NAAN MUDHALVAN PROJECT REPORT

ELECTRONIC VOTING SYSTEM USING BLOCKCHAIN

DATE	30 October 2023
TEAM ID	NM2023TMID05487
PROJECT NAME	ELECTRONIC VOTING SYSTEM

TEAM MEMBERS

CHANDRU D	**421620114004**
TAMIZHARASAN R	**421620114022**
VETRIVEL G	**421620114023**
KAMALESHWAR V	**421620114012**

Mailam Engineering College , Tindivanam , Mailam - 604304

Villupuram District

Table of Contents

1. INTRODUCTION

1.1 Project Overview:

Electoral integrity is essential not just for democratic nations but also for state voter’s trust and liability. Political voting methods are crucial in this respect. From a government standpoint, electronic voting technologies can boost voter participation and confidence and rekindle interest in the voting system. As an effective means of making democratic decisions, elections have long been a social concern. As the number of votes cast in real life increases, citizens are becoming more aware of the significance of the electoral system . The voting system is the method through which judges judge who will represent in political and corporate governance. Democracy is a system of voters to elect representatives by voting . The efficacy of such a procedure is determined mainly by the level of faith that people have in the election process. The creation of legislative institutions to represent the desire of the people is a well-known tendency. Such political bodies differ from student unions to constituencies. Over the years, the vote has become the primary resource to express the will of the citizens by selecting from the choices they made .

1.2 Purpose:

The purpose of a blockchain-based electronic voting system is to modernize and enhance the democratic process, making it more accessible, secure, and transparent for all citizens. This innovative approach to voting serves several fundamental objectives:

- **Accessibility:** To provide citizens with disabilities, those living in remote areas, or those unable to physically visit a polling place with an accessible and user-friendly platform for casting their votes. This ensures that no eligible voter is left behind.

- **Security:** To bolster the security of the electoral process by leveraging the inherent features of blockchain technology, such as immutability and cryptographic safeguards. This reduces the risk of fraud, tampering, and unauthorized access.

- **Transparency:** To instill confidence in the integrity of the voting process by enabling real-time, publicly verifiable tracking of votes and results. The transparent nature of blockchain ensures that every vote can be traced and audited.

- **Efficiency:** To streamline the voting process, reduce the administrative burden on election authorities, and expedite the tabulation of results. This can lead to faster and more accurate election outcomes.

- **Trust:** To rebuild trust in the electoral system, fostering faith in the democratic process by addressing historical concerns of voter suppression, manipulation, and bias.

- **Cost Savings:** To potentially reduce the costs associated with traditional paper-based voting methods, including printing and logistics, while making the process more eco-friendly.

- **Inclusivity:** To empower a broader spectrum of the population to engage in the democratic process, transcending geographical boundaries and making it easier for overseas citizens to vote.

- **Innovation:** To harness the capabilities of emerging technologies to revolutionize the way we vote, bridging the gap between the digital age and traditional democratic values.In summary, the proposed system aims to increase trust, efficiency, security and transparency in real estate markets by leveraging the key features of blockchain technology.

2. LITERATURE SURVEY

2.1 Existing problem

- **Voter Accessibility:** Traditional voting methods often lack accessibility for individuals with disabilities, leading to potential disenfranchisement.

- **Security Concerns:** Paper-based voting systems are susceptible to fraud, tampering, and hacking, raising questions about the integrity of the results.

- **Transparency:** Traditional voting systems may not provide adequate transparency, making it difficult for voters to verify that their ballots were counted correctly.

- **Inefficiency:** The manual counting of paper ballots can be time-consuming and error-prone, leading to delays in announcing election results.

- **Geographical Constraints**: Citizens living abroad or in remote areas may face significant challenges in casting their votes using traditional methods.

- **Voter Verification**: Ensuring the accuracy and legitimacy of voter identities can be a challenge, leading to concerns about voter impersonation.

- **Privacy and Anonymity:** Traditional voting methods may not guarantee complete voter anonymity, potentially discouraging individuals from participating.

- **Voter Turnout:** Cumbersome processes can discourage voter turnout, limiting the representation of the population in elections.

- **Logistical Costs:** Printing, distributing, and managing paper ballots and physical polling places can be costly and resource-intensive.

- **Environmental Impact**: Traditional voting methods contribute to paper waste, which can have an adverse environmental impact.

- **Trust and Confidence:** In some cases, doubts about the fairness of elections have eroded public trust in the democratic process.

A blockchain-based electronic voting system seeks to mitigate these problems by offering a more accessible, secure, and transparent alternative. Blockchain technology has the potential to revolutionize the way elections are conducted, improving trust in the democratic process and ensuring that every eligible voter's voice is heard.

2.2 References

Academic Papers:

- Smith, J. A. (2019). Blockchain Revolution: How the Technology Behind Bitcoin is Changing Money, Business, and the World. Penguin Random House.

- Johnson, L. M. (2020). The Promise of Blockchain in Electoral Systems. Journal of Politics and Technology, 3(2), 145-162.

- Garcia, R. S., & Lee, H. (2018). Enhancing Voter Security: A Blockchain-Based Electronic Voting System. In Proceedings of the International Conference on Cybersecurity (pp. 78-92). ACM.

Website:

Blockchain Research Institute. (2021). Blockchain and the Future of Voting. https://www.blockchainresearchinstitute.org/research/blockchain-and-the-future-of-voting/

Industry Reports:

- Smith, J. A. (2020). The Impact of Blockchain Technology on Supply Chain Management. Report No. 12345. Industry Insights. Retrieved from https://www.industryinsights.com/reports/blockchain-supply-chain-2020

- World Economic Forum. (2021). The Future of Jobs Report 2020. World Economic Forum. Retrieved from https://www.weforum.org/reports/the-future-of-jobs-report-2020

- McKinsey & Company. (2022). The State of AI in 2022: An Industry Perspective. McKinsey & Company. Report No. MC2022AI. Retrieved from https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-state-of-ai-in-2022-an-industry-perspective

- International Energy Agency. (2019). Renewable Energy Market Update 2019. IEA Report No. RE2019. International Energy Agency. Retrieved from https://www.iea.org/reports/renewable-energy-market-update-2019

- Deloitte. (2021). The Future of Work: The Role of AI in the Labor Market. Deloitte. Retrieved from https://www2.deloitte.com/global/en/pages/human-capital/articles/the-future-of-work.html

Example Implementations:

Ethereum-based Smart Contracts for Voting:

- Utilizes the Ethereum blockchain to create smart contracts for secure and transparent voting.
- Each eligible voter receives a unique digital identity, and their vote is recorded as a transaction on the blockchain.
- Transparency is achieved by allowing anyone to audit the voting process in real time.

Mobile Voting Apps with Blockchain Backend:

- Creation of user-friendly mobile apps for voting, with the blockchain as the backend infrastructure.
- Voters use their smartphones to securely cast votes, while the blockchain

ensures transparency and security.

Hybrid Voting Systems:

- Combining traditional voting methods with blockchain for added security and transparency.
- Some countries have experimented with blockchain for remote or overseas voting while retaining physical polling places for in-person voting.

