



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SHARDA SCHOOL OF ENGINEERING AND TECHNOLOGY
SHARDA UNIVERSITY, GREATER NOIDA**

Scam-free Election using Blockchain

A project submitted

*in partial fulfillment of the requirements for the degree of Bachelor of
Technology in Computer Science and Engineering*

by

Aman Verma (2019007415)

Harshit Garg (2019587586)

Arpit Pahwa (2019557429)

Supervised by:

Mr. Bal Saraswat

Asst. Professor

CERTIFICATE

This is to certify that the report entitled “**Scam-free Election using Blockchain**” submitted by “Aman Verma (2019007415), Harshit Garg (2019587586) and Arpit Pahwa (2019557429)” to Sharda University, towards the fulfillment of requirements of the degree of “**Bachelor of Technology**” is record of bonafide final year Project work carried out by them in the “Department of Computer Science & Engineering, Sharda School of Engineering and Technology, Sharda University”.

The results/findings contained in this Project have not been submitted in part or full to any other University/Institute forward of any other Degree/Diploma.

Signature of the Guide

Name: Mr. Pradeep Kumar Mishra

Designation: Asst. Professor

Signature of Head of Department

Name: Prof.(Dr.)Nitin Rakesh

Place: Sharda University

Date:

Signature of External Examiner

Date:

ACKNOWLEDGEMENT

A major project is a golden opportunity for learning and self-development. We consider ourselves very lucky and honored to have so many wonderful people lead us through in completion of this project.

First and foremost we would like to thank Dr. Nitin Rakesh, HOD, CSE who gave us an opportunity to undertake this project.

Our grateful thanks to Mr. Pradeep Kumar Mishra for her guidance in our project work. Mr. Pradeep Kumar Mishra, who in spite of being extraordinarily busy with academics, took timeout to hear, guide and keep us on the correct path. We do not know where we would have been without her help.

CSE department monitored our progress and arranged all facilities to make life easier. We choose this moment to acknowledge their contribution gratefully.

Name and signature of Students:

Aman Verma (2019007415)

Harshit Garg (2019587586)

Arpit Pahwa (2019557429)

ABSTRACT

In today's democracy, elections are very important but large sections of society around the world do not trust their electoral system which is a major concern for democracy.

Even the world's largest democracies like India, the United States, are still plagued by flawed electoral systems. Voter fraud, EVM (electronic voting machine), electoral fraud, and confiscation of polling stations are major problems in the current voting system.

Blockchain is a emerging, distributed, and distributed technology that promises to improve the various aspects of many industries. Increasing e-voting in blockchain technology could be a solution to the concerns of the e-voting system.

Blockchain is a powerful tool because of its smart contracts and many features that override traditional systems.

Blockchain, with smart contracts, emerges as a viable candidate for the development of secure, cheap, secure, transparent, and easy-to-use electronic voting systems. Due to its consistency, extensive use, and logic provision of smart contracts, Ethereum and its network are one of the most appropriate.

TABLE OF CONTENTS

Title Page	I
Certificate	II
Declaration	III
Acknowledgement	IV
Abstract	V
List of Figures	VIII
List of Tables	IX
1. INTRODUCTION	
1.1 Problem Definition , significance and objective	10
1.2 Methodologies	11
1.3 Outline of the project	11
1.4 Scope of the project	12
1.5 Organization of the report	13
2. LITERATURE SURVEY	
2.1 Introduction to the problem domain terminology	14
2.2 Existing solutions for decentralized e-voting system	17
2.3 Related works	18
2.4 Tools/Technologies used	20
3. DESIGN OF THE PROPOSED SYSTEM	
3.1 Block Diagram	21
3.2 Module Description	23
3.3 Theoretical Foundation/Algorithms	
4. IMPLEMENTATION	
4.1 Flowcharts /DFDs / ER Diagrams	25
4.2 Design and Test Steps / Criteria	30
4.3 Algorithms / Pseudo Code	30
4.4 Data Set description	32
4.5 Testing Process	32

5. RESULTS	33
6. CONCLUSIONS	
6.1 Conclusions	
6.1.1 Limitations	46
6.2 Recommendations /Future Work / Future Scope	47
REFERENCES	48
APPENDICES	51

LIST OF FIGURES

Figure	Page No.
3.1 System Architecture	21
3.2.1 Frontend and backend modules	23
3.2.2 Dependencies	24
4.1 ER diagram	25
4.2 Case diagram	26
4.3 Class diagram	27
4.4 Sequence diagram	28
4.5 Block Chain Deployment	29
5.1 Homepage	33
5.2 User login	33
5.3 Signup page	34
5.4 User dashboard	34
5.5 Voter area	35
5.6 Results of Voting	35
5.7 About page	36
5.8 Add candidates	36
5.9 Candidate details	37
5.10 Change Election Phases	37
5.11 Analytics of Voting	38

5.12 Metamask wallet	39
5.13 Contract information	40
5.14 Etherscan	41
5.15 Gas fees	42
5.16 Ganache Suite	43
5.17 Account Information for each key	43
5.18 website on google search	44

List of Tables

Table	Page No
1- What a Transaction Contain	45

1.INTRODUCTION

A set of regulations that govern how elections and referendums are held and how their outcomes are determined is known as an electoral system or voting system. Governments conduct political electoral systems, whereas non-political elections can occur in business, non-profit organisations, and unofficial organisations. On election day, e-voting systems may fail, lose, or scatter votes, or perform improperly. Because a defect in an electronic system has the ability to be systematic and centralised, as opposed to a fault in a manual system, which is more likely to be localised and random, even small errors can have disastrous results.

Since a defect in an electronic system has the ability to be systematic and centralised, as opposed to a fault in a manual system, which is more likely to be localised and random, even small errors can have catastrophic results. Even the biggest democracies in the world, including India and the United States, still have unreliable voting procedures. The main problems with the existing voting system include vote rigging, EVM hacking, election manipulation, and polling place capturing.

1.1 Problem Definition , significance and objective

Problem Definition : The current voting processes, including ballot box voting and electronic voting, are labor-intensive, time-consuming, and vulnerable to a variety of security risks, including DDoS assaults, voting booth hacking, vote-rigging, and fraud. As a result, existing systems become distrustful of one another.

Disadvantages: Long lines during elections, Security breaches such as data leaks, vote rigging, It is difficult for disabled voters to reach the polling station, The cost of elections is high.

Significance : This project report is to provide comprehensive information about the technical aspects and real-time design of the project - Extended E Voting System Using Block Chain Technology (Web application for citizens of each country).

Objective: The objective of this project is to build a platform using Block chain smart contracts.

Detailed research of electoral processes as it relates to voting. Design and develop software forums for voter registration, election voting, real-time election collection and monitoring and especially voters reaching remote elections.

1.2 Methodologies

This project will be a web application with front end, back end and database. Registration of voters and candidates must be done in advance. Prior to creating accounts, personal verification must be completed. The authorised person must confirm the eligible users by presenting a coin or token after checking the identity document. Each user may only cast a single vote using this coin or token. The blockchain verification procedure will make sure that it is impossible to use this symbol twice. Therefore, a user cannot vote more than once. The electronic voting system built on blockchain has been enhanced. Elections are not managed by a central body. Votes are cryptographically protected.

An app dedicated to electronic voting, or dApp, built into the Ethereum blockchain. A smart Ethereum contract was signed with Solidity to get the votes cast. User communication on the client side is designed to use Ethereum accounts for voting.

1.3 Outline of the project

At the end of the project, our application accomplishes all the functional requirements.[16]

- Votes are cryptographically protected.
- The votes cast are unchanged and uninterrupted.
- Keeps voter privacy and anonymity.
- The E voting system can improve voter participation.
- It can improve efficiency and allow faster results.
- Promotes transparency and clarity in the system.
- Eliminates misunderstandings arising from incorrect / vague selections made on ballot papers.
- Voting results read publicly.

1.4 Scope of the Project

This project is intended to work on various devices such as mobile, PC / Laptop. This project will only work once the main function in the backend has been completed. Blockchain is a distributed spreadsheet program that accesses, verifies, and transmits network data through distributed nodes.

First of all, the objectives of the project will be determined. Then the software requirements will be analyzed. The planning languages involved, the frameworks used, the technology to be used etc. will be analyzed.

Then the hardware requirements are understood, what type of hardware is required, etc., will be understood.

Hardware will be acquired and configured to operate according to written software requirements. The final product will be tested, and the results will be recorded.

1.5 Organization Of The Report

This report will clearly explain the terminologies used, tools used, proposed implementations, results of the implementations, and conclusions drawn from the results. After this introduction section,

section 2, the literature survey, will introduce the problem in detail. It will go over the necessary domain terminology, mention the tools, and discuss how they will be used. We will then go over the existing solutions that have already been presented along with any other related works.

Section 3 ,We will look into the details of how exactly we implement this tool and how this tool is run and implemented differently from the existing solutions. Block diagrams, implementation architectures, module descriptions, and any other relevant descriptions of the tool's design will be clearly presented in this section.

Next, section 4 will show the implementation of the proposed system. This section will depict the implementation of the tool through flowcharts, DFDs, ER Diagrams, etc. The algorithms or pseudo code will also be presented in this section. Any data sets would also be presented and described in this section.

In section 5, We'll look at how the tool run at each step and displaying all the outputs and results obtained from the training and execution of our proposed model.

Finally in section 6, we'll come to a conclusion, based on our implementation and results, whether the tool successful. Finally, after observing all the results and drawing conclusions, we present our recommendations on how to use this tool, who should use this tool, what scenarios is it most effective in, how we can improve it, and so on. The future works and scopes of this project will also be discussed in this section.

2. LITERATURE SURVEY

2.1 Introduction to the problem domain terminology

Let us discuss the terminology related to the problem domain.

2.1.1 BlockChain

A blockchain is a website that computer network nodes share with one another. In the same way that a website saves digital data, a blockchain does the same. The crucial role that blockchains play in cryptocurrency systems like Bitcoin by maintaining a secure and independent record of transactions is what makes them most well-known.

2.1.2 SmartContract

Smart contracts are simply blockchain-based programs that work where predetermined conditions are met. They are usually used to automatically execute a contract so that all participants can be immediately assured of the outcome, without the involvement of any arbitrator or loss of time.

2.1.3 Ethereum

Ethereum is a blockchain platform with its own cryptocurrency, called Ether (ETH) or Ethereum, and its own programming language, called Solidity. Ethereum is an all-inclusive public bridge to verify and record activity.[12]

2.1.4 Dapp

DApps works similarly to standard web applications; However, they do get their status and data on a blockchain network (or multiple blockchain networks). DApps does not require a central web server to operate and can only communicate via the blockchain network messaging protocol connected to it.

2.1.5 Solidity

Solidity is a programming language used to develop smart contracts in Ethereum and other blockchain platforms, such as Monax and its Hyperledger Burrow blockchain.[12]

2.1.6 Metamask

A cryptocurrency wallet called MetaMask is used to interact with the Ethereum network. enables users to communicate with isolated programmes by giving them access to their Ethereum wallet through a browser extension or mobile application.

2.1.7 Truffle

Intelligent Contract development area that integrates features such as an exploration framework and a supply chain for blockchains using Ethereum virtual machine (EVM) which makes engineer unit testing faster and easier.

2.1.8 web3.js

Web3.js is used to connect a localhost or remote ethereum node using HTTP, IPC or WebSocket.[13]

2.1.9 Hyperledger sawtooth

Hyperledger Sawtooth provides a flexible and modular architecture that separates the main program from the application domain, so smart contractors can specify the business rules of the applications without knowing the basic design of the main system.

