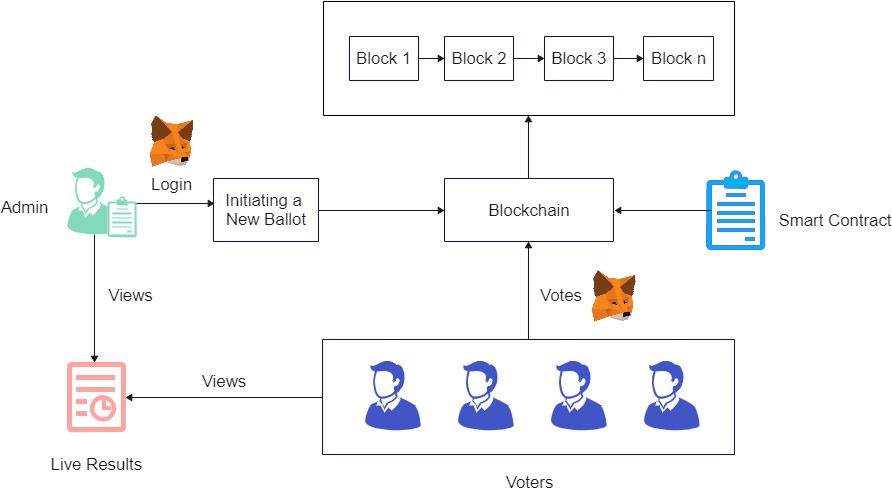


**Figure 4.7 – Collaboration Diagram**

# SYSTEM ARCHITECTURE

## SYSTEM ARCHITECTURE

The below diagram is nothing but a simple description of all the entities that have been incorporated into the system. The diagram represents the relations between each of them and involves a sequence of decision-making processes and steps. You can simply call it a visual or the whole process and its implementation. All functional correspondences are explained in this diagram.



**Figure 5.1 – Architecture Diagram**

## MODULE DESIGN SPECIFICATION

## LOGIN

Unlike traditional web2 applications where the user logs in to their account using a username and a password, web3 applications use wallets to connect to a user account. The public key of an account uniquely identifies the user whereas the private key is used for authorization. In our voting dapp, we use meta mask wallet which is available as a browser extension. To log in, the user has to simply connect the app to an account from the wallet.

## INITIATING A NEW BALLOT

Only admin can start a new ballot by specifying the information regarding the ballot and by authorizing the transaction using the wallet. The transaction is sent to the blockchain network and upon success is appended to the ledger permanently. The ballot creator has to add a list of addresses (public address of user accounts) that are eligible to vote.

## VOTING PROCESS

A user who is eligible to vote can connect their accounts in a similar manner using metamask. If a match is identified, the voter is given a list of candidates to choose from, with the option to vote against them. By signing a transaction in the wallet, the user's vote is recorded on the ledger as a transaction is delivered to the network. To this end, a successful vote cast is considered as a transaction within the blockchain of the voting application. Furthermore, once a vote is made, the same voter cannot cast several votes.

## LIVE RESULTS

Users can access the ballot results at any time. A cast vote is recorded in data tables at the backend of the database and added to the blockchain as a new block (after successful mining). The proposed system assures that each person can only vote once. During execution, the asset is transmitted to the address, and a transaction hash containing the vote transfer is produced.

## ALGORITHMS

Here we use Ethereum blockchain which comes under public blockchain where we can create Dapps (Decentralized Application)

* A blockchain is a distributed database that is shared among the nodes of a computer network. As a database, a blockchain stores information electronically in digital format.
* Blockchains are best known for their crucial role in cryptocurrency systems, such as Bitcoin, for maintaining a secure and decentralized record of transactions.
* The innovation with a blockchain is that it guarantees the fidelity and security of a record of data and generates trust without the need for a trusted third party.
* Ethereum is a platform powered by blockchain technology that is best known for its native cryptocurrency, called Ether, or ETH, or simply Ethereum.
* The distributed nature of blockchain technology is what makes the Ethereum platform secure, and that security enables ETH to accrue value.
* The Ethereum platform supports Ether in addition to a network of decentralized apps, otherwise known as dApps.
* Smart contracts, which originated on the Ethereum platform, are a central component of how the platform operates.

Here we use Smart Contract, Smart Contracts refer to computer protocols that digitally facilitate the verification, control, or execution of an agreement. Smart contracts run on the blockchain platform, which will process all the transactions in a contract; hence, middle men are not required for executing the transactions. This is written using Solidity (Programming language). Following are the reasons to consider this algorithm

* Open Access - Anyone in the Network can access and cast their vote.
* Security - As we use Ethereum Blockchain technology each and every action(transaction) will be stored in the network in a secured way(hashing).
* Anonymity - One can participate without revealing his identity. Here each and every one will be represented using an address.
* Accuracy - Using smart contracts results in the elimination of errors that occur due to manual filling of numerous forms.
* Verified Transactions - Transactions will be verified and validated by the network.

# SYSTEM IMPLEMENTATION

## SYSTEM IMPLEMENTATION

## CLIENT-SIDE CODING

### App.js

import React, { useEffect, useState } from "react"; import Voting from "./contracts/Voting.json";

import { getWeb3 } from "./utils.js"; import "./css/App.css";

function App() {

const [web3, setWeb3] = useState(undefined);

const [accounts, setAccounts] = useState(undefined); const [contract, setContract] = useState(undefined); const [admin, setAdmin] = useState(undefined); const [ballots, setBallots] = useState([]);

useEffect(() => {

const init = async () => {

const web3 = await getWeb3();

const accounts = await web3.eth.getAccounts(); const networkId = await web3.eth.net.getId();

const deployedNetwork = Voting.networks[networkId]; const contract = new web3.eth.Contract(

Voting.abi,

deployedNetwork && deployedNetwork.address

);

const admin = await contract.methods.admin().call();

setWeb3(web3); setAccounts(accounts); setContract(contract); setAdmin(admin);

};

init();

window.ethereum.on("accountsChanged", (accounts) => { setAccounts(accounts);

});

}, []);

const isReady = () => { return (

typeof contract !== "undefined" && typeof web3 !== "undefined" && typeof accounts !== "undefined" && typeof admin !== "undefined"

);

};

useEffect(() => { if (isReady()) { updateBallots();

}

}, [accounts, contract, web3, admin]);

async function updateBallots() {

const nextBallotId = parseInt(await contract.methods.nextBallotId().call());

const ballots = [];

for (let i = 0; i < nextBallotId; i++) {

const [ballot, hasVoted] = await Promise.all([ contract.methods.getBallot(i).call(), contract.methods.votes(accounts[0], i).call(),

]);

ballots.push({ ...ballot, hasVoted });

}

setBallots(ballots);

}

async function createBallot(e) { e.preventDefault();

const name = e.target.elements[0].value;

const choices = e.target.elements[1].value.split(","); const duration = e.target.elements[2].value;

await contract.methods

.createBallot(name, choices, duration)

.send({ from: accounts[0] }); await updateBallots();

}

async function addVoters(e) { e.preventDefault();

const voters = e.target.elements[0].value.split(",");

await contract.methods.addVoters(voters).send({ from: accounts[0] });

}

async function vote(e, ballotId) { e.preventDefault();

const select = e.target.elements[0];

const choiceId = select.options[select.selectedIndex].value;

await contract.methods.vote(ballotId, choiceId).send({ from: accounts[0] }); await updateBallots();