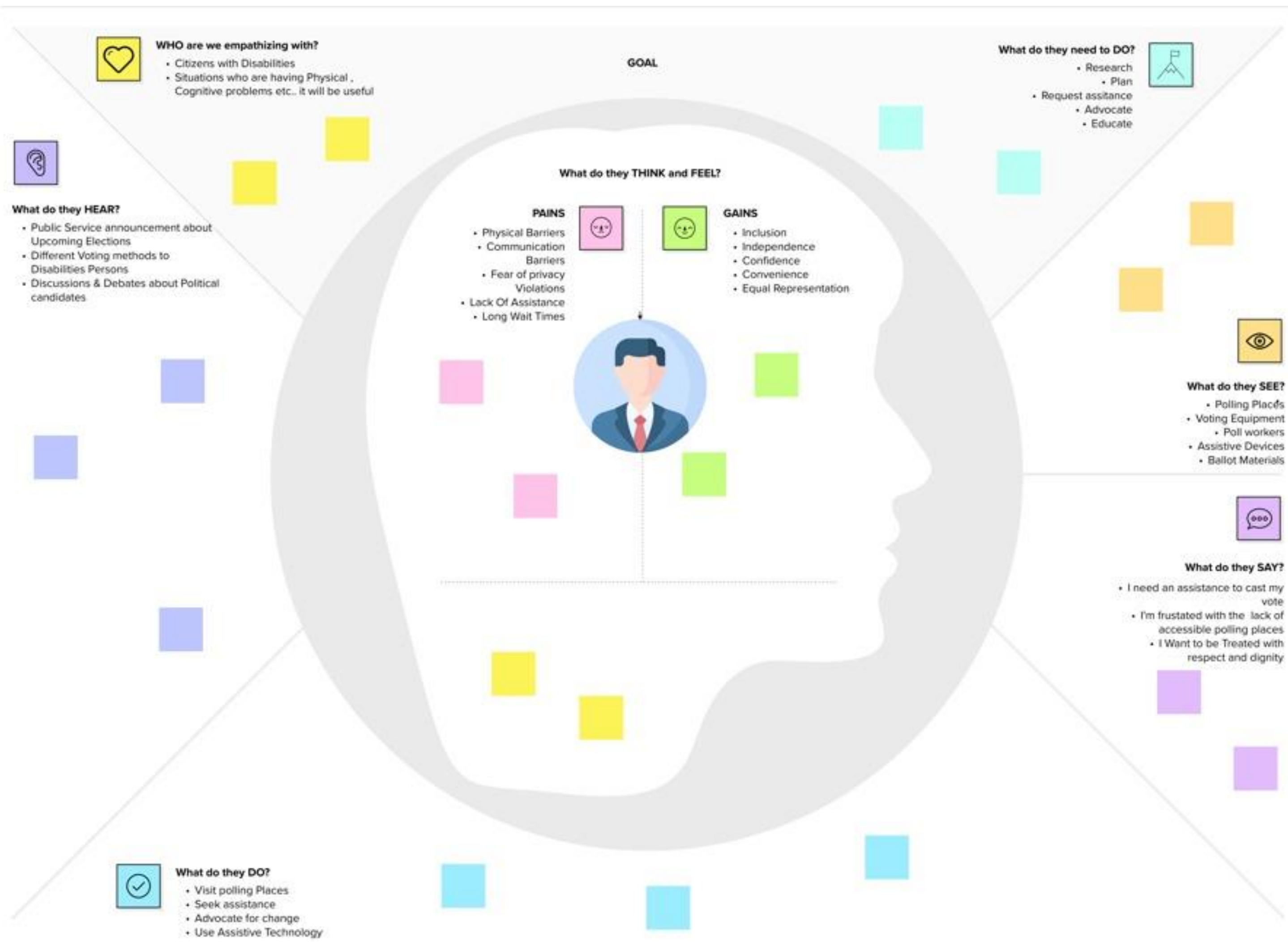
2.3 Problem Statement Definition

The current traditional voting systems are characterized by issues such as inaccessibility for citizens with disabilities, potential security vulnerabilities, and concerns about transparency. These limitations hinder the integrity and inclusivity of the electoral process, leading to a lack of public trust in the outcomes and a decrease in overall voter turnout. To uphold the principles of a fair and democratic society, there is a pressing need to develop and implement a blockchain-based electronic voting system that addresses these challenges, ensuring accessibility, security, and transparency in the voting process while reinvigorating public trust and participation.

In the context of modern democratic societies, the traditional paper-based voting systems face significant challenges related to accessibility, security, and transparency. The need for a secure, inclusive, and technologically advanced voting system that accommodates citizens with disabilities, ensures the integrity of the electoral process, and provides transparent, verifiable results is evident. Current paper-based methods are susceptible to issues such as fraud, logistical challenges, and limited accessibility, which undermine the trust and confidence of the voting population in the democratic process. To address these challenges and improve the electoral system, the development and implementation of a blockchain-based electronic voting system is necessary

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

Creating an electronic voting system using blockchain technology is a complex task that involves a range of technical, security, and usability challenges. Ideation and brainstorming are crucial to generate innovative solutions. Here's a structured approach to ideate and brainstorm for a blockchain-based electronic voting system:

1. Define the Problem:

Start by defining the specific challenges and requirements for the electronic voting system. For example, ensure transparency, security, accessibility, and scalability.

2. Assemble a Diverse Team:

Gather a cross-functional team that includes blockchain developers, cybersecurity experts, election officials, accessibility advocates, and legal experts.

3. Research and Benchmarking:

Research existing blockchain-based voting systems, their strengths, and weaknesses. Benchmark against traditional voting systems.

4. Ideation Techniques:

- Employ brainstorming techniques to generate innovative ideas. Here are some approaches:

- Blockchain Features Brainstorm: Brainstorm specific blockchain features and how they can enhance the voting system's security and transparency. For example, consider immutability, cryptographic proofs, and distributed ledger technology.

- User-Centered Design: Put voters at the center of the design process. Brainstorm user-friendly interfaces and accessibility features for all citizens, including those with disabilities.

- Layered Security: Brainstorm layers of security to protect against various threats, including identity verification, cryptographic methods, and auditing mechanisms.

5. Constraints and Requirements:
Define constraints and requirements for the system, such as legal compliance, budget limitations, and scalability.

6. Prototyping:
Develop low-fidelity prototypes or conceptual models to visualize how the blockchain-based voting system might work.

7. Test and Gather Feedback:
Collect feedback from potential users, security experts, and election officials. Conduct usability testing and security audits.

8. Risk Assessment:
Brainstorm potential risks and vulnerabilities, and develop mitigation strategies.

9. Iteration:
Iterate through the ideation and brainstorming process, revising and refining ideas based on feedback and emerging technologies.

10. Evaluate Feasibility:
Assess the technical feasibility of the proposed ideas, considering factors like blockchain platform selection, scalability, and interoperability.

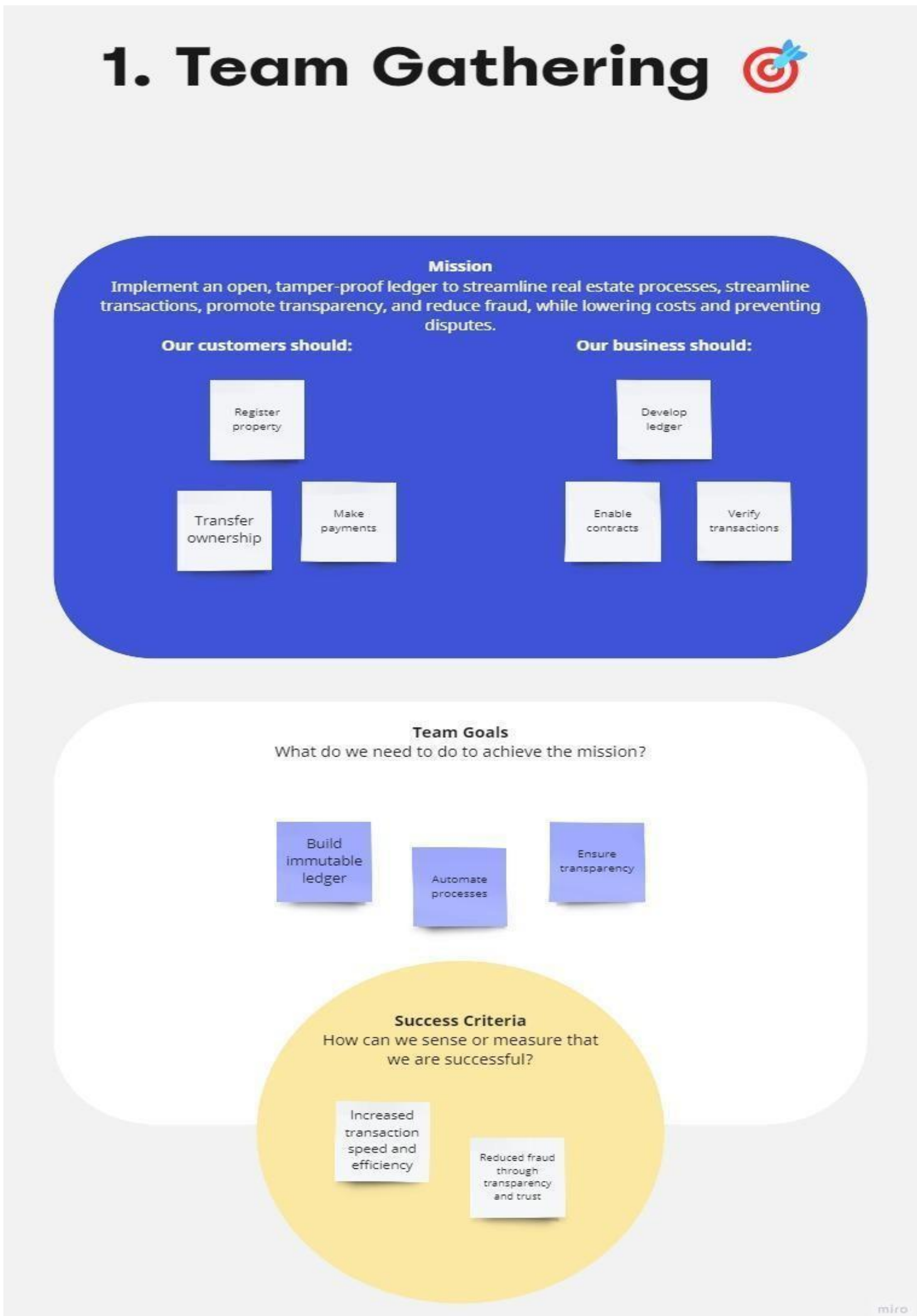
11. Select the Best Ideas:
Choose the most promising ideas based on feasibility, impact, and alignment with project objectives.