2.1.10 Invisible signature

A blind signature is a type of digital signature where the message's content is first encrypted or "blinded" before being signed. A typical digital signature can be used to publicly verify the resulting blind signature against an actual, error-free communication.

2.1.11 Merkel Tree

The Merkle Tree is an integral part of blockchain technology. It is a mathematical data structure composed of hashes of different data blocks, and serves as a summary of everything that is done in the block.

2.1.12 DAO

DAO represents the Decentralized Autonomous Organization. As the name implies, it is an independent and divided organization. At one time, it was also known as the Decentralized Autonomous Corporation (DAC), but the term DAO is widely used because not all corporations are corporations.

2.1.13 Distributed Manual

A distributed ledger is a type of shared database that is agreed upon, duplicated, and synchronized between members of a shared network. All information in this file is stored securely and accurately using cryptography.

2.2 Existing solutions for decentralized E-voting system

Voting online is a growing practice in today's society.

One-on-one risk can lead to massive voter turnout. When utilised in elections, electronic voting systems must be reliable, accurate, secure, and competent. Acquisitions, however, might be restricted to issues that might arise with electronic voting systems. Due to the advantages of end-to-end voting, blockchain technology is utilized to resolve these problems, provides electronic voting booths, and produces electronic voting systems. This technology effectively replaces conventional electronic voting solutions with distributed, unrestricted features, and security features.[15]

This study offers an introduction to the fundamental structure and blockchain features related to electronic voting as well as an overview of the blockchain-based targeted electronic voting application.

As a consequence of this investigation, it has been discovered that blockchain programmes can assist in resolving some of the issues that electoral systems are currently facing. The protection of anonymity and transaction speed, on the other hand, are the problems with blockchain systems that are most frequently raised. Remote participatory security must be operational and the transaction speed must be regulated in an electronic voting system built on a blockchain. These issues led to the decision that the current structures needed to be enhanced before being employed in voting systems.

The voting procedure became clearer and more accessible, illegal voting was halted, data protection was increased, and voting results were reviewed thanks to Blockchain technology, which fixed the flaws in the current electoral system.

2.3 Related works

[1] **“Implementation of a Blockchain Enabled E-Voting Application Within IoT-Oriented Smart Cities”**- their approach is a two-pronged approach, namely, both national election bodies and the rest of the organization can ensure security when IoT devices are compromised using the blockchain method. The Blockchain voting system uses not only electoral bodies but also informed voters in case they interfere with their votes before the scheduled counting date.

[2] **“Votereum: An Ethereum-Based E-Voting System”**- On the other hand, EthVote is a site-specific programme that utilises a smart contract and operates on the Ethereum blockchain. Only those who are eligible to vote in this process—whose identities are unknown—are permitted to do so. When a communication is signed, unless its content is revealed and confirmed as a regular digital signature, this is accomplished by utilising a Blind Signature.

[3] **“A blockchain based E-voting system”**- They have analyzed three blockchain frameworks for implementation and use our smart election contracts. Those are Exonum, Quorum and Geth. One of the three primary applications of the Ethereum protocol is Go-Ethereum, or Geth. It employs smart contract strategies exactly as intended without the risk of inefficiency, research, fraud, or outside meddling. This framework is the most suitable one for the developer and supports the expansion of the Geth protocol. .The level of performance depends on the blockchain being used as a public or private network.they use a region-based voting system.

[4] The Hyperledger Sawtooth framework is used to ensure consistency using the same transaction processing, and to use two different stages in a single blockchain, to ensure anonymity and fairness in the voting process. and blockchain business logic, a smart contract, to be implemented.

[5] **“ElectionBlock: An Electronic Voting System using Blockchain and Fingerprint Authentication”**- ElectionBlock adheres to the standard principles of central blockchain technology and integrated biometric authentication. Registered voters will have their biometrics registered on the website. will verify that the user is a registered voter, and secondly, will check the blockchain to ensure that the user has not cast a vote. Voting is accelerated using the Merkle Tree algorithm.

[6] **“Decentralized Voting Platform Based on Ethereum Blockchain”**- The system's essential components include safeguarding data transparency and integrity and requiring one vote per phone number for each poll with privacy assured. As a platform for executing the Blockchain, Ethereum Virtual Machine (EVM) is employed in order to do this.

[7] **“Comparitive Analysis on E-Voting System Using Blockchain”.****“Fraud-Resistance”**- The system should verify the identity of each potential voter and determine his or her status, but should not allow this information to be associated with their vote. The “Easy to Use” option should work for the whole community. It should be designed in such a way that it can be used with minimal training and some technical skills.

[8] **“A Smart Contract For Boardroom Voting with Maximum Voter Privacy”** - doped an online voting protocol with separate features and greater voting privacy using Open Vote Network (OVN) .OVN is a smart Ethereum Blockchain contractor. After using the program the creators concluded that it cost \$ 0.73 per voter in the program. Researchers soon found that OVN was at risk of being attacked by DOS and traffic jams at the time of purchase.

[9] **“Securing e-voting based on blockchain in P2P network”**- Designed a harmonized model of DLT-based voting records to avoid vote-rigging.Design an ECC-based user verification model to provide authenticity and non-refusal.Design a withdrawal model that allows voters to change their vote before the deadline.

[10] **“Implementation of an E-Voting Scheme Using Hyperledger Fabric Permissioned Blockchain”**- Compared to latency it has been found that with a low transaction value, the Ethereum delay is 2x times the Hyperledger.And in the transition no transaction the Hyperledger rate output is greater than Ethereum.

1.4 Tools/Technologies Used

React.js - React is a JavaScript library for building user links. Used for building single page applications. React allows us to create reusable UI components.

We used React.js for developing front-end of the E-voting Application.

3.DESIGN OF THE PROPOSED SYSTEM

3.1 Block diagram

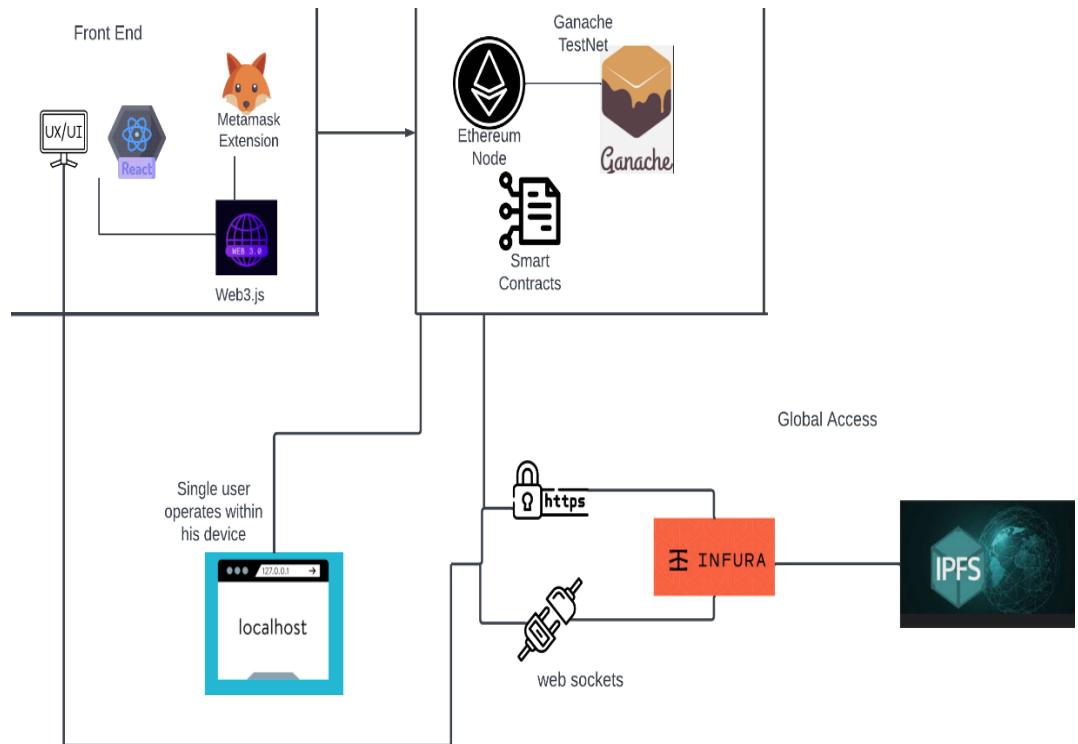


Figure 3.1 System Architecture

1.Frontend: The part visible to the user. It is an interface through which users will interact the components/pages available in the website . It also contains all the stylized pages. This is created mainly using React.js and styled using MaterialUI. The Metamask extension is the wallet which is used by the user for initiating transactions.

2.Backend:The website is fully functional in localhost.To make it get accessed by everyone we have used Infura IPFS server where we hosted the website on it,and it is accessible everywhere.

IPFS is a distributed system .Designed for speed and simplicity, the IPFS API for Infura and the dedicated gateway connects applications of all sizes to a secure, distributed environment, opening up a web-resistant webpage.

3.Blockchain: A **smart contract** is a tamper-proof program that runs on a blockchain network.we have used Ethereum blockchain Network in this project.[11]

3.2 Module description

- **Metamask:** A plugin for the browser that serves as an Ethereum wallet and is installed similarly to other plugins. After being deployed, it gives users the ability to hold Ethereum and other ERC-20 tokens, allowing them to conduct transactions at any Ethereum address. The website can be accessed in the local computer or globally using infura IPFS.
- The voting system is deployed on infura so as to access globally.

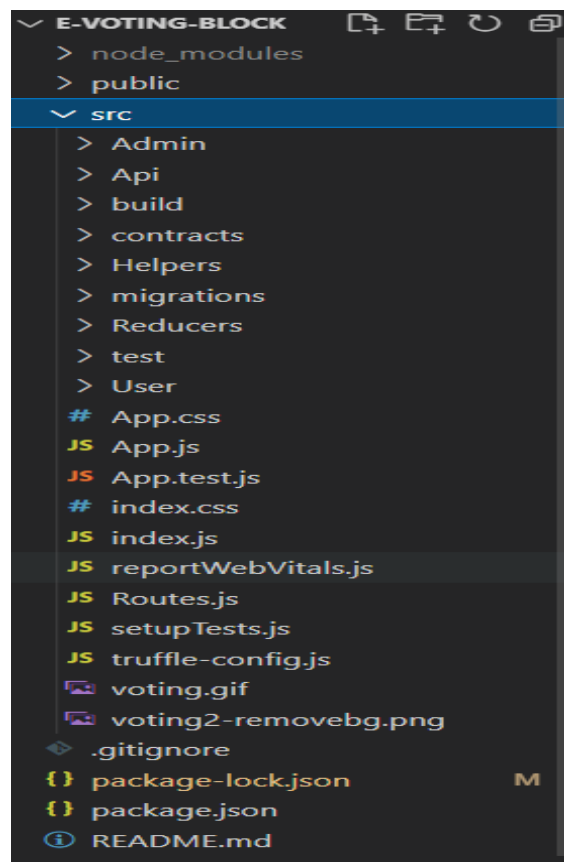


Figure 3.2.1 Frontend and Backend Modules

The project is developed with React.js where many files have been coded and many libraries have been imported. It has many components like Admin, User, Contracts and soon. The project has been deployed on Infura Ipfs platform.