}

function isFinished(ballot) {

const now = new Date().getTime();

const ballotEnd = new Date(parseInt(ballot.end) \* 1000).getTime(); return ballotEnd - now > 0 ? false : true;

}

if (!isReady()) {

return <div>Loading...</div>;

}

function Voting\_Cards(isVoter) { let fragment = (

<div className={isVoter ? "container" : null}>

<div className={ isVoter

? "app\_cards vote\_container row"

: "app\_cards admin\_vote\_container row"

}

>

<div className="col-sm-12">

<div className="votes\_heading">

<h2>Votes</h2>

{isVoter ? null : <div id="blink\_dot"></div>}

</div>

<table id="table1" className="table">

<thead>

<tr>

<th>Election ID</th>

<th>Election Name</th>

<th>Polling Status</th>

{isVoter ? <th>Cast Vote</th> : null}

<th>Ends on</th>

</tr>

</thead>

<tbody>

{ballots.map((ballot) => (

<tr key={ballot.id}>

<td>{parseInt(ballot.id) + 1}</td>

<td>{ballot.name}</td>

<td className="td\_view">

<table id="table2">

<th>Party ID</th>

<th>Party Name</th>

<th>Live Results</th>

<>

{ballot.choices.map((choice) => (

<tr key={choice.id}>

<td>{parseInt(choice.id) + 1}</td>

<td>{choice.name}</td>

<td> {choice.votes}</td>

</tr>

))}

</>

</table>

</td>

{isVoter ? (

<td>

{isFinished(ballot) ? ( "Vote finished"

) : ballot.hasVoted ? ( "You already voted"

) : (

<form onSubmit={(e) => vote(e, ballot.id)}>

<div className="form-group">

<label htmlFor="choice">Choice</label>

<select className="form-control" id="choice">

{ballot.choices.map((choice) => (

<option key={choice.id} value={choice.id}>

{choice.name}

</option>

))}

</select>

</div>

<button type="submit" className="btn btn-primary"> Submit

</button>

</form>

)}

</td>

) : null}

<td>

{new Date(parseInt(ballot.end) \* 1000).toLocaleString()}

</td>

</tr>

))}

</tbody>

</table>

</div>

</div>

</div>

);

return fragment;

}

return (

<div className="app\_body">

<h1 className="app\_title">

<span> Transparent Voting System</span>

</h1>

{accounts[0].toLowerCase() === admin.toLowerCase() ? (

<div className="container">

<div className="row app\_cards">

<div className="col-sm-12">

<h2>Create Ballot</h2>

<form onSubmit={(e) => createBallot(e)}>

<div className="form-group">

<label htmlFor="name">Name</label>

<input type="text" className="form-control" id="name" />

</div>

<div className="form-group">

<label htmlFor="choices">Choices</label>

<input type="text" className="form-control" id="choices" />

</div>

<div className="form-group">

<label htmlFor="duration">Duration (s)</label>

<input type="text" className="form-control" id="duration" />

</div>

<button type="submit" className="btn btn-primary"> Submit

</button>

</form>

</div>

</div>

<hr />

<div className="app\_cards row">

<div className="col-sm-12">

<h2>Add Voters</h2>

<form onSubmit={(e) => addVoters(e)}>

<div className="form-group">

<label htmlFor="voters">Voters</label>

<input type="text" className="form-control" id="voters" />

</div>

<button type="submit" className="btn btn-primary"> Submit

</button>

</form>

</div>

</div>

<hr />

{Voting\_Cards(false)}

</div>

) : (

<div style={{ minHeight: "80vh" }}>{Voting\_Cards(true)}</div>

)}

</div>

);

}

export default App;

### App.css

@import url("https://fonts.googleapis.com/css2?family=Koulen&family=Satisfy&display

=swap"); @import

url("https://fonts.googleapis.com/css2?family=Alfa+Slab+One&family=Koulen &family=Satisfy&display=swap");

#root {

min-height: 100vh; background: linear-gradient(

0deg,

rgba(255, 255, 255, 0.3),

rgba(194, 194, 194, 0.493)

),

url("https://t4.ftcdn.net/jpg/02/32/32/27/360\_F\_232322711\_Q1f4rMOpFU91xC EP0v1l46cTIFywH3RX.jpg");

background-repeat: no-repeat; background-size: cover;

}

.app\_title { display: flex;

justify-content: center; align-items: center;

color: rgba(255, 255, 255, 0.882);

background-color: rgb(246, 183, 36);

box-shadow: 1px 0px 30px rgba(0, 0, 0, 0.396);

border-radius: 0 0 15px 15px;

font-weight: 400;

font-family: "Koulen", cursive; text-transform: uppercase; letter-spacing: 3px;

padding: 10px 0;

width: 100%;

margin-bottom: 30px;

}

.app\_body { width: 100%;

min-height: 100vh;

}

.app\_cards {

border-radius: 5px;

box-shadow: 1px 0px 30px rgba(90, 89, 89, 0.396); padding: 20px;

margin: 10px 0; background: white; width: 100%;

}

.btn\_color {

background-color: rgb(246, 183, 36);

.vote\_container { min-height: 70vh;

}

table {

background-color: rgba(255, 255, 255, 0.445);

}

#table1 {

margin-top: 20px;

border-radius: 0 0 10px 10px;

}

#table2 { width: 100%;

}

tr,

th, thead, td {

text-align: center; color: rgb(50, 30, 0);

}

.admin\_vote\_container { background-color: #5cb25d;

.td\_view { display: flex;

justify-content: center; align-items: center;

text-align: left !important;

}

.votes\_heading { display: flex;

flex-direction: row; align-items: center;

}

#blink\_dot { width: 20px; height: 20px;

background-color: red; border-radius: 50%; margin-left: 10px;

animation: blink 1s linear infinite;

}

@keyframes blink { 0% {

opacity: 0;

}

50% {

opacity: 1;

}

100% {

opacity: 0; }}

## SERVER-SIDE CODING

### Voting.sol

pragma solidity ^0.5.2;

pragma experimental ABIEncoderV2;

contract Voting {

mapping(address => bool) public voters; struct Choice {

uint id; string name; uint votes;

}

struct Ballot { uint id; string name;

Choice[] choices; uint end;

}

mapping(uint => Ballot) public ballots; uint public nextBallotId;

address public admin;

mapping(address => mapping(uint => bool)) public votes;

constructor() public { admin = msg.sender;

}

function getBallot(uint id) external view returns(Ballot memory) { return ballots[id];

}

function addVoters(address[] calldata \_voters) external onlyAdmin() { for(uint i = 0; i < \_voters.length; i++) {

voters[\_voters[i]] = true;

}

}

function createBallot( string memory name, string[] memory \_choices, uint offset

) public onlyAdmin() { ballots[nextBallotId].id = nextBallotId; ballots[nextBallotId].name = name; ballots[nextBallotId].end = now + offset; for(uint i = 0; i < \_choices.length ; i++) {

ballots[nextBallotId].choices.push(Choice(i, \_choices[i], 0));

}

nextBallotId++;

}

function vote(uint ballotId, uint choiceId) external { require(voters[msg.sender] == true, 'only voters can vote'); require(votes[msg.sender][ballotId] == false, 'voter can only vote once for a

ballot');

require(now < ballots[ballotId].end, 'can only vote until ballot end date'); votes[msg.sender][ballotId] = true; ballots[ballotId].choices[choiceId].votes++;