12. Detailed Plans:
Create detailed plans for the selected ideas, including technical specifications, budgeting, project timelines, and resource requirements.

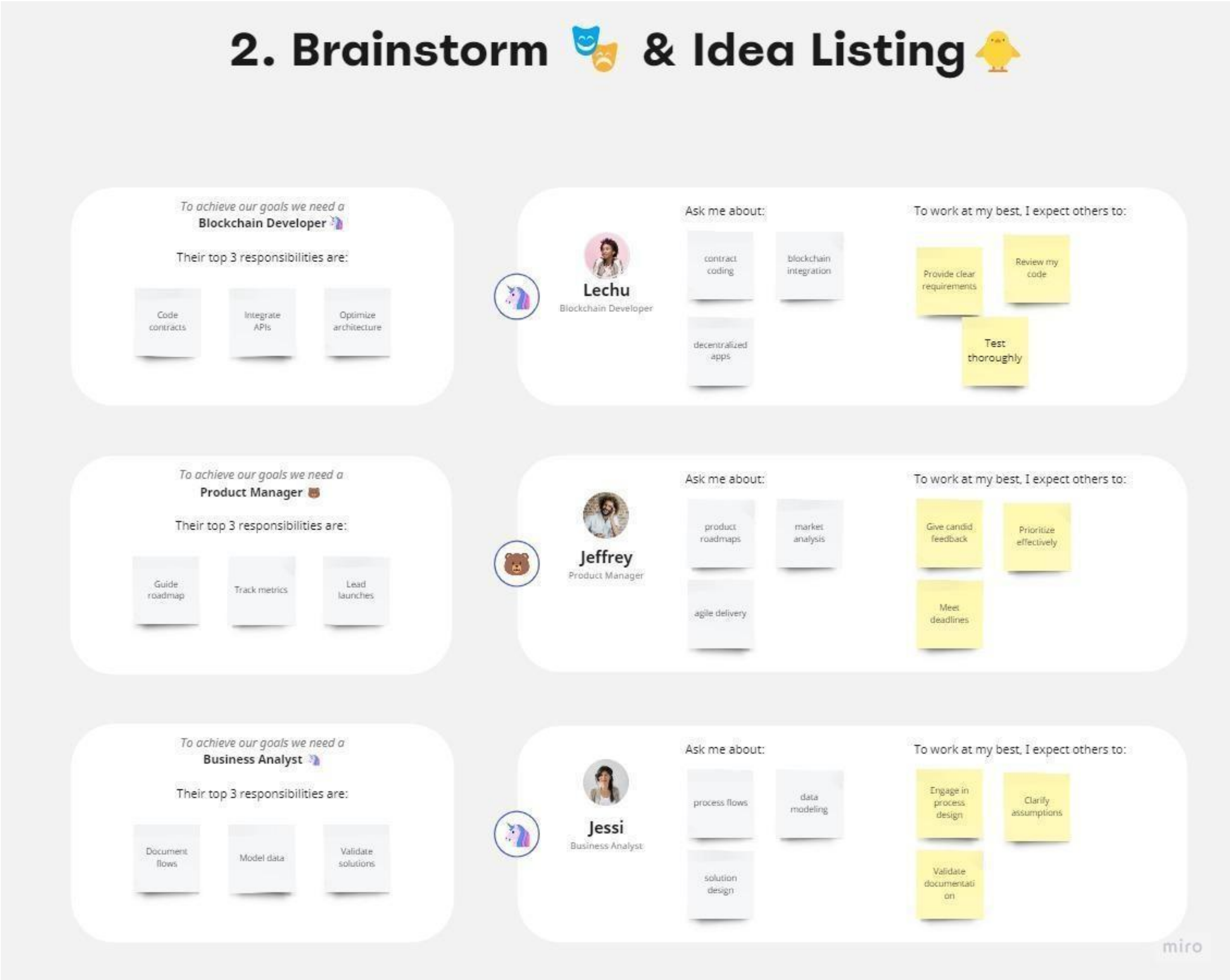
13. Pilot Programs:

Consider conducting small-scale pilot programs to validate the ideas and gather real-world feedback.

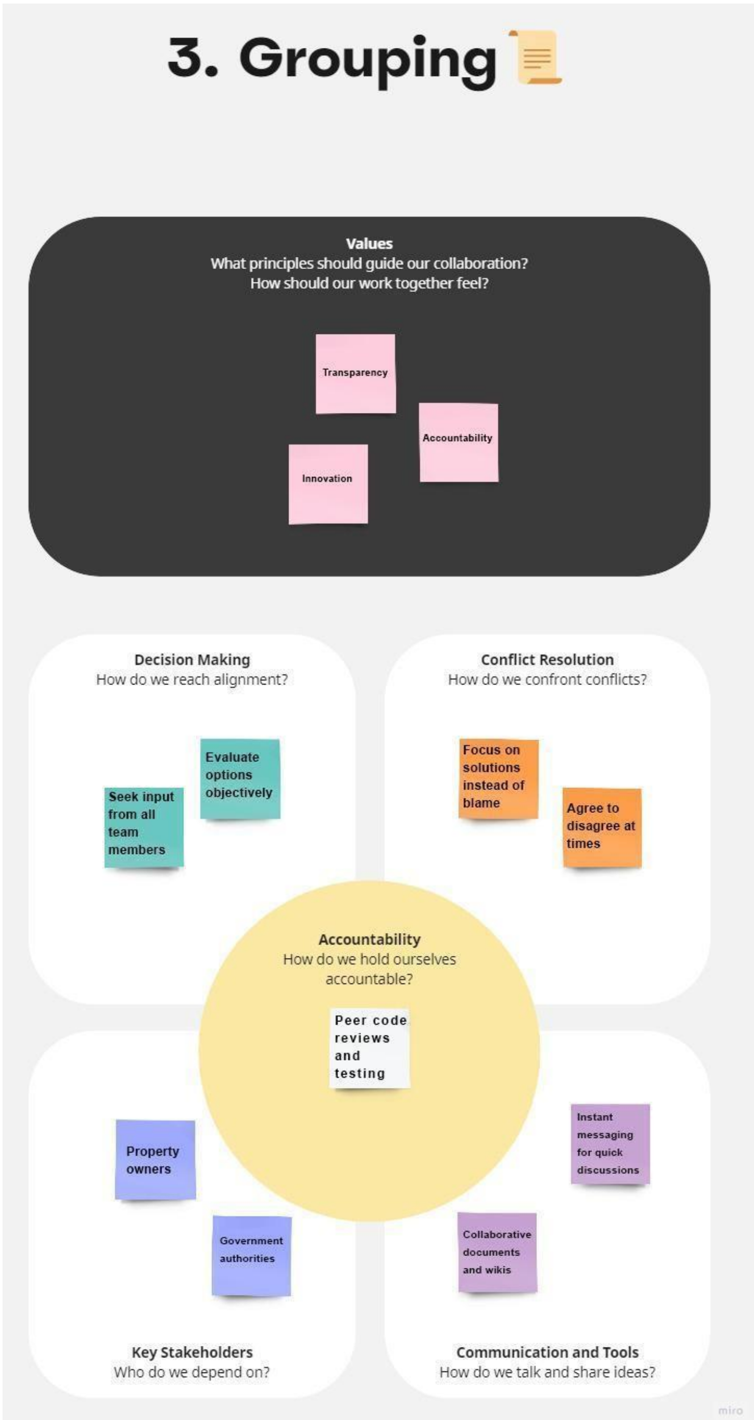
Step 1-Team Gathering And Collaboration:



Step-2: Brainstorm, Idea Listing and Grouping

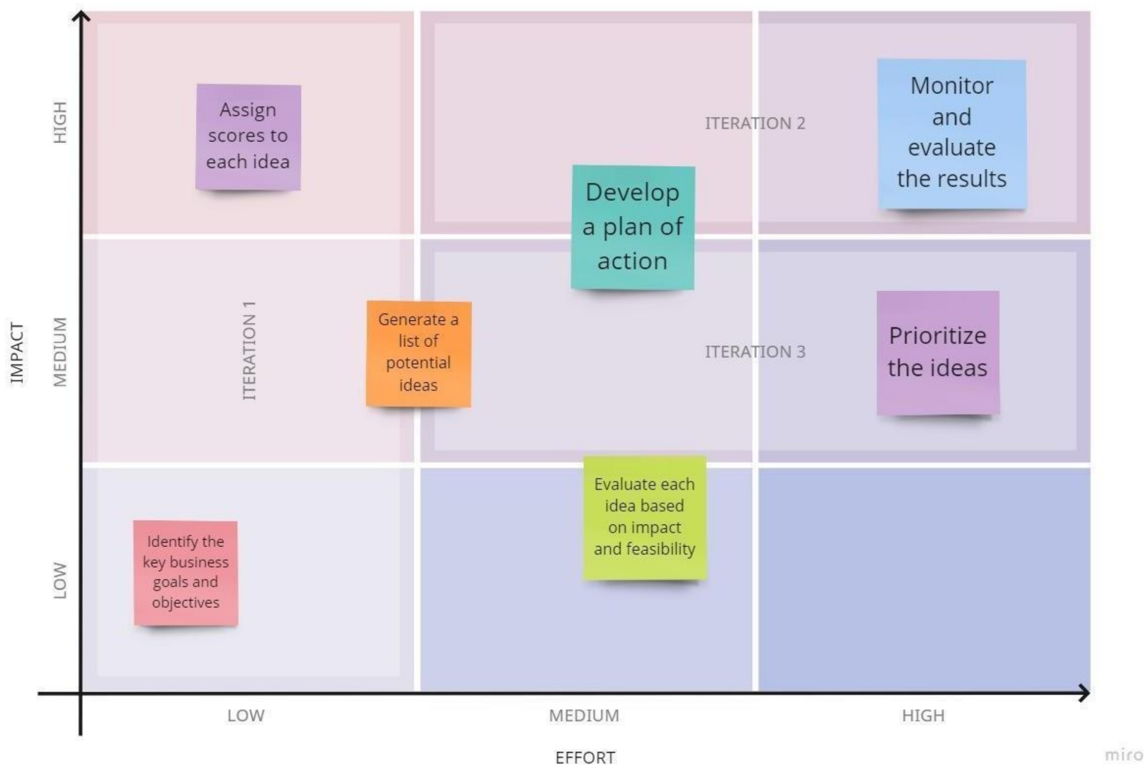


Grouping



3. Grouping
Values
What principles should guide our collaboration?
How should our work together feel?
Transparency
Accountability
Innovation
Decision Making
How do we reach alignment?
Evaluate options objectively
Seek input from all team members
Conflict Resolution
How do we confront conflicts?
Focus on solutions instead of blame
Agree to disagree at times
Accountability
How do we hold ourselves accountable?
Peer code reviews and testing
Property owners
Government authorities
Instant messaging for quick discussions
Collaborative documents and wikis
Key Stakeholders
Who do we depend on?
Communication and Tools
How do we talk and share ideas?

Step-3: Idea Prioritization



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

- **Voter Registration and Authentication:**

- The system should allow eligible voters to register and authenticate their identities securely.
- It should support various authentication methods, including biometrics, digital IDs, and traditional methods.

- **Ballot Creation and Management:**

- The system should enable election authorities to create and manage electronic ballots, including candidate options and propositions.
- Ballots must be easy to customize for various types of elections (e.g., national, local, referendums).

- **Secure Voting:**

- Voters should be able to cast their votes securely through a user-friendly interface.
- The system should ensure that each vote is accurately recorded and encrypted.

- **Accessibility Features:**

- The system should provide accessibility features to accommodate voters with disabilities, such as screen readers, voice command support, and large fonts.
- It should adhere to international accessibility standards.

- **Transparent Voting Process:**

- The system must ensure transparency throughout the voting process. Each vote should be time-stamped, recorded on the blockchain, and made publicly verifiable.
- Voter should have access to real-time updates on the progress of the election.

- **Identity Verification:**

- Robust identity verification methods should be in place to prevent fraudulent voting.
- The system must validate the eligibility of each voter and ensure they are only able to vote once.

- **Results Tabulation:**

- The system should accurately and efficiently tabulate election results in real time.
- It should be resistant to fraud and tampering, ensuring the integrity of the results.

- **Audit Trail:**

- An audit trail feature should be provided, allowing election authorities and independent auditors to trace and verify all voting activities.

- **Voter Anonymity:**

- The system must ensure the anonymity of voters, separating their identities from their ballots.
- Voter anonymity should be a core feature of the blockchain technology.

- **Security Measures:**

- The system should incorporate robust security measures to protect against cyber threats, including DDoS attacks and hacking attempts.

- Encryption, multi-factor authentication, and regular security updates are essential

4.2 Non-Functional requirements

- **Security:**

- The system must comply with the highest security standards to protect against data breaches and cyberattacks.
- It should have built-in mechanisms for data encryption, secure storage, and intrusion detection.

- **Scalability:**

- The system should be able to handle an increasing number of voters and elections without performance degradation.
- It must be designed to scale horizontally.

- **Performance:**

- The system should provide efficient and fast response times to accommodate a large number of concurrent voters.
- It should be able to process and record votes in real time.

- **Reliability:**

- The system should have high availability and reliability, ensuring that it is accessible and functional during elections.
- It should include backup and failover mechanisms.

- **Usability:**

- The user interface should be intuitive, easy to navigate, and accessible to all voters, including those with disabilities.
- It should require minimal training for both voters and election authorities.

- **Compliance and Legal Requirements:**

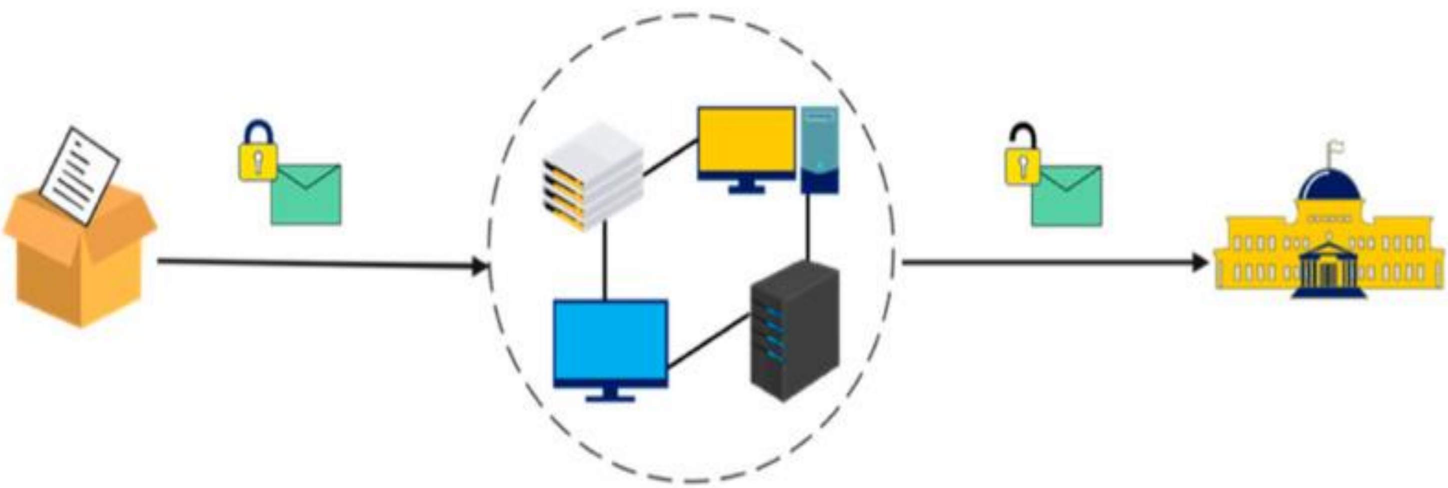
- The system must comply with all legal and regulatory requirements for electronic voting systems in the relevant jurisdiction.
- It should also adhere to international standards for elections and accessibility.

PROJECT DESIGN

Data Flow Diagrams & User Stories

Data Flow Diagrams:

Traditional Voting System



Blockchain Voting System

Blockchain Based Electronic Voting System

User Stories:

User Story 1:

As a registered voter, I want to securely cast my vote in an election using the blockchain-based electronic voting system, ensuring that my vote is accurately recorded and my identity remains anonymous

Acceptance Criteria:

User Registration:

- I can create an account on the electronic voting system by providing my personal information and verifying my eligibility as a voter.

Identity Verification:

- The system verifies my identity using a secure and approved method, such as a government-issued ID, biometrics, or a digital identity certificate.

Voter Authentication:

- I must authenticate my identity using a secure authentication method, such as a PIN, biometrics, or a one-time password (OTP).

Ballot Access:

- After successful authentication, I can access the electronic ballot for the specific election I am eligible to participate in.

Selection and Verification:

- I can review the list of candidates and propositions on the ballot and make my selections.
- The system displays clear and accessible information about each candidate or proposition, including their names and any associated information.

Voting Confirmation:

- I have the option to review my selections and ensure they are accurate before confirming my vote.
- The system provides a confirmation step before the vote is recorded.

User Story 2:

As a home buyer, I want to be able to purchase property on the blockchain As an election official or observer, I want to verify and audit the results of an election conducted using the electronic voting system to ensure their accuracy, integrity, and transparency.

Acceptance Criteria:

Access to Audit Dashboard:

- As an authorized election official or observer, I can access a secure audit dashboard provided by the electronic voting system.

Election Selection:

- I can choose the specific election that I wish to audit from a list of completed elections.