Dependencies:

```
5  "dependencies": {  
6    "@emotion/react": "^11.8.2",  
7    "@emotion/styled": "^11.8.1",  
8    "@mui/icons-material": "^5.5.0",  
9    "@mui/material": "^5.5.0",  
10   "@testing-library/jest-dom": "^5.16.2",  
11   "@testing-library/react": "^12.1.4",  
12   "@testing-library/user-event": "^13.5.0",  
13   "@truffle/hdwallet-provider": "^2.0.4",  
14   "firebase": "^9.6.8",  
15   "radium": "^0.26.2",  
16   "react": "^17.0.2",  
17   "react-confetti": "^6.0.1",  
18   "react-dom": "^17.0.2",  
19   "react-loading": "^2.0.3",  
20   "react-redux": "^7.2.6",  
21   "react-router-dom": "^6.2.2",  
22   "react-scripts": "5.0.0",  
23   "recharts": "^2.1.9",  
24   "web-vitals": "^2.1.4",  
25   "web3": "^1.7.1"  
26 }
```

Figure 3.2.2 Dependencies

These are the dependencies that have been installed to develop the project. There are many versions but we have used the ones which are compatible for our project.

4. IMPLEMENTATION

4.1 Flowcharts / DFDs / ER Diagrams

ER Diagram

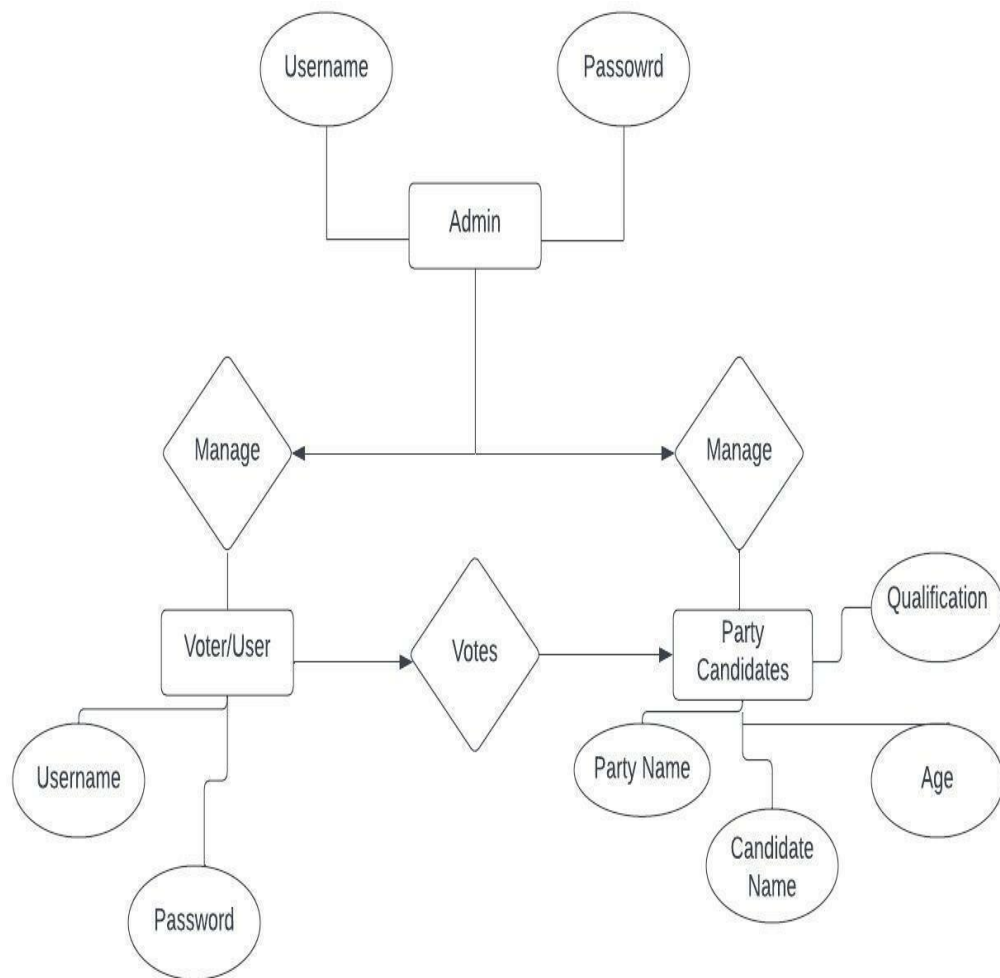


Figure 4.1 ER-Diagram

Here the Admin will manage the Voters and Party Candidates, the user will cast his vote. Only the voters who are registered will be eligible for voting.

Class Diagram

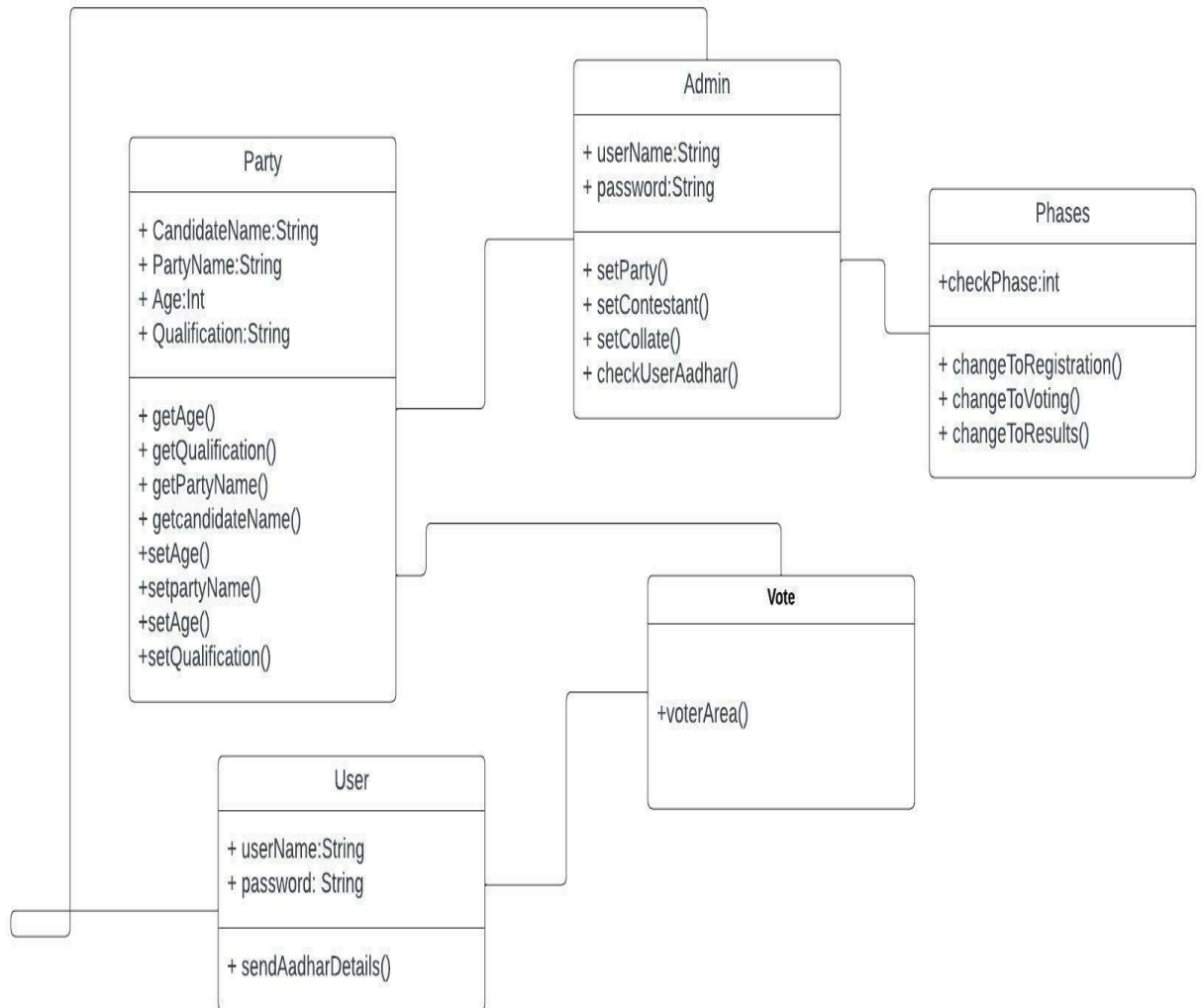


Figure 4.2 Class Diagram

This is a class Diagram where classes have been created for party, Admin, Phases, User, Vote. Each class has different Attributes and Methods.

Sequence Diagram

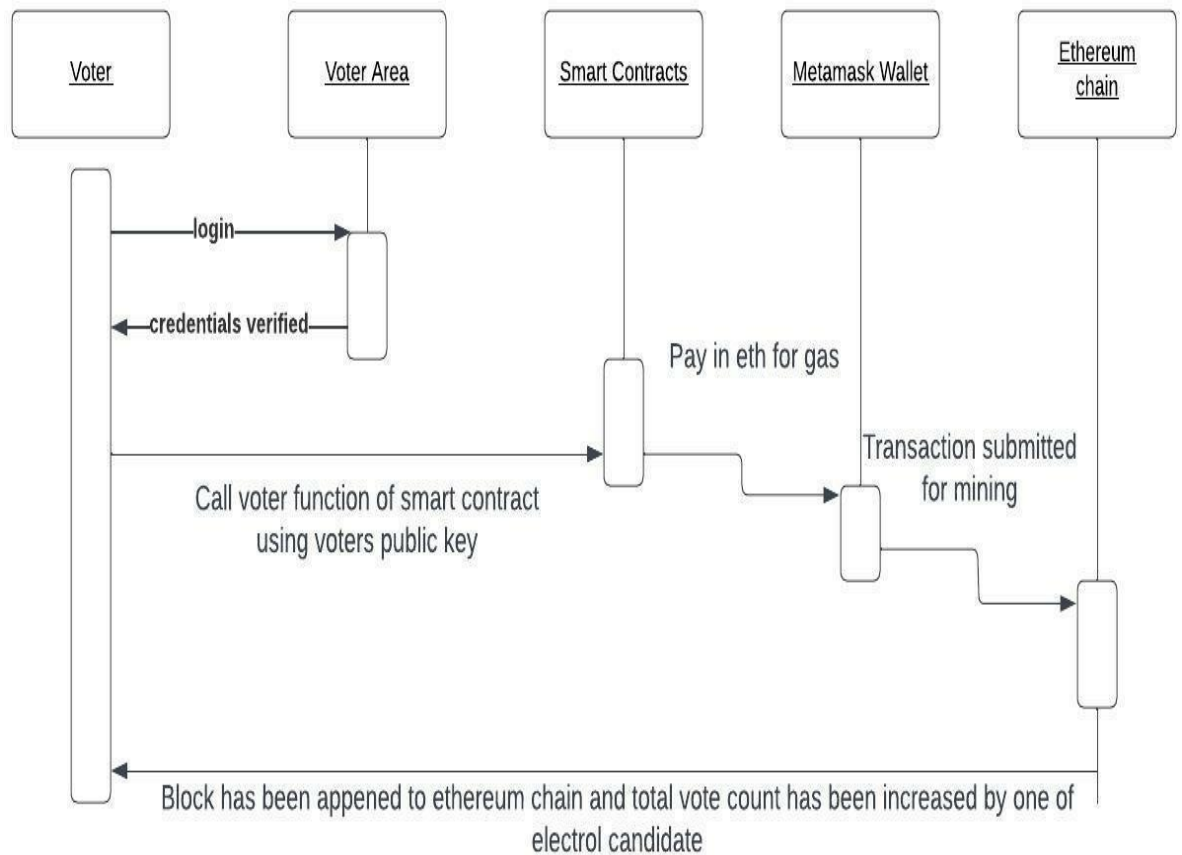


Figure 4.3 Sequence Diagram

This is a sequence diagram which shows the flow of casting of a vote from user/voter to Ethereum chain. The Transaction will be submitted for mining and block will be appended to Ethereum chain and total vote count will be increased by one on electrol candidate.

