ELCTRONIC VOTING SYSTEM-BLOCKCHAIN

1. INTRODUCTION

In an era marked by rapid technological advancements, the traditional methods of casting and counting votes in elections have begun to show their limitations. Paper ballots and manual tabulation processes have long been the backbone of the democratic process, but they are not immune to issues of security, transparency, and accessibility. The need for a more robust, secure, and efficient voting system has led to the exploration of innovative solutions, and one such solution is the "Electronic Voting Machine by Blockchain."

The concept of integrating blockchain technology with electronic voting machines is a promising step towards modernizing and fortifying the democratic electoral process. Blockchain, originally designed to underpin cryptocurrencies like Bitcoin, has proven itself as a groundbreaking technology for its ability to provide a decentralized, immutable, and transparent ledger of transactions. By harnessing the attributes of blockchain, electronic voting machines can potentially address many of the challenges faced by traditional voting systems, and in this introduction, we will delve into the fundamental principles and advantages of this transformative approach.

1.1 Project Overview:

The primary objective is to create a voting system that ensures the accurate recording of votes while offering complete transparency in the process. The "Electronic Voting System by Blockchain" project is a groundbreaking initiative designed to enhance the security, transparency, and efficiency of the voting process by harnessing the power of blockchain technology. Traditional voting systems have long been plagued by concerns related to fraud, tampering, and lack of transparency. This project aims to overcome these challenges by developing an electronic voting system that leverages blockchain's inherent qualities to provide a highly secure and trustworthy platform for conducting elections.

We aim to design an electronic voting system that is accessible to a diverse range of voters, including those with disabilities, through intuitive and user-friendly interfaces. Building a system that is resilient to fraud, manipulation, and cyberattacks is crucial to restore public trust in the electoral process. The project adopts a decentralized approach by utilizing blockchain technology, making it difficult for any single point of failure to disrupt the system's integrity.

1.2 Purpose:

The purpose of an electronic voting system is to modernize and optimize the electoral process by harnessing technology. It aims to improve the democratic voting experience in several ways. Firstly, it enhances accessibility, making it easier for citizens to participate in the democratic process, including those with disabilities or those living abroad. Secondly, electronic voting systems streamline the voting process, reducing long lines and wait times at polling stations, which can discourage voter participation. Moreover, these systems increase transparency by providing real-time results and verifiable voting records, helping build trust in the electoral process. Security is also a primary focus, with robust measures in place to safeguard against fraud, tampering, and hacking. By minimizing human errors in vote counting, electronic voting systems contribute to more accurate election results.

Quick results are another benefit, as the tallying of votes can be expedited, providing prompt election outcomes. Additionally, they have the potential to reduce the costs associated with conducting traditional elections, which often involve printed materials and extensive manpower. Overall, electronic voting systems aim to encourage voter participation, provide audit trails, and ensure the integrity of elections through technological advancements.

2. LITERATURE SURVEY

2.1 Existing Problem:

Electronic voting systems, particularly those utilizing blockchain technology, have gained attention due to their potential to address various issues associated with traditional voting methods. However, they also come with their own set of challenges and concerns. Here are some existing problems related to electronic voting systems using blockchain

2.1.1 Security Concerns: Ensuring the security of electronic voting systems is a primary concern. While blockchain offers immutability and transparency, it's not immune to attacks. Issues like 51% attacks, vulnerabilities in smart contracts, and the risk of compromised private keys must be addressed.

2.1.2 Privacy and Anonymity: Maintaining voter privacy is crucial in any voting system. Blockchain provides transparency, which can be at odds with voter anonymity. Striking the right balance between these two aspects is a challenge.

2.1.3 Accessibility and Inclusivity: Not everyone has access to the technology required to participate in electronic voting, potentially excluding certain demographics. Ensuring inclusivity is a problem that needs to be addressed.

2.1.4 Scalability: As the number of voters and transactions increases, scalability becomes a major issue. Blockchain networks like Ethereum have experienced congestion during high-demand events, affecting the speed and cost of transactions.

2.1.5 User-Friendliness: Making electronic voting systems user-friendly is essential to encourage adoption. Many voters, especially older individuals, may struggle with new technology.

2.2 References for Electronic Voting System using Blockchain:

1. "A Survey on the Use of Blockchain in the Voting System" by Elisa Bernardini, et al. - This survey provides an overview of the challenges and opportunities of using blockchain in voting systems.

2. "Towards Secure and Verifiable Electronic Voting using Blockchain Technology" by Pieter Hartel and Srinivas Padmanabhuni - This paper discusses the use of blockchain for secure and verifiable electronic voting.

3. "Securing Mobile E-Voting Systems Using Blockchain Technology" by Noureddine Lasla and Houda Hachami - This paper explores the use of blockchain to secure mobile electronic voting systems.

4. "Decentralized Voting System Using Ethereum Blockchain" by Ali Dorri, et al. - The paper describes a decentralized voting system built on the Ethereum blockchain.

5. "Design of a Secure E-Voting System Using Blockchain" by Emmanuel O. Akanbi and Akinyokun Oluwadare - This paper focuses on the design and security aspects of an electronic voting system using blockchain.

2.3 Problem Statement Definition:

An electronic voting system utilizing blockchain technology is a cutting-edge approach to modernizing and securing the electoral process. This innovative system combines the efficiency and accessibility of electronic voting with the transparency, integrity, and immutability of blockchain, offering a potential solution to long-standing challenges in traditional voting systems. Traditional voting systems often face issues such as voter fraud, ballot manipulation, and mistrust in election results. The electronic voting system by blockchain seeks to address these problems by introducing a tamper-resistant and decentralized ledger technology.

In this system, each vote cast is recorded as a transaction on a blockchain, which is a distributed and immutable digital ledger. The blockchain stores a record of every vote, ensuring that once a vote is cast, it cannot be altered or deleted, thereby safeguarding the integrity of the electoral process. Furthermore, blockchain technology allows for the creation of a transparent and publicly accessible database of votes. This transparency fosters trust among voters, as they can independently verify the accuracy and legitimacy of election results.

In addition to enhancing security and transparency, an electronic voting system by blockchain offers convenience to voters. Eligible voters can cast their ballots electronically from the comfort of their homes or at designated polling stations using secure devices. This reduces the need for physical infrastructure and paper ballots,

potentially lowering the cost of elections and minimizing the environmental impact. The system also tackles the issue of voter identification and authentication. Through cryptographic techniques, voters can be securely identified while maintaining their anonymity, preventing fraudulent votes and ensuring that each voter's identity remains confidential.

While the electronic voting system by blockchain offers numerous advantages, it is not without challenges. Scalability, privacy concerns, and accessibility for all citizens, particularly those without access to technology, must be carefully considered in its implementation.

The problem statement for an electronic voting system by blockchain can be defined as follows: "The current electoral systems worldwide face issues of insecurity, lack of transparency, and inefficiency. To address these problems, an electronic voting system built on blockchain technology is proposed, which aims to provide a secure, transparent, and convenient method for voters to cast their ballots while maintaining the integrity of the electoral process. This innovative approach seeks to revolutionize the way we conduct elections, ensuring the accuracy of results and the trust of the voting populace, while also reducing the environmental footprint of traditional paper- based systems. However, careful consideration of challenges related to scalability, privacy, and accessibility is essential in the development and implementation of this system."

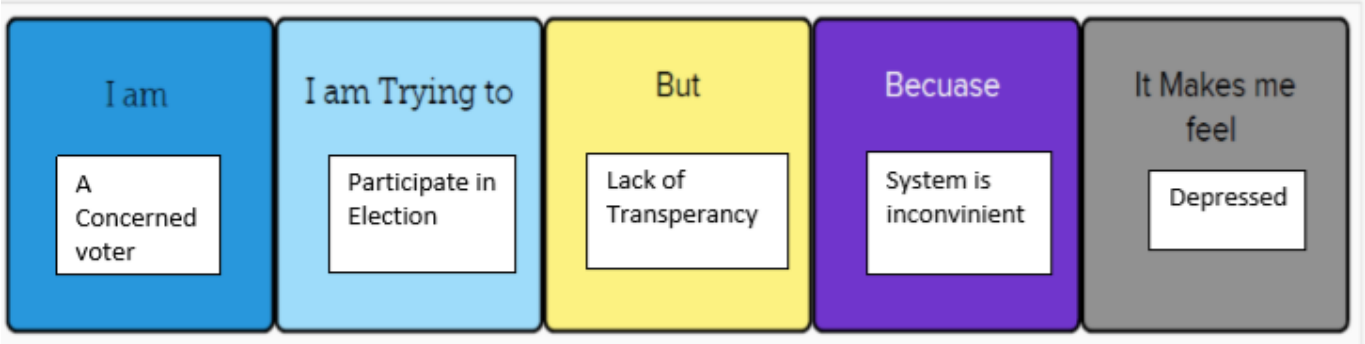


Fig 2.1.customer problem statement

3. IDEATION AND PROPOSED SOLUTION

3.1.Empathy map canvas:

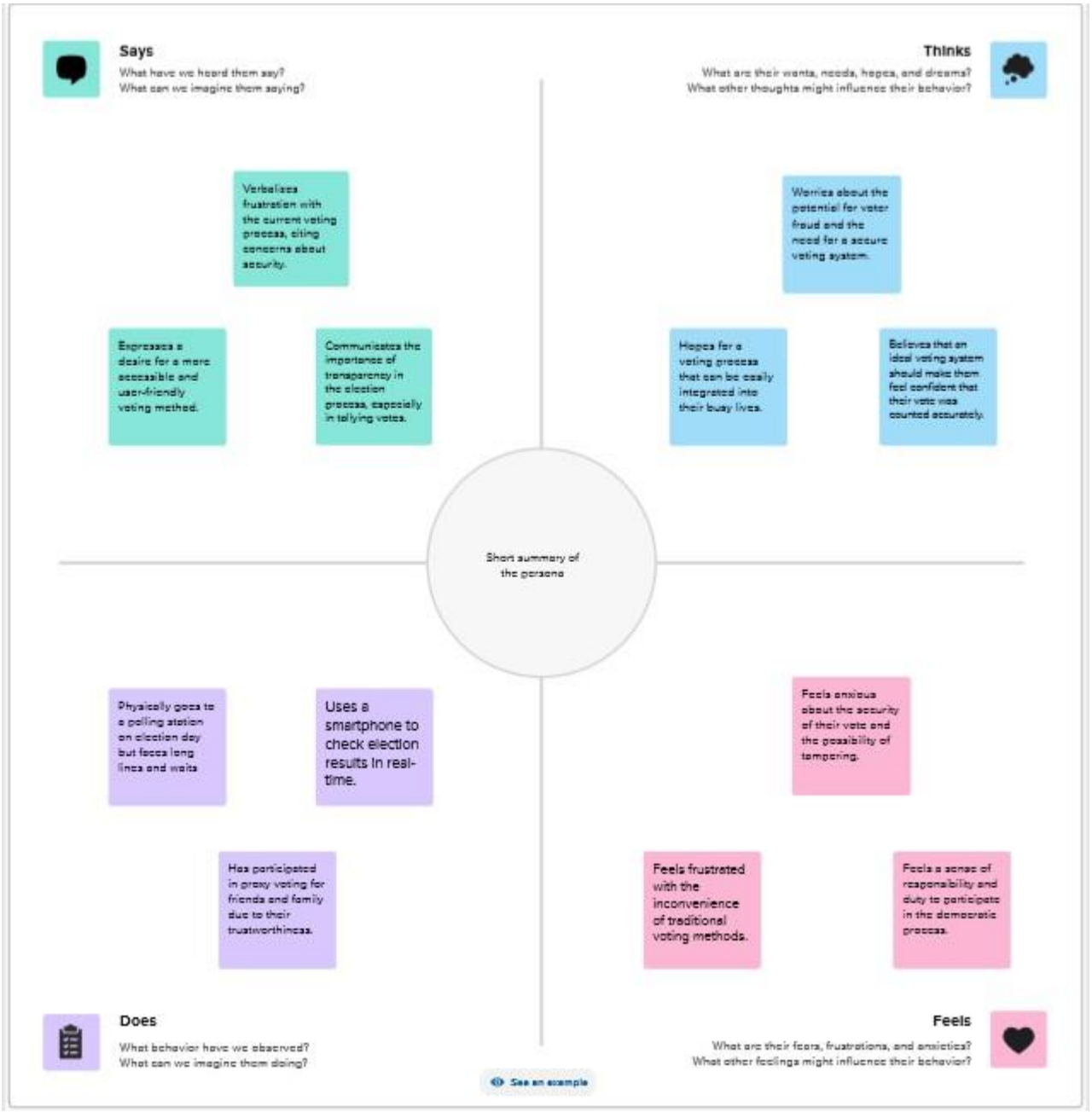
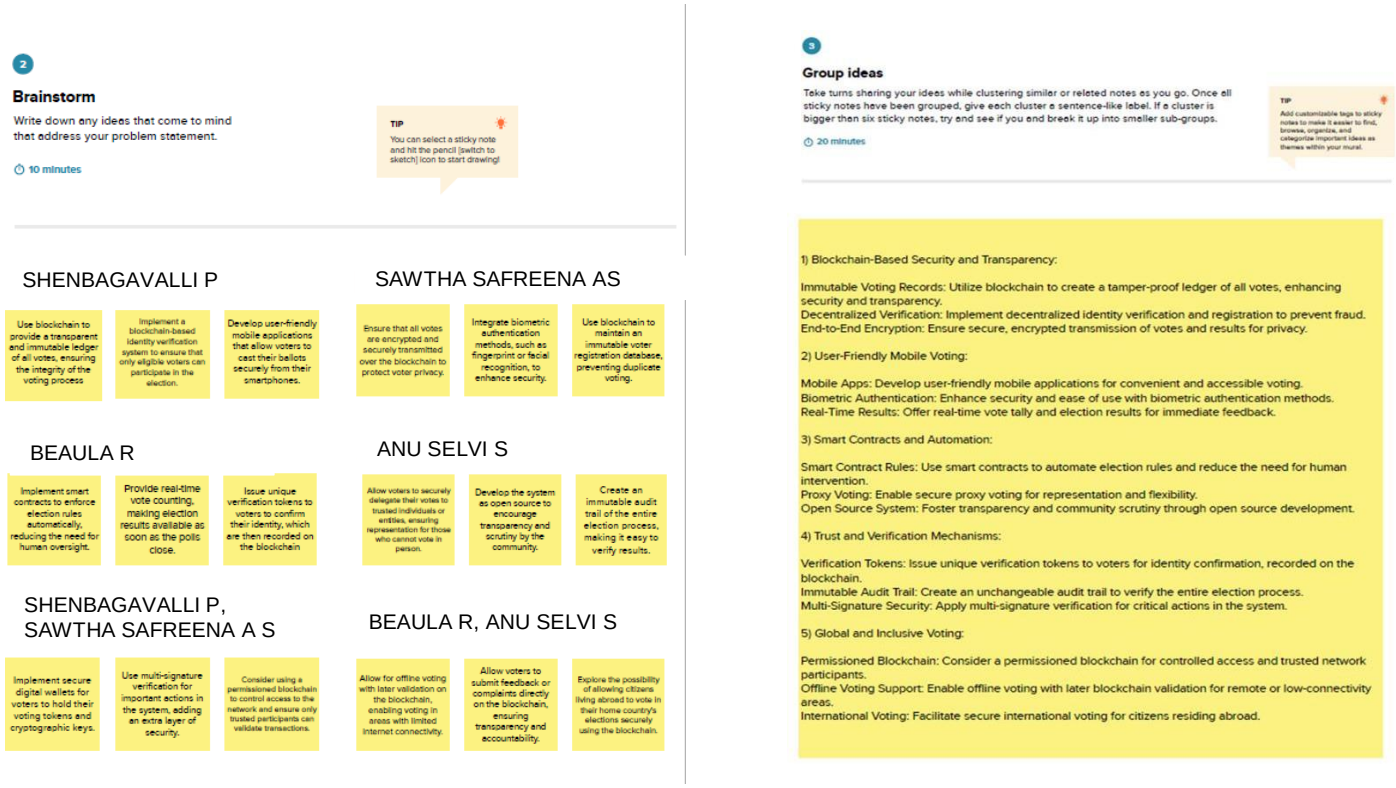


Fig 3.1 Empathy map canvas

3.2. Ideation & Brainstorming :



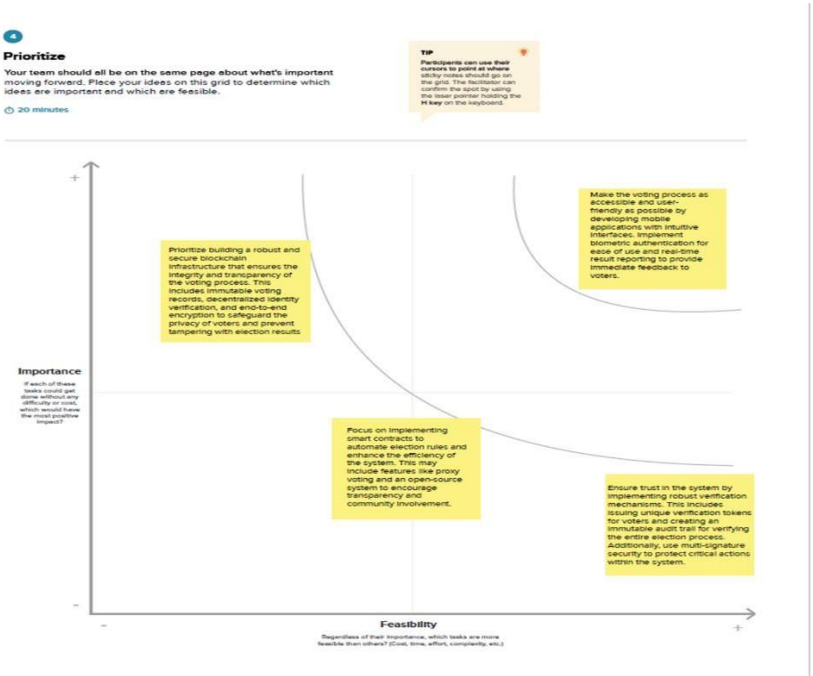


Fig .3.2 Brainstorming and idea prioritization for the project

4. REQUIREMENT ANALYSIS

4.1. Functional requirement:

1. **Voter Authentication:** Secure and user-friendly authentication methods, such as biometrics or multi-factor authentication, to verify the identity of the voters.
2. **Ballot Creation:** Ability to create, design, and customize digital ballots for different elections.
3. **Vote Casting:** User-friendly interfaces for voters to cast their votes electronically and ensuring that voters can verify their votes before submission.
4. **Vote Recording:** Recording and time-stamping of votes on the blockchain to ensure transparency and immutability.
5. **Privacy and Anonymity:** Ensuring voter privacy and anonymity by using cryptographic techniques to protect voter identities.
6. **Blockchain Integration:** Integration with a blockchain network for secure and tamper-proof storage of voting data.
7. **Security:** Robust security measures to prevent tampering, fraud, and unauthorized access.
8. **Auditability:** A mechanism for voters and relevant authorities to audit the voting process and verify the results.
9. **Accessibility:** Ensuring that the system is accessible to all eligible voters, including those with disabilities.
10. **Scalability:** The ability to handle a large number of users and transactions, especially during high-traffic election periods.
11. **Results Reporting:** Real-time or near real-time reporting of election results.