}

//If `pragma experimental ABIEncoderV2` function results(uint ballotId)

view external

returns(Choice[] memory) {

require(now >= ballots[ballotId].end, 'cannot see the ballot result before ballot end');

return ballots[ballotId].choices;

}

modifier onlyAdmin() {

require(msg.sender == admin, 'only admin');

\_;

}

}

# TESTING

## TESTING

### UNIT TESTING

**System Testing**

During testing, errors are detected. It's used for quality assurance. After we implement smart contracts in our dApp, we must check that all of the conditions are met, such as only the admin may make ballots, only added voters can vote, and only added voters can vote. Various tests were conducted in order to make this practicable. Software testing is an important part of the development and maintenance process. The goal of testing at this stage is to ensure that the specification has been accurately and thoroughly included into the design, as well as the correctness of the design. For example, any logic flaws in the design must be recognized before coding begins; otherwise, the cost of resolving the flaws will be significantly higher, as reflected. Inspection and a walkthrough are both effective methods for detecting design flaws.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

* Static analysis is used to investigate the structural properties of the Source code.
* Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

### Unit Testing

Unit testing is used to ensure that each modular component of the software works properly. Before deploying the smart contract throughout the entire app, each module is unit tested. The smallest unit of software design (i.e., the module) is the focus of unit testing. Unit testing was mainly reliant on white-box testing approaches.

### Functional Tests

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

* Performance Test
* Stress Test
* Structure Test

### Performance Test

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

### Stress Test

Stress tests are those that are intended to break the unit. Examining how a programmer breaks a program unit can reveal a lot about a program's power and limitations. Here, a stress test was performed to ensure that the constructed polling concludes on time, and that no further voting can take place after the polling time has expired.

### Structure Test

Structure tests are focused with putting a program's underlying logic to the test and traversing specific execution routes. The White-Box testing approach that was used to ensure that all independent pathways within a module had been executed at least once.

Admin initiates a ballot, voter’s voting process, and live results were some of the structures provided in our dApp. For the application to perform properly, all of the structures involved in it must be traversed at least once. Exercise all logical decisions on their true or false sides.

* Execute all loops at their boundaries and within their operational bounds.
* Exercise internal data structures to assure their validity.
* Checking attributes for their correctness.

### INTEGRATION TESTING

Integration testing is a method of building a program's structure while also doing tests to find interface issues. i.e., integration testing is the entire testing of the set of modules that makes up the product. The goal is to develop a application structure from of untested modules. The tester should identify key modules. As soon as feasible, critical modules should be tested. One method is to wait until all of the components have passed testing before combining and testing them. The unstructured testing of small apps gave birth to this method. Another option is to build the product in increments of the units that have been tested. A small number of components are combined and tested, after which another module is added and tested in tandem. And so on. The benefits of this method include the ease with which interface errors can be identified and remedied.

A linkage issue was the most serious mistake made during the project. When all of the modules are integrated, the link to all of the support files is broken. After that, we examined for interconnection and linkages. The new module and its intercommunications are the source of the errors. Modules can be included into the product development process once unit testing is completed. When the last module has been integrated and tested, testing is complete.

## TEST CASES AND REPORTS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO.** | **Test Cases** | **Expected Output** | **Actual Output** | **Status** |
| 1. | Admin creates ballot. | All eligible voters will see the ballot that has been created. | The ballot was prepared and displayed to all  voters successfully. | PASS |
| 2. | Admin adds eligible voters. | Voters with voting ID will be added successfully. | Accounts for eligible voters will be successfully  generated. | PASS |
| 3. | Voters login in to their account. | When genuine voter credentials are entered successfully. Voters can use their voting ID to log in. | The voter's login credentials have been confirmed, and they have been successfully entered  into their account. | PASS |
| 4. | Voters can view live polling  results. | Before casting their votes, voters can  check live results. | The live results were successfully  displayed. | PASS |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5. | Cast vote. | Voters can cast their  vote and observe the results immediately. | Voter’s vote will be  reflected in ballot immediately. | PASS |
| 6. | Avoid vote reputation. | When a voter attempts to cast numerous votes, the deployed smart contract should prevent the voter  from doing so. | When voters attempt to cast numerous votes, the message "You have already voted" will appear. | PASS |
| 7. | Ends voting on specified time. | After the polling period has ended, voters are unable to vote. | When people attempt to vote after the polling period has ended, they will get the message  "Voting concluded." | PASS |
| 8. | Transparent  display of results. | Everyone will be able  to see the voting results. | The voting results  were successfully shown. | PASS |

**Table 7.1 – Test Cases & Reports**

# CONCLUSION

## CONCLUSION

## RESULTS AND DISCUSSION

The outcome indicates how blockchain technology can be utilized to remotely conduct elections. We investigate the role of blockchain technology in the smart contract of a transparent voting system. For an election process to take place, the existing blockchain voting system comprises three phases: authentication, voting, and counting. Where several user’s authentication and voting can be done in concurrently, and the counting phase begins after the voting phase is completed. In the proposed system, the authentication phase will be predefined, and the voting and result phases will be run simultaneously, allowing the proposed system to realize the main notion of openly showing live results to each and every voter. This allows individuals to choose their preferred leader.

|  |  |
| --- | --- |
| **EXISTING BLOCKCHAIN VOTING**  **SYSTEM** | **PROPOSED TRANSPARENT VOTING**  **SYSTEM** |
|  |  |

## CONCLUSION AND FUTURE ENHANCEMENT

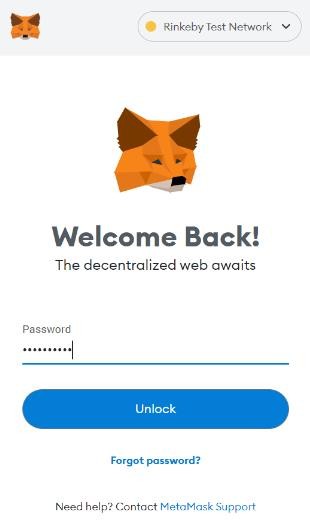
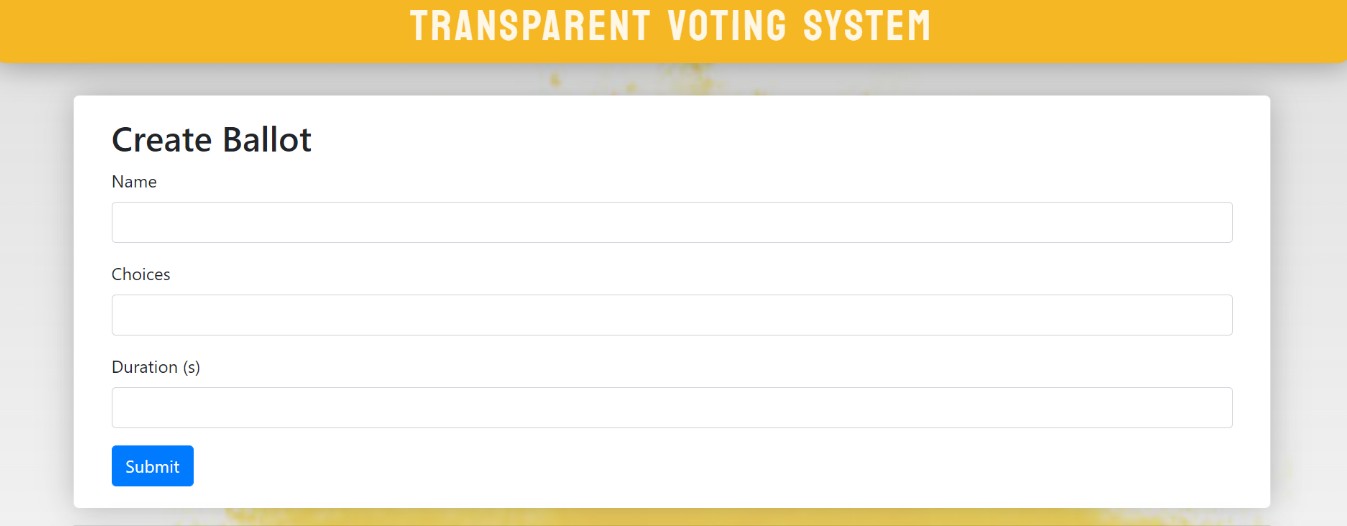
As a decentralized voting system, we employ blockchain in our implementation. For the voting process, the system will use candidate or voter information. The voting procedures and outcomes will be managed using smart contracts. Our method also improves the efficiency of the validation phase as well as the voting assignment. Our main goal is to create a more efficient and comprehensive approach for accomplishing Transparent Voting System using blockchain technology and its associated variable tools.