Use Case Diagram

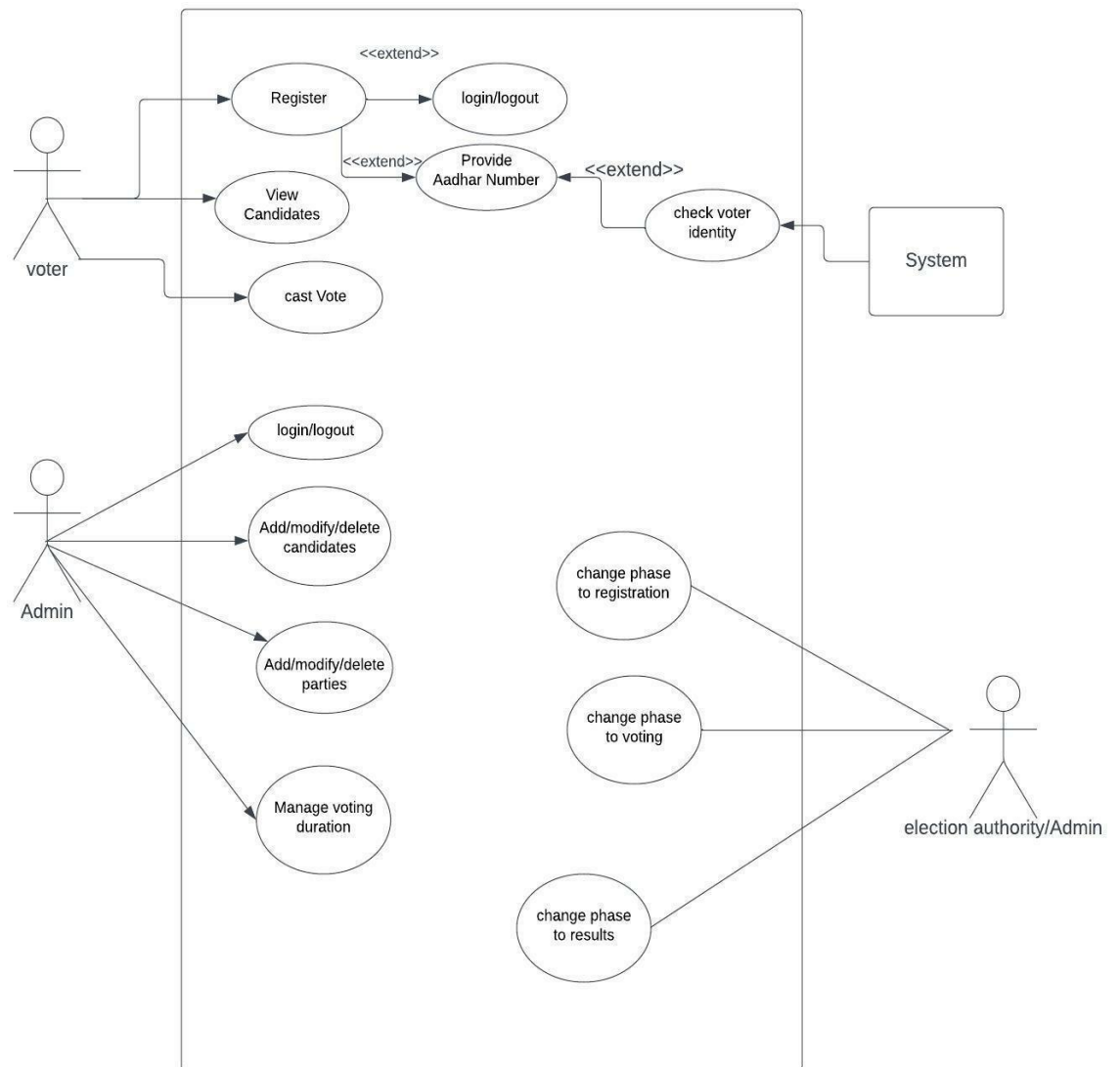


Figure 4.4 Use Case Diagram

This is a use case diagram which describes the role of each of the voter and Admin. The Admin manages the overall elections. The system checks voter identity and Admin will verify the voter details.

BLOCK CHAIN DEPLOYMENT

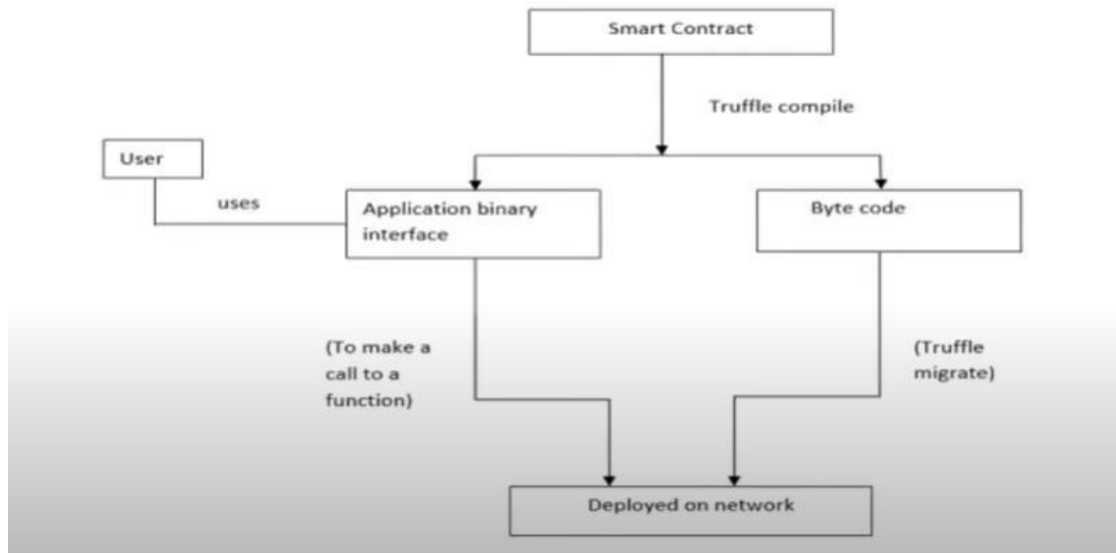


Figure 4.5 Block Chain Deployment

The Truffle compile the smart contract and is converted to byte code the truffle will migrate the byte code and will be deployed on the network.

The user uses a Application binary interface to make a call to a function.

4.2 Design and Test Steps / Criteria

- User has to create an account with Email and login to website
- Then user need to register with aadhar number and admin checks whether the aadhar is eligible to register or not.
- Admin will activate the user account.
- Admin will add the candidates who are participating in elections and change the phase to “voting”
- Now Voting will become live and users start voting to candidates and results are hidden .
- Admin has to change the phase to “Election is done” and results are shown.
- The system is non hackable and is fully secured.
- The system does not allow user to cast his vote more than once.

4.3 Algorithms / Pseudo Code

4.3.1 Creating accounts and verification process

Step1: Start

Step2: User will create his account with email and password.

Step3: User need to give his Aadhar number after login ,admin will verify the number.

Step4:Stop

4.3.2 Adding Candidates interested in elections and election phase

Step1: Start

Step2: Admin will add the candidates and turn the phase to Election Phase on Election day.

Step3: Users start casting their votes and once voted cannot cast again the system shows warning.

Step4: The transaction will happen with metamask wallet and Ethereum will be deducted from user account.

Step5: Stop

4.3.3 Result Phase

Step1: Start

Step2: Admin will change the phase of elections to results and users can't cast their vote.

Step3: users can view the results by going to the results tab in the website.

Step4: Stop

4.4.4 Ethereum deduction

Step1: Start

Step2: Ethereum will be deducted for user creation, Admin creation, Changing election phases.

Step3: to view the transactions, open etherscan.

Step4: Stop

4.4 Data Set description

This is a web based application and it doesn't require any data set.

4.5 Testing Process

- Different User accounts have been created and added candidates participating in elections.
- Casted votes from created account and checked whether user can cast the vote more than once or not.
- All the block chain transactions have been checked carefully and no problem arised.

5 . RESULTS

In this project we used React.js for developing front-end and solidity to write smart Contracts. we used Metamask wallet for performing transactions using ether. The application is working and is deployed on Netlify hosting site, everyone can access the website from any location. The transactions in the application are immutable, transparent and are secured.

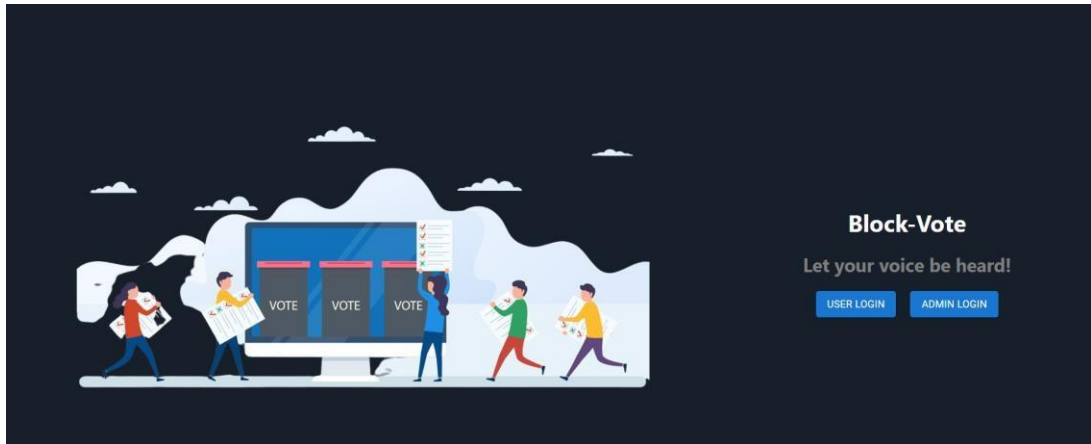


Figure 5.1 Homepage

This is the Homepage of the E-Voting web application where there are User login and Admin Login Features.