12. **Verifiability:** Enabling independent third-party verification of the election results.

13. **Backup and Recovery:** Regular backups and disaster recovery procedures to ensure the system's availability.

14. **Regulatory Compliance:** Compliance with relevant legal and regulatory requirements for elections.

15. **User Support:** Providing help and support for voters who may encounter issues with the system.

16. **User Education:** Educational materials and support to help voters understand how to use the system securely.

17. **Redundancy and Failover:** Building in redundancy and failover mechanisms to ensure system availability even in the event of technical failures.

18. **Immutable Voting History:** Ensuring that once a vote is recorded on the blockchain, it cannot be altered or deleted.

19. **Token Distribution:** A method for distributing voting tokens to eligible voters.

20. **Voter Feedback:** A mechanism for voters to provide feedback or report issues with the system.

4.2. Non-Functional requirements :

Host Name: CSE01

OS Name: Microsoft Windows 10 Education

OS Version: 10.0.19045 N/A Build 19045

OS Manufacturer: Microsoft Corporation

OS Configuration: Standalone Workstation

OS Build Type: Multiprocessor Free

System Manufacturer: Acer

System Model: Veriton M200-H610

System Type: x64-based PC

Processor(s): 1 Processor(s) Installed.

[01] : Intel64 Family 6 Model 151 Stepping 2 GenuineIntel ~2100 Mhz

BIOS Version: American Megatrends International, LLC. 5.27, 23-12-2022

Total Physical Memory: 15,962 MB

Available Physical Memory: 9,884 MB

Virtual Memory: Max Size: 18,394 MB

Virtual Memory: Available: 10,683 MB

Virtual Memory: In Use: 7,711 MB

5. PROJECT DESIGN

5.1 Data flow diagram

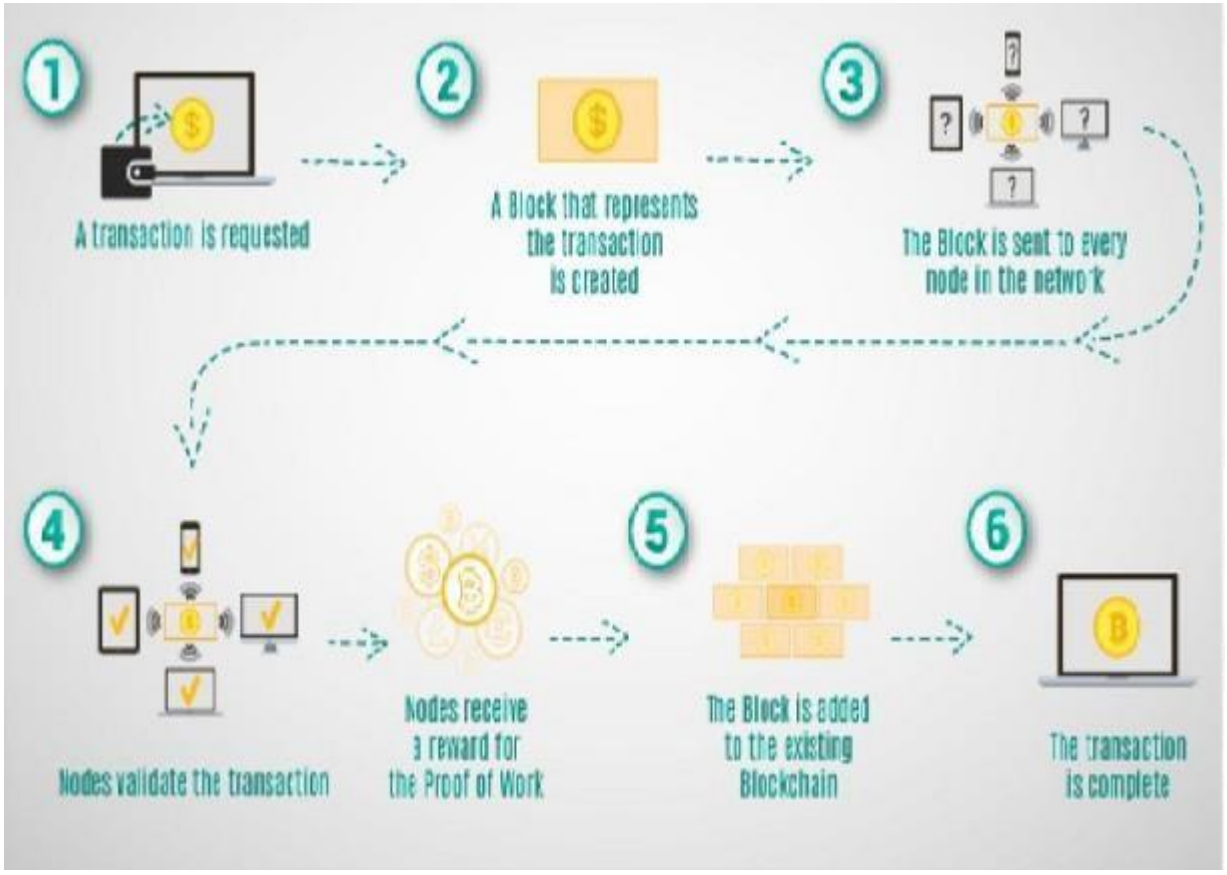


Fig 5.1 Data flow diagram

User Stories:

Use the below template to list all the user stories for the product.

User Time	Functional Requirements (Epic)	User story Number	Use story/ task	Acceptance Criteria	Priority	Team Member
Political Elections	Accuracy	USN-1	As a user, I want the accurate rate of votes which was voted by the people.	I can get an accurate rate of voting and fake votes can be avoided	High	SHENBAG AVALLI P
Corporate Elections	Time efficiency	USN-2	As a user, Chief head of the board members and shareholders can be elected	I can easily elect the head of board.	Moderate	SAWTHA SAFREENA A S
Student Union Elections	Transparency	USN-3	As a user, Student leader can be elected.	I can easily choose the head for our classes.	Low	BEAULA R

Trade Union Elections	Accuracy	USN-4	As a user, Head of Union members can be elected.	I can surely selected Union head.	High	ANU SELVI S
Referendums and Plebiscites	Time efficiency	USN-5	As a user, issues and opinions can be collected through this system.	I can definitely accommodate the issues of the people.	Moderate	SHENBAGAVALLI P, SAWTHA SAFREENA A S
Surveys and Polls	Transparency	USN-6	As a user, I can collect the data through this system		Low	BEAULA R, ANU SELVI S

6. PROJECT PLANNING & SCHEDULING

6.1.Technical Architecture :

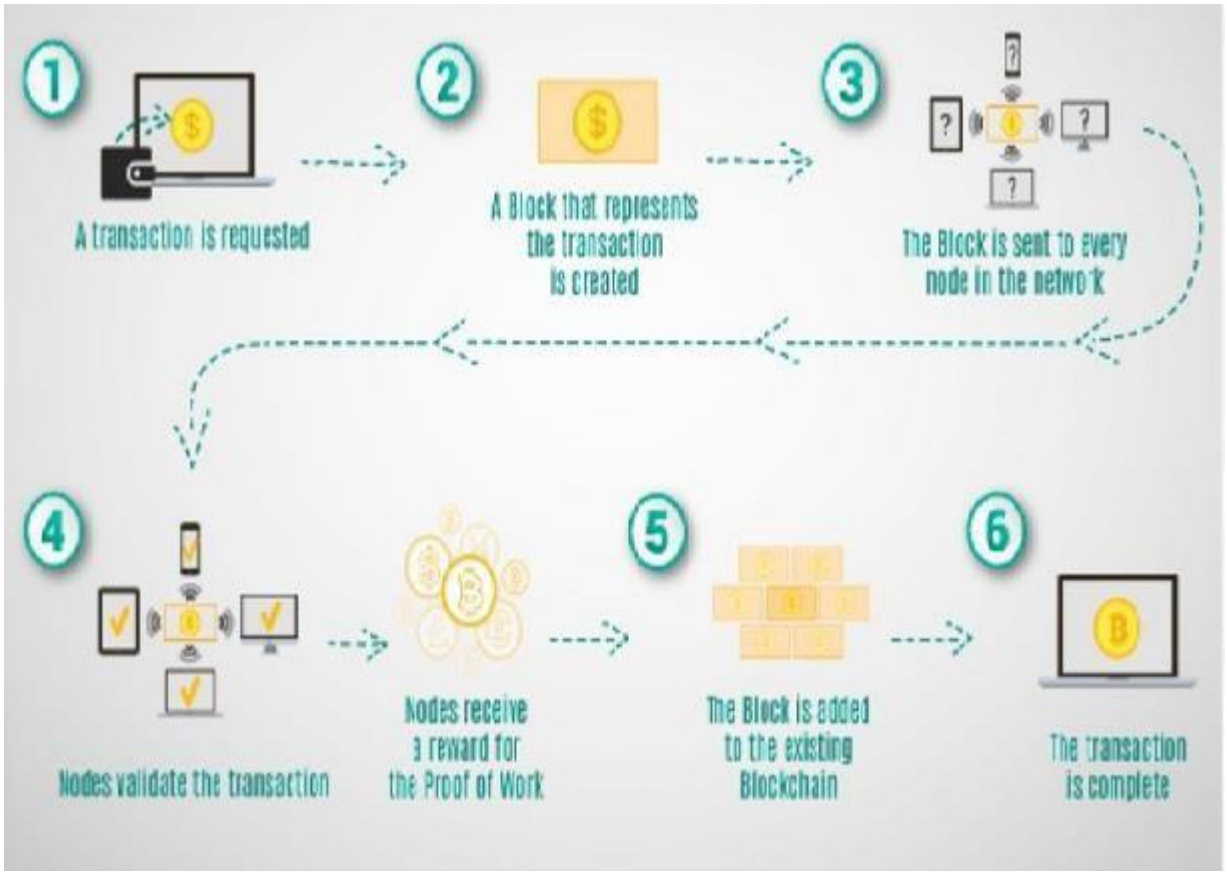


Fig 6.1 simplified technical architecture

6.2 Sprint Planning & Estimation:

1. Product Backlog: Create a detailed product backlog with all the features and tasks required for the project, including blockchain development, security measures, user interface, testing, and more.