Integration of the Machine Learning concept by adding a facial recognition system is one of the project's planned developments. Face recognition would be a great addition because it improves security and prevents duplicate and forged votes. Our application can also include two-factor authentication. Even if the credentials provided to the voters are lost, the usage of two factor authentication will help to avoid cast proxy votes.

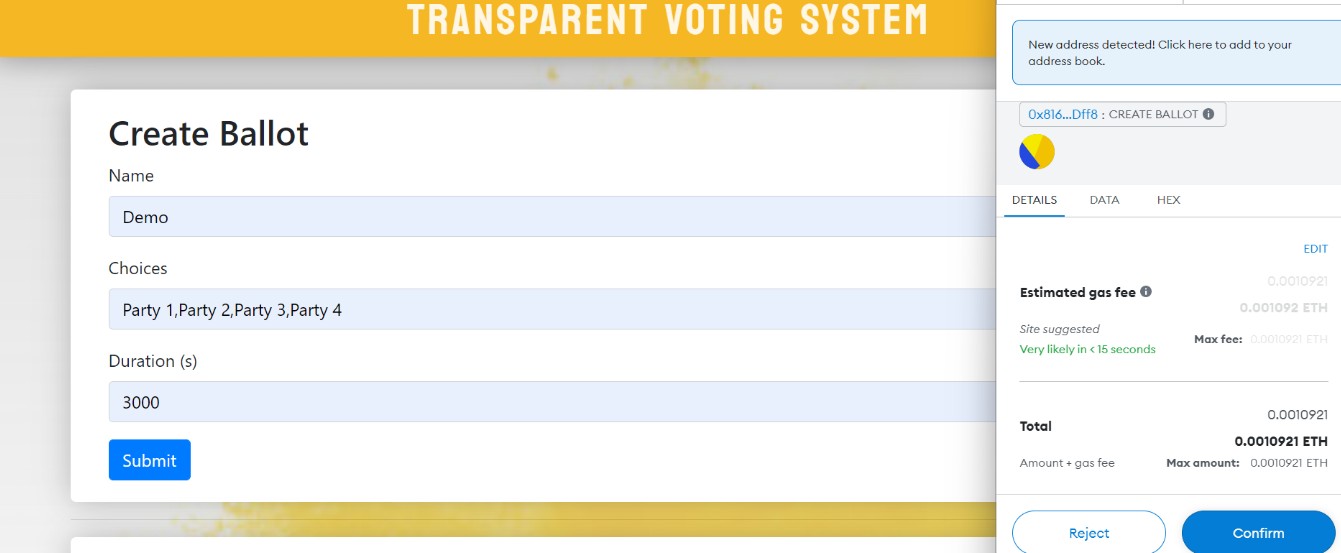
# APPENDICES

## APPENDICES

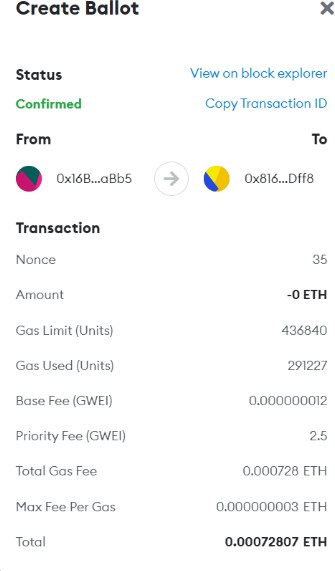
* + - 1. **SAMPLE SCREENSHOTS**



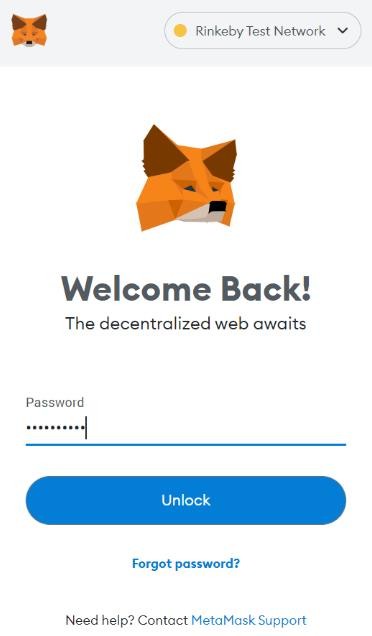
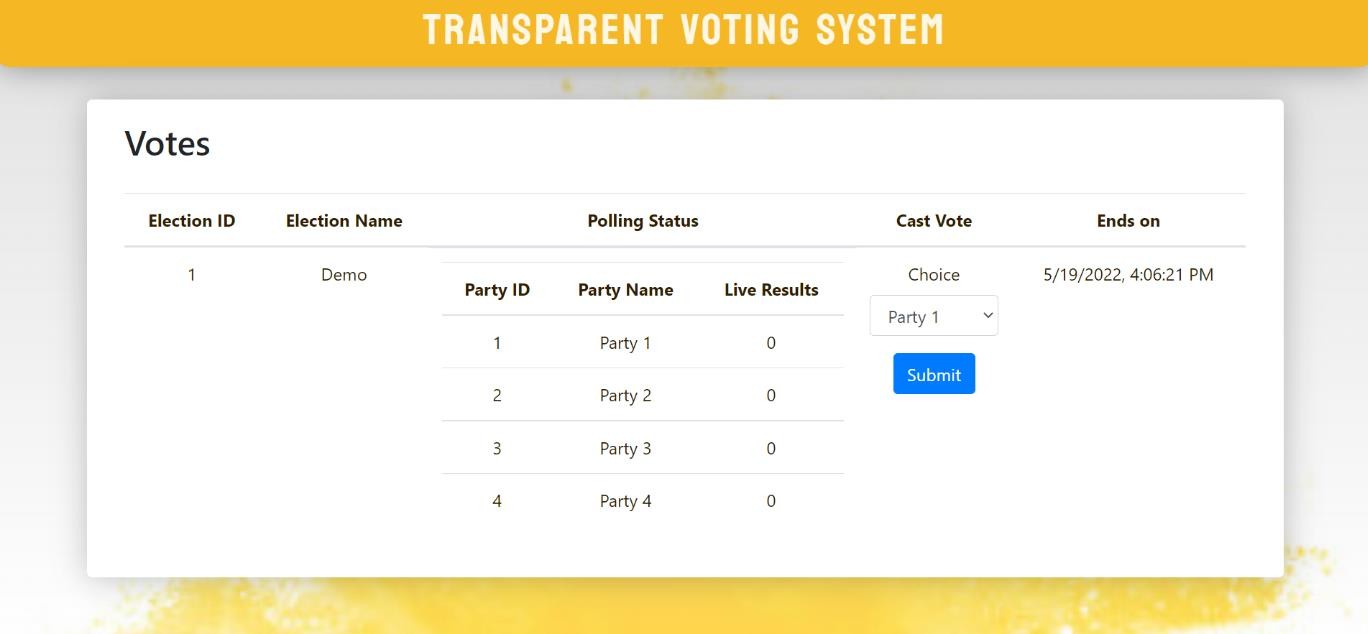
### Figure A.1 – Admin Login



