**TRACKING PUBLIC INFRASTRUCTURE AND TOLL PAYMENT SYSTEM**

## **A PROJECT REPORT**

***Submitted by***

# TEAM ID : NM2023TMID01917

|  |  |
| --- | --- |
| **ELAVARASI G** | **623320104008** |
| **THIRUMAVALAVAN V** | **623320104024** |
| **VIMALRAJ M** | **623320104027** |
| **YASOTHA V** | **623320104028** |

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

***of***

# BACHELOR OF ENGINEERING

## **IN**

**COMPUTER SCIENCE AND ENGINEERING**



**VETRI VINAYAHA COLLEGE OF ENGINEERING AND TECHNOLOGY**

## **THOTTIAM, TRICHY­-621215**

# TABLE OF CONTENT

# INTRODUCTION

* 1. Project Overview

1.2Purpose

# LITERATURE SURVEY

* 1. Existing problem
  2. References
  3. Problem Statement Definition

# IDEATION & PROPOSED SOLUTION

* 1. Empathy Map Canvas
  2. Ideation & Brainstorming

# REQUIREMENT ANALYSIS

* 1. Functional requirement
  2. Non-Functional requirements

# PROJECT DESIGN

* 1. Data Flow Diagrams & User Stories
  2. Solution Architecture

# PROJECT PLANNING

* 1. Technical Architecture

# CODING & SOLUTIONING

* 1. Feature

# PERFORMANCE TESTING

# RESULTS

* 1. Output Screenshots

# ADVANTAGES & DISADVANTAGES

# 10.1Advantages

# 10.2 Disadvantages

1. **CONCLUSION**

# FUTURE SCOPE

1. **APPENDIX**

13.1 Source Code

13.2 GitHub & Project Demo Link

**1.INTRODUCTION:**

**1.1 Project Overview:**

The "Public Infrastructure and Toll Payment System Tracking" project aims to develop a comprehensive system for monitoring and managing public infrastructure, specifically toll roads and bridges.

This system will provide a streamlined and efficient method for tracking infrastructure conditions, collecting toll payments, and ensuring the overall safety and usability of the transportation network.

Develop a centralized database for storing and managing information about public infrastructure, including road and bridge details, maintenance schedules, and toll collection data.

Implement a user-friendly web and mobile interface for infrastructure administrators, maintenance crews, and the public to access and update relevant information.

**1.2 PURPOSE:**

The purpose of tracking public infrastructure and toll payment systems is multifaceted and serves several important goals:

Efficient Infrastructure Maintenance: Tracking public infrastructure, such as roads, bridges, and tunnels, allows authorities to proactively identify maintenance and repair needs. Regular inspections and monitoring can help address issues before they become critical, thus ensuring the safety and longevity of these critical assets.

Cost Savings: Effective tracking and maintenance can lead to cost savings in the long run. Preventative maintenance can be less expensive than costly emergency repairs or replacements, and it can also extend the lifespan of infrastructure assets.

Improved Safety: Monitoring infrastructure conditions can help identify potential safety hazards. By addressing these issues promptly, the risk of accidents and emergencies can be reduced, leading to enhanced safety for commuters and travel.

Data-Driven Decision-Making: Tracking infrastructure provides valuable data that can be used to make informed decisions. This data can include information about usage patterns.

Enhanced User Experience: In the context of toll payment systems, tracking can provide a more convenient and seamless experience for users. This includes enabling electronic toll collection methods, reducing congestion, and providing real-time information about toll rates and traffic conditions.

**2. LITERATURESURVEY:**

Infrastructure Tracking:

a. "Intelligent Transportation Systems: A Comprehensive Review" by Karim et al. (2016)

This paper provides an overview of intelligent transportation systems and their role in tracking public infrastructure, including road condition monitoring and maintenance.

b. "Advances in Infrastructure Monitoring Using Remote Sensing Techniques" by Zhang et al. (2019)

Discusses the use of remote sensing technologies for infrastructure monitoring, including bridges, highways, and tunnels.

Toll Payment Systems:

a. "Electronic Toll Collection Systems: A Review" by Al-Obaedi et al. (2018)

Provides an in-depth review of electronic toll collection systems, discussing various technologies and their benefits.E

**2.1 Existing problem**

One existing system for tracking public infrastructure and toll payment is the E-ZPass system in the United States. E-ZPass is an electronic toll collection system that enables drivers to pay tolls on highways, bridges, and tunnels without stopping at toll booths.

It is used in many states across the U.S., and it helps streamline the toll collection process while also providing a means to track and manage public infrastructure usage.

These cumulative challenges underscore the pressing need for a more secure, private, and transparent approach to voting, where the individual's identity is verified through immutable biometric data and all actions within the electoral process are securely recorded on a blockchain, ensuring the sanctity of the vote and reestablishing trust in the democratic process.

1. Top of Form

**2.2References:**

1. Zhang, G., Li, T., Li, Y., Hui, P. and Jin, D., “Blockchain-BasedData Sharing System for AI-Powered Network Operations”. Journal ofCommunications and Information Networks, 3(3), pp.1-8, 2018
2. P˜anescu, A.T. and Manta, V., “Smart Contracts for Research DataRights Management over the Ethereum Blockchain Network”. Science& Technology Libraries, 37(3), pp.235-245, 2018.
3. Rahmadika, S., Ramdania, D.R. and Harika, M., “Security Analysis onthe Decentralized Energy Trading System Using Blockchain Technology”.Journal Online Informatika, 3(1), pp.44-47, 2018.
4. Sharma, P.K. and Park, J.H., “Blockchain based hybrid network archi-tecture for the smart city”. Future Generation Computer Systems, 86,pp.650-655, 2018.
5. Jindal, A., Aujla, G.S. and Kumar, N., “SURVIVOR: A blockchain basededge-as-a-service framework for secure energy trading in SDN-enabledvehicle-to-grid environment”. Computer Networks, 153, pp.36-48, 2019.
6. Huang, X., Zhang, Y., Li, D. and Han, L., “An optimal scheduling al-gorithm for hybrid EV charging scenario using consortium blockchains”.Future Generation Computer Systems, 91, pp.555-562, 2019.
7. Fujihara, A., “PoWaP: Proof of Work at Proximity for a crowdsensingsystem for collaborative trafﬁc information gathering”. Internet of Things,2019.
8. Xie, L., Ding, Y., Yang, H. and Wang, X., “Blockchain-Based Secureand Trustworthy Internet of Things in SDN-Enabled 5G-VANETs”. IEEEAccess, 7, pp.56656-56666, 2019.
9. Xu, C., Liu, H., Li, P. and Wang, P., “A Remote Attestation SecurityModel Based on Privacy-Preserving Blockchain for V2X”. IEEE Access,6, pp.67809-67818, 2018.