2. User Stories: Break down the features into user stories or tasks that can be completed within one sprint.

3. Prioritize: Prioritize the user stories based on their importance and dependencies.

4. Sprint Length: Decide on the sprint length, typically 2-4 weeks.

5. Estimation: Use story points, ideal days, or another estimation technique to assign effort to each user story.

6. Sprint Planning: Select a reasonable number of user stories to fit within the sprint's capacity (considering the team's velocity).

7. Commitment: The team commits to completing the selected user stories within the sprint.

8. Daily Stand-ups: Conduct daily stand-up meetings to monitor progress and address any issues.

9. Review and Retrospective: At the end of the sprint, review what was accomplished and hold a retrospective to discuss improvements.

10. Repeat: Continue this process for subsequent sprints, adjusting based on feedback and progress.

6.3 Sprint Delivery Schedule:

Project Duration: 6 weeks

Sprint Length: 2 weeks per sprint

Sprint 1 - 2 (Week 1-4): Project Setup and Initial Development

Sprint 1: Project kickoff, requirements gathering, and initial architecture design.

Sprint 2: Start blockchain development (smart contracts, basic blockchain infrastructure).

Sprint 3 - 4 (Week 5-8): Development and Testing

Sprint 3: Continue blockchain development and start front-end development.

Sprint 4: Integrate the front-end with the blockchain, conduct initial testing.

Sprint 5 - 6 (Week 9-12): Testing, Feedback, and Deployment

Sprint 5: Comprehensive testing (security, performance, usability), gather feedback.

Sprint 6: Address feedback, make final adjustments, and prepare for system deployment.

7. CODING & SOLUTIONING

7.1 Feature 1 :

Optimized Front-End Interface

Enhanced User Experience: A streamlined and responsive front-end interface ensures a user-friendly experience, making the voting process effortless.

Improved Accessibility: The optimized front end caters to a wider audience, including those with varying levels of tech-savviness.

Faster Interaction: Swift, intuitive design enables quicker interactions, allowing users to cast their votes efficiently.

Mobile Compatibility: The front end is now fully compatible with mobile devices, enhancing accessibility for voters on the go.

Visual Appeal: A polished and attractive front-end design engages users and encourages participation in the voting system.

Seamless Integration: The optimized front end seamlessly integrates with the blockchain back end for a secure and efficient voting experience.

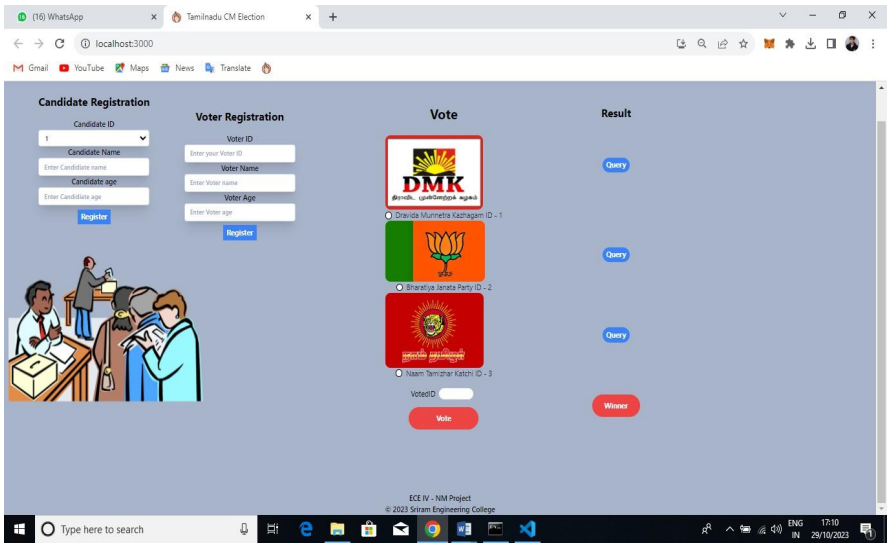


Fig 7.1 Optimized User-Interface

7.2 Feature 2 :

Ganache Testnet Integration

Efficient Testing Environment: Ganache, a local blockchain emulator, provides a fast and efficient testing environment for the electronic voting system.

Realistic Simulations: It replicates real blockchain behaviors, enabling thorough testing of smart contracts and interactions without the cost and latency of a live blockchain.

Enhanced Security Testing: The integration of Ganache allows for extensive security testing, identifying vulnerabilities and weaknesses in a controlled environment.

Resource Cost Savings: By using Ganache, the project minimizes the cost associated with deploying and testing on a live blockchain network.

Agile Development: Developers can rapidly iterate and fine-tune the system, ensuring robustness and reliability before deploying to the production network.

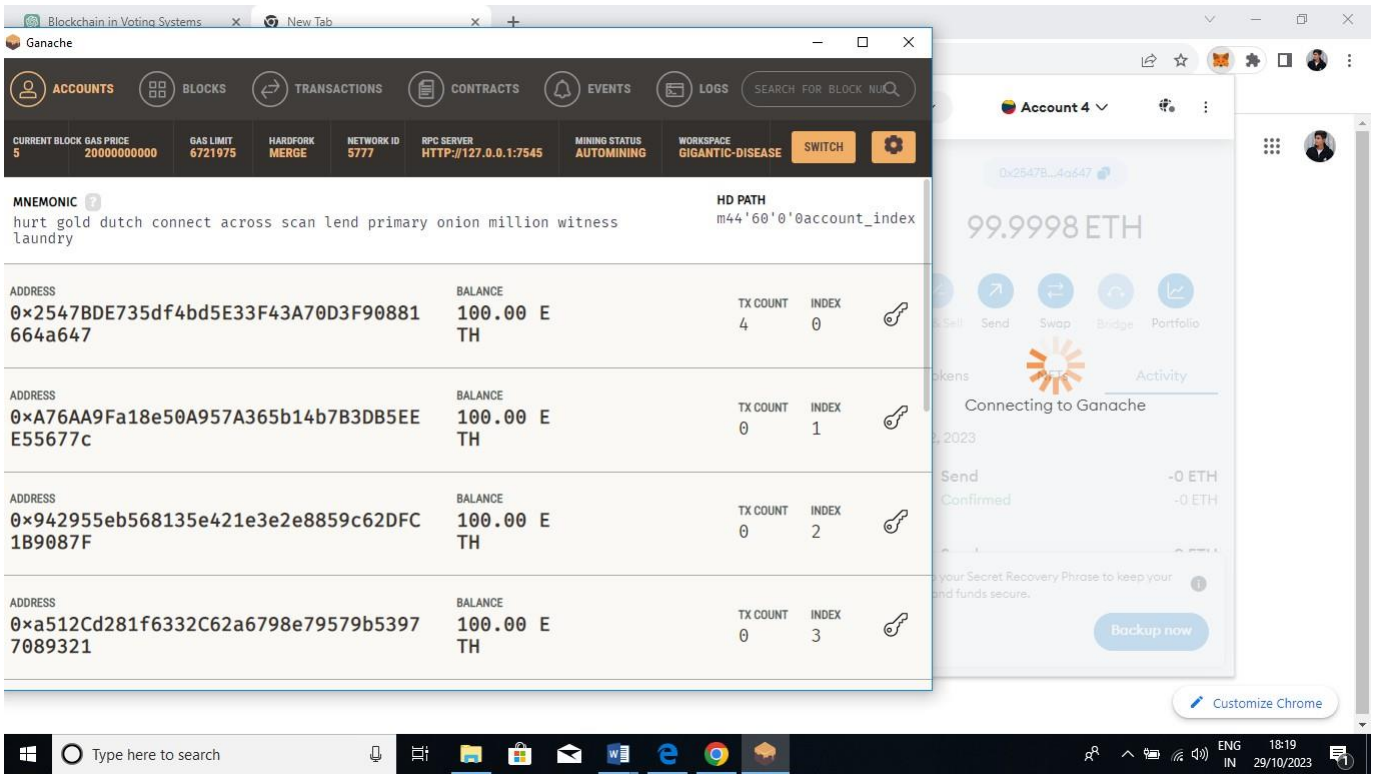


Fig 7.2 Ganache testnet integration

8. PERFORMANCE TESTING

8.1 Performance Metrics:

A performance matrix for an electronic voting system based on blockchain technology encompasses various critical aspects that ensure its functionality, security, and reliability.