**Figure A.2 – Creation of a new Ballot**

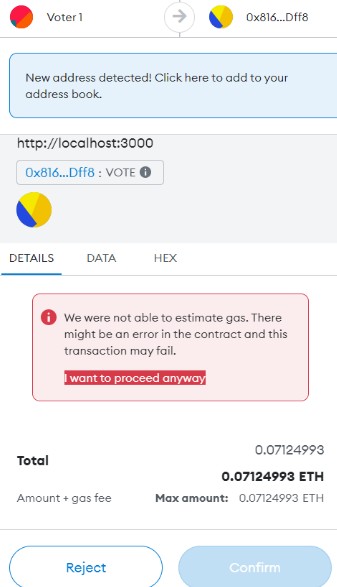


### Figure A.3 – Ballot creation confirmation page

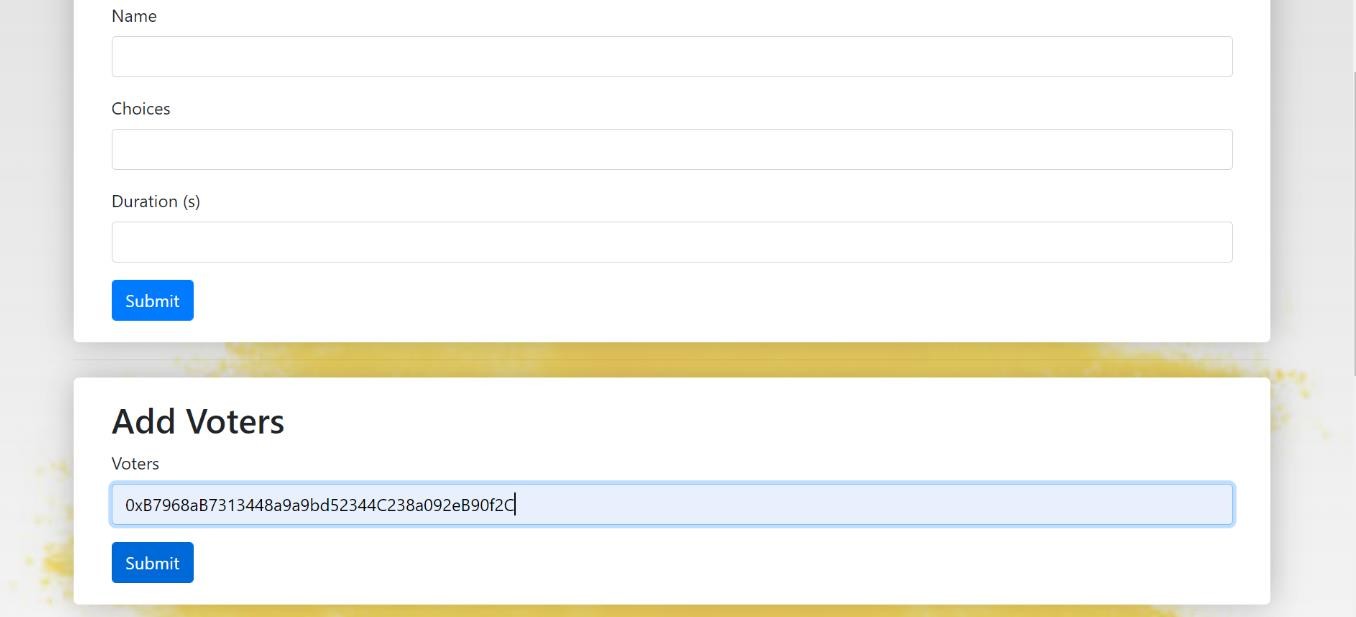


**Welcome!**

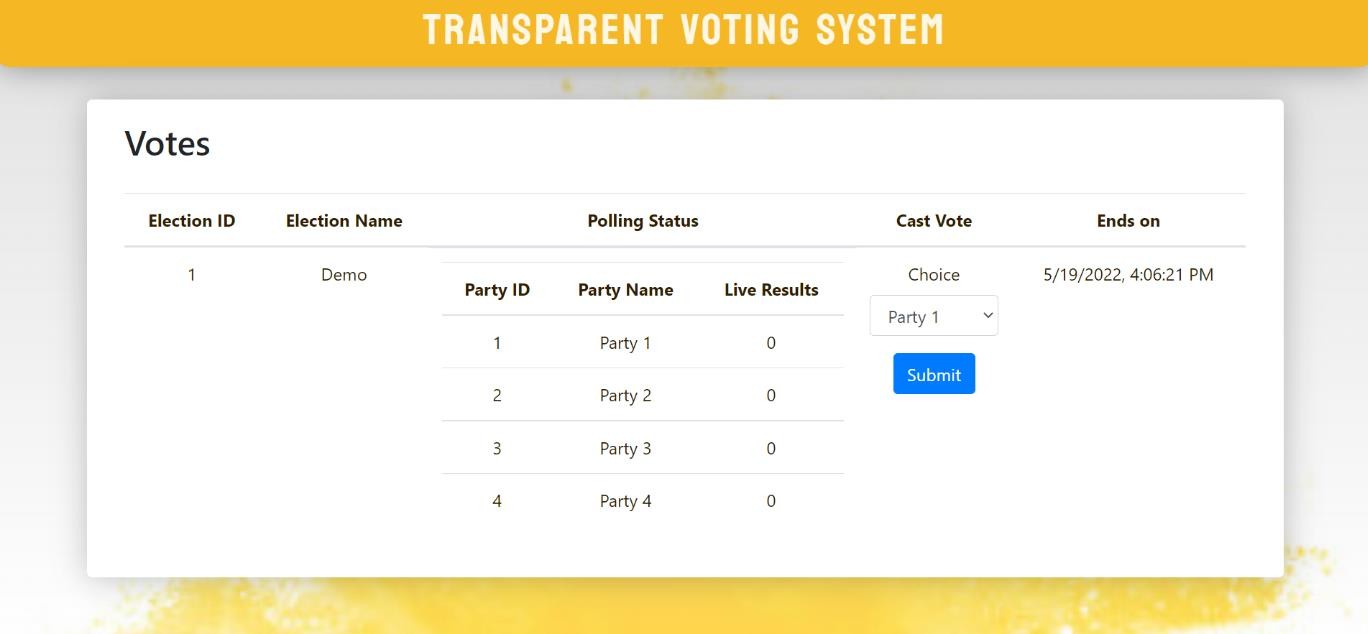
**Figure A.4 – Voter Login**



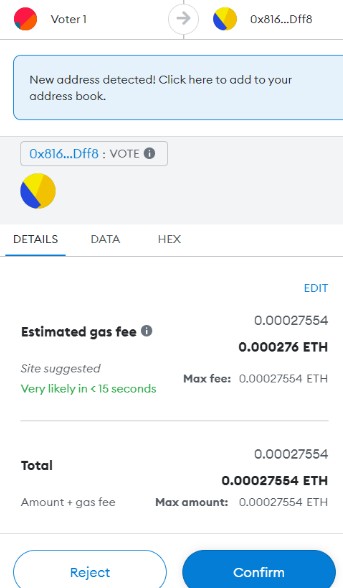
### Figure A.5 – Ineligible voter’s prompt