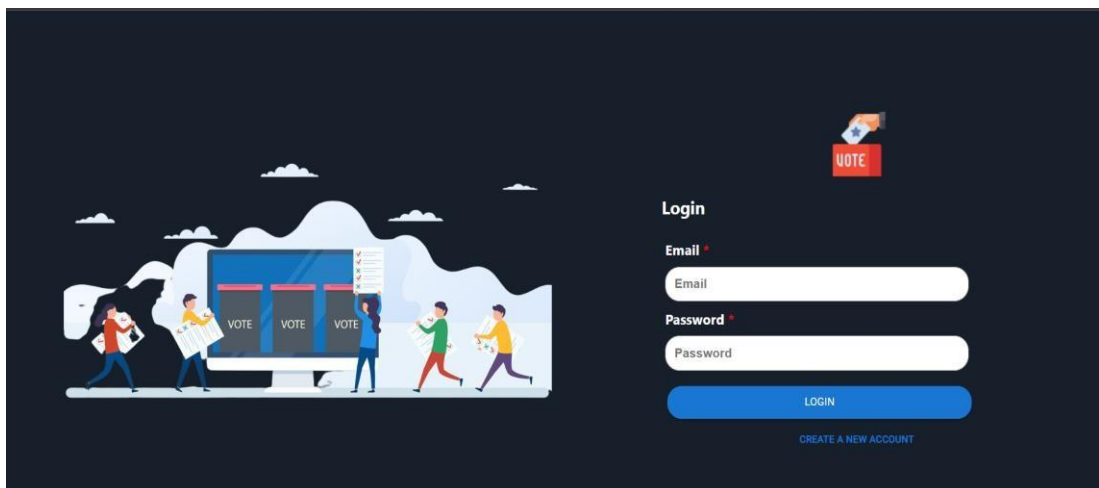
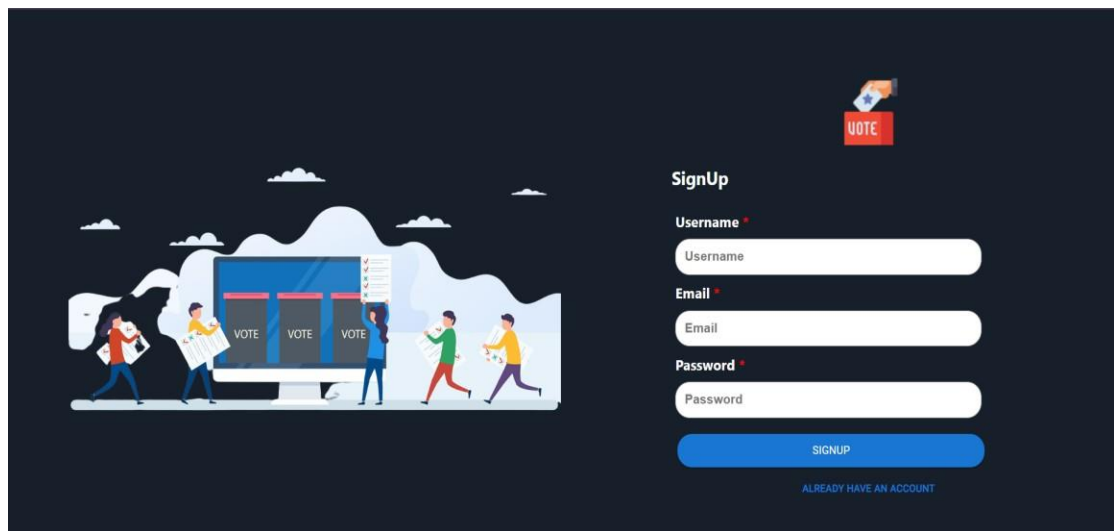


Figure 5.2 User Login

User need to create his account by providing his Email Id and a unique Password.



The image shows a dark-themed web page for a voting system. On the left, there is a colorful illustration of several people standing around a large digital screen that displays the word 'VOTE' multiple times. On the right, there is a 'SignUp' form. At the top right of the form area is a small icon of a ballot box with a star and the word 'VOTE'. The form contains three input fields: 'Username', 'Email', and 'Password', each with a red asterisk indicating a required field. Below these fields is a blue 'SIGNUP' button. At the bottom of the form, there is a link that says 'ALREADY HAVE AN ACCOUNT'.

SignUp

Username *

Email *

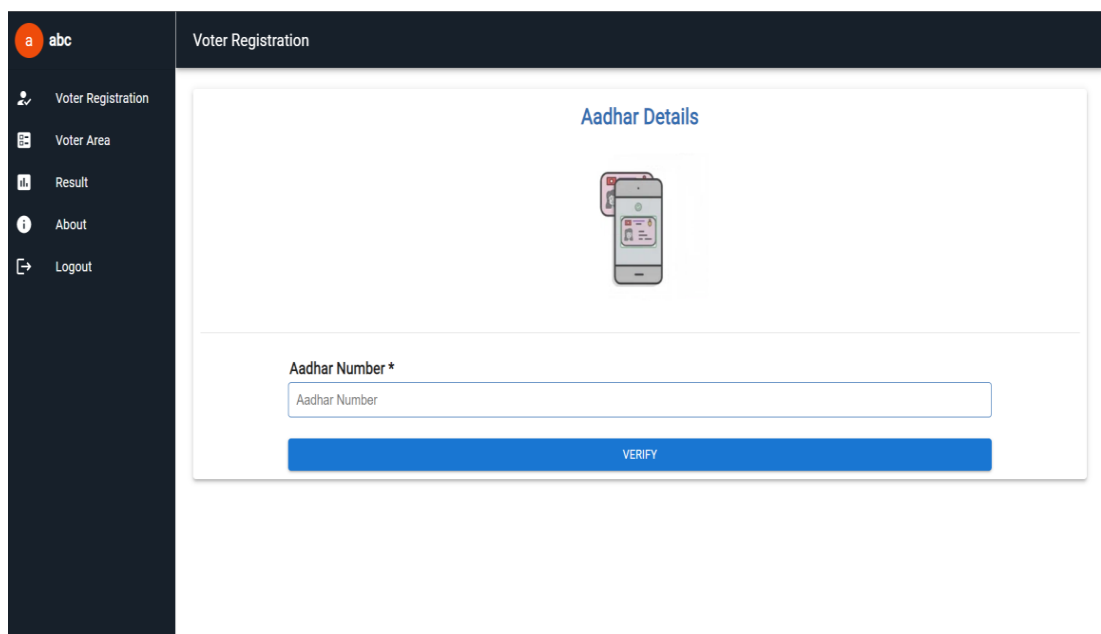
Password *

SIGNUP

[ALREADY HAVE AN ACCOUNT](#)

Figure 5.3 SignUp

If a user is new to the website he/she must create a new account by providing a username, Email Id, Password. If Email Id already exists then the website will not allow user to create the account.



The image shows a user dashboard for a voting system. On the left is a dark sidebar with a user profile icon and the text 'a abc'. Below this are navigation links: 'Voter Registration', 'Voter Area', 'Result', 'About', and 'Logout'. The main content area has a dark header with the text 'Voter Registration'. Below the header, there is a section titled 'Aadhar Details' with an illustration of a smartphone displaying an Aadhar card. Underneath the illustration is a form with a label 'Aadhar Number *' and a text input field containing the placeholder 'Aadhar Number'. At the bottom of the form is a blue 'VERIFY' button.

a abc

Voter Registration

Voter Area

Result

About

Logout

Voter Registration

Aadhar Details

Aadhar Number *

Aadhar Number

VERIFY

Figure 5.4 User Dashboard

Once the user login successfully the user will be redirected to User Dashboard, After which the user need to provide his Aadhaar Details to be eligible for voting.The Admin will check the Age from Aadhaar details and may allow the user to cast the vote if eligible else will be rejected.

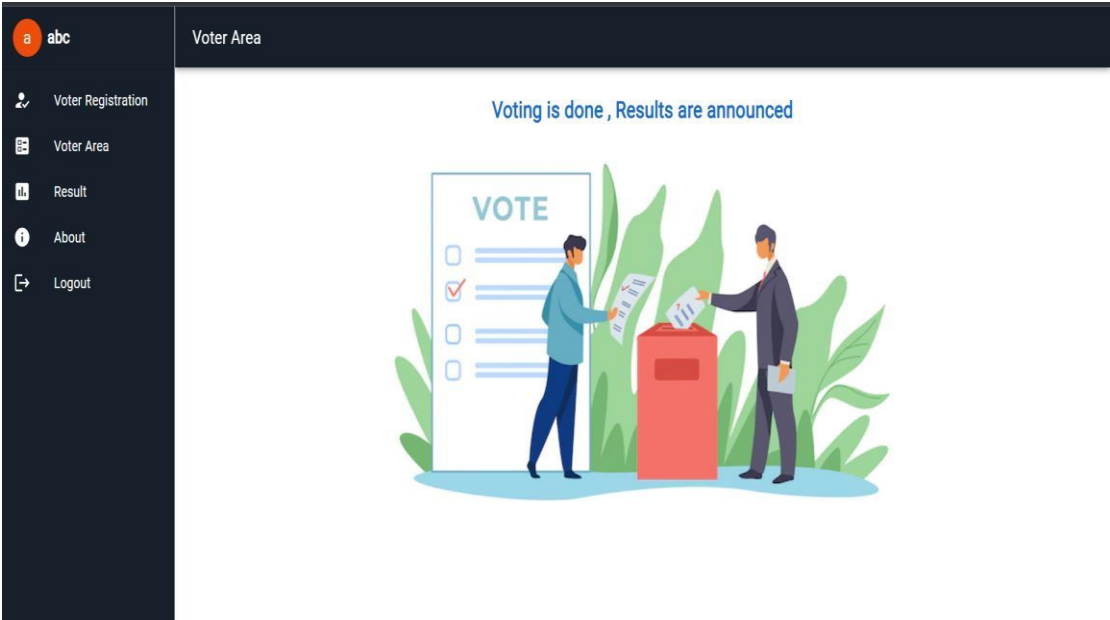


Figure 5.5 Voter Area

This is the area where user will cast his vote once the time has come for Elections. The Candidates who are Participating in elections will will be displayed here.

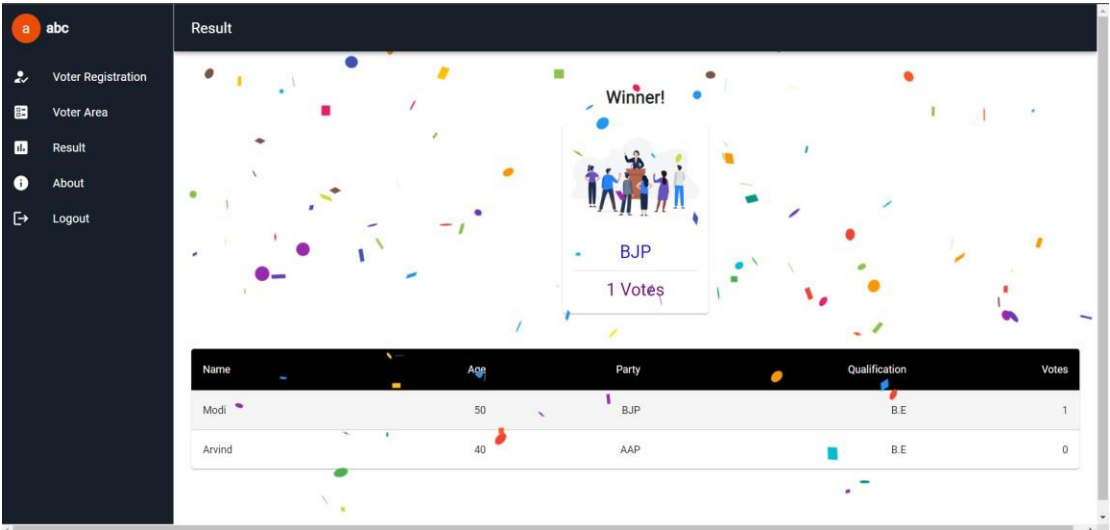


Figure 5.6 Results of Voting

The Results page is where the candidate who won in the elections will be displayed and it also shows each candidate got how many votes.