1. **Security:** It is paramount, encompassing tamper resistance to prevent unauthorized alterations, strong encryption to safeguard voter identity and ballot data, robust identity authentication to verify voters securely, the use of a permissioned blockchain to control access, and smart contracts for defining and enforcing voting rules.
2. **Scalability:** It involves assessing the system's ability to handle a significant number of transactions efficiently, including evaluating transaction throughput and the scalability of the underlying blockchain network.
3. **Accessibility and Usability:** They are vital for ensuring that all eligible voters can participate. This includes assessing the user-friendliness of the system, incorporating accessibility features to accommodate individuals with disabilities, and ensuring mobile compatibility for broader access.
4. **Transparency and Auditability:** They are essential for instilling trust. A public ledger makes all transactions transparent and open source code allows for public scrutiny, while verifiability ensures that voters can independently confirm the accuracy of their votes.
5. **Anonymity:** It must be preserved to protect voter identities and ensure a secret ballot where individual votes cannot be linked back to voters.
6. **Resilience:** It is critical, covering disaster recovery procedures, redundant system nodes to ensure reliability, and safeguards against Distributed Denial of Service (DDoS) attacks.
7. **Cost Efficiency:** It involves evaluating both the cost per transaction and the overall infrastructure expenses to ensure an economically viable system.

8. **Compliance:** It encompassing adherence to local and national election laws, as well as data protection and privacy regulations.
9. **Voting Process:** It includes assessing the ease and security of voter registration, the process of casting votes, and the verification of votes.
10. **Audit Trail:** It records all changes and transactions, with precise timestamps for accountability.
11. **Blockchain Consensus**: Proof of Work or Proof of Stake, are chosen based on performance and security, and the time it takes for a vote to be considered final (finality) is also evaluated.
12. **Election Management:** It aspects include candidate management and the procedures for publishing election results.
13. **End-to-End Testing:** Their protocols including security testing like penetration testing and vulnerability assessments, ensure system reliability.
14. **Support and Maintenance:** It involve providing user support through a help desk and maintaining the system with regular updates and security patches.
15. **Blockchain Network Health:** It assesses the availability of nodes and the stability of the blockchain's consensus mechanism.
16. **Voter Turnout**: It measures how many voters use the electronic system compared to traditional voting methods.
17. **Feedback and Improvement:** It entail gathering user feedback for system enhancements and ensuring continuous updates to address vulnerabilities. In summary, this comprehensive performance matrix covers critical aspects of an electronic voting system based on blockchain, ensuring its security, efficiency, accessibility, and compliance while maintaining transparency and usability.

9. RESULTS

Output :