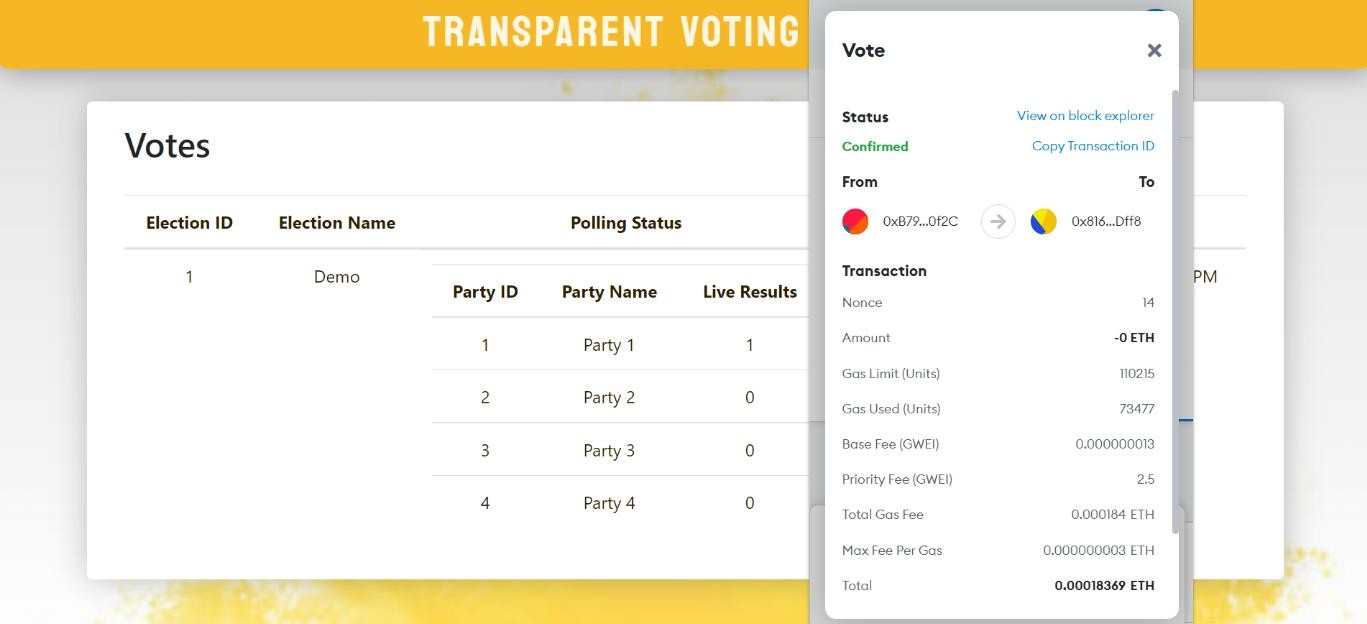
**Figure A.6 – Add Voters**



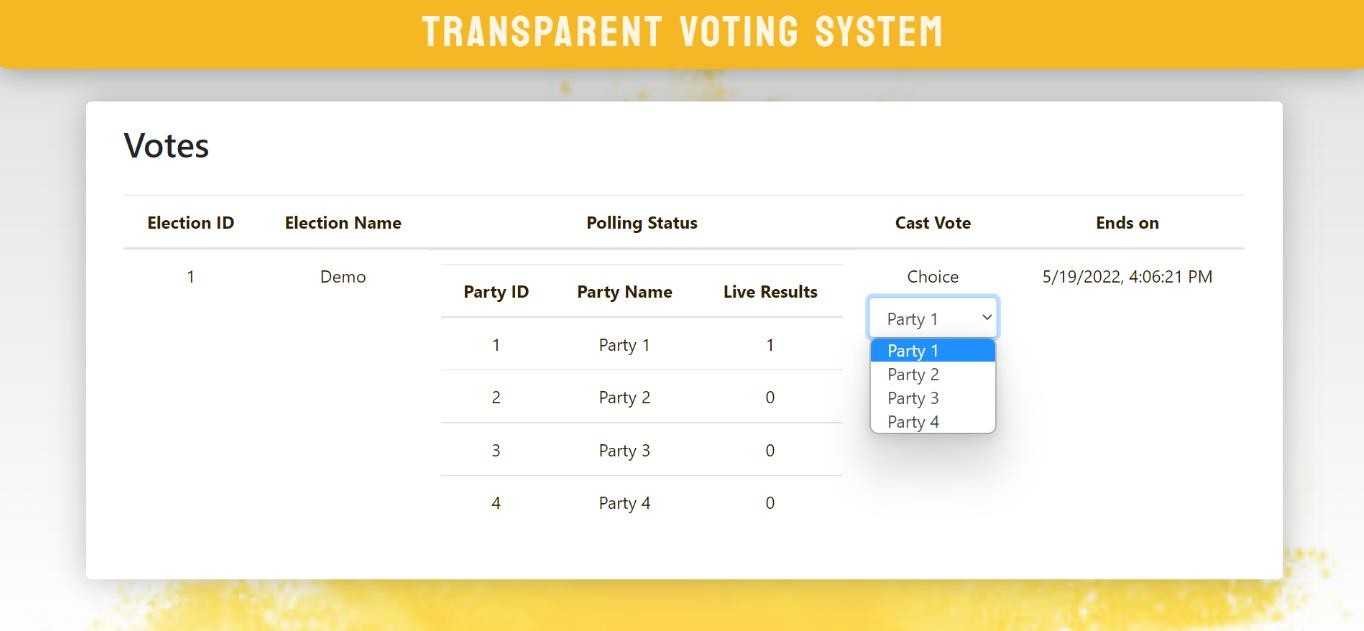
### Figure A.7 – Eligible voter’s page



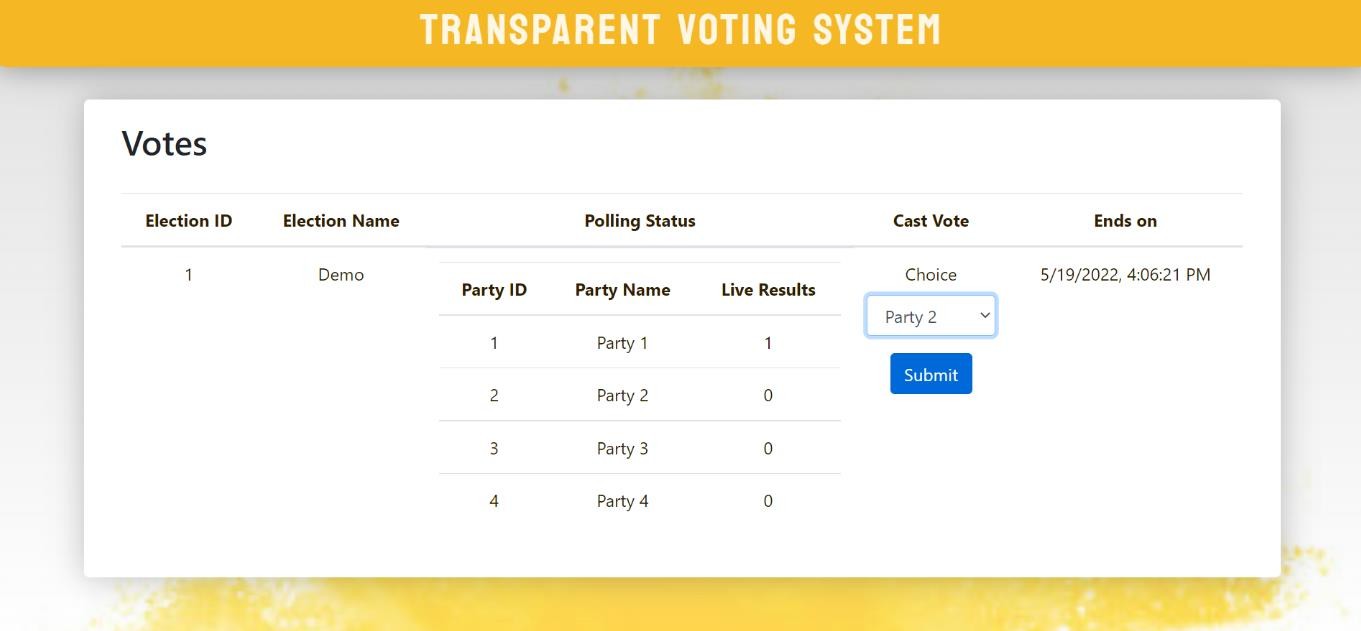