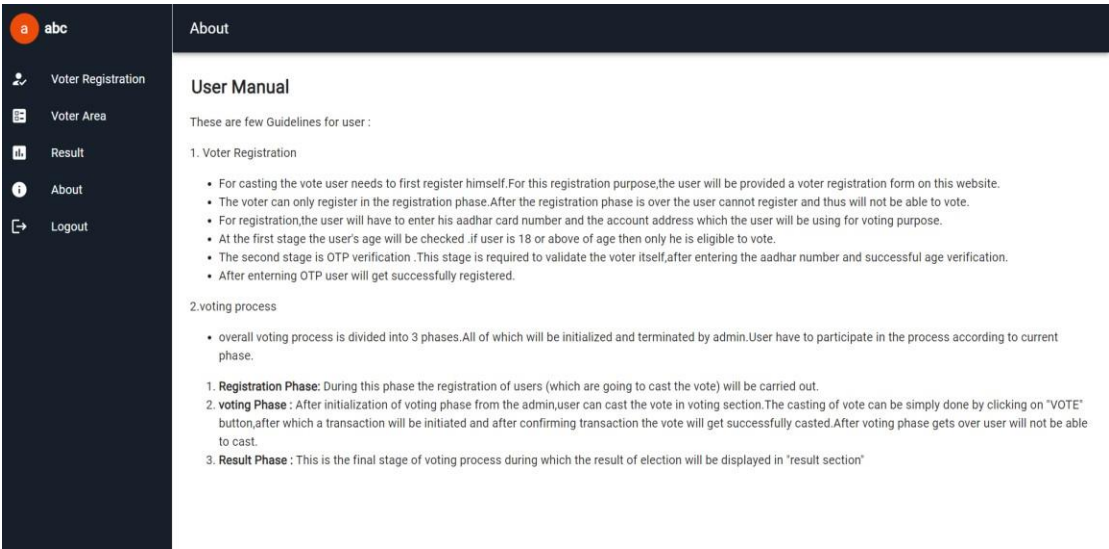


Figure 5.7 About Page

This is the page for user reference where it gives a clear idea for the user how election process carry on.

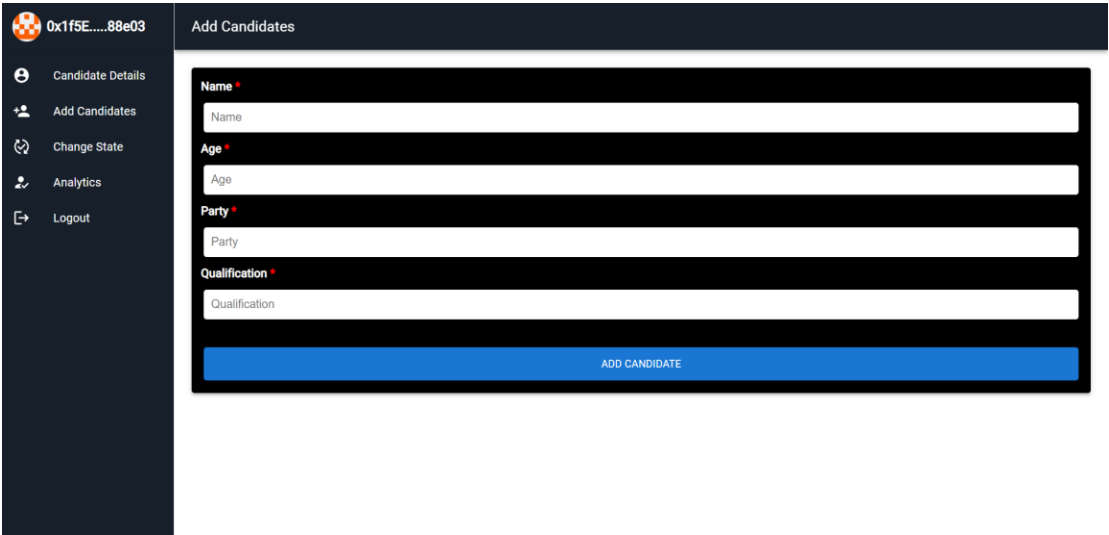
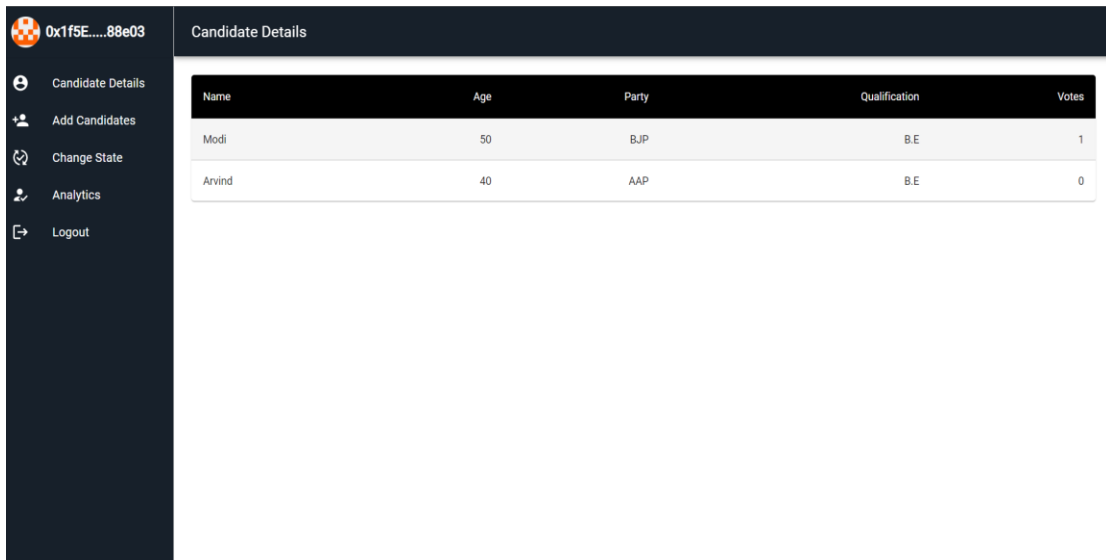


Figure 5.8 Add Candidates

This is the Admin dashboard where the contesting candidate details will be added by the admin.

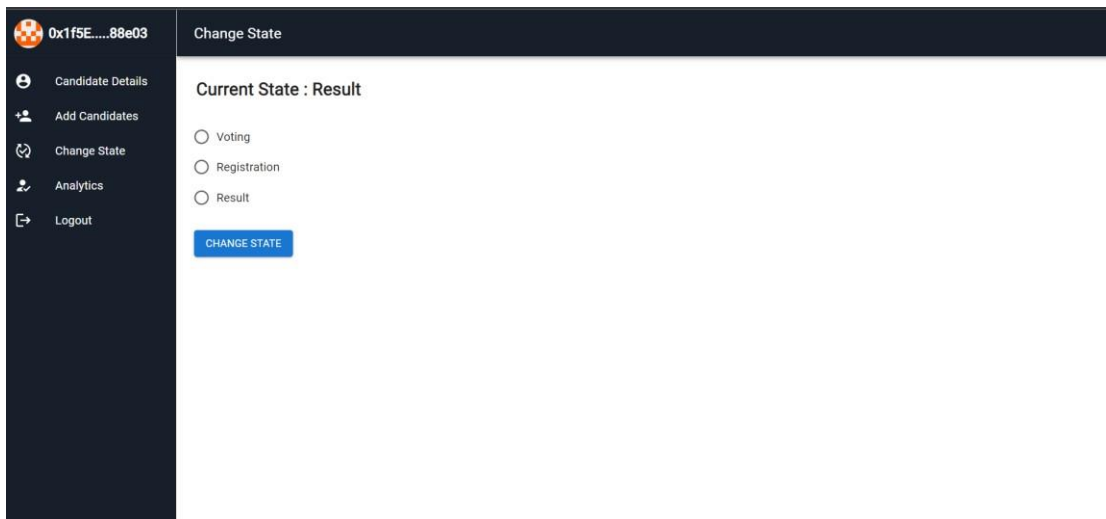


Name	Age	Party	Qualification	Votes
Modi	50	BJP	B.E	1
Arvind	40	AAP	B.E	0

Figure 5.9 Candidate details

Once the admin add the candidate details the data will be shown in the Candidate details page.

The attributes are Name, Age, Party Name, Qualification, No of votes.



Change State

Current State : Result

☐ Voting
☐ Registration
☒ Result

CHANGE STATE

Figure 5.10 Change state

This is the area where admin will change the state of elections to Registration, Voting, Result.

If the state is Registration means the admin is adding the eligible candidates for elections where user cant see the vote button.

If the state is Voting implies that users can cast their vote.A user can cast a vote only once.

If the state is Results implies that the elections are completed and the results are announced where users can see the results.



Figure 5.11 Analytics of Voting

This is the page which describes total no of candidates and total votes casted as of now.

The Graph shows who has got more votes as of now i.e till a particular time.

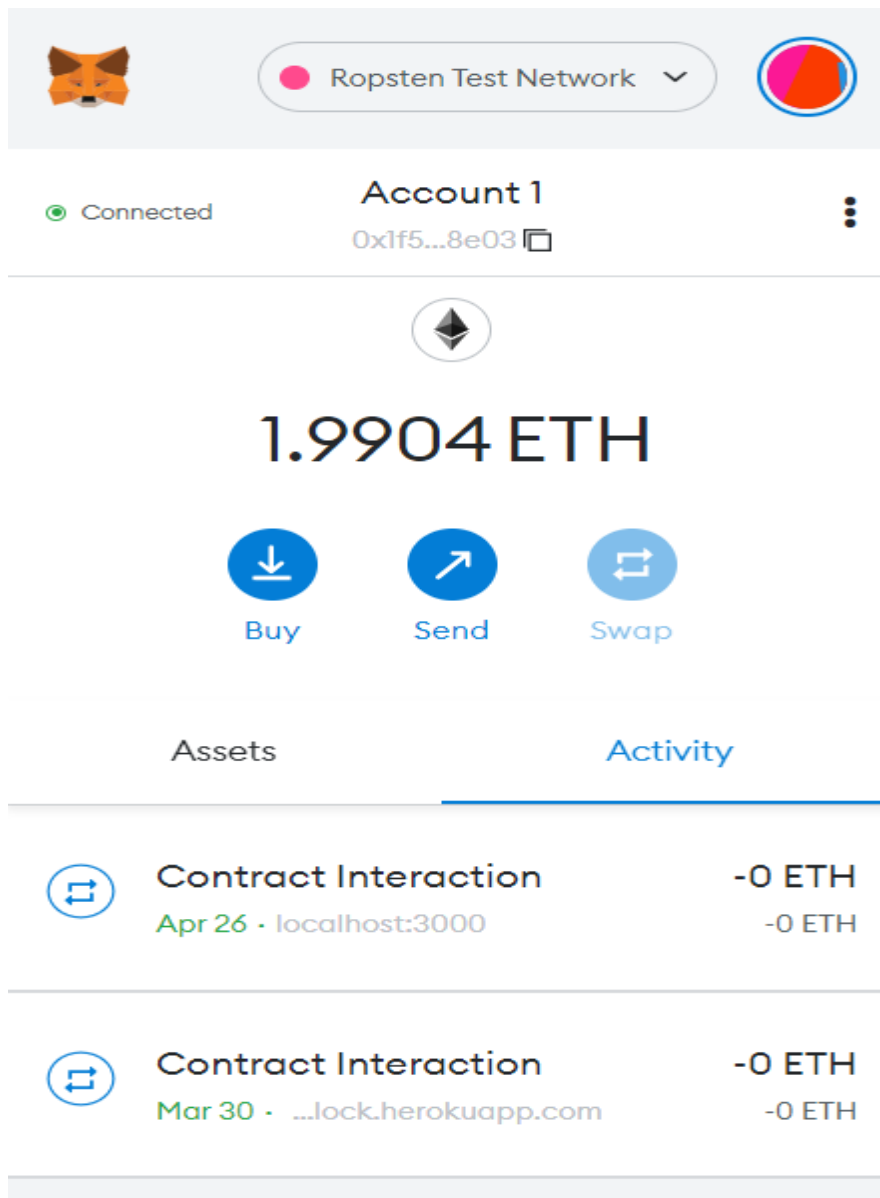


Figure 5.12 Metamask wallet

MetaMask is used to store and manage account keys, streaming, send and receive cryptocurrencies and Ethereum-based tokens, and is securely connected to nationally distributed systems via a compatible web browser or built-in mobile browser.