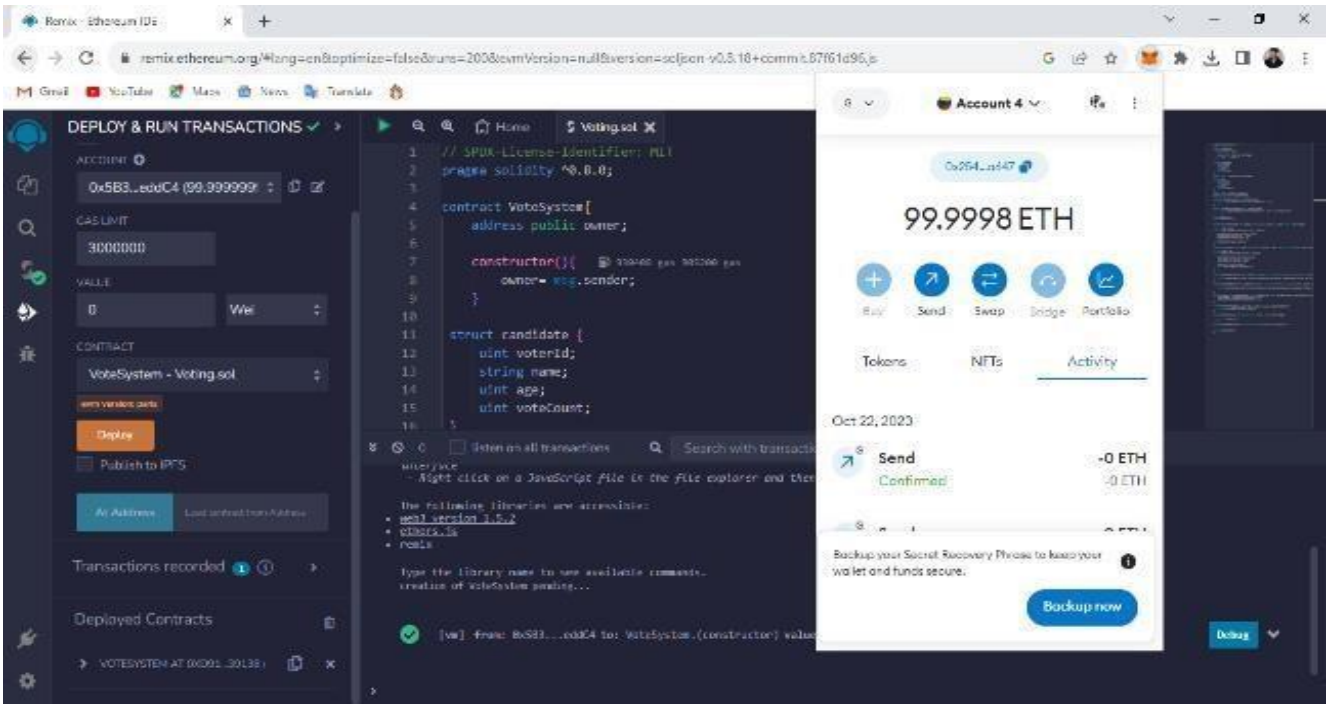


Fig 9.1 Deploying Smart Contract & Ganache Test Net Balance

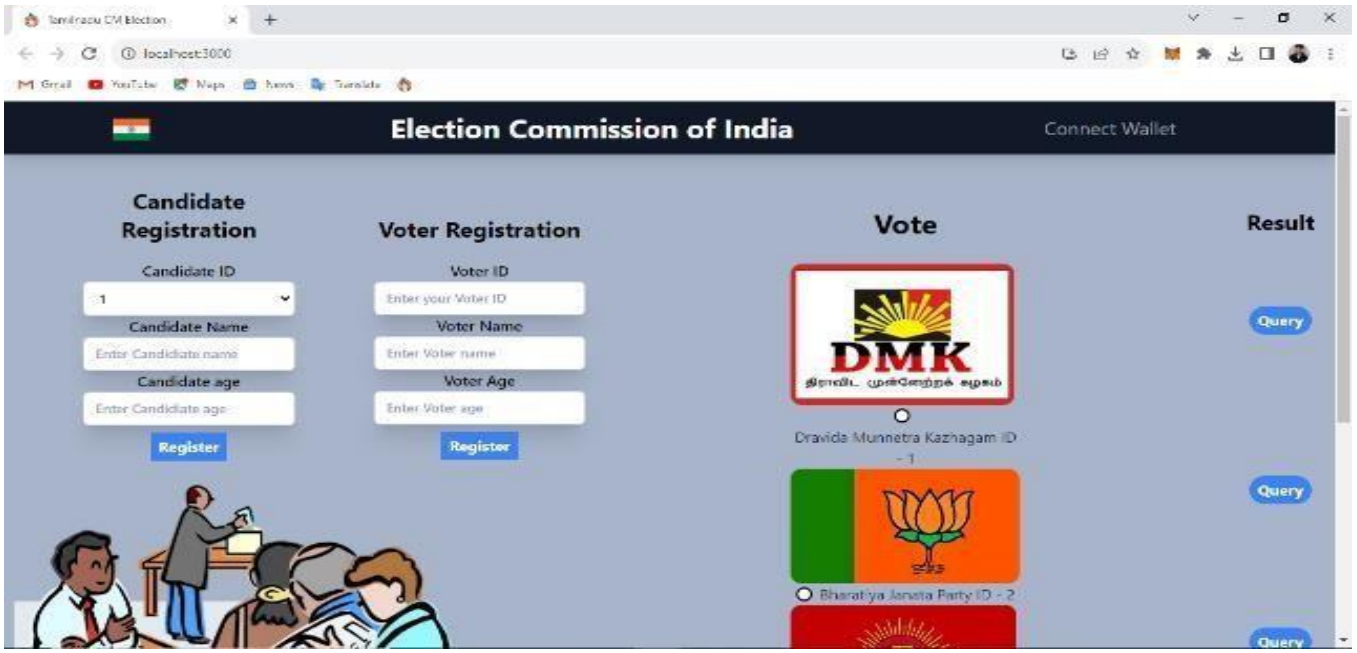


Fig 9.2 User-Interface of Decentralized Voting Web App

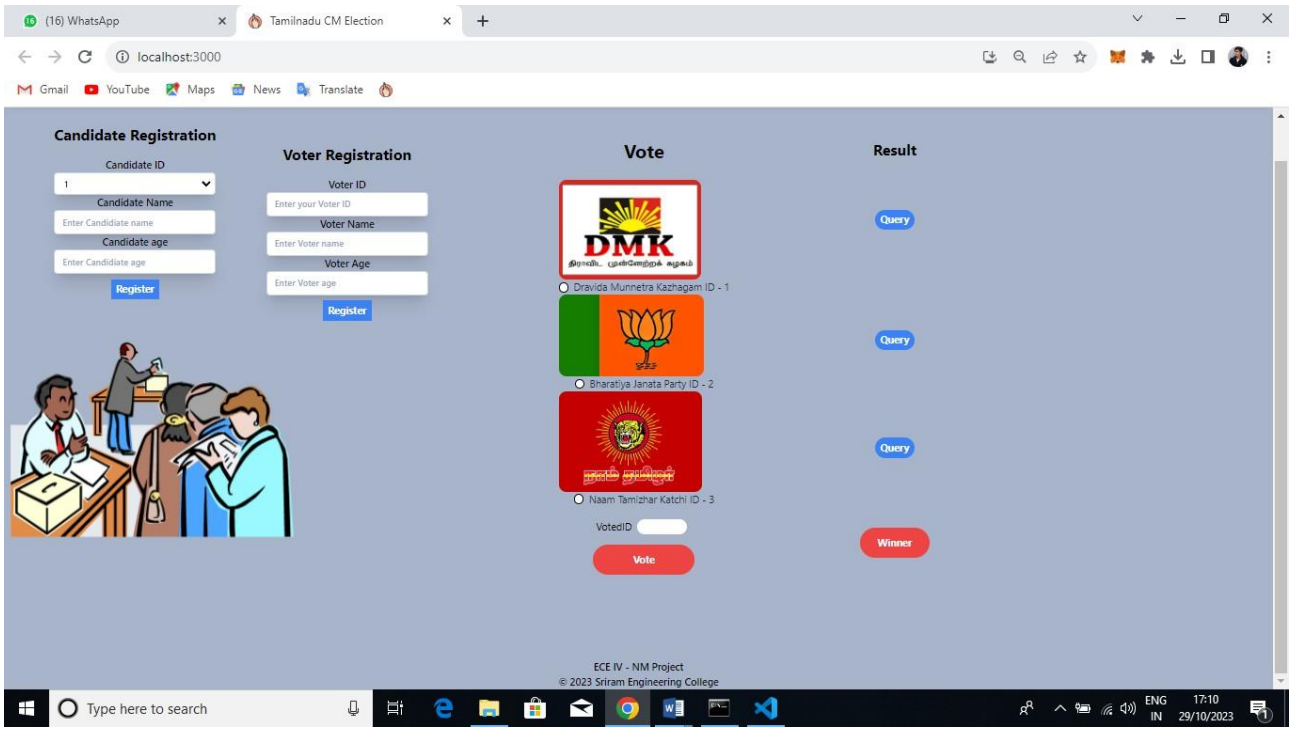


Fig 9.3 User-Interface of Decentralized Voting Web App

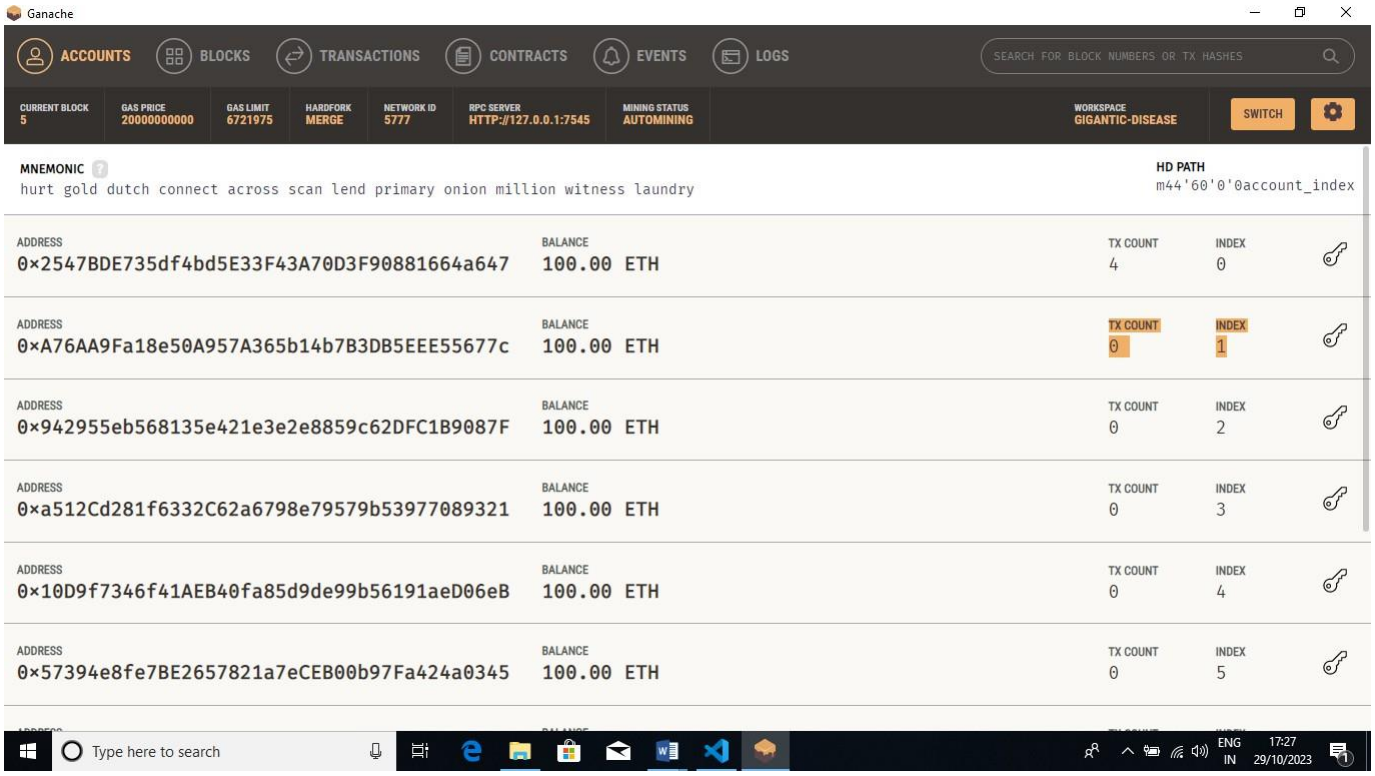


Fig 9.4 Ganache test network

10. ADVANTAGES & DISADVANTAGES

Advantages:

An electronic voting system based on blockchain technology offers several advantages:

1. Security: Blockchain's decentralized and immutable nature makes it highly resistant to tampering and fraud. Votes are recorded in a transparent and secure manner, reducing the risk of hacking or manipulation.

2. Transparency: The entire voting process is transparent and auditable, as anyone can verify the blockchain's public ledger. This enhances trust in the electoral system.

3. Accessibility: Electronic voting can be more accessible, allowing voters to cast their ballots remotely, which can increase voter participation, especially among those with physical limitations or living in remote areas.

4. Reduced Costs: Traditional paper-based voting systems can be expensive to administer, while blockchain-based systems can potentially reduce costs associated with printing, counting, and transporting paper ballots.

5. Real-time Results: Blockchain voting systems can provide real-time or near-real-time election results, reducing the time and effort needed for counting and reporting.

6. Enhanced Privacy: Blockchain can offer a balance between transparency and voter privacy. Votes can be kept anonymous, and voters can verify their choices without revealing their identities.

7. Voter Identity Verification: Blockchain can facilitate secure identity verification, reducing the risk of voter impersonation and ensuring that each vote is cast by an eligible voter.

8. Immutable Records: Once a vote is recorded on the blockchain, it cannot be altered or deleted, ensuring the integrity of the election results.

9. Decentralization: Blockchain operates on a distributed network, reducing the risk of centralized control or manipulation by a single entity.

10. Trust in Elections: By providing a secure and transparent method of conducting elections, blockchain-based voting systems can help restore trust in the electoral process.

Disadvantages:

While electronic voting systems based on blockchain offer several advantages, they also come with some disadvantages and challenges:

1. Technology Complexity: Implementing and maintaining a blockchain-based voting system can be technically complex and require specialized expertise. This complexity may lead to higher initial costs and ongoing maintenance expenses.

2. Accessibility and Inclusivity: Not all voters may have access to the technology required for electronic voting, potentially excluding some individuals, particularly those who are less tech-savvy, have limited access to the internet, or live in areas with poor connectivity.

3. Identity Verification: Ensuring the secure and accurate verification of voter identities in an electronic system can be challenging. Protecting against identity theft or voter impersonation is crucial.

4. Security Risks: While blockchain technology is considered secure, it is not immune to cyberattacks. Any vulnerabilities in the voting system's software or hardware can be exploited by malicious actors. Ensuring the security of the entire infrastructure is essential.

5. Voter Privacy: Maintaining voter privacy while using blockchain can be a challenge. Striking the right balance between transparency and anonymity is crucial to prevent coercion, vote buying, or undue influence.

6. Initial Setup Costs: The initial investment required to set up a blockchain-based voting system can be significant. This includes costs for hardware, software, security measures, and staff training.

7. Scalability: Ensuring that the system can handle a large number of votes in a timely manner can be difficult. Blockchain networks may experience scalability issues, leading to slower transaction processing times during high-demand periods.

8. Legal and Regulatory Challenges: Adapting existing election laws and regulations to accommodate blockchain-based voting can be a complex process. Legal frameworks need to be established or revised to address the unique aspects of electronic voting.

9. Error Handling: Unlike paper-based systems, where voters can easily correct mistakes, electronic voting systems can be less forgiving if a voter makes an error when casting their ballot. Clear procedures for error correction and dispute resolution are essential.

10. Trust and Acceptance: Trust in new technology can take time to build, and some voters may be hesitant to accept blockchain-based voting due to concerns about its security, reliability, or susceptibility to manipulation.

11. Long-Term Data Storage: Managing and archiving election data on a blockchain over the long term can present challenges. Ensuring that historical voting records remain accessible and secure is important.

These disadvantages and challenges should be carefully considered when developing and implementing blockchain-based voting systems to ensure the integrity and accessibility of the electoral process.

11. CONCLUSION

In conclusion, the concept of an electronic voting system based on blockchain technology presents a promising solution to enhance the security, transparency, and efficiency of the electoral process. The advantages of such a system include increased security, transparency, accessibility, and reduced costs. However, there are also notable challenges and disadvantages, including technological complexity, concerns about accessibility and inclusivity, and the need for robust identity verification and privacy protection.

The successful implementation of a blockchain-based voting system requires a careful balance of these factors, along with a strong legal and regulatory framework. As technology continues to evolve, and as society becomes increasingly digital, blockchain-based voting systems may hold significant potential to revolutionize elections, but they must be approached with caution and a commitment to addressing the associated challenges to ensure a trustworthy and inclusive democratic process.