**Figure A.8 – Confirmation of vote**



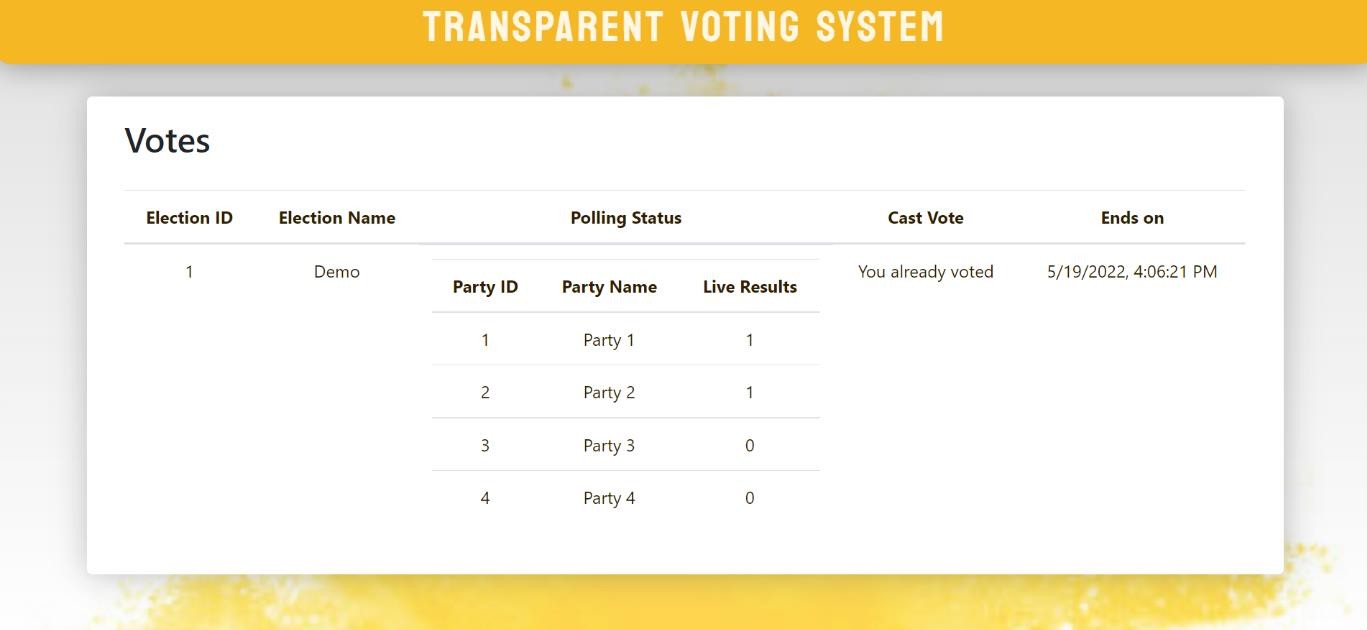
### Figure A.9 – Vote confirmation page with immediate ballot updation



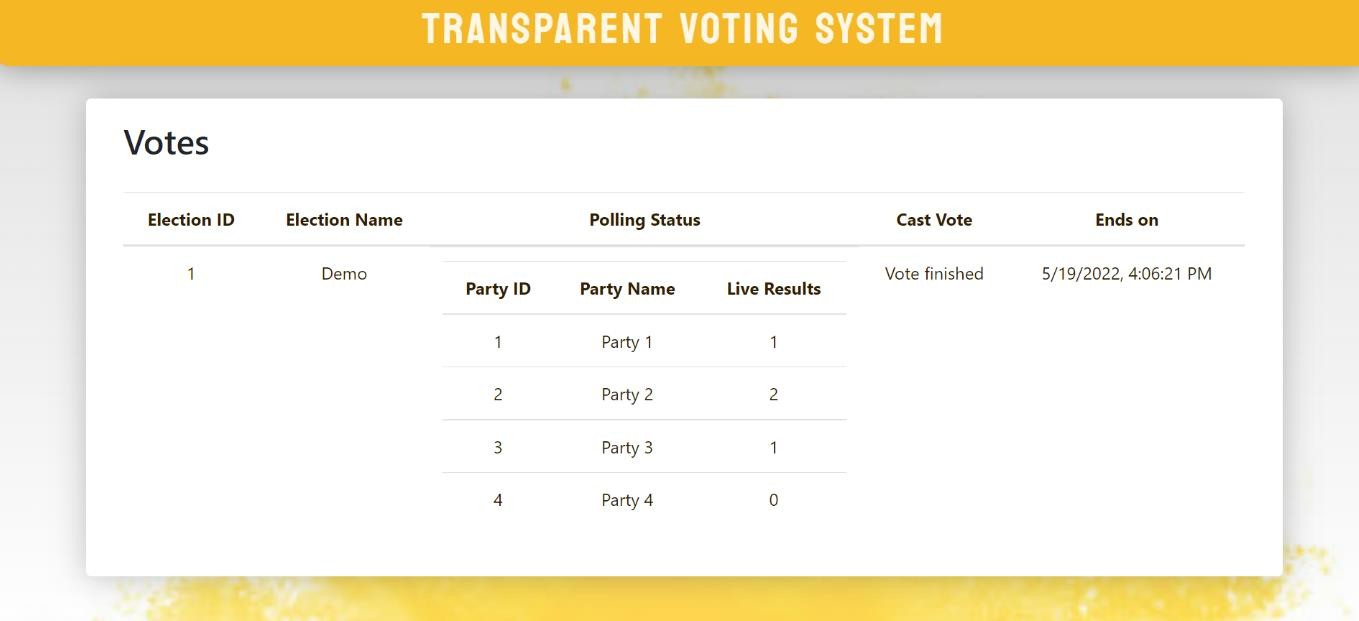
**Figure A.10 – Another voter's page with real-time results**