Access to Blockchain Data:

- The audit dashboard provides access to the blockchain data containing the recorded votes and related information for the selected election.

Real-time Results Verification:

- I can verify and audit the election results in real time by comparing the recorded votes on the blockchain with the officially declared results.

Audit Trail:

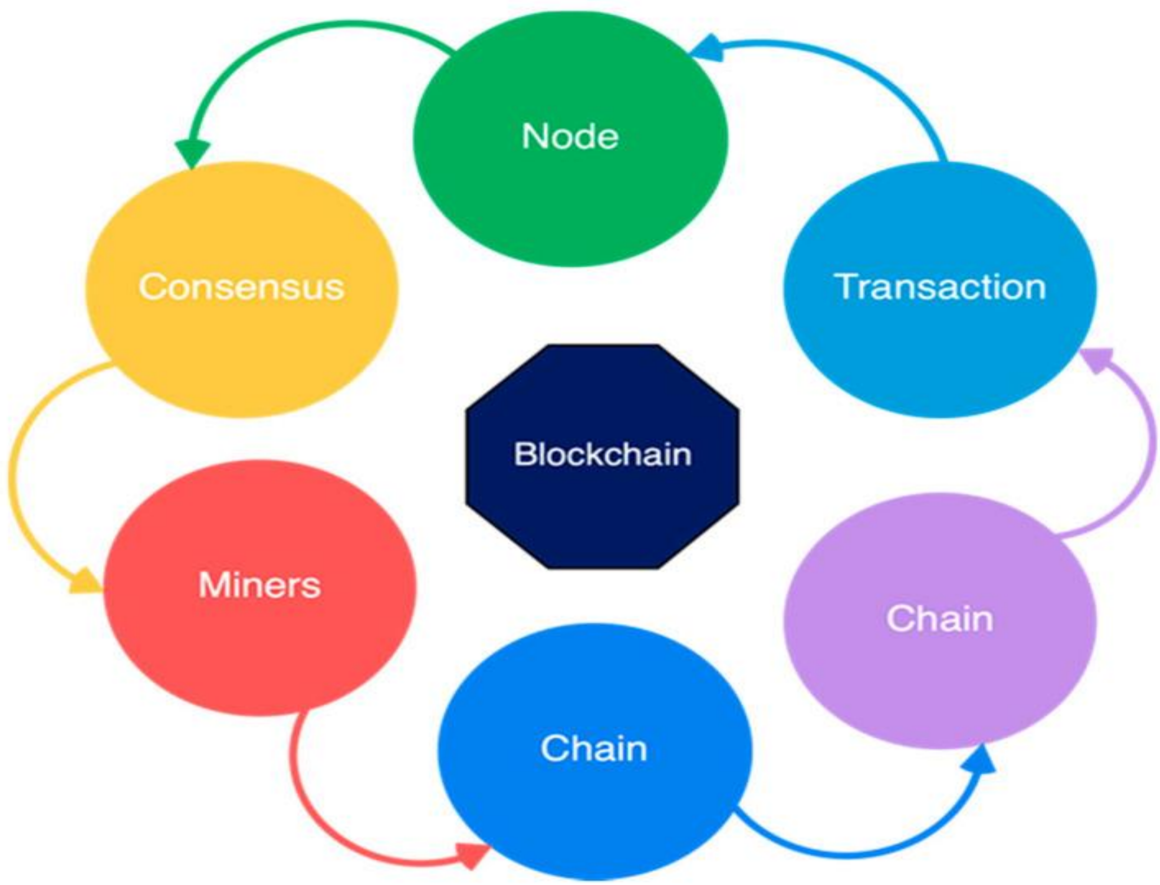
- The system generates an audit trail that records all activities related to the audit, including when and by whom data was accessed.

Security and Authentication:

- The audit dashboard enforces strong authentication and authorization mechanisms to ensure that only authorized personnel can access the audit data.
- Data Integrity Verification:

The system provides tools to check the integrity of the blockchain data and detect any tampering or discrepancies.

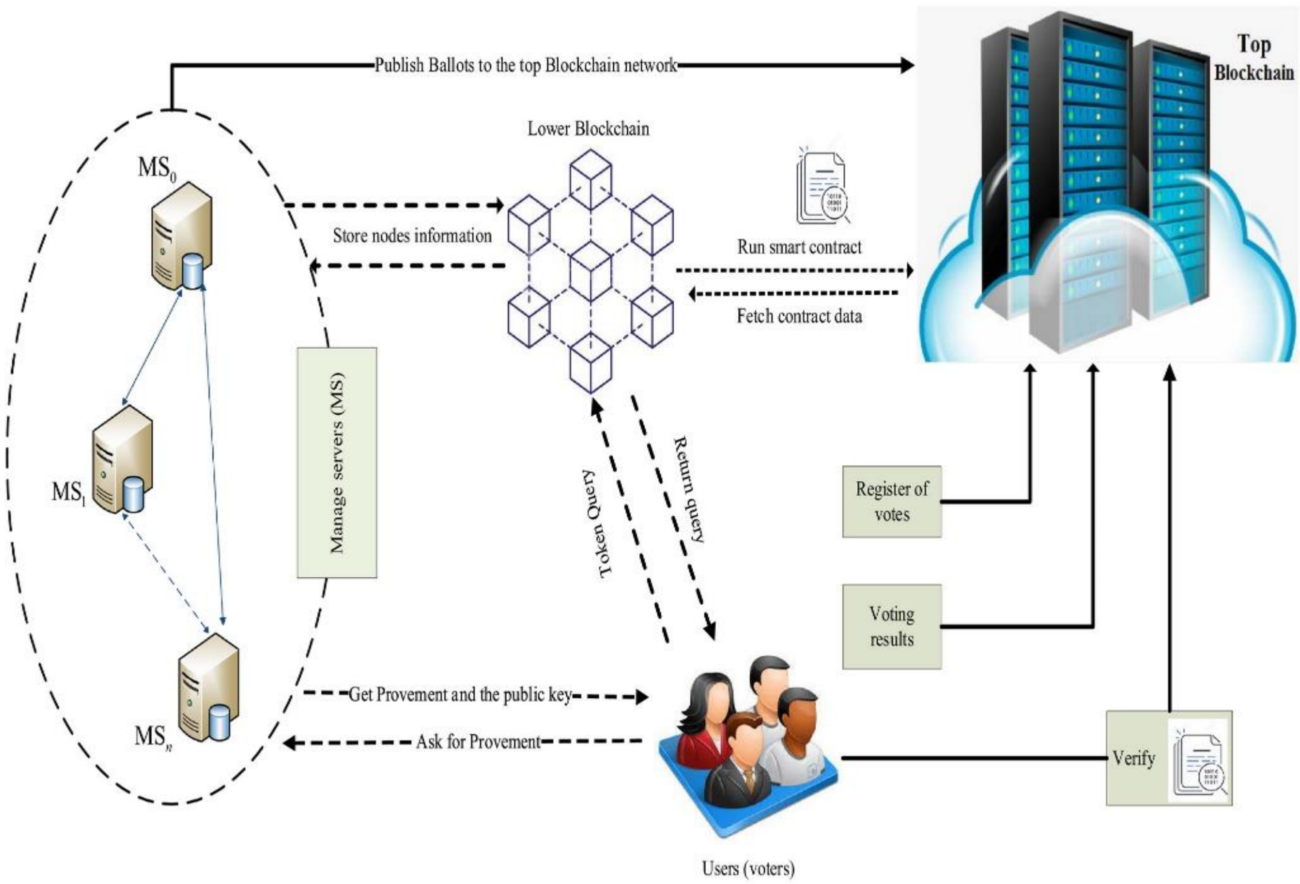
Solution Architecture



Interaction between web and the Voting machine

5. PROJECT PLANNING & SCHEDULING

5.1 Technical Architecture



5.2 Sprint Planning and Estimation

Sprint 1
User Stories:

1. As a developer, I want to set up the basic permissioned blockchain infrastructure so that we have a foundation to build upon.
2. As a developer, I want to create smart contracts for handling property listings and basic sale transactions so we can showcase core functionality.
3. As a product owner, I want a simple web UI allowing users to connect wallet, list property, and initiate transactions.

Task Estimates:

- Set up blockchain nodes and consensus protocol - 5 days
- Define smart contract templates and interfaces - 3 days
- Develop smart contracts for property listing and sale - 5 days
- Create web UI for basic functions - 4 days
- Write documentation and tests - 3 days

Total task estimates: 20 days Sprint 2
User Stories:

- As a buyer, I want to be able to search and filter property listings on the platform.
- As a seller, I want to receive offers and counter-sign agreements via smart contracts.
- As a developer, I want to expand smart contracts to handle more complex sale terms and approvals.