12. FUTURE SCOPE

The future scope of an electronic voting system based on blockchain technology is vast and holds the potential to revolutionize the way elections are conducted. Here are some key areas where we can expect significant developments:

1. **Enhanced Security**: As blockchain technology evolves, it can offer even more robust security features, making it increasingly resistant to cyberattacks and fraud. This will be critical in safeguarding the integrity of elections.

2. Wider Adoption: More countries and regions may adopt blockchain-based voting systems as they become more mature and trusted. This could lead to a global shift toward electronic voting, further improving election efficiency and transparency.

3. Remote Voting: The ability for voters to cast their ballots remotely from anywhere with internet access may become more prevalent. This could significantly increase voter participation, especially in regions with low voter turnout.

4. Mobile Voting Apps: User-friendly mobile apps could make electronic voting even more accessible, providing a convenient and familiar interface for voters. This could further boost participation among tech-savvy populations.

5. Improved Identity Verification: Innovations in identity verification methods, such as biometrics or zero-knowledge proofs, could enhance the security and privacy of electronic voting systems.

6. Interoperability: Efforts to create interoperable blockchain standards for voting could facilitate cross-border and cross-regional voting, potentially paving the way for international elections or referendums.

7. Auditable Elections: Blockchain's inherent transparency will continue to improve trust in the electoral process, as voters, candidates, and the public can independently verify the results.

8. Integration with Smart Contracts: Smart contracts on blockchains can automate various aspects of the voting process, reducing the need for manual oversight and streamlining administrative tasks.

9. Accessibility Features: Ongoing development can ensure that electronic voting systems are inclusive and accessible to all, regardless of physical abilities, age, or location.

10. Environmental Benefits: Blockchain-based voting can contribute to sustainability by reducing the need for paper ballots and minimizing the carbon footprint associated with traditional elections.

11. Regulatory Frameworks: Governments and international organizations may establish clear legal and regulatory frameworks to govern the use of blockchain in elections, ensuring consistency and security.

12. Research and Innovation: Ongoing research and innovation in blockchain technology, cryptography, and cybersecurity will continue to advance the field of electronic voting, addressing existing challenges and pushing the boundaries of what's possible.

The future scope of electronic voting systems based on blockchain is promising, and while challenges persist, ongoing developments will likely lead to more secure, accessible, and efficient electoral processes in the years to come.

13. APPENDIX

Source Code :

App.js

```
import "./App.css";
import Home from "./pages/Home";
import { Navbar } from "./components/Navbar";
import Footer from "./components/Foooter";
import Anime from "./components/Anime";
function App() {
   return (
      <div className="App">
         <Navbar />
         <Home />
         <Anime />
         <Footer />
      </div>
   );
}

export default App;
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
reportWebVitals();
```

Home.js

```
import React from "react";
import Voting from "../components/Voting";

function Home() {
   return <Voting />;}

export default Home;
```

Constants.js

```
import { ethers } from "ethers";
import abi from "./voting.json";

export const contractAddress = "0xd8b934580fcE35a11B58C6D73aDeE468a2833fa8";
export const provider = new ethers.providers.Web3Provider(window.ethereum);
export const signer = provider.getSigner();

export const votingContract = new ethers.Contract(contractAddress, abi, signer);
```

Anime.js

```
import React, { Component } from 'react'

export default class Anime extends Component {
  render() {
    return (
      <div className='Anime'>
<img src="https://i.ibb.co/ry4qBnB/Vote-Anime.png" alt="Vote-Anime" className='animate-
bounce'/>

      </div>
    )
  }
}
```

Footer.js

```
import React from "react"
export default class Footer extends React.Component{

    render(){
    return(

<div>
  <div className="mt-40 font-semibold">
    <p>ECE IV - NM Project</p>
    <span>&copy; 2023 Sriram Engineering College </span>
  </div>

</div>

    )
    }
}
```

Navbar.js

```
import { useState } from "react";

export const Navbar = () => {
  const [CurrentAccount, setCurrentAccount] = useState("");

  const walletConnect = async () => {
    if (!window.ethereum) {
      return alert("please install metamask");
    }
    const addr = await window.ethereum.request({ method: "eth_requestAccounts" });
    setCurrentAccount(addr[0]);
  };

  return (
    <nav className="bg-gray-900 shadow-lg">
      <div className="container mx-auto">
        <div className="sm:flex justify-around">
        <img src="https://i.ibb.co/GT1fynf/National-flag.png" alt="India" className="h-10 mt-3 -
ml-10"/>
          <a href="/" className="text-white text-3xl font-bold p-3">
            Election Commission of India
          </a>

          <ul className="text-gray-400 sm:self-center text-xl border-t sm:border-none">
            <li className="sm:inline-block">
              {!CurrentAccount ? (
                <button onClick={walletConnect} className="p-3 hover:text-white">
                  Connect Wallet
                </button>
              ) : (
                <p>{CurrentAccount}</p>
              )}
            </li>
          </ul>
        </div>
      </div>
    </nav>
  );
};
```