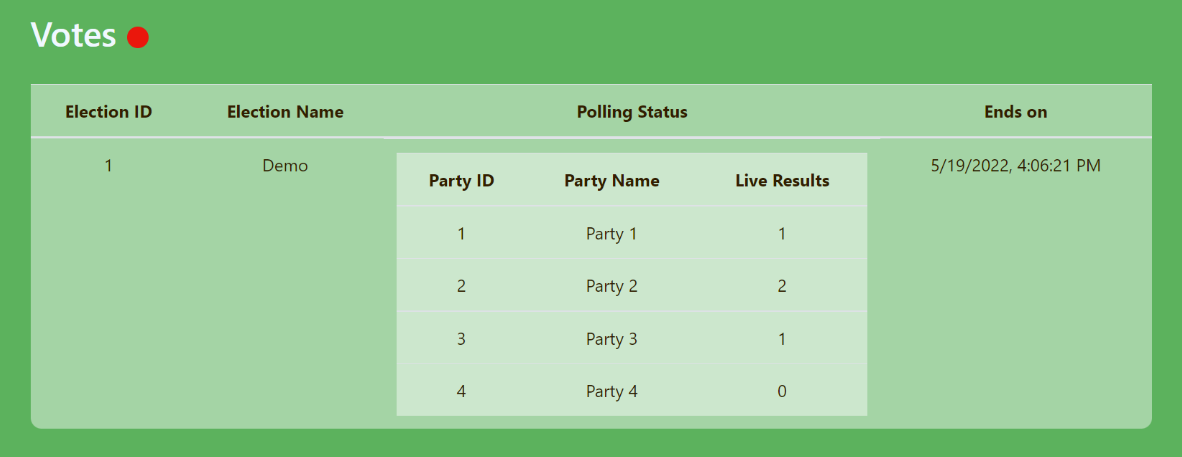
### Figure A.11 – Selection of different parties



**Figure A.12 – Restriction of multiple votes from same voter ID**

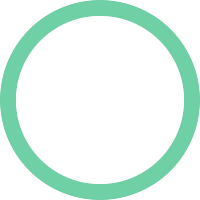


### Figure A.13 – Voter’s page at the end of polling



**Figure A.14 – Admin’s Final results page**

**Plagiarism Scan Report**



**Report Generated on: May 22,2022**

|  |  |
| --- | --- |
| Total Words: | **689** |
| Total Characters: | **4581** |
| Plagiarized Sentences: | **0** |
| Unique Sentences: | **35 (100%)** |

**0%**

**100%**

Plagiarised

Unique

**Content Checked for Plagiarism**

TRANSPARENT VOTING SYSTEM USING BLOCKCHAIN ABSTRACT

The aim of this project is to create a decentralized transparent voting and analysis system that can be implemented with blockchain to provide an efficient and highly secure justifiable method of implementing election systems in countries where traditional physical voting with gameable securities is used, increasing the chances of rigged elections. This system is designed to focus on a secure voting system, lower costs, faster wait times, no disparities due to various erroneous proxies, high scalability, and geographic independence. Overall, an effective election mechanism to strengthen the democratic process. Our dApp allows voters to vote from the comfort of their own homes, saving time and reducing the number of false votes registered.

INTRODUCTION

Many elections conducted on electronic voting machines are hand-counted, and many jurisdictions that use lever voting machines tally absentee ballots by hand. Because these devices are not meant to endure a long time, they must be serviced on a regular basis by only trained mechanics. Apart from equipment maintenance issues, there has also been a problem with only a small fraction of individuals voting. Because when we have to store large amount of sensitive information in the local database, it is difficult to keep it secure. There is also the possibility of casting multiple ballots. To tackle these challenges, we've implemented a decentralised database and a smart contract in our dApp, which stops voters from casting duplicate votes.

EXISTING SYSTEM

The current system lacks a secure blockchain architecture, and existing E-voting applications rely on government-issued data, which isn't the most reliable method of authentication. And also, for an election process to take place, the existing blockchain voting system comprises three phases: authentication, voting, and counting. Where several users' authentication and voting can be done in concurrently, and the counting phase begins only after the voting phase is completed.

PROPOSED SYSTEM

The suggested system features a secure blockchain architecture, as well as well-protected security layers, ensuring that data is safe from all threats. In the proposed system, the authentication phase will be predefined, and the voting and result phases will be run simultaneously, allowing the proposed system to realize the main notion of openly showing live results to each and every voter. The web application in the proposed system is also built utilizing the React8 framework, which makes the system scalable. Our project is efficient in terms of loading speed and performance because we use React JS. In our dApp, we've also included a decentralized database and a smart contract to prevent people from voting twice.

RESULTS AND DISCUSSION



The outcome indicates how blockchain technology can be utilized to remotely conduct elections. We investigate the role of blockchain technology in the smart contract of a transparent voting system. For an election process to take place, the existing blockchain voting system comprises three phases: authentication, voting, and counting. Where several user’s authentication and voting can be done in concurrently, and the counting phase begins after the voting phase is completed. In the proposed system, the authentication phase will be predefined, and the voting and result phases will be run simultaneously, allowing the proposed system to realize the main notion of openly showing live results to each and every voter. This allows individuals to choose their preferred leader.

CONCLUSION AND FUTURE ENHANCEMENT

As a decentralized voting system, we employ blockchain in our implementation. For the voting process, the system will use candidate or voter information. The voting procedures and outcomes will be managed using smart contracts. Our method also improves the efficiency of the validation phase as well as the voting assignment. Our main goal is to create a more efficient and comprehensive approach for accomplishing Transparent Voting System using blockchain technology and its associated variable tools. Integration of the Machine Learning concept by adding a facial recognition system is one of the project's planned developments. Face recognition would be a great addition because it improves security and prevents duplicate and forged votes. Our application can also include two-factor authentication. Even if the credentials provided to the voters are lost, the usage of two factor authentication will help to avoid cast proxy votes.

**No Plagiarism Found**

# REFERENCES

65

## REFERENCES

1. Azaria, Asaph, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman. "MedRec: Using Blockchain for Medical Data Access and Permission Management." In Open and Big Data (OBD), International Conference on, pp. 25-30. IEEE, 2016.
2. E-Voting Systems using Blockchain: An Exploratory Literature Survey, Second International conference on Inventive Research in Computing Applications (ICIRCA), 2020.
3. Guo, Ye, and Chen Liang. “Blockchain application and outlook in the banking industry.” Financial Innovation 2, (no. 1: 24), 2016.
4. Ikhsan Darmawan E-voting adoption in many countries: A literature review, First Published October 12, 2021.
5. Nakamoto Satoshi, inventing bitcoin, implementing the first blockchain, deploying the first decentralized digital currency “A Peer-to-Peer Electronic Cash System” original 20 March 2014.
6. Nicole J. Goodman; Jon H. Pammett, “Internet Voting in a Local Election in Canada”, in Internet and Democracy in Global Perspective, Studies in Public Choice 31, Eds. Bernard Grofman, Alex Trechsel, and Mark Franklin, Springer Verlag, 2014.
7. Rafer Cooley; Shaya Wolf; Mike Borowczak Conference: IEEE International Smart Cities Conference (ISC2), 2018.
8. R.S. Yashank E-Voting System using HyperledgerSawtooth, Communication & Materials (ICACCM), International Conference on 2020.
9. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, Computer Networks: The International Journal of Computer and Telecommunications Networking, December 2017.

66