For casting the vote, changing the phase of elections, adding the candidates we need to spend some ether to perform a transaction.

In today's global market the cost of 1 Ether is around 2.5 lakhs to 3 lakhs. In this application we have got some free ether from different testing providers. The ether has no value that is we can't sell to gain the money.

Instead we can make use of Ganache Truffle suite which provides 10 free accounts with 100 Ether each for one particular project. We can create any number of projects in Ganache Truffle suite.

The screenshot shows a 'Contract Interaction' window with a close button (X) in the top right corner. The window displays the following information:

- Status:** Confirmed (in green). Links for 'View on block explorer' and 'Copy Transaction ID' are provided.
- From:** 0x1f5...8e03 (with a red and blue circular icon).
- To:** 0xaDe...46DE (with a red and blue circular icon).
- Transaction Details:**

Field	Value
Nonce	7
Amount	-0 ETH
Gas Limit (Units)	165058
Gas Used (Units)	110039
Base Fee (GWEI)	43.3903239
Priority Fee (GWEI)	2.5
Total Gas Fee	0.00505 ETH
Max Fee Per Gas	0.000000072 ETH
Total	0.00504973 ETH

Figure 5.13 Contract Information

When a transaction is performed we can get to know all the transaction information but we can't track the transaction made by a user.

The transaction has the following information:

Nonce

Amount:

The total amount used in performing a transaction.

Gas Limit:

Gas Limit is the maximum amount a cryptocurrency user is willing to pay when sending a job, or doing smart contract work, to the Ethereum blockchain.

Gas used:

It is the fee or the pricing value that is used to perform the transaction.

Base Fee, Priority fee, Total gas fee, Max fee for gas and total Ethereum spent on the transaction.

The screenshot shows the Etherscan website interface for a transaction on the Ropsten Testnet. The page title is "Transaction Details" with tabs for "Overview", "Logs (1)", and "State". A warning message states: "[This is a Ropsten Testnet transaction only]". The transaction details are as follows:

Field	Value
Transaction Hash:	0x3671eae617ab0f9da6adb9672259ca7dc602d0fc936a5b228becf9a3d3666e45
Status:	Success
Block:	12219645 (65128 Block Confirmations)
Timestamp:	24 days 19 hrs ago (Apr-26-2022 02:33:43 PM +UTC)
From:	0x1f5ec3c22a6b11e8a576734cd22bcae28808e03
To:	Contract 0xadecf1fb073d85f7062d0efa004d6e49271a46de
Value:	0 Ether (\$0.00)
Transaction Fee:	0.0050497253516321 Ether (\$0.00)
Gas Price:	

A cookie notice at the bottom states: "This website uses cookies to improve your experience and has an updated Privacy Policy." with a "Got it" button.

Figure 5.14 Etherscan

We can also view the transaction details on the etherscan website which contains all the information about the transaction.

Gas Price:	0.0000000458903239 Ether (45.8903239 Gwei)
Gas Limit & Usage by Txn:	165,058 110,039 (66.67%)
Gas Fees:	Base: 43.3903239 Gwei Max: 71.965580332 Gwei Max Priority: 2.5 Gwei
Burnt & Txn Savings Fees:	<div>Burnt: 0.0047746278516321 Ether (\$0.00)</div> <div>Txn Savings: 0.002869295142520848 Ether (\$0.00)</div>

Figure 5.15 Gas Fees

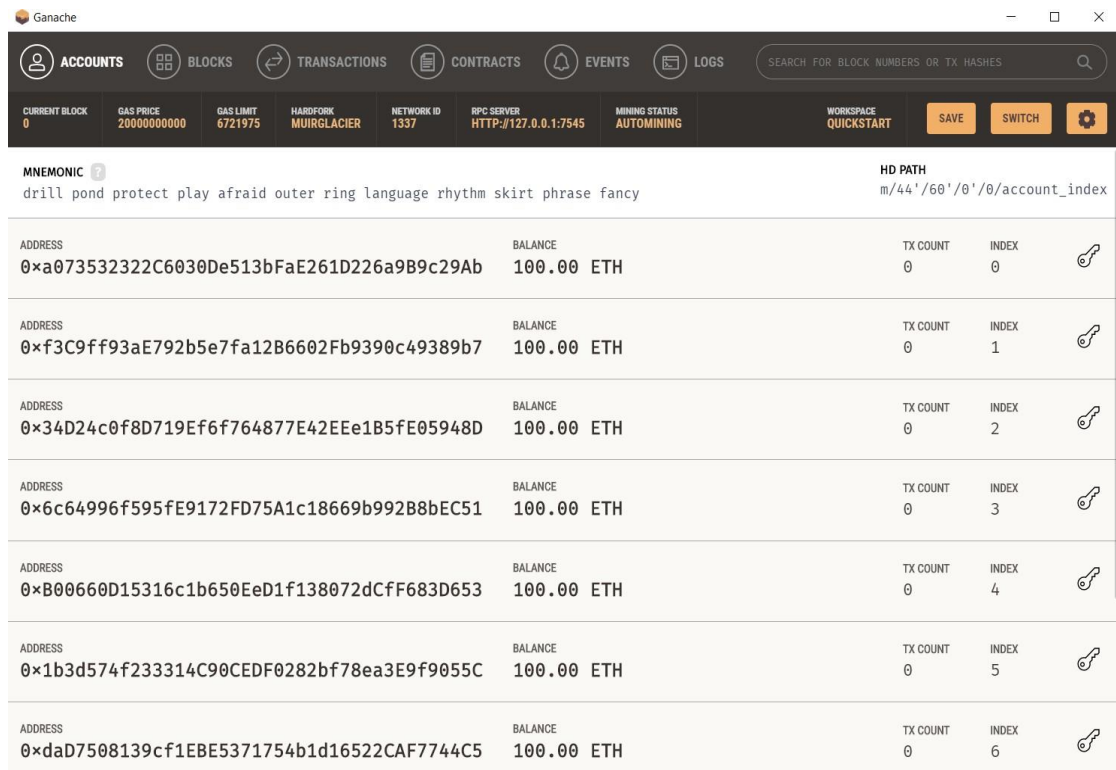


Figure 5.16 Ganache Suite

Ganache is an instant blockchain for Ethereum and Corda 's distributed application development. We can use Ganache for the development, which enables us to develop, implement, and test our dApps in a secure and secure environment.[14]

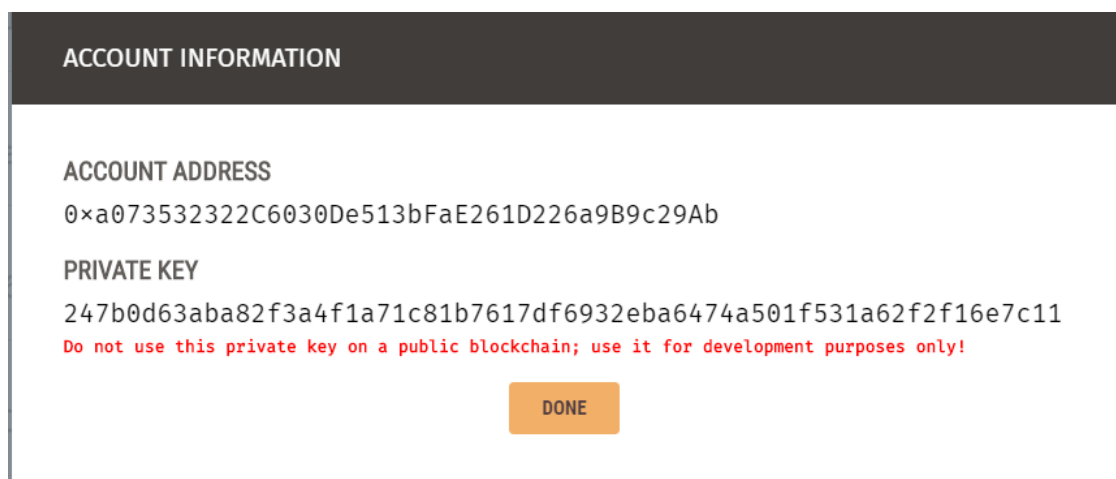


Figure 5.17 Account Information for each key

Ganache Provides 10 free accounts which we can use in our application, the account address is 42 characters length which is in hexadecimal format.

The private key is a unique key for each and every key. We need to use this in order to perform a transaction.



Figure 5.18 website on google search

We have successfully deployed the application on Netlify, where users can access the application by living any place. This only works if they have Metamask extension on their browser and have some Eth coins with them.

<https://e-voting-blockchain.netlify.app/>

TRANSACTION

Nonce	How many time the sender has sent a transaction
To	Address of account the money is going to
Value	Amount of Ether to send to the target address
Gasprice	Amount of Ether the sender is willing to pay per unit gas to get this transaction processed
Startgas/gaslimit	Units of ga that this transaction can consume
V	Cryptographic pieces of datathat can be used to generatethe senders account address. Generated from the users private key.
T	
S	

Table 5.1 What a Transaction contain

6. CONCLUSIONS

6.1 Conclusions

Active E-Voting platform created using HTML / CSS / JavaScript, React.js, Material UI, Ganache, Truffle Integrated Environment, Infura and Metamask. Users were able to register for the application and the administrator was able to add candidates to the election. The administrator was able to control the entire election process. The main objective of the project and achieved is that the user can only vote once. In each case the work will be done using the Metamask wallet and the details of the transaction will be displayed on the ether scan, but no one can track the user who did the work because it is completely protected by cryptographic hashing. User votes are cryptographically protected. Once the votes have been saved they have not changed and there is evidence of disruption. The system maintains voter privacy and anonymity. The program also promotes transparency. The system allows for quick results.

Voting results are publicly auditable.

6.1.1 Limitations

- There are no form validators present to validate form fields.
- Error handling is very limited.
- The Transaction speed is somewhat slower.
- We currently do not have Aadhaar Api and everyone who accesses a website and provides an Aadhaar number after logging in can vote irrespective of the age.
- Every one can't buy Ethereum because it is costly.
- A better user Interface need to be designed.

6.2 Recommendations/Future Work/Future Scope

The application should have a proper Domain name instead of using Hosting systems like Heroku and Netlify. The application can have translation feature using Google Api. We can also add Face Recognition feature to have more security when ever user login to the application. The Aadhaar Card validation is not added in the current application as the Api is not open source and will not be given for un-authorized users.

We can add more Stylings to improve the website and can make the system robust.

We need to work on or look on how to make the transaction speed faster.