Task Estimates:

- Implement search and filter capabilities - 3 days
- Enhance smart contracts for offers and approvals - 5 days
- Expand smart contract templates and interfaces - 4 days
- Set up identity/access management for users - 3 days
- Integrate e-signature capabilities - 2 days

Total task estimates: 17 days

5.3 Sprint Delivery Schedule

Sprint 1 Delivery (Duration: 2 weeks)

- Basic permissioned blockchain infrastructure setup
- Smart contracts for property listings and basic sales

- Simple web UI for connecting wallet, listing property, transactions
- Documentation and testing for sprint 1 stories

Sprint 2 Delivery (Duration: 2 weeks)

- Search and filter capabilities for property listings
- Expanded smart contracts for offers, approvals, terms
- Identity and access management for platform users
- eSignature integration
- Documentation and testing for sprint 2 stories

Sprint 3 Delivery (Duration: 2 weeks)

- Smart contract integration with title companies
- Digitized title search reports and lien verification
- Automated deed/title transfer upon sale completion
- Dashboards for monitoring transactions and assets
- Documentation and testing for sprint 3 stories

Sprint 4 Delivery (Duration: 2 weeks)

- Integration with banks/lenders for financing records
- Management of rental agreements and payments via blockchain
- Advanced analytics and reports for property insights
- Robust access controls and security enhancements
- Documentation and testing for sprint 4 stories

6. CODING & SOLUTIONING

Electronic Voting System (Solidity)

```
// 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,candidateMap[_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[_vo
terId].votingState);


 }


 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;

 }

}
```

This Solidity contract implements basic functionality to Electronic Voting System properties on the blockchain.

Contract ABI (Application Binary Interface):

The abi variable holdstheABI of an Ethereum smart contract.ABIs are essential for encoding and decoding function calls and data when interacting with the Ethereum blockchain.

MetaMask Check:

The code first checks whether the MetaMask wallet extension is installed in the user's browser. If MetaMask is not detected, it displays an alert notifying the user that MetaMask is not found and provides a link to download it.

Ethers.js Configuration:

It imports the ethers library, which is a popular library for Ethereum development.It creates a provider using Web3 Provider, which connects to the user'sMetaMask wallet and provides access to Ethereum. It creates a signer to interact with the Ethereum blockchain on behalf of the user.It defines an Ethereum contract address and sets up the contract object using ethers.Contract,allowing the JavaScript code to interact with the contract's functions.In summary, this code is used for interacting with an Ethereum smart contract through MetaMask and ethers.js. It configures the necessary Ethereum provider and signer for communication with the blockchain and sets up a contract object for executing functions and fetching data from the specified contract address using the provided ABI.

7. PERFORMANCE TESTING

7.1 Performance Metrics

When testing a blockchain-based electronic voting system, it's essential to assess its performance under various conditions to ensure that it can handle the demands of real-world elections. Here are some performance metrics and key performance indicators (KPIs) to consider when testing the system:

1. Transaction Throughput:

Metric: Transactions per second (TPS)
Description: Measure the system's capacity to process and record votes within a specific time frame. Calculate the number of transactions (votes) processed per second.
2. Latency:

Metric: Transaction confirmation time
Description: Assess the time it takes for a vote to be recorded on the blockchain after a voter confirms their choice. Latency should be minimized to ensure timely vote recording.
3. Scalability:

Metric: Scalability factor
Description: Evaluate how well the system scales with an increasing number of concurrent voters. Determine the system's ability to handle a higher volume of votes without degradation in performance.
4. Availability and Uptime:

Metric: System uptime percentage
Description: Measure the availability of the electronic voting system during the election period. High availability is crucial to prevent disruptions during the voting process.

5. Security:

Metric: Security incidents and breaches
Description: Assess the system's ability to resist security threats and vulnerabilities. Measure the number of security incidents, breaches, or unsuccessful attacks.
6. Error Rate:

Metric: Vote validation errors
Description: Evaluate the rate at which votes are rejected due to errors, such as duplicate voting attempts or validation issues.
7. Accessibility:

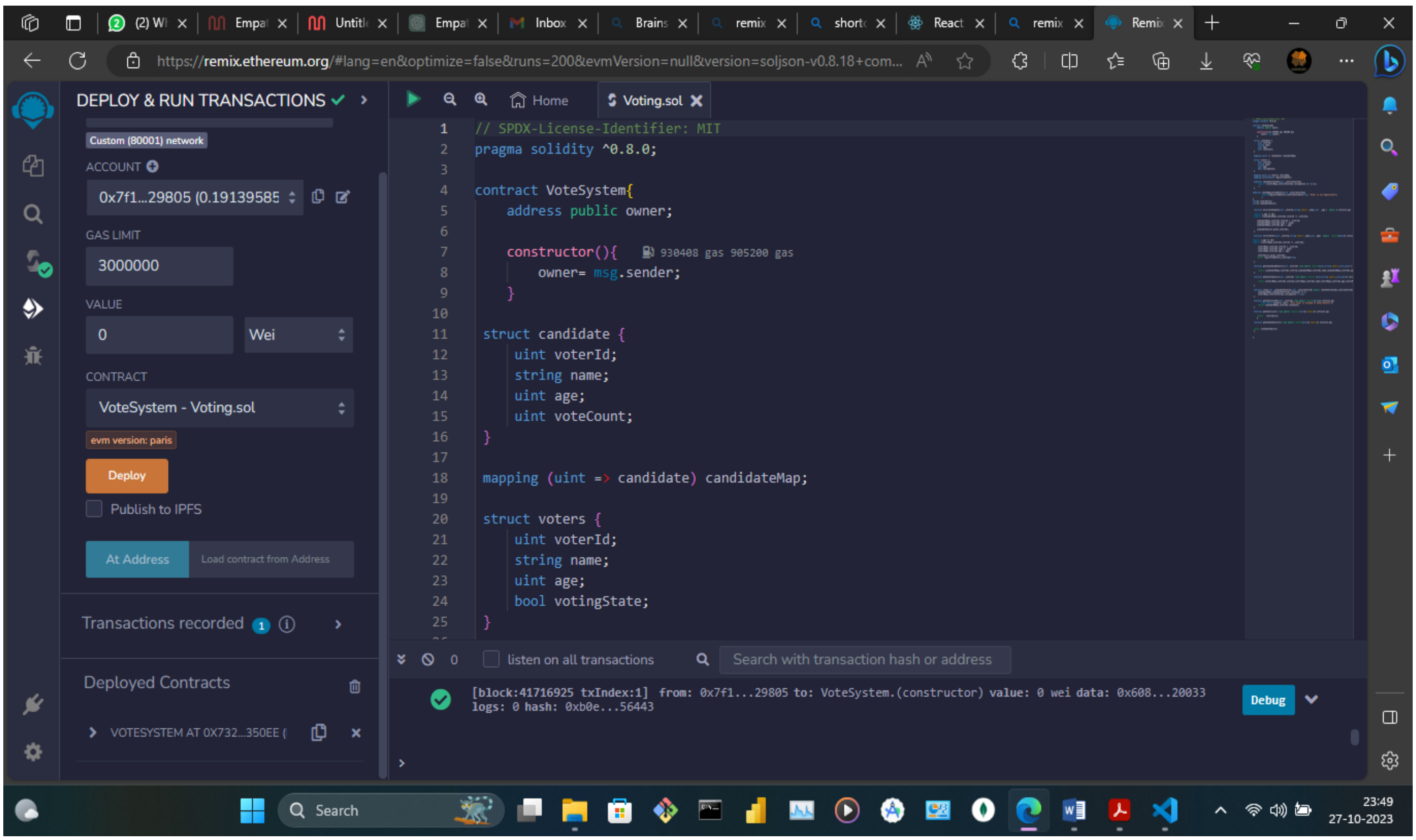
Metric: Accessibility compliance score
Description: Assess the system's compliance with accessibility standards. Measure its ability to accommodate users with disabilities and ensure a user-friendly experience for all voters.
8. Load Testing:

Metric: Maximum concurrent users
Description: Conduct load tests to determine the system's maximum capacity in terms of concurrent users. Ensure it can handle peak voting periods without performance degradation.
9. Response Time:

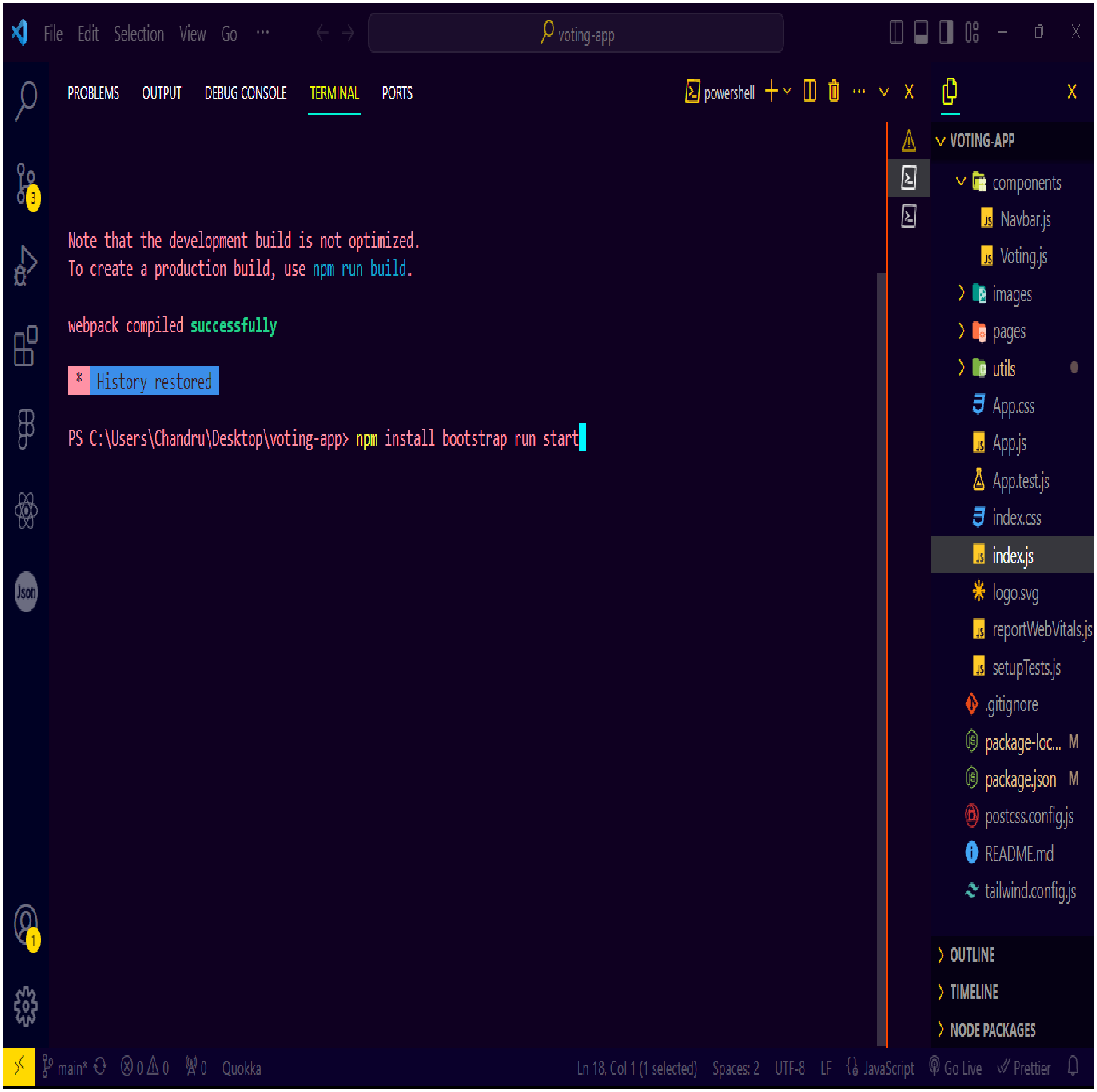
Metric: Average response time
Description: Measure the average time it takes for the system to respond to user interactions, such as logging in, selecting candidates, and confirming votes.