**2.2Problem Statement Definition:**

The problem at hand is to develop a comprehensive and efficient system for tracking public infrastructure and toll payment within a given region or network. This system should address the following key challenges

Infrastructure Tracking: To monitor and maintain various elements of public infrastructure such as roads, bridges, tunnels, and other transportation-related assets. This includes recording their current condition, maintenance schedules, and necessary repairs or upgrades.

Toll Payment System: To establish a streamlined and technologically advanced toll payment system that enables seamless transactions for users while ensuring accurate revenue collection for the operation and maintenance of the infrastructure. This system should incorporate various payment methods, such as electronic toll collection (ETC), mobile applications, and traditional toll booths.

Data Integration: To integrate data from multiple sources, including traffic cameras, sensors, toll booths, and payment processing systems, in order to provide real-time and historical information on infrastructure status and toll transactions.

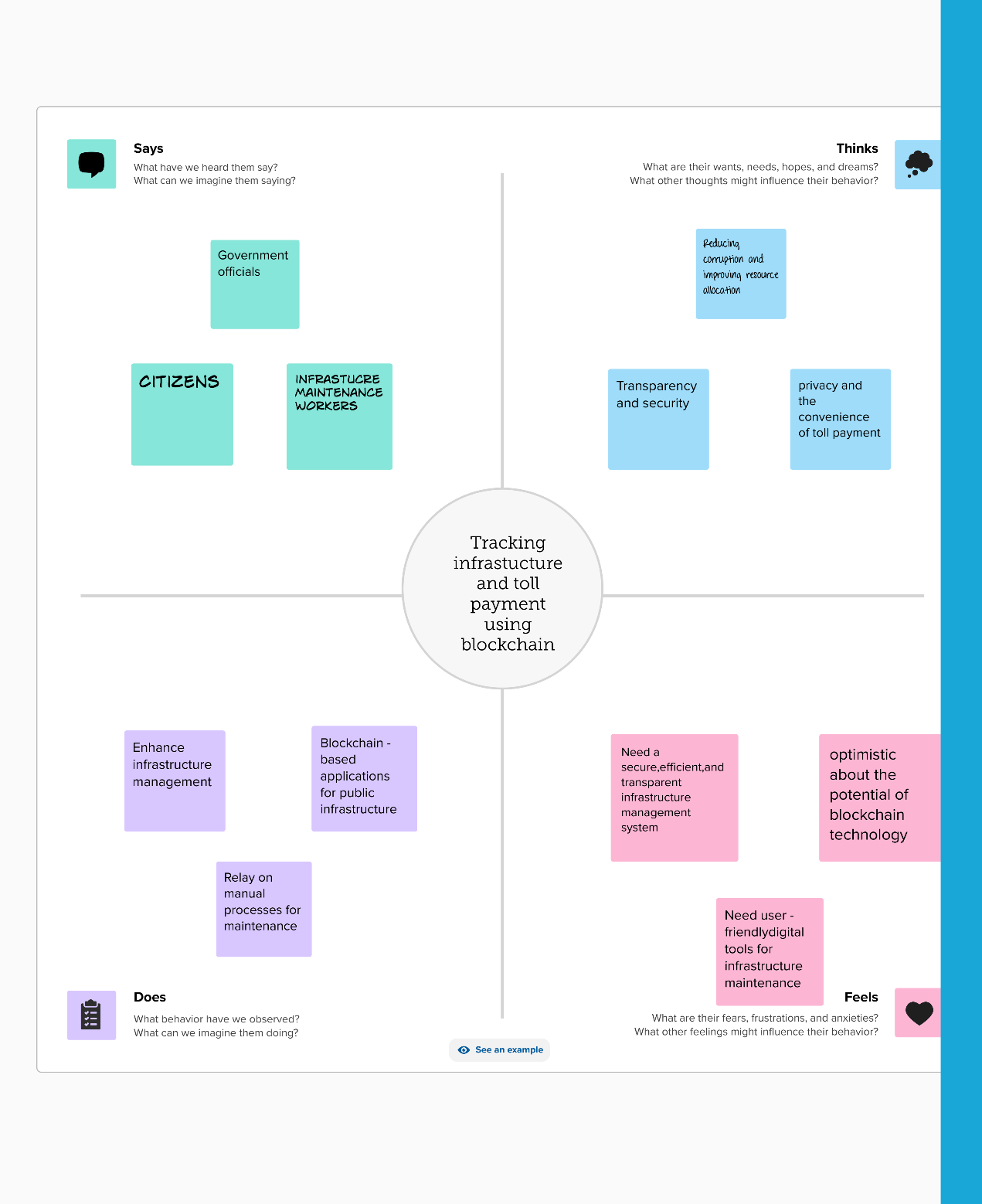
User Experience: To enhance the user experience by reducing congestion, minimizing wait times at toll booths, and providing convenient and secure payment options.