REFERENCES

- [1] G. Rathee, R. Iqbal, O. Waqar and A. K. Bashir, "On the Design and Implementation of a Blockchain Enabled E-Voting Application Within IoT- Oriented Smart Cities," in *IEEE Access*, vol. 9, pp. 34165-34176, 2021, doi: 10.1109/ACCESS.2021.3061411.
- [2] L. V. Thuy, K. Cao-Minh, C. Dang-Le-Bao and T. A. Nguyen, "Votereum: An Ethereum-Based E-Voting System," 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF), 2019, pp. 1-6, doi: 10.1109/RIVF.2019.8713661.
- [3] F. P. Hjalmarsson, G. K. Hreiðarsson, M. Hamdaqa and G. Hjalmtýsson, "Blockchain-Based E-Voting System," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), 2018, pp. 983-986, doi:10.1109/CLOUD.2018.00151.
- [4] S. K. Vivek, R. S. Yashank, Y. Prashanth, N. Yashas and M. Namratha, "E- Voting Systems using Blockchain: An Exploratory Literature Survey," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020, pp. 890-895, doi: 10.1109/ICIRCA48905.2020.9183185..
- [5] M. Ibrahim, K. Ravindran, H. Lee, O. Farooqui and Q. H. Mahmoud, "ElectionBlock: An Electronic Voting System using Blockchain and Fingerprint Authentication," 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C), 2021, pp. 123-129
- [6] D. Khoury, E. F. Kfoury, A. Kassem and H. Harb, "Decentralized Voting Platform Based on Ethereum Blockchain," 2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET), 2018, pp.1-6, doi: 10.1109/IMCET.2018.8603050.

[7]K. Garg, P. Saraswat, S. Bisht, S. K. Aggarwal, S. K. Kothuri and S. Gupta, "A Comparative Analysis on E-Voting System Using Blockchain," 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), 2019, pp. 1-4, doi: 10.1109/IoT-SIU.2019.8777471.

[8]K. Patidar and S. Jain, "Decentralized E-Voting Portal Using Blockchain," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2019, pp. 1-4, doi: 10.1109/ICCCNT45670.2019.8944820.

[9]Yi, H. Securing e-voting based on blockchain in P2P network. J Wireless Com Network 2019, 137 (2019). <https://doi.org/10.1186/s13638-019-1473-6>

[10]A. Kaudare, M. Hazra, A. Shelar and M. Sabnis, "Implementing Electronic Voting System With Blockchain Technology," 2020 International Conference for Emerging Technology (INCET), 2020, pp. 1-9, doi: 10.1109/INCET49848.2020.9154116.

[11]Geth.ethereum.org. (2018). Go Ethereum. Available at: <https://geth.ethereum.org/>

[12](documentation) - **Solidity v0.8.10**, 2016-2021

Available:<https://docs.soliditylang.org/en/v0.8.10/>

[13](documentation) - **web3.js v1.5.2**, 2016 Available:

<https://web3js.readthedocs.io/en/v1.5.2/>

[14] MDPI and ACS Style

Jafar, U.; Aziz, M.J.A.; Shukur, Z. Blockchain for Electronic Voting System—Review and Open Research Challenges. *Sensors* **2021**, *21*, 5874. <https://doi.org/10.3390/s21175874>

[15] <https://www.geeksforgeeks.org/decentralized-voting-system-using-blockchain/>

APPENDICES

Solidity code

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.22 <0.9.0;
contract Evote{
    uint public candidatesCount=0;
    uint public userCount=0;
    string public changeState="";
    mapping(uint => candidates) public candidatesList;
    mapping(string => email) public emailList;
    mapping(string => aadhar) public aadharList;
    mapping(string => user) public usersList;
    struct user{
        string username;
        string email;
        string password;
        bool isVoted;
        string aadhar;
    }
    struct email{
        string accountAddress;
        string aadhar;
    }
    struct aadhar{
        string accountAddress;
        string email;
    }

    struct candidates{
        uint id;
        string name;
```

```

event aadharCreated(
    string accountAddress,
    string email
);
event candidatesCreated([
    uint id,
    string name,
    string age,
    string party,
    string qualification,
    uint voteCount
]);

function createUser(string memory _username,string memory _email,string memory _password,string memory _aadhar )
{
    userCount++;

    usersList[_email] = user(_username,_email,_password,false,_aadhar);

    emit userCreated(_username,_email,_password,false,_aadhar);
}

function createCandidate(string memory _name,string memory _age,string memory _party,string memory _qualification)
{
    candidatesCount++;

    candidatesList[candidatesCount] = candidates(candidatesCount,_name,_age,_party,_qualification,0);
}

```

```

function createCandidate(string memory _name,string memory _age,string memory _party,string memory _qualification)
{
    candidatesCount++;

    candidatesList[candidatesCount] = candidates(candidatesCount,_name,_age,_party,_qualification,0);

    emit candidatesCreated(candidatesCount,_name,_age,_party,_qualification,0);
}

function addVote(uint id,string memory _email) public {
    candidatesList[id].voteCount++;
    usersList[_email].isVoted=true;
}

function changeStates(string memory _name) public {
    changeState=_name;
}

function appendIsVote(string memory _email) public {
    usersList[_email].isVoted=true;
}

function createAdharEmail(string memory _aadharNum, string memory _accountAddress,string memory _email) public {
    aadharList[_aadharNum] = aadhar(_accountAddress,_email);
}

```

Web3.js

```
import Web3 from "web3/dist/web3.min.js";
import { accountAction, eVoteAction } from "../Api/action";
import Evote from "../build/contracts/Evote.json";
export const loadWeb3 = async () => {
  if (window.ethereum) {
    window.web3 = new Web3(window.ethereum);
    await window.ethereum.enable();
  } else if (window.web3) {
    window.web3 = new Web3(window.web3.currentProvider);
  } else {
    window.alert(
      "Non-Ethereum browser detected. You should consider trying MetaMask!"
    );
  }
};
export const loadBlockchainData = async (dispatch) => {
  //setLoading(true);
  const web3 = window.web3;
  // Load account
  const accounts = await web3.eth.getAccounts();

  dispatch(accountAction(accounts[0]));
  // Network ID
  const networkId = await web3.eth.net.getId();
  // Network data

  if (networkId) {
    const eVote = new web3.eth.Contract(
      Evote.abi,
```

```
      Evote.networks[networkId].address
    );
    dispatch(eVoteAction(eVote));
  } else {
    alert("not deployed");
  }
};
```

Frontend

Step 1: Create React app

Step 2: Install required dependencies

Step 3: Create App.js file and create routes

Step 4: Create a screens folder to store page components

Step 5: Create components folder to store general components

Step 6: Create a user context

Step 7: Initialize Metamask using web3.js

Step 8: Check if Metamask is loading Ganache accounts

Step 9: After building main functionality add styling

Below code in smart in contract adds all users using map function in solidity language

```
function createUser(string memory _username,string memory _email,string memory _password,string memory _aadhar )
{
    userCount++;
    usersList[_email] = user(_username,_email,_password,false,_aadhar);
    emit userCreated(_username,_email,_password,false,_aadhar);
}
```

Authentication

Authentication using normal Databases

- Firebase
- Mongodb
- Mysql

Above public databases there is a high chance of hacking , admin has chance to manipulate database , here user details are at high risk

Authentication using Blockchain

```

const login = async () => {
  setLoading(true);
  if (!email || !password) {
    alert("please fill all details");
    setLoading(false);
    return;
  }

  try {
    const res = await eVote.methods.usersList(email).call();

    if (res.password === password) {
      navigate("/UserHome/Voter-Registration");
      localStorage.setItem("email", email);
    } else {
      alert("wrong user credintinals or please signup");
    }
    setLoading(false);
  } catch (error) {
    setLoading(false);
    alert(error.message);
  }
};

```

Declared function in smart contract

```

function changeStates(string memory _name) public {
  changeState=_name;
}

```

Calling smart contract function from frontend

```

await eVote.methods.changeStates(value).send({ from: account });

```

Modules used to develop website

```
"@emotion/react": "^11.8.2",
"@emotion/styled": "^11.8.1",
"@mui/icons-material": "^5.5.0",
"@mui/material": "^5.5.0",
"@testing-library/jest-dom": "^5.16.2",
"@testing-library/react": "^12.1.4",
"@testing-library/user-event": "^13.5.0",
"@truffle/hdwallet-provider": "^2.0.4",
"firebase": "^9.6.8",
"radium": "^0.26.2",
"react": "^17.0.2",
"react-confetti": "^6.0.1",
"react-dom": "^17.0.2",
"react-loading": "^2.0.3",
"react-redux": "^7.2.6",
"react-router-dom": "^6.2.2",
"react-scripts": "5.0.0",
"recharts": "^2.1.9",
"web-vitals": "^2.1.4",
"web3": "^1.7.1"
```

Frontend Routes

Since we are using React JS it is a single page application here we use routes.


```

import SignUp from "../User/Screens/SignUp";
import AdminLogin from "../Admin/Screens/AdminLogin";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import AdminHome from "../Admin/Components/AdminHome";
import UserHome from "../User/Components/UserHome";
export default function RoutesComponent() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/Login" element={<Login />} />
        <Route path="/SignUp" element={<SignUp />} />
        <Route path="/UserHome/:name" element={<UserHome />} />
        <Route path="/AdminLogin" element={<AdminLogin />} />
        <Route path="/AdminHome/:name" element={<AdminHome />} />
      </Routes>
    </Router>
  );
}

```

Calculating the winner

Smart Contract code

```

function addVote(uint id,string memory _email) public {

    candidatesList[id].voteCount++;
    usersList[_email].isVoted=true;

}

```

Frontend Code

```
const list = [];  
var winner = "";  
var maxCount = 0;  
var winnerPartyName = "";  
for (var i = 1; i <= count; i++) {  
    const res = await eVote.methods.candidatesList(i).call();  
  
    if (res.voteCount > maxCount) {  
        maxCount = res.voteCount;  
        winner = res.name;  
        winnerPartyName = res.party;  
    }  
    list.push(res);  
}
```

ABI- Application Binary Interface

List of ABI contains all the functions of smart contract

```

"abi": [
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": false,
        "internalType": "string",
        "name": "accountAddress",
        "type": "string"
      },
      {
        "indexed": false,
        "internalType": "string",
        "name": "email",
        "type": "string"
      }
    ],
    "name": "aadharCreated",
    "type": "event"
  },
  {
    "anonymous": false,

```

```

"anonymous": false,
"inputs": [
  {
    "indexed": false,
    "internalType": "uint256",
    "name": "id",
    "type": "uint256"
  },
  {
    "indexed": false,
    "internalType": "string",
    "name": "name",
    "type": "string"
  },
  {
    "indexed": false,
    "internalType": "string",
    "name": "age",
    "type": "string"
  },
  {
    "indexed": false

```

Finally smart contract is deployed to the ropsten network which is provided by Infura

```
ropsten: {
  provider: () =>
    new HDWalletProvider(
      "trouble collect first hunt expire taxi board truth reward vital almost rally",
      `https://ropsten.infura.io/v3/14ce10a5a4d84ca683d2ac857e74a3f5`
    ),
  network_id: 3, // Ropsten's id
  gas: 5500000, // Ropsten has a lower block limit than mainnet
  confirmations: 2, // # of confs to wait between deployments. (default: 0)
  timeoutBlocks: 200, // # of blocks before a deployment times out (minimum/default: 50)
  skipDryRun: true, // Skip dry run before migrations? (default: false for public nets )
},
```