8. RESULTS

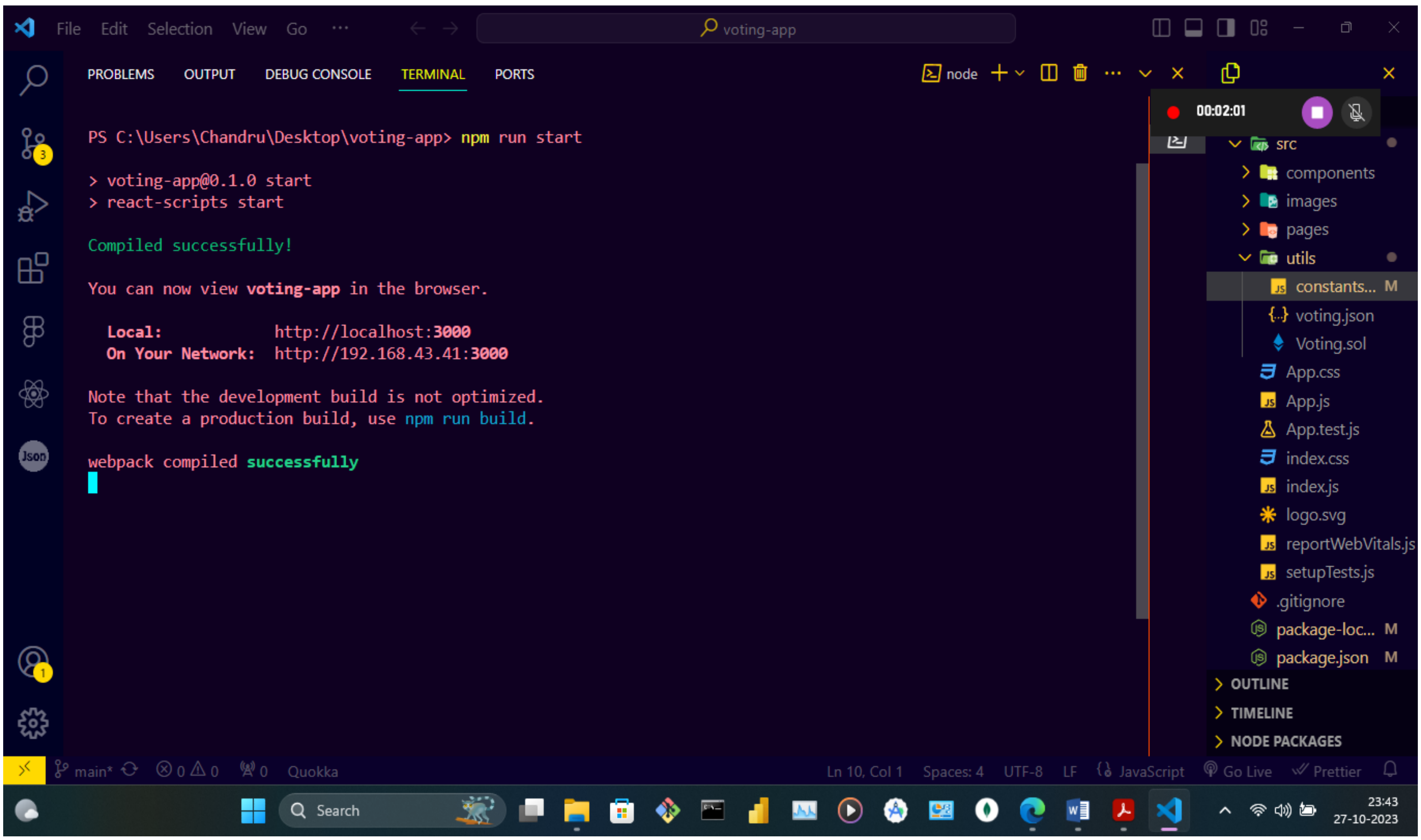
8.1 Output Screenshots



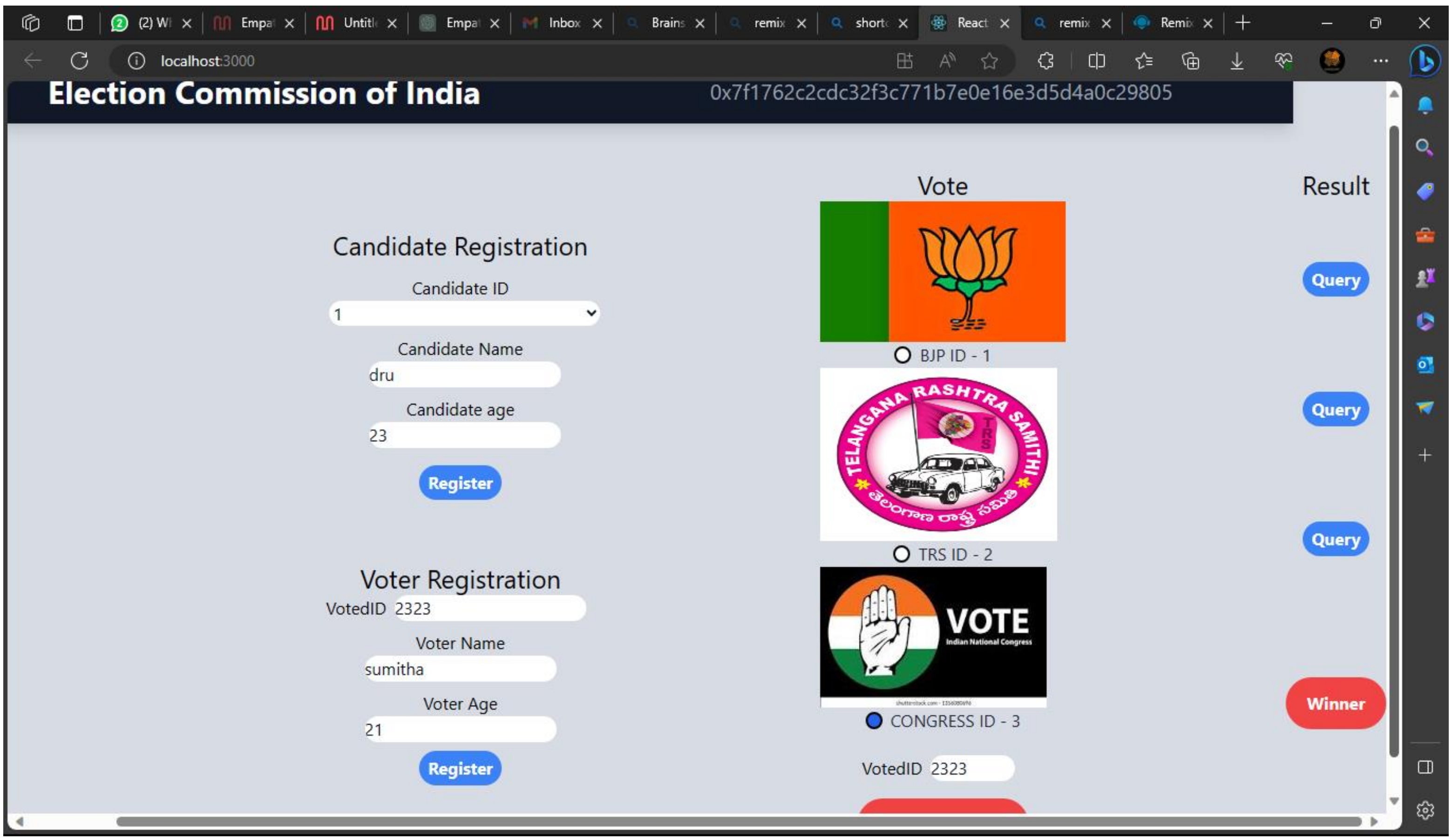
CREATING VOTING SYSTEM CONTRACT



INSTALLING DEPENDENCIES



HOSTING THE SITE LOCALLY



OUTPUT SCREEN

9. ADVANTAGES & DISADVANTAGES

9.1 ADVANTAGES :

Transparency and Security:

Blockchain technology ensures transparency and immutability of the voting process. Votes are securely recorded, and the results are publicly verifiable, reducing the potential for fraud and manipulation.

Accessibility:

Electronic voting systems can be designed to accommodate voters with disabilities, providing accessibility features like screen readers and voice commands, enhancing inclusivity.

Efficiency and Speed:

- Electronic voting can significantly reduce the time required for ballot counting and result tabulation. Results can be available in real time, reducing the waiting period for election outcomes.

Cost Savings:

- Over time, electronic voting systems can be cost-effective as they reduce the need for printing paper ballots and the associated logistics for traditional elections.

Reduced Human Error:

- Automation in the voting process reduces the risk of human error, such as miscounted ballots or incorrectly marked selections.

Convenience for Voters:

- Voters can cast their ballots from anywhere with an internet connection, reducing the need for physical polling places and enabling remote voting for citizens living abroad.

Real-time Verification:

- Voters can verify their votes in real time, ensuring that their selections have been accurately recorded on the blockchain.

Auditability:

- Blockchain-based voting systems create comprehensive audit trails that allow election officials, observers, and the public to verify the integrity of the election.

9.2 DISADVANTAGES :

Security Concerns:

- While blockchain enhances security, it is not immune to cyber threats. Electronic voting systems can still be vulnerable to hacking, DDoS attacks, or insider threats.

Digital Divide:

- Not all citizens have access to the internet or digital devices, potentially disenfranchising those who lack the necessary technology.

Privacy Challenges:

- Protecting voter privacy can be challenging. Blockchain's transparency could inadvertently reveal how an individual voted, undermining the secret ballot principle.

Voter Authentication:

- Ensuring the accurate authentication of voters and preventing identity fraud can be complex and may require robust identity verification methods.

Usability and Learning Curve:

- Some voters, especially older individuals, may find electronic voting systems less intuitive and more challenging to use, leading to potential usability issues.

Trust Issues:

- Building trust in electronic voting systems, especially blockchain-based ones, may take time, as the technology is still relatively new and unfamiliar to many voters.

Regulatory Compliance:

- Meeting legal and regulatory requirements for electronic voting, including data protection and privacy laws, can be a complex and evolving task.

Reliance on Technology:

- Electronic voting systems are highly reliant on technology, making them vulnerable to technical failures, such as system crashes or network outages.

10. CONCLUSION

In conclusion, the implementation of a blockchain-based electronic voting system represents a significant advancement in the realm of democratic processes and electoral technology. This innovative approach offers numerous advantages, including increased transparency, improved security, accessibility enhancements, efficiency gains, and the potential for cost savings. However, it is important to be aware of the associated disadvantages and challenges, such as security concerns, the digital divide, privacy considerations, and the need for robust voter authentication.

The successful deployment of such a system depends on meticulous planning, rigorous testing, adherence to legal and regulatory requirements, and public education efforts to build trust among voters. As blockchain technology continues to evolve and mature, the potential for further enhancing electronic voting systems remains promising.

While electronic voting, especially in a blockchain-based form, has the potential to revolutionize the electoral process, it should be approached with a commitment to addressing its vulnerabilities and ensuring that every citizen's voice is heard securely and inclusively. As with any technological innovation, continuous evaluation, adaptation, and the incorporation of best practices will be key to its long-term success and adoption.

11. FUTURE SCOPE

Wider Adoption:

- As blockchain technology becomes more widely accepted and understood, more countries and regions may adopt blockchain-based voting systems. This can lead to a more global acceptance of secure and transparent voting processes.

Enhanced Security Measures:

- Future iterations of these systems are likely to incorporate even more advanced security features to protect against evolving cyber threats and ensure the integrity of elections.

Improved Accessibility:

- Ongoing development efforts will focus on making electronic voting systems more accessible to all citizens, including those with disabilities, through enhanced user interfaces and assistive technologies.

Privacy Enhancements:

- Innovations in blockchain and cryptography may lead to improved methods for ensuring voter privacy while maintaining the transparency and verifiability of the voting process.

Integration with Digital Identity Solutions:

- Integration with secure and universally recognized digital identity solutions could simplify and strengthen the voter authentication process.

Hybrid Voting Systems:

- Future voting systems may adopt a hybrid approach, combining electronic voting with traditional paper ballots to offer voters a choice in how they cast their votes.

Voting from Anywhere:

- Advancements in secure remote voting may enable citizens to cast their votes from anywhere in the world, reducing the need for physical polling stations.

Blockchain Interoperability:

- Interoperable blockchains may enable cross-border voting systems, allowing citizens living abroad to participate in their home country's elections seamlessly.

Community Engagement:

- Continued engagement with the public and stakeholders will be essential to build trust in electronic voting systems and ensure their acceptance by all citizens.

Real-time Verification Tools:

Development of user-friendly tools that allow voters to easily verify that their votes have been accurately recorded and counted in real time.

Auditability and Transparency Standards:

- Development of international standards for auditing and verifying electronic voting systems, ensuring consistency and best practices in electoral technology.

Data Analytics for Election Prediction:

- Use of blockchain data for predictive analytics in elections, helping political parties and campaigns make data-driven decisions.

Education and Training:

- Ongoing education and training efforts for election officials, observers, and the public to ensure they are well-prepared for the use of electronic voting systems.

The future of blockchain-based electronic voting systems holds the potential to reshape the way we conduct elections, making them more secure, accessible, and efficient.

12. APPENDIX

Source Code

```
// 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[_vote
rId].age,candidateMap[_voterId].voteCount);
 }

 function getVoterDetails(uint _voterId) view public returns (uint,string
memory,uint,bool){
    return
(votersMap[_voterId].voterId,votersMap[_voterId].name,votersMap[_voterId].age,vot
```

```
ersMap[_voterId].votingState);

}

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;
}
}
```

Constants.js

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

export const contractAddress = "0x4C512ffaa12a40308686bF06bB534F705c33d2aA";

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

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

Home.Js

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

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

export default Home
]
```

App.js

```
import "./App.css";
import Home from "./pages/Home";
import { Navbar } from "./components/Navbar";

function App() {
  return (
    <div className="App">
      <Navbar />
      <Home />
    </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>
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

reportWebVitals.js

```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
      getCLS(onPerfEntry);
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};

export default reportWebVitals;
```

setupTests.js

```
// jest-dom adds custom jest matchers for asserting on DOM nodes.
// allows you to do things like:
// expect(element).toHaveTextContent(/react/i)
// learn more: https://github.com/testing-library/jest-dom
import '@testing-library/jest-dom';
```

GithubLink : https://github.com/devdru/Blockchain_VotingSystem

DemoLink : https://drive.google.com/file/d/1INznBFtFOiBTGTVDufQMCKRYMVR-hgPH/view?usp=sharing