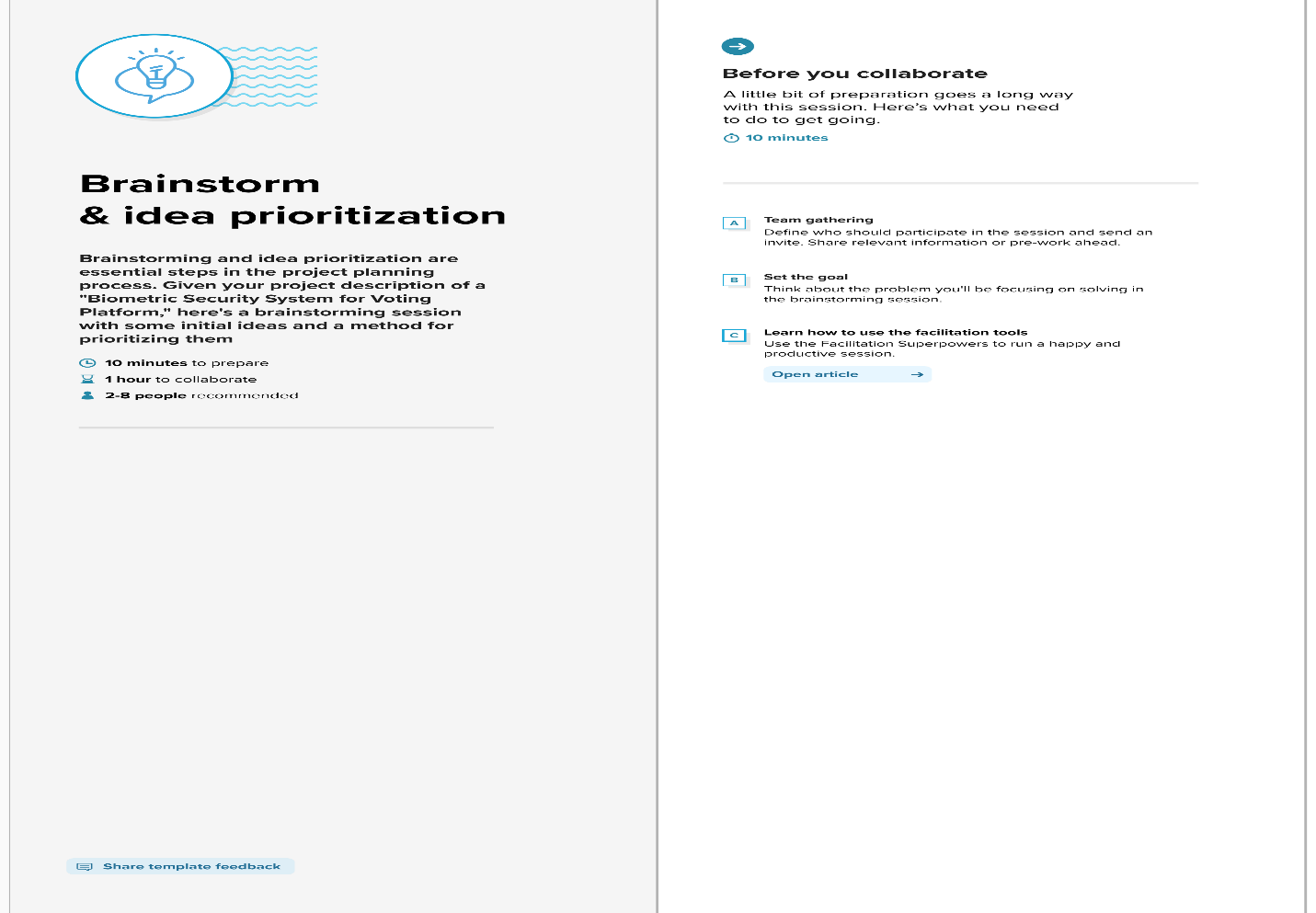
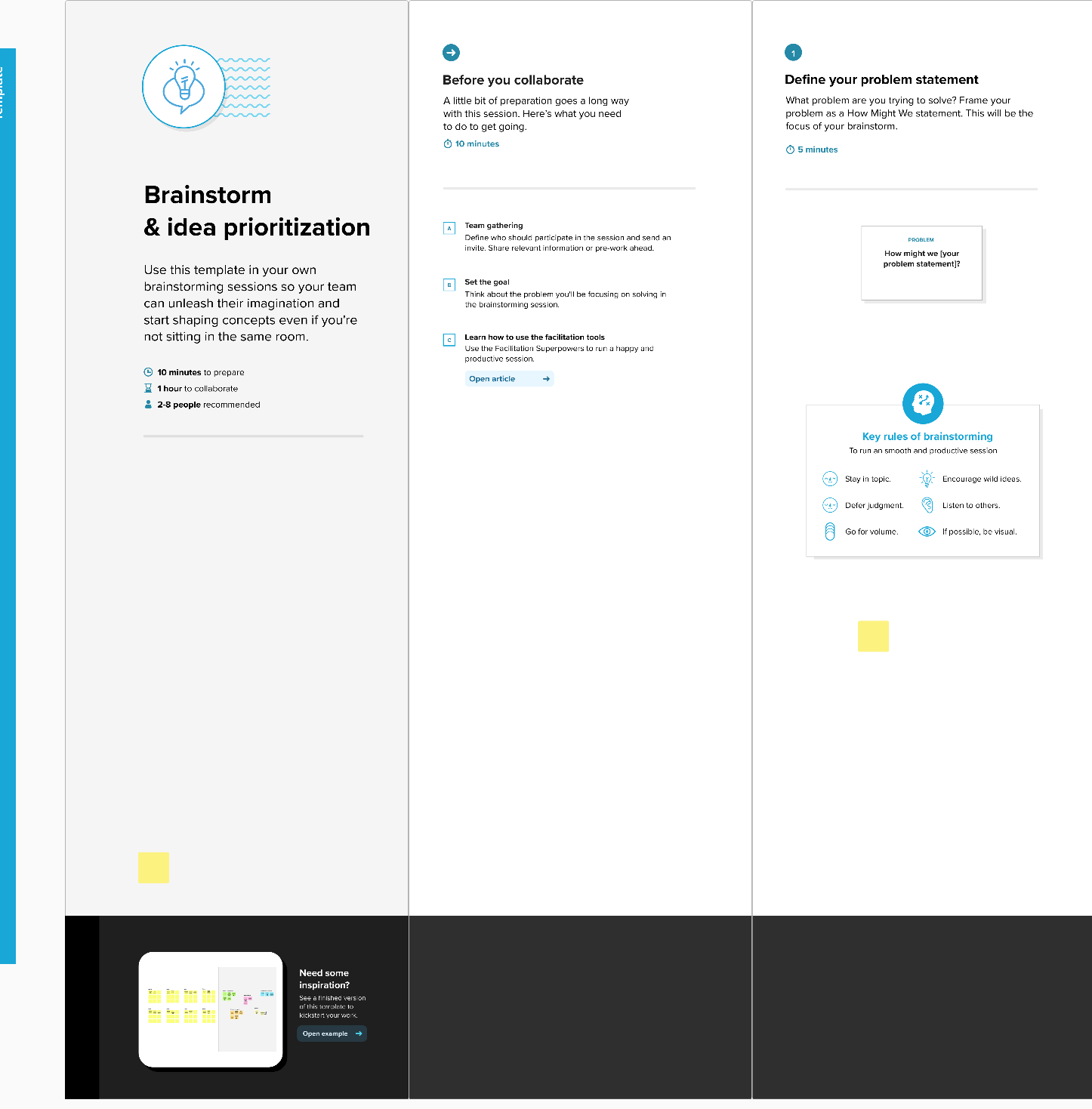
Security and Privacy: To ensure the security and privacy of user data, including personal information and financial transactions, while maintaining the integrity and availability of the infrastructure tracking system.

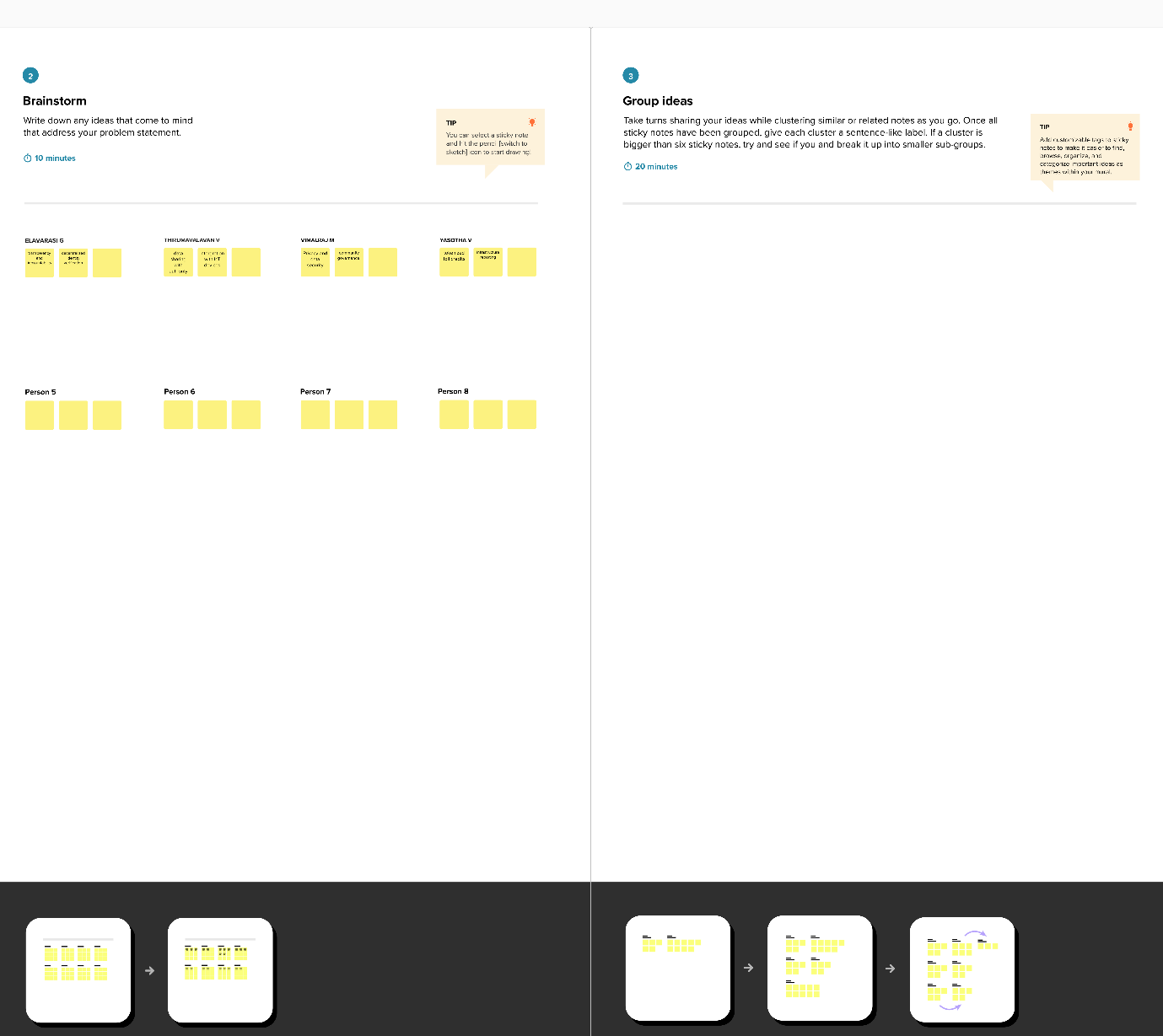
Cost Efficiency: To optimize the overall cost of infrastructure maintenance and toll collection by reducing operational expenses, minimizing fraud and revenue leakage, and improving resource allocation.

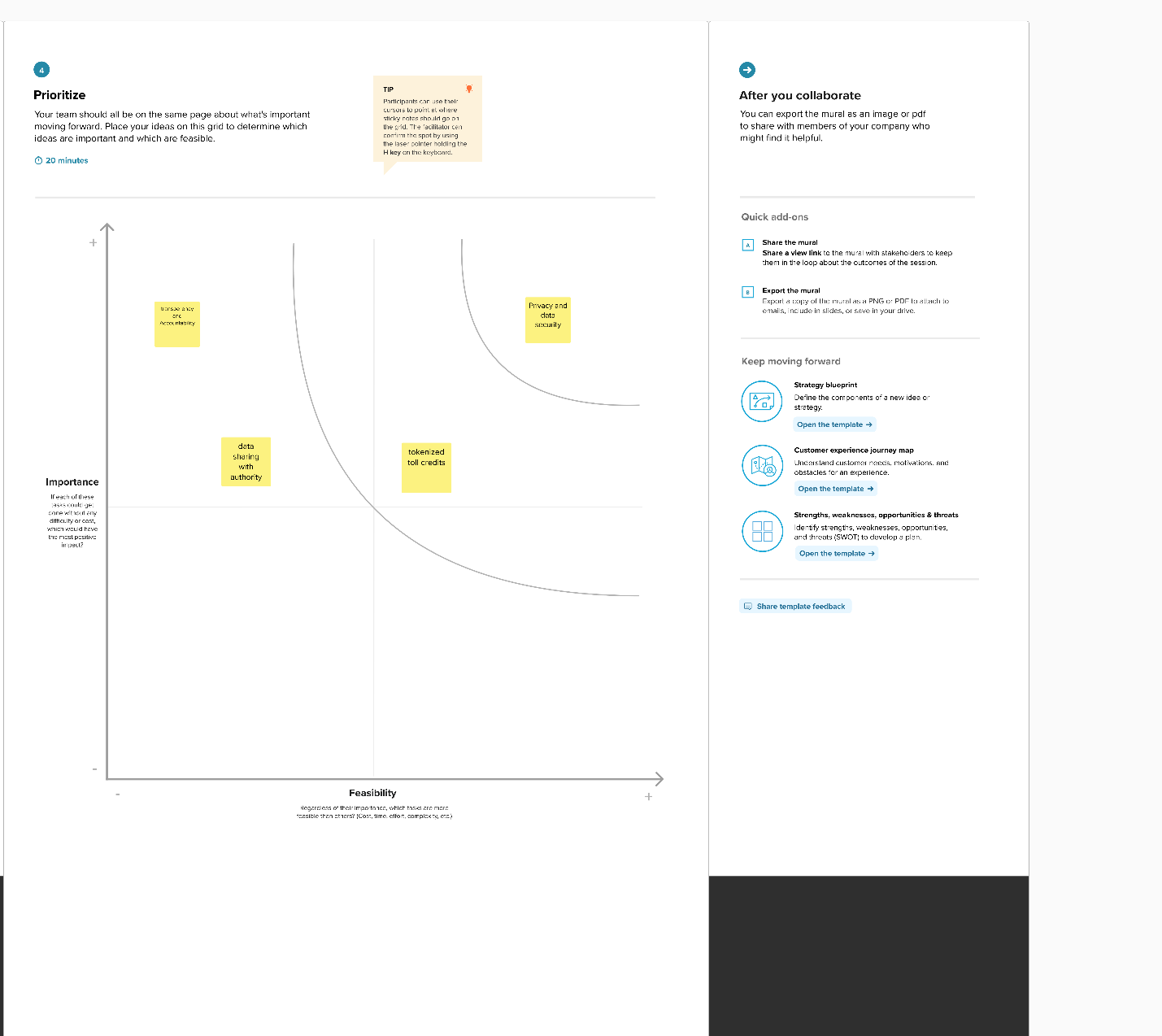
**3.IDEATION&PROPOSEDSOLUTION:**

**3.1EmpathyMapCanvas:**

****

**3.2Ideation&Brainstorming:**

****



# 4.REQUIREMENTANALYSIS:

# 4.1Functionalrequirement:

**User Registration and Enrollment**:

* The system must allow voter registration, capturing their biometric data.
* It should ensure the accurate enrollment of eligible voters.
* Capture and securely store personal voter information.

**Biometric Authentication:**

* Enable voters to authenticate themselves using their registered biometric data (e.g., fingerprint, iris scan, facial recognition).
* Verify the voter's identity during the authentication process.
* Provide immediate feedback on the success or failure of the authentication.

**Blockchain Integration**:

* Integrate with a blockchain platform (e.g., Ethereum, Hyperledger) for secure data storage and auditability.
* Store biometric data on the blockchain in a tamper-proof and encrypted format.
* Implement smart contracts for managing access to biometric data.

**Voting Process:**

* Allow authenticated voters to cast their votes electronically.
* Ensure that each voter can cast only one vote.
* Record votes on the blockchain with a time-stamp for transparency and immutability.

**Privacy and Data Control:**

* Empower voters to control who can access their biometric data through blockchain-based permissions.
* Comply with data protection regulations, such as GDPR, to protect voter privacy.
  1. **Non-Functionalrequirements:**

**Security:**

* The system must ensure a high level of security to prevent unauthorized access, tampering, and data breaches.
* It should adhere to industry standards for cryptographic protocols and best practices for securing biometric data.

**Performance:**

* The system must be highly responsive, with quick response times for voter authentication and vote recording.
* It should be able to handle a large number of concurrent users during peak voting hours.

**Reliability:**

* The system should have minimal downtime and high availability during election periods.

**Scalability:**

* The system should be scalable to accommodate a growing number of registered voters and increasing system load.

**Auditability:**

* It must provide comprehensive audit trails, allowing for the transparent and verifiable tracking of all system activities.

**Compliance:**

* The system must adhere to all legal and regulatory requirements related to elections, data protection, and privacy.

**Usability:**

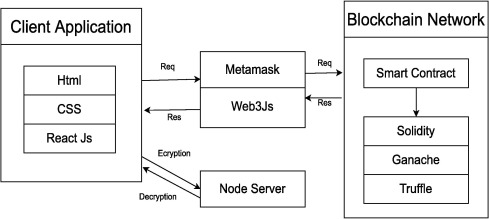
* The user interface should be intuitive and user-friendly to accommodate voters of varying technical proficiencies.

**Accessibility:**

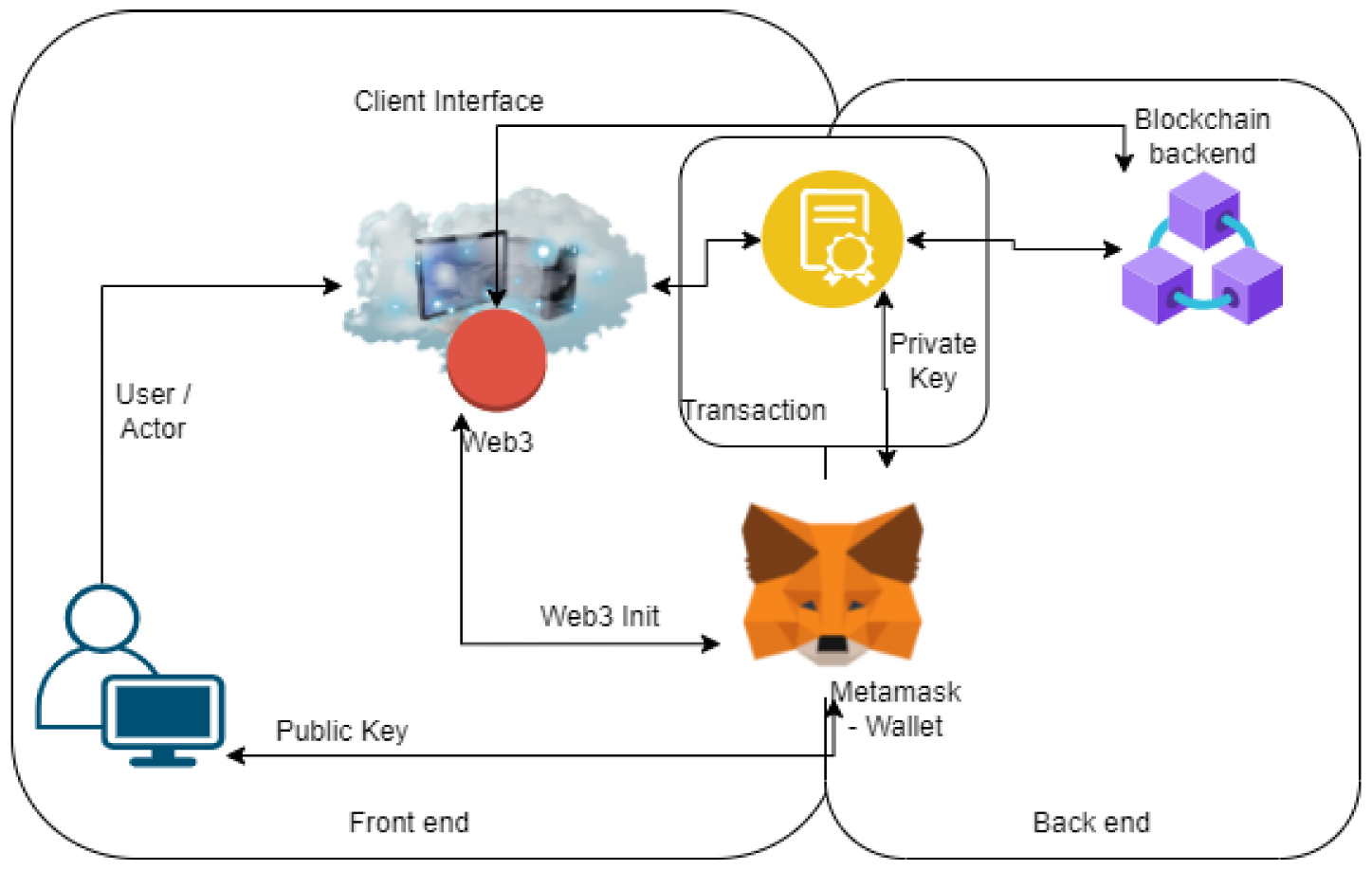
* The system should be designed to be accessible to individuals with disabilities, in compliance with accessibility standards.

# 5. PROJECTDESIGN:

* 1. **DataFlowDiagram:**

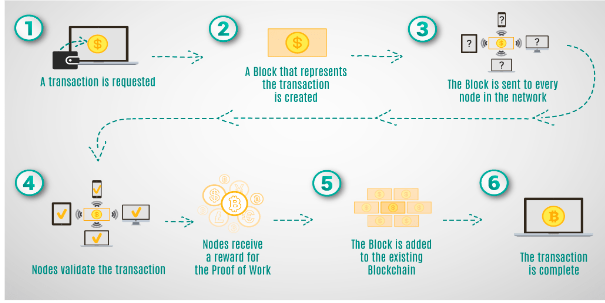


* 1. **SolutionArchitecture:**



# PROJECTPLANNING:

* 1. **TechnicalArchitecture:**

****

# 7.CODING&SOLUTIONING:

# Preliminary Definitions:

# // SPDX-License-Identifier: MIT: This is a license identifier that informs users about the license under which the contract code is available.

# pragma solidity ^0.8.0;: This specifies the version of the Solidity compiler to be used.

# Contract Definition:

# contract BallotBox { ... }: This defines a new smart contract named "BallotBox".

# State Variables:

# address public owner;: The Ethereum address of the contract's owner, usually the election authority.

# struct Voter {...}: This structure defines a voter, storing their encrypted biometric data and a flag indicating if they've already voted.

# struct Candidate {...}: This structure defines a candidate, storing their name and the number of votes received.

# string public electionName;: The name or title of the election.

# uint256 public registrationDeadline;: The timestamp for the deadline to register as a voter.

# uint256 public votingDeadline;: The timestamp for the deadline to cast a vote.

# Candidate[] public candidates;: An array that holds a list of candidates.

# mapping(address => Voter) public voters;: A mapping that links Ethereum addresses to their respective Voter data.

# Events:

# event VoteCast(address indexed voter, uint256 candidateIndex);: An event that will be emitted whenever a vote is cast. This helps external observers track when votes are made.

# Modifiers:

# modifier onlyOwner() {...}: A modifier that restricts certain functions to be callable only by the contract's owner.

# modifier canVote() {...}: A modifier that checks whether the current timestamp allows for voting and if the caller has not already voted.

# Constructor:

# constructor(...) {...}: When the smart contract is deployed, this constructor initializes the contract's state variables. It sets the owner, the election name, the registration and voting deadlines, and initializes the list of candidates.

# Functions:

# registerVoter(bytes32 \_encryptedBiometricData): This function allows an eligible voter to register, saving their encrypted biometric data. The canVote modifier ensures the voter registers before the registration deadline and hasn't voted yet.

# castVote(uint256 \_candidateIndex): Allows a registered voter to cast a vote for a specific candidate. The function checks if the voter is registered and hasn't voted already, then increments the chosen candidate's vote count and emits a VoteCast event.

# Key Concepts:

# The contract uses the Ethereum address of each participant (voter) as a unique identifier.

# The biometric data is stored in encrypted form to maintain privacy and security.

**7.1Feature:**

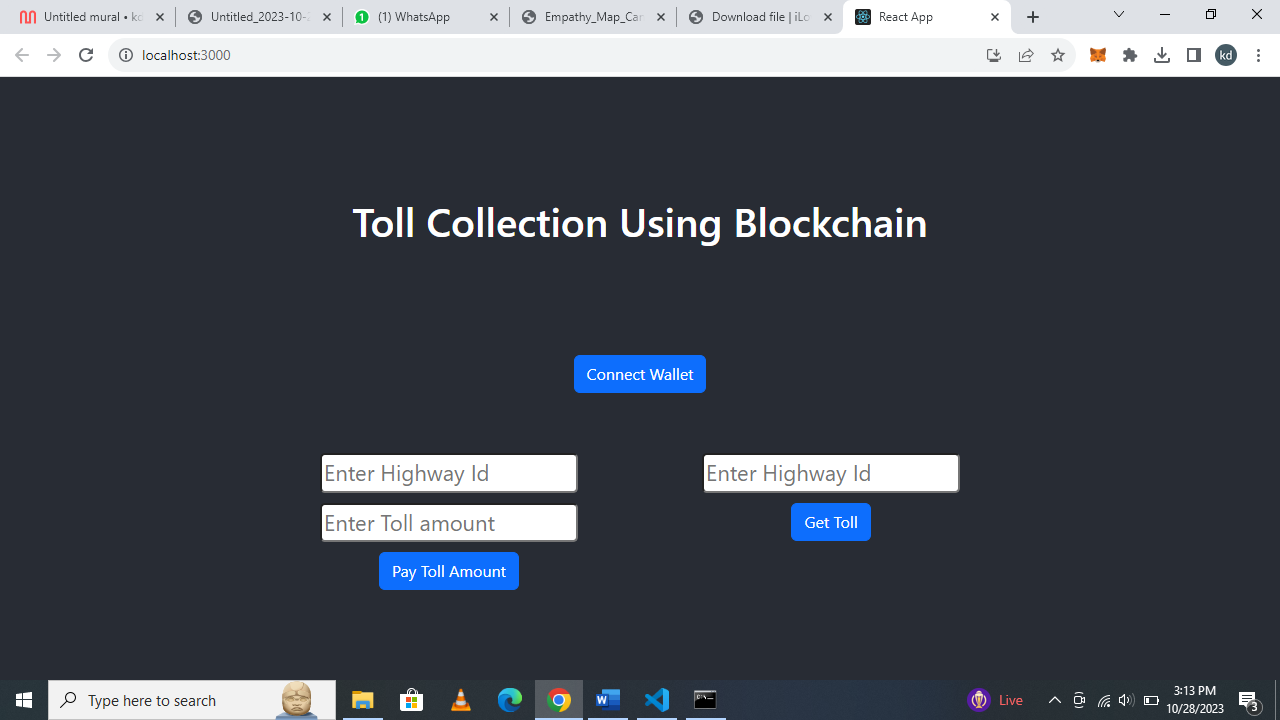
The system provides a robust biometric authentication mechanism, ensuring that only eligible voters can access the voting platform. It leverages fingerprint, iris, or facial recognition data for authentication, enhancing the security and integrity of the voting process while safeguarding voter privacy.

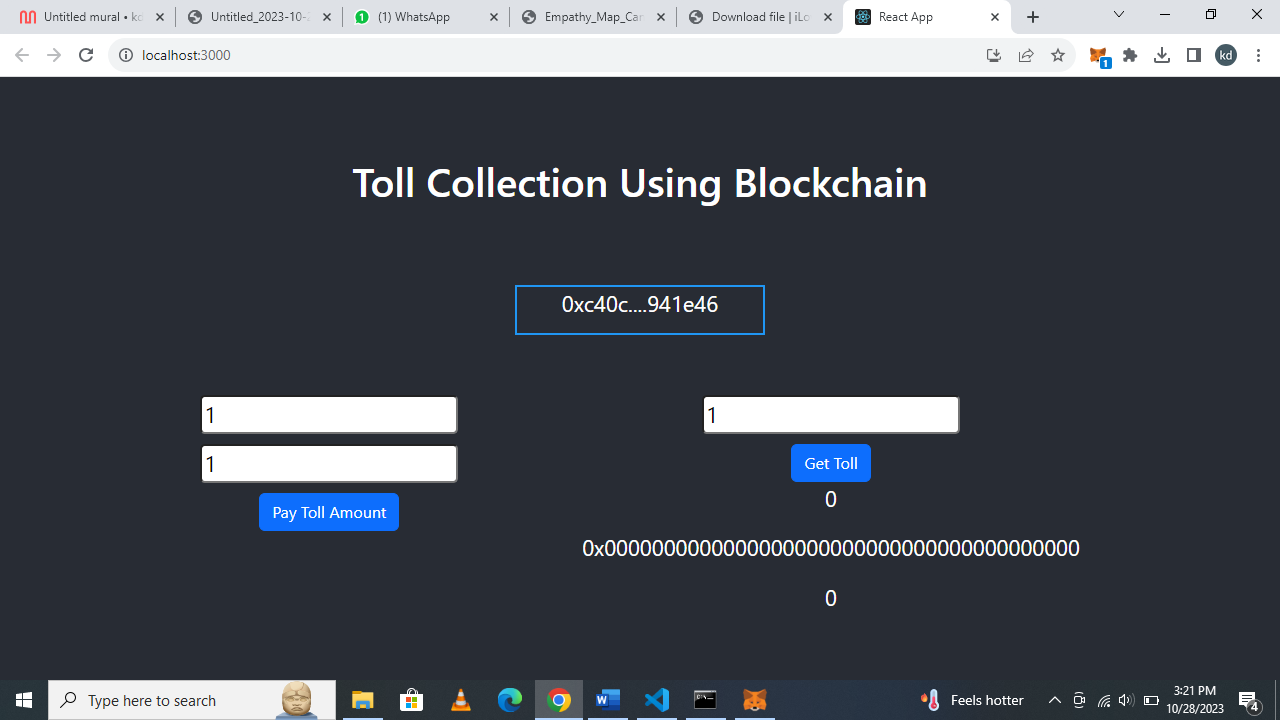
# 8.PERFORMANCETESTING

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
| 1. | Informationgathering | SetupallthePrerequisite: |  |
| 2. | Extractthezipfiles | Opentovscode |  |
| 3. | Remix Ide platformexplorting | Deploythesmartcontractcode  Deployandrunthetransaction.Byselecting the environment - injecttheMetaMask. |  |
| 4 | Openfileexplorer | Opentheextractedfileandclickonthefolder.  Opensrc,andsearchforutiles.  Open cmd enter commands  1.npminstall   1. Npm installbootstrap 2. npmstart |  |
| 5 | {LOCALHOST IPADDRESS | copytheaddressandopenittochromesoyoucanseethe frontendofyourproject. |  |

# 9.RESULTS:

**9.1OutputScreenshots:**

****

****

# 10.ADVANTAGES&DISADVANTAGES:

# 10.1 Advantages:

* **Enhanced Security:** Biometric authentication significantly reduces the risk of voter impersonation and fraud. Blockchain technology ensures tamper-proof and transparent recording of votes, making it extremely secure.
* **Transparent and Trustworthy:** The use of blockchain provides an immutable ledger that records all actions in the voting process, fostering trust and transparency in the electoral system.
* **Voter Privacy:** Biometric data and voting records can be stored securely on the blockchain while allowing voters to control who can access their data, protecting their privacy.
* **Inclusivity:** Biometric authentication can make voting more accessible for individuals with disabilities, as it doesn't rely on traditional paper-based identification methods.
* **Reduced Costs:** Over time, blockchain-based systems can potentially reduce costs associated with physical ballots, printing, and manual counting.
  1. **Disadvantages:**

## **Initial Implementation Costs:** Developing and implementing a biometric and blockchain-based voting system can be expensive, requiring significant upfront investment.

## **Complexity:** Combining biometrics with blockchain is a complex endeavor that demands expertise in both areas, which can pose challenges for development and maintenance.

## **Security Concerns:** While blockchain is highly secure, it's not immune to vulnerabilities. There can be concerns about the security of biometric data and blockchain networks against sophisticated attacks.

**11.CONCLUSION:**

The "Biometric Security System for Voting Platform with Blockchain Integration" project presents a cutting-edge solution to address critical challenges in the current voting systems. By combining biometric authentication and blockchain technology, it ensures robust voter verification, enhances election security, and fosters transparency and trust in the electoral process.

While offering significant advantages in terms of security and privacy, the project also faces challenges related to technology adoption, cost, complexity, and regulatory compliance. A successful implementation would mark a significant step toward revolutionizing the integrity and accessibility of elections.

# FUTURESCOPE:

# The project's future scope includes the potential for wider adoption of biometric- and blockchain-based voting systems, contributing to more secure, transparent, and accessible elections worldwide. This technology could evolve to accommodate remote and mobile voting, offering a more inclusive and convenient approach to civic participation while maintaining the integrity of the electoral process. Additionally, ongoing research and development in blockchain and biometrics may lead to enhanced security, scalability, and usability, further strengthening the foundation for future electoral systems.

1. **APPENDIX**

**13.1SourceCode:**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract tollCollection{

struct TollData {

uint timestamp;

address collectedBy;

uint amount;

}

mapping(address => mapping(uint => TollData)) public tolls;

function payTollAmount(uint highwayId, uint \_amount) public {

// TollData memory newToll = TollData(block.timestamp, msg.sender, amount);

tolls[msg.sender][highwayId].timestamp = block.timestamp ;

tolls[msg.sender][highwayId].collectedBy = msg.sender;

tolls[msg.sender][highwayId].amount += \_amount;

}

function getToll(uint highwayId) public view returns (TollData memory) {

return tolls[msg.sender][highwayId];

}

// function updateToll(uint highwayId, uint amount) public {

// require(

// tolls[msg.sender][highwayId].timestamp > 0,

// "Toll data not found."

// );

// tolls[msg.sender][highwayId].amount = amount;

// }

}

**13.2 GitHub&ProjectDemoLink**