Voting.js

```
import React, { useState } from "react";
import { votingContract } from "../utils/constants";

function Voting() {
   const [CandidateName, setCandidateName] = useState("");
   const [CandidateAge, setCandidateAge] = useState("");
   const [CandidateID, setCandidateID] = useState("");
   const [VoterID, setVoterID] = useState("");
   const [VoterName, setVoterName] = useState("");
   const [VoterAge, setVoterAge] = useState("");
   const [VoterVoteID, setVoterVoteID] = useState("");
   const [PartyID, setPartyID] = useState("");
   const [VoteCount1, setVoteCount1] = useState("");
   const [VoteCount2, setVoteCount2] = useState("");
   const [VoteCount3, setVoteCount3] = useState("");
   const [HighestCount, setHighestCount] = useState("");

   const handleCandidatename = (e) => {
      setCandidateName(e.target.value);
   };

   const handleCandidateAge = (e) => {
      const value = e.target.value.replace(/\D/g, "");
      setCandidateAge(Number(value));
   };

   const handleCandidateID = async (e) => {
      const value = e.target.value.replace(/\D/g, "");
      setCandidateID(Number(value));
   };

   const handleCandidateRegistration = async (e) => {
      e.preventDefault();
      const enrollCanddidateTx = await votingContract.enrollCandidate(CandidateID,
CandidateName, CandidateAge);
      await  enrollCanddidateTx.wait();
      console.log(enrollCanddidateTx);
      alert(enrollCanddidateTx.hash);
   };

   const handleVoterID = async (e) => {
      const value = e.target.value.replace(/\D/g, "");
      setVoterID(Number(value));
   };
   const handleVoterName = (e) => {
      setVoterName(e.target.value);
   };
```

```
const handleVoterAge = async (e) => {
   const value = e.target.value.replace(/\D/g, "");
   setVoterAge(Number(value));
};

const handleVoterRegistration = async (e) => {
   e.preventDefault();
   const enrollVoterTx = await votingContract.enrollVoter(VoterID, VoterName, VoterAge);
   await enrollVoterTx.wait();
   console.log(enrollVoterTx);
   alert(enrollVoterTx.hash);
};

const handlePartyID = async (e) => {
   setPartyID(Number(e.target.value));
};

const handleVoterVoteID = async (e) => {
   const value = e.target.value.replace(/\D/g, "");
   setVoterVoteID(Number(value));
};

const handleVote = async (e) => {
   e.preventDefault();
   const voteTx = await votingContract.vote(PartyID, VoterVoteID);
   await voteTx.wait();
   console.log(voteTx);
   alert(voteTx.hash);
};

const handleQuery1 = async (e) => {
   let vote = Number(e.target.id);
   const voteCountTx = await votingContract.getVotecountOf(vote);
   setVoteCount1(voteCountTx.toString());
};

const handleQuery2 = async (e) => {
   let vote = Number(e.target.id);
   const voteCountTx = await votingContract.getVotecountOf(vote);
   setVoteCount2(voteCountTx.toString());
};
const handleQuery3 = async (e) => {
   let vote = Number(e.target.id);
   const voteCountTx = await votingContract.getVotecountOf(vote);
   setVoteCount3(voteCountTx.toString());
};
const handleResult = async () => {
   let number1 = await votingContract.getVotecountOf(1);
   let number2 = await votingContract.getVotecountOf(2);
   let number3 = await votingContract.getVotecountOf(3);
```

```
    let num1 = number1.toString();
    let num2 = number2.toString();
    let num3 = number3.toString();

    if (num1 > num2 && num1 > num3) {
      setHighestCount("DMK");
    } else if (num2 > num1 && num2 > num3) {
      setHighestCount("BJP");
    } else if (num3 > num1 && num3 > num2) {
      setHighestCount("NTK");
    } else {
      setHighestCount("");
    }
  };
  return (
    <div>
      <div className="flex flex-row-3 space-x-52 mt-5 ml-20">
      <div className="grid grid-cols-2 ">
          <div className="mt-4 mr-10">
            <h3 className="text-2xl font-bold antialiased">Candidate Registration</h3>
            <form onSubmit={handleCandidateRegistration}>
              <div className="form-group ">
                <div className="mt-5"></div>
                <div>
                  <div>
                    <label className="font-semibold">
                      Candidate ID
                      <select className="px-3 py-2 bg-white border shadow-sm border-slate-
300 placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-full
rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Select Candidiate ID"
value={CandidateID} onChange={handleCandidateID}>
                        <option name="DMK">1</option>
                        <option name="BJP">2</option>
                        <option name="NTK">3</option>
                      </select>
                    </label>
                  </div>
                  <div>
                    <label className="font-semibold">
                      Candidate Name
                      <span>
                        <input className="px-3 py-2 bg-white border shadow-sm border-
slate-300 placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-
full rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Enter Candidiate name"
value={CandidateName} onChange={handleCandidatename} />
                      </span>
                    </label>
                  </div>
                  <div>
                    <label className="font-semibold">
```

```
                                Candidate age
                                <span>
                                    <input className="px-3 py-2 bg-white border shadow-sm border-
slate-300 placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-
full rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Enter Candidiate age"
value={CandidateAge} onChange={handleCandidateAge} />
                                </span>
                            </label>
                        </div>
                    </div>
                    <input className="bg-blue-500 hover:bg-blue-900 text-white font-bold py-1 px-
2 mt-2" type="submit" value="Register" />
                </div>
            </form>
        </div>
        <div className="mt-12 ml-10 p-0">
            <h3 className="text-2xl font-bold">Voter    Registration</h3>
            <form onSubmit={handleVoterRegistration}>
            <div className="mt-5"></div>
                <div>
                    <label className="font-semibold">
                        Voter ID
                        <span>
                            <input className="px-3 py-2 bg-white border shadow-sm border-slate-300
placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-full
rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Enter your Voter ID"
value={VoterID} onChange={handleVoterID} />
                        </span>
                    </label>
                </div>
                <div>
                    <label className="font-semibold">
                        Voter Name
                        <span>
                            <input className="px-3 py-2 bg-white border shadow-sm border-slate-300
placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-full
rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Enter Voter name"
value={VoterName} onChange={handleVoterName} />
                        </span>
                    </label>
                </div>
                <div>
                    <label className="font-semibold">
                        Voter Age
                        <span>
                            <input className="px-3 py-2 bg-white border shadow-sm border-slate-300
placeholder-slate-400 focus:outline-none focus:border-sky-500 focus:ring-sky-500 block w-full
rounded-md sm:text-sm focus:ring-1 shadow-xl" placeholder="Enter Voter age" value={VoterAge}
onChange={handleVoterAge} />
                        </span>
```

```
                    </label>
                  </div>

                  <button className="bg-blue-500 hover:bg-blue-900 text-white font-bold py-1 px-2
mt-2">Register</button>
                </form>
              </div>
            </div>


            <div className="left-100">
              <form onSubmit={handleVote}>
                <p className="text-3xl mb-7 font-bold mt-10">Vote</p>
                <div>
                  <div>
                    <img
                      className="max-w-56 max-h-40 rounded-xl"
                      src="https://i.ibb.co/ChqK7nm/dmk-final.jpg"
                      alt="DMK"
                    />
                  </div>
                  <div>
                    <input
                      className="form-check-input appearance-none rounded-full h-4 w-4 border
border-black border-x-2 border-y-2 bg-white checked:bg-blue-600 checked:border-black
focus:outline-none transition duration-200 mt-1 align-top bg-no-repeat bg-center bg-contain  mr-2
cursor-pointer"
                      type="radio"
                      name="flexRadioDefault"
                      value="1"
                      onChange={handlePartyID}
                    />
                    <label className="form-check-label inline-block text-gray-800"
htmlFor="flexRadioDefault1">
                      Dravida Munnetra Kazhagam ID - 1
                    </label>
                  </div>
                </div>
                <div>
                  <div>
                    <img
                      className="max-w-56 max-h-40 rounded-xl"
                      src="https://i.ibb.co/JRkrwd0/bjp.jpg"
                      alt="BJP"

                    />
                  </div>
                  <div>
                    <input
                      className="form-check-input appearance-none rounded-full h-4 w-4 border
```

```
border-black border-x-2 border-y-2 bg-white checked:bg-blue-600 checked:border-black
focus:outline-none transition duration-200 mt-1 align-top bg-no-repeat bg-center bg-contain  mr-2
cursor-pointer"
                        type="radio"
                        name="flexRadioDefault"
                        value="2"
                        onChange={handlePartyID}
                    />

                    <label className="form-check-label inline-block text-gray-800"
htmlFor="flexRadioDefault1">
                        Bharatiya Janata Party ID - 2
                    </label>
                </div>
            </div>
            <div>
                <div>
                    <img
                        className="max-w-56 max-h-40 rounded-xl"
                        src= "https://i.ibb.co/jWQK4g9/NTK.jpg"
                        //imgbb - website
                        alt="NTK"
                    />
                </div>
                <div>
                    <input
                        className="form-check-input appearance-none rounded-full h-4 w-4 border
border-black border-x-2 border-y-2 bg-white checked:bg-blue-600 checked:border-black
focus:outline-none transition duration-200 mt-1 align-top bg-no-repeat bg-center bg-contain  mr-2
cursor-pointer"
                        type="radio"
                        name="flexRadioDefault"
                        value="3"
                        onChange={handlePartyID}
                    />
                    <label className="form-check-label inline-block text-gray-800"
htmlFor="flexRadioDefault1">
                        Naam Tamizhar Katchi ID - 3
                    </label>
                </div>
            </div>
            <div className="mt-5">
                <label>
                    VotedID
                    <span className="ml-2 mr-2 ">
                        <input className="rounded-full w-20 text-slate-900" value={VoterVoteID}
onChange={handleVoterVoteID} />
                    </span>
                </label>
            </div>
```

```
                <input className="bg-red-500 hover:bg-blue-900 text-white font-bold py-3 px-16
rounded-full mt-4" type="submit" value="Vote" />
              </form>
            </div>
            {/* =>>>>>>>.....................................................................*/}
            <div className="Result">
              <div >
                <p className="text-2xl font-bold mt-10 mb-20">Result</p>
                <div></div>
                <button onClick={handleQuery1} id="1" className="bg-blue-500 hover:bg-blue-900
text-white font-bold py-1 px-2 rounded-full mt-30 mb-40">
                  Query
                </button>
                <p>{VoteCount1}</p>
              </div>
              <div>
                <div></div>
                <button onClick={handleQuery2} id="2" className="bg-blue-500 hover:bg-blue-900
text-white font-bold py-1 px-2 rounded-full mt-25 mb-10">
                  Query
                </button>
                <p>{VoteCount2}</p>
              </div>
              <div className="mt-20">
                <button onClick={handleQuery3} id="3" className="bg-blue-500 hover:bg-blue-900
text-white font-bold py-1 px-2 rounded-full mt-5 ">
                  Query
                </button>
                <p>{VoteCount3}</p>
              </div>
              <div className="mt-28">
                <button onClick={handleResult} className="bg-red-500 hover:bg-blue-900 text-
white font-bold py-3 px-7 rounded-full ">
                  Winner
                </button>
                <p className="text-3xl">{HighestCount}</p>
              </div>
            </div>
            {/* =>>>>>>>.....................................................................*/}

          </div>
        </div>
    );
}

export default Voting;
```

Voting.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract VoteSystem{
   address public owner;
   constructor(){
       owner= msg.sender;
    }
 struct candidate {
     uint voterId;
     string name;
     uint age;
     uint voteCount;
 }
 mapping (uint => candidate) candidateMap;
 struct voters {
     uint voterId;
     string name;
     uint age;
     bool votingState;
 }
 mapping (uint => voters) votersMap;
 mapping (uint=>bool) registeredVoter;

 modifier checkVoterVoted(uint _votersVoterId){
     require (votersMap[_votersVoterId].votingState == false);
     _;
 }
modifier checkRegisteredVoter(uint _votersVoterId){
       require(registeredVoter[_votersVoterId]==true, "Voter is not Registered");
       _;
}
uint[] voterIdlist;
uint[] candidateIdList;

 function enrollCandidate(uint _voterId,string memory _name,uint  _age )  public {
 require (_age >= 25);
 require (candidateMap[_voterId].voterId != _voterId);

    candidateMap[_voterId].voterId = _voterId;
    candidateMap[_voterId].name = _name;
    candidateMap[_voterId].age = _age;
    candidateIdList.push(_voterId);
 }

 function enrollVoter(uint _voterId,string memory _name,uint _age)  public  returns(bool){
require (_age >= 18);
```

```
require (votersMap[_voterId].voterId != _voterId);

    votersMap[_voterId].voterId = _voterId;
    votersMap[_voterId].name = _name;
    votersMap[_voterId].age = _age;

    voterIdlist.push(_voterId);
   return registeredVoter[_voterId]=true;

 }

 function getCandidateDetails(uint _voterId) view public returns(uint,string memory,uint,uint) {

    return
(candidateMap[_voterId].voterId,candidateMap[_voterId].name,candidateMap[_voterId].age,candidateMa
p[_voterId].voteCount);
 }

 function getVoterDetails(uint _voterId) view public returns (uint,string memory,uint,bool){

    return
(votersMap[_voterId].voterId,votersMap[_voterId].name,votersMap[_voterId].age,votersMap[_voterId].v
otingState);

 }

 function vote(uint _candidateVoterId,uint _votersVoterId) public checkVoterVoted(_votersVoterId)
checkRegisteredVoter(_votersVoterId) {
    candidateMap[_candidateVoterId].voteCount += 1;
    votersMap[_votersVoterId].votingState = true;
 }

 function getVotecountOf(uint _voterId) view public returns(uint){
       require(msg.sender== owner, "Only owner is allowed to Check Results");
    return candidateMap[_voterId].voteCount;
 }

 function getVoterList() view public returns (uint[] memory){

   return  voterIdlist;
   }

 function getCandidateList() view public returns(uint[] memory){


 return candidateIdList;
 }


}
```

Demo Video Link:

https://drive.google.com/file/d/1_DEc0wWvxJKtsFnTcrNnvbnYjYtLEUtq/view?usp=drive_link