INDEX

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. EXISTING SYSTEM

2.1 Existing Problem

2.2 Problem Statement Definition

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation and Brainstorming

4. REQUIRMENT ANALYSIS

4.1 Functional Requirements

4.2 Non-Functional Requirements

5. PROJECT DESIGN

5.1 Data Flow Diagram and User Stories

5.2 Solution Architecture

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

6.2 Sprint Planning and Estimation

6.3 Sprint Delivering

7. CODING AND SOLUTIONING

7.1 Feature 1

7.2 Feature 2

8. PERFORMANCE TESTING

8.1 Performance Metrics

9. RESULTS

9.1 Output Screenshots

10. ADVANTAGES AND DISADVANTAGES

10.1 Advantages

10.2 Disadvantages

11. CONCLUSION

12. FUTURE SCOPE

VACCINE TRACKING - TRANSPARENT

1. INTRODUCTION

1.1 PROJECT OVERVIEW

In today's ever-evolving global landscape, the Ethereum Vaccine Tracking system stands out as a truly groundbreaking and transformative solution, addressing a critical need for transparency in the distribution of vaccines. This innovative platform leverages the incredible power of the Ethereum blockchain to establish a secure and entirely transparent ecosystem for meticulously tracking the journey of vaccines, from their very inception in production to their ultimate administration. The use of blockchain technology in this context serves a dual purpose: not only does it safeguard the integrity of vaccine data, but it also instills a profound sense of trust in the vaccination process itself, resonating among stakeholders and the broader public alike. In a world where vaccine distribution has taken center stage in the face of unprecedented global challenges, the Ethereum Vaccine Tracking system presents a dependable and highly accountable approach to the recording and verification of vaccine-related information. Authorized users are granted access to real-time data, thereby empowering them to make well-informed decisions. This system carries the immense potential to revolutionize the way vaccines are distributed, underpinning and strengthening public health initiatives while paving the way for a future that is safer and more secure for all. As our world grapples with the ever-present challenges tied to vaccine distribution, this project emerges as a guiding light of transparency and reliability, offering a visionary solution to a global issue of paramount importance.

1.2 PURPOSE

The primary purpose of the Ethereum Vaccine Tracking system is to address the critical need for transparency and trust in the distribution of vaccines. In today's interconnected world, vaccines have become an essential tool in safeguarding public health, and their effective and equitable distribution is of paramount importance. This project aims to create a secure and transparent ecosystem that leverages blockchain technology to monitor every stage of the vaccine journey, from production facilities to end-users. By ensuring the integrity of vaccine data and providing real-time access to authorized stakeholders, this system fosters confidence in the vaccination process. It enables regulatory bodies, healthcare providers, and the general public to track and verify the authenticity of vaccines, reducing the risks of counterfeiting, mismanagement, and inequitable distribution. Moreover, the Ethereum Vaccine Tracking system can revolutionize vaccine distribution practices, making them more efficient, accountable, and responsive to the needs of various populations. Ultimately, this project's purpose is to strengthen public health initiatives, instill trust in vaccines, and contribute to a safer and more secure global future by ensuring that vaccines reach those who need them most while upholding the highest standards of transparency and accountability.

2. EXISTING SYSTEM

2.1 EXISTING PROBLEM

The prevailing issues within vaccine distribution paint a picture of opacity and inefficiency across the entire supply chain. This systemic challenge spans from the initial production of vaccines to their eventual administration to end-users, where a web of vulnerabilities and obstacles undermines the process. One of the core predicaments lies in the inadequacy of tracking and verification systems, resulting in a conspicuous lack of transparency regarding the authenticity and voyage of vaccines. This opacity in the system creates a breeding ground for counterfeit vaccines, mismanagement, and disparities in distribution that can significantly undermine public health efforts.

Furthermore, an additional facet of this predicament is the stark absence of real-time data accessibility for stakeholders. The inability to promptly access critical information profoundly hampers their capacity to make well-informed decisions, leading to delays and inefficiencies at various junctures. These issues have become acutely apparent and pressing during recent global health crises, none more so than the COVID-19 pandemic. The urgent need to establish an equitable and efficient vaccine distribution system, underpinned by robust transparency, is underscored by these challenges.

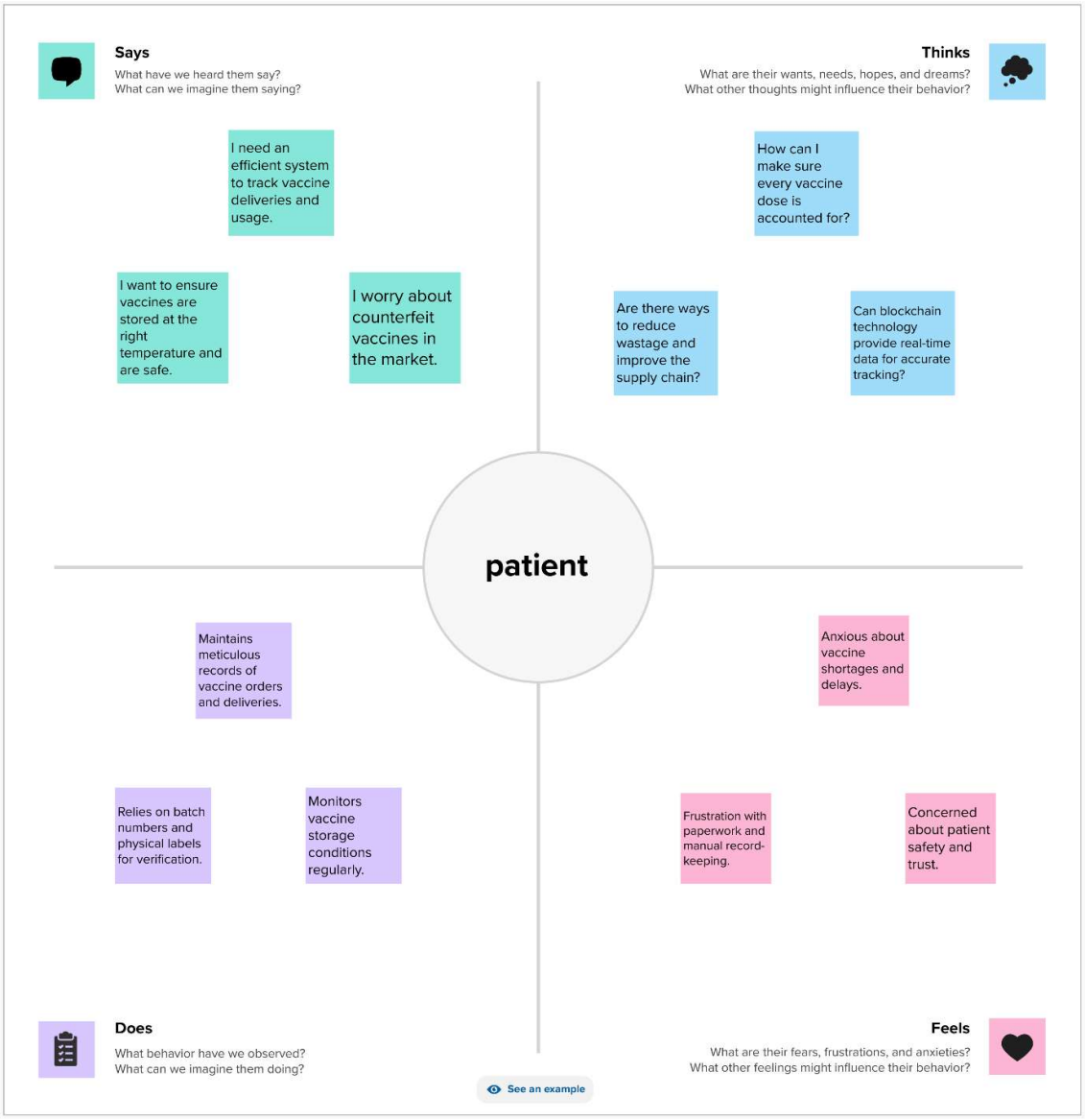
2.2 PROBLEM STATEMENT

The present-day landscape of vaccine distribution is marred by a glaring lack of transparency and pervasive inefficiencies that hinder the assurance of authentic and equitable vaccine allocation. This predicament stems from a multitude of challenges that permeate the entire vaccine supply chain, from production facilities to the eventual recipients. The deficiencies in tracking and verification systems are particularly noteworthy, contributing to the murkiness surrounding the origins and journey of vaccines. This opacity in the system serves as a fertile breeding ground for issues like counterfeit vaccines, supply chain

mismanagement, and glaring disparities in distribution, casting a pall on global public health endeavours. Adding to the complexity, stakeholders involved in vaccine distribution grapple with the onerous task of monitoring and verifying the complex supply chain, exacerbating issues related to counterfeit vaccines, mishandling, and the uneven dissemination of vaccines. The issue is further compounded by the notable absence of real-time data accessibility, which severely constrains the ability of decision-makers to operate with precision and timeliness. This problem takes on an acute urgency, particularly in the throes of global health crises, as was acutely demonstrated during the harrowing COVID-19 pandemic. The imperative for a prompt and effective solution becomes abundantly clear: a solution that can deliver transparency, security, and real-time monitoring capabilities within the vaccine distribution framework. Such a solution would stand as an embodiment of public health and safety enhancement, ensuring the efficient and equitable distribution of vaccines.

3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND BRAINSTORMING

Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

The challenge is to establish a secure, transparent, and trustworthy vaccine tracking system using Ethereum blockchain to enhance public confidence, ensure data integrity, and revolutionize vaccine distribution.

Need some inspiration?

See a finished version of this template to kickstart your work.

Open example →

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

HASSAN RIYAS

- Implement a secure, permissioned Ethereum blockchain for vaccine data storage.
- Develop a user-friendly web portal for vaccine stakeholders to access real-time data.
- Explore the use of smart contracts to automate vaccine tracking and alerts.

AKASH

- Implement robust encryption and access controls to protect sensitive patient information.
- Implement robust encryption and access controls to protect sensitive patient information.
- Explore options for pseudonymization to maintain privacy.

KISHORE

- Collaborate with national health agencies to ensure regulatory compliance and standardization.
- Implement a tamper-proof audit trail for regulators to monitor vaccine distribution.
- Explore ways to use the system for public health campaigns and crisis response

BALAJI

- Create an awareness campaign to educate the public about the benefits of blockchain in vaccine tracking.
- Establish a transparent communication channel for addressing public concerns and inquiries.
- Develop a user-friendly public-facing dashboard for vaccine transparency.

LOHITHKUMAR

- Ensure compatibility with existing healthcare record systems for seamless data integration.
- Establish protocols for secure data sharing and access permissions.
- Create a mobile app for healthcare professionals to scan and verify vaccines easily.

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Blockchain Infrastructure

- Implement a secure, permissioned Ethereum blockchain.
- Develop a user-friendly web portal for stakeholders.
- Utilize smart contracts for automation.

Data Privacy and Security

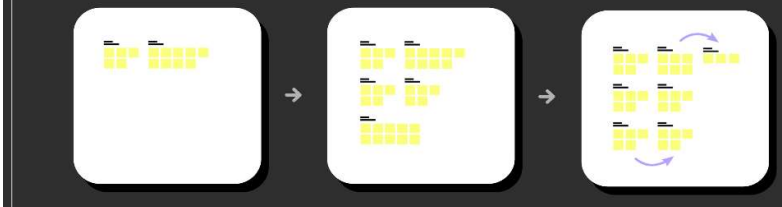
- Implement robust encryption and access controls.
- Develop mechanisms for patient consent and control.
- Explore pseudonymization options

Integration and Collaboration

- Ensure compatibility with existing healthcare record systems.
- Collaborate with national health agencies for regulatory compliance.
- Explore crisis response and public health campaign applications.

User-Friendly Tools

- Create a mobile app for healthcare professionals.
- Develop a public-facing dashboard for transparency.
- Establish transparent communication channels.



4

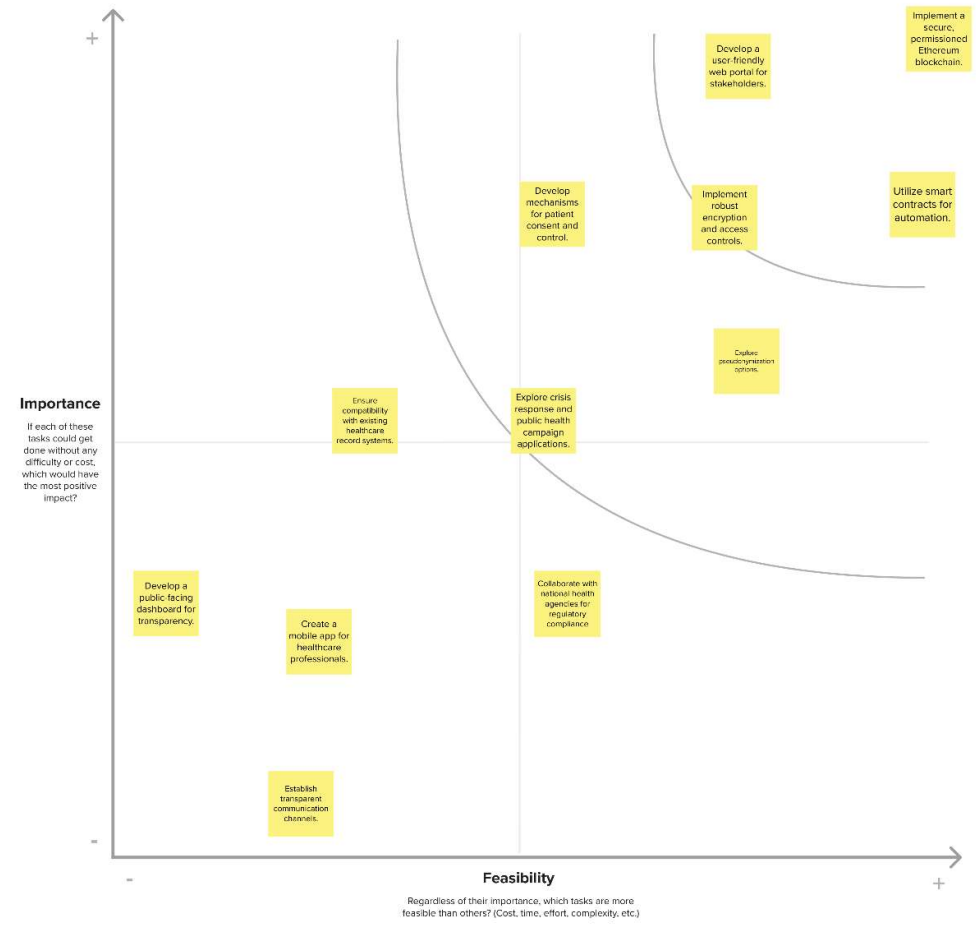
Prioritize

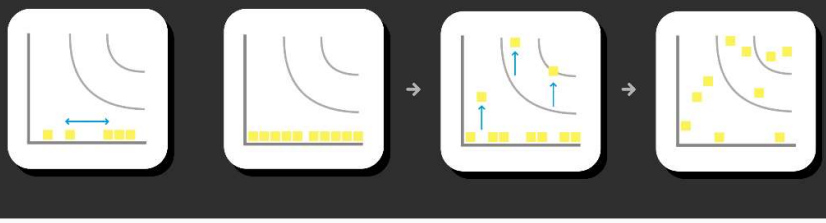
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.





4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

User Registration and Authentication: The system should enable users, including healthcare providers, regulatory bodies, and vaccine manufacturers, to securely register and authenticate themselves. This is crucial to ensure that only authorized individuals or organizations can access and interact with the vaccine tracking system.

User Wallet Integration: Users should have the capability to integrate digital wallets, such as Metamask, to sign transactions securely and manage any associated cryptocurrency. This feature supports the financial aspects of vaccine distribution and tracking.

User-Friendly Frontend: The frontend of the system should feature an intuitive and user-friendly interface. It should include buttons for actions like connecting the wallet, recording new vaccine information, updating existing data, and retrieving vaccine details. The interface should be designed for ease of use, allowing stakeholders to navigate and interact with the system without difficulty**.**

Record Vaccine Information: Users should be able to input vaccine-related data into the system. This includes fields for vaccine identification, vaccine name, description, batch numbers, and quantities. The system should validate and securely store this information on the blockchain.

Update Vaccine Information: Users must have the ability to update vaccine information, which may include modifying batch numbers, quantities, or other relevant details. This feature is crucial to ensure that the data remains accurate and up-to-date as vaccines move through the supply chain**.**

Query Vaccine Data: Users should be able to retrieve detailed information about a specific vaccine by entering its unique identification number. The system will then fetch and display the stored data, providing quick access to vital vaccine-related information.

Blockchain Smart Contract: The core of the project is an Ethereum blockchain-based smart contract, written in Solidity. It should handle functions such as recording vaccine data, querying data, and updating information in a secure and transparent manner.

Data Validation: The system should incorporate data validation mechanisms to ensure that only accurate and relevant vaccine information is stored. Invalid or inconsistent data should be rejected to maintain data quality.

Security Measures: Robust security measures, including encryption, digital signatures, and access controls, must be implemented to safeguard sensitive vaccine data. Only authorized users and entities should have access to specific information to protect the integrity of the system.

Error Handling: Comprehensive error-handling procedures should be in place to provide informative feedback to users and organizations in case of data entry or transaction errors. This ensures a smooth user experience and helps resolve issues promptly.

Compliance: The solution should adhere to relevant data privacy and regulatory requirements, ensuring that vaccine-related data is handled in a compliant and ethical manner. This is essential to maintain trust and legality in vaccine distribution.

Scalability and Performance: As the volume of vaccine data grows, the system should scale efficiently while maintaining optimal performance. It should be capable of handling an increasing number of users, vaccine records, and transactions without compromising on speed and responsiveness. This ensures the system's effectiveness as the vaccine distribution landscape evolves**.**

These functional requirements are pivotal in the development of a robust and user-friendly vaccine tracking transparency system, aligning with the project's overarching objectives of secure, efficient, and accessible vaccine data management on the blockchain.

4.2 NON - FUNCTIONAL REQUIREMENT:

Performance: The system must exhibit high performance, efficiently handling a substantial volume of simultaneous users and data transactions with minimal latency, ensuring that vaccine data is processed promptly**.**

Security: Robust security measures are imperative to safeguard against unauthorized access, data breaches, and potential security threats. The use of data encryption, secure authentication methods, and security protocols should be prioritized.

Reliability: The system should be highly reliable, minimizing downtime and data loss to maintain consistent operation. This is crucial to ensure uninterrupted access to vaccine information**.**

Scalability: The solution should be designed to scale seamlessly as the volume of vaccine data and the number of users grow over time, ensuring long-term effectiveness and responsiveness**.**

Usability: User interfaces, both web and mobile, should be intuitively designed to cater to individuals with varying levels of technical expertise. Accessibility and user-friendliness are key to promoting widespread adoption**.**

Data Privacy: Compliance with data privacy regulations is mandatory. The system must protect sensitive user and vaccine-related information, maintaining privacy and data security.

Auditability: The system should maintain a comprehensive audit trail, recording all data transactions and interactions with the blockchain smart contract. This feature fosters transparency and accountability**.**

Disaster Recovery: A robust disaster recovery plan is essential to ensure data recovery and system restoration in the event of unforeseen incidents or system failures, guaranteeing business continuity.

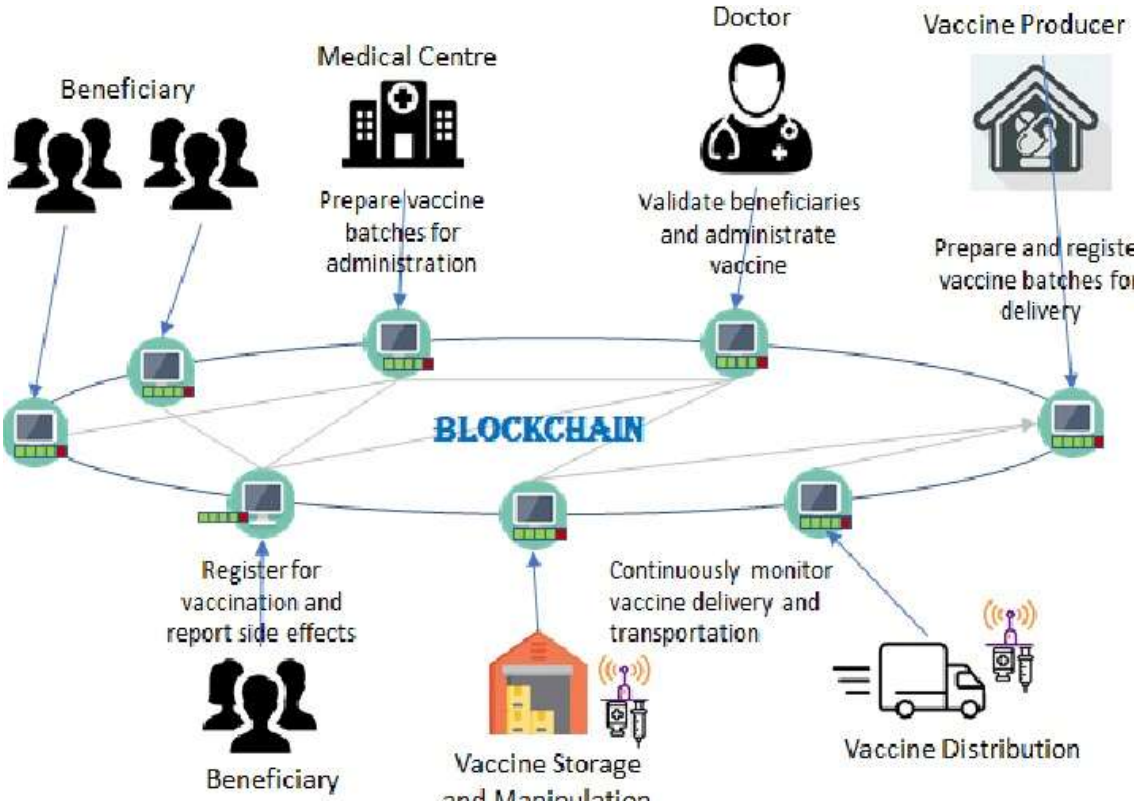
Cost-Effectiveness: The project should adopt a cost-effective approach, optimizing blockchain gas costs for transactions and operational expenses to ensure efficient resource utilization.

Interoperability: The system should be designed with interoperability in mind, enabling seamless data exchange and collaboration with other vaccine tracking and healthcare information systems to promote effective coordination among stakeholders.

Regulatory Compliance: Ensuring compliance with relevant vaccine distribution, healthcare, and blockchain-related regulations and standards is critical to establish trust among users and authorities. It is essential for maintaining transparency and adherence to legal requirements.

5. PROJECT DESIGN

5.1 DATAFLOW DIAGRAM AND USER STORY



STORY 1

As a healthcare provider, I want to utilize the Vaccine Tracking Transparency Blockchain system to ensure the secure and transparent tracking of vaccine distribution. Using a user-friendly web portal, I aim to connect my digital wallet to the system for secure access through a process similar to Metamask. Through the frontend interface, I need the capability to input essential vaccine details, including the vaccine's unique identification, name, description, and quantity in our inventory.

Moreover, it's crucial that I can easily update the vaccine records when necessary by providing the vaccine's unique ID and new information. This functionality is essential

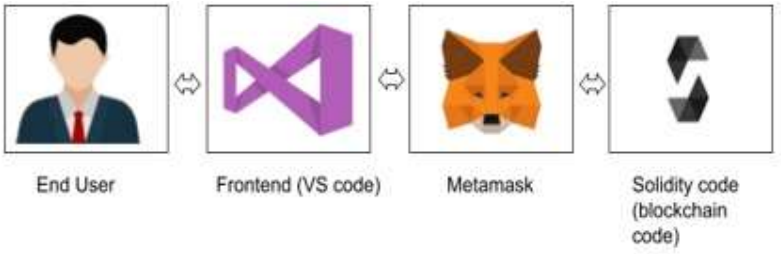
to maintaining the accuracy and real-time integrity of our vaccine records, which contributes to trust and accountability in the vaccination process. With this system, I can ensure that vaccines are administered securely and transparently, enhancing public health and safety.

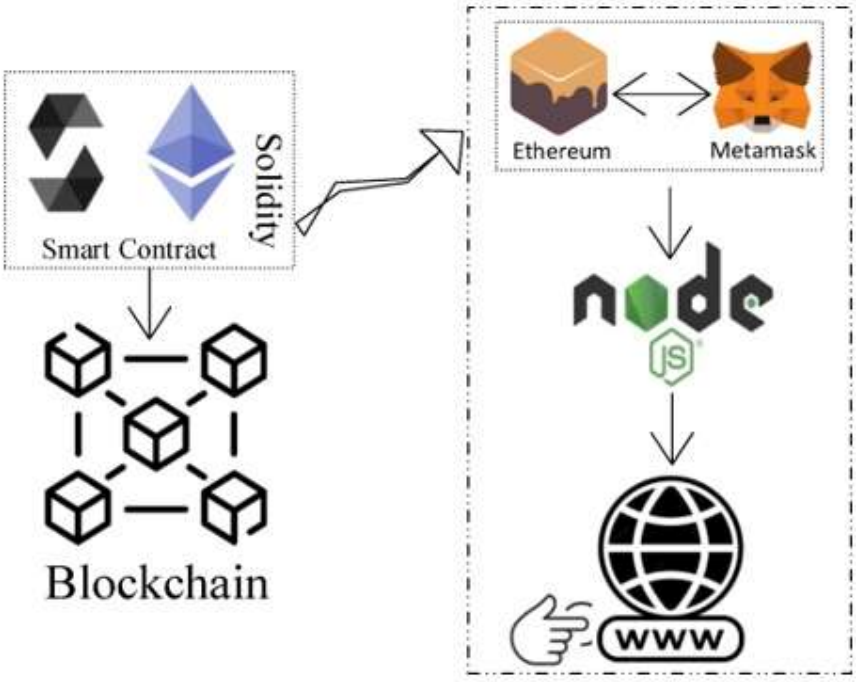
STORY 2

As a healthcare professional and a user of the Vaccine Tracking Transparency Blockchain system, I want the ability to retrieve vaccine details efficiently from the blockchain. To achieve this, I would use the system's user-friendly frontend interface, where I can enter the vaccine's unique ID and click the "Get Vaccine Details" button. This feature would provide me with instant access to comprehensive information about a specific vaccine, including its source, storage conditions, and administration history.

Having this functionality at my fingertips is crucial for making well-informed decisions regarding vaccine distribution, ensuring their safety and authenticity. Moreover, it simplifies the process of tracking a vaccine's journey, allowing me to verify their history quickly and efficiently. This ease of access empowers healthcare professionals to deliver vaccines with confidence, bolstering public trust in the vaccination process and ultimately promoting better public health outcomes.

5.2 SOLUTION ARCHITECTURE





Interaction between web and the Contract

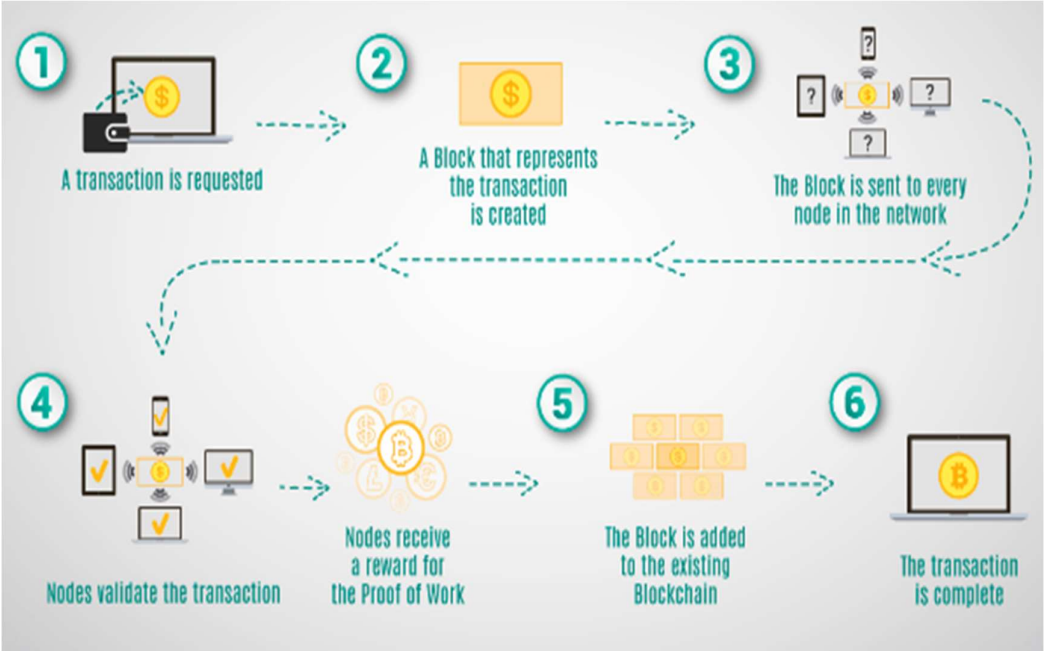
In the diagram, the end user is depicted at the top, using the web interface (Frontend) to access the system.

Metamask serves as the bridge between the user and the Ethereum blockchain, ensuring secure interactions with the smart contract developed in Solidity using the Remix IDE.

The Ethereum module connects to the Ethereum network, while Node.js and www (web servers) facilitate the communication between the frontend and the blockchain. This architecture ensures data security, accessibility, and reliability, aligning with the project's goals of solving agriculture data management challenges with a robust and user-friendly solution.

6. PROJECT PLANNING & SCHEDULING

6.1 TECHNICAL ARCHITECTURE



6.2 Sprint Planning and Estimation

Sprint planning plays a pivotal role in our project's agile development process, ensuring the efficient selection and commitment to work items from the product backlog for each sprint. Here's how we approach sprint planning and estimation in the context of vaccine tracking transparency:

Reviewing Product Backlog: Our project team, consisting of the product owner, scrum master, and development team, regularly evaluates the items in the product backlog. We carefully assess user stories and technical tasks, taking into account evolving requirements and project priorities.

Setting Sprint Goals: Sprint goals are established based on the product backlog. These goals serve as guiding principles for the sprint, ensuring that the team's efforts align with the broader objectives of the vaccine tracking transparency project.

Breaking Down User Stories: User stories and tasks are further deconstructed into smaller, actionable sub-tasks. This detailed breakdown is essential for creating a comprehensive sprint plan, ensuring that all necessary steps are considered.

Estimating Work: Our development team utilizes agile estimation techniques to estimate the effort required for each task. Two primary techniques are employed:

Story Points: Tasks are assigned story point values, providing a relative measure of complexity and effort needed for completion. This allows us to gauge the scope and intricacy of the work compared to reference tasks.

T-Shirt Sizes: To provide a quick and high-level estimate of effort, tasks are categorized into t-shirt sizes, such as small, medium, and large. This approach simplifies estimation, especially for less complex tasks, and provides a rapid overview of the work involved.

Sprint Backlog: The selected user stories and tasks, along with their corresponding estimates, collectively form the sprint backlog. This backlog acts as the foundation for the work the team will undertake during the sprint, serving as a management and tracking tool to monitor progress effectively.

Estimation Techniques

Story Points: Story points serve as a relative measure of the complexity and effort required to complete tasks within the vaccine tracking transparency project. Tasks are assigned story point values based on their complexity when compared to reference tasks. This allows the team to gauge the relative intricacy of each task and estimate the effort needed accurately.

T-Shirt Sizes: To provide a quick and high-level estimate of effort, tasks in the vaccine tracking transparency project can be categorized into t-shirt sizes. These sizes, such as small, medium, and large, simplify the estimation process,

particularly for tasks of varying complexity. This approach offers a rapid and straightforward way to assess the effort required for each task and provides a broad overview of the work involved in the project.

6.3 Sprint Delivering Schedule

Week 1: Project Initiation and Setup**

- Set up the foundational blockchain infrastructure for the vaccine tracking system.
- Initialize the core smart contract on the Ethereum blockchain.
- Begin the implementation of user registration and authentication system.
- Outline the project's security framework and identify key measures.

Week 2: Functional Expansion and Refinement**

- Expand the smart contract to accommodate vaccine data recording and tracking.
- Implement real-time updates for vaccine status, allowing for dynamic monitoring.
- Enhance user authentication with multi-factor authentication (MFA) for heightened security.
- Develop initial components of the user dashboard, focusing on essential functionalities.

Week 3: System Finalization and Testing**

- Finalize the user dashboard with additional features for a comprehensive user experience.
- Implement minimal compliance checks to ensure alignment with relevant regulations.
- Conduct basic testing to identify and resolve issues, optimizing system performance.
- Create essential user support resources, including documentation and user guides.

This sprint delivery schedule provides a well-structured plan for the project development team, guiding them through the critical tasks and milestones over a three-week period. It ensures that the vaccine tracking transparency project progresses efficiently and systematically.

7. CODING AND SOLUTIONING

7.1 FEATURE 1

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Vaccination {
    address public owner;

    constructor() {
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can perform this action");
        _;
    }

    struct Vaccine {
        string vaccineName;
        string manufacturer;
        uint256 manufacturingDate;
        string batchNumber;
        uint256 quantity;
        address customerAddress;
    }
```

```
    mapping(uint256 => Vaccine) public vaccines;
    uint256 public vaccineCount;

    event VaccineAdded(uint256 indexed vaccineId, string vaccineName, string
manufacturer, uint256 manufacturingDate, string batchNumber, address
customerAddress);

    function addVaccine(uint256 vaccineId, string memory _vaccineName, string
memory _manufacturer, uint256 _manufacturingDate, string memory
_batchNumber,uint256 _qty, address _customerAddress) external onlyOwner {



        vaccines[vaccineId] = Vaccine(_vaccineName, _manufacturer,
_manufacturingDate, _batchNumber, _qty, _customerAddress);
        vaccineCount++;

        emit VaccineAdded(vaccineId, _vaccineName, _manufacturer,
_manufacturingDate, _batchNumber, _customerAddress);
    }

    function getVaccineDetails(uint256 _vaccineId) external view returns (string
memory, string memory, uint256, string memory,uint256, address) {

        Vaccine memory vaccine = vaccines[_vaccineId];
        return (vaccine.vaccineName, vaccine.manufacturer,
vaccine.manufacturingDate, vaccine.batchNumber, vaccine.quantity,
vaccine.customerAddress);
    }
}
```

The provided Solidity smart contract, named "Vaccination," serves as a foundation for a blockchain-based vaccine tracking system. The contract is equipped with an owner variable, allowing the contract owner to have exclusive access rights. Within the

contract, a struct named "Vaccine" is defined to encapsulate key information about each vaccine, including its name, manufacturer, manufacturing date, batch number, quantity, and the associated customer's address. These details are stored in a mapping, associating each vaccine with a unique ID. The contract features functions to add vaccine records, restricted to the owner using the "onlyOwner" modifier, and retrieve vaccine details based on the vaccine's ID. An event, "VaccineAdded," is emitted when a new vaccine is added to the system, capturing essential information. This smart contract provides a fundamental framework for managing and retrieving vaccine information securely and transparently on the Ethereum blockchain.

7.2 FEATURE 2

FRONTEND *(JAVASCRIPT)*

```
import React, { useState } from "react";
import { Button, Container, Row, Col } from 'react-bootstrap';
import 'bootstrap/dist/css/bootstrap.min.css';
import { contract } from "./connector";

function Home() {
  const [Id, setId] = useState("");
  const [DrugName, setDrugName] = useState("");
  const [Manufacturer, setManufacturer] = useState("");
  const [date, setDate] = useState("");
  const [TranId, setTranId] = useState("");
  const [Owner, setOwner] = useState("");
  const [BookId, setBookId] = useState("");
  const [BookDet, setBookDet] = useState("");
  const [Batch, setBatch] = useState("");
  const [Qty, setQty] = useState("");
  const [Cus, setCus] = useState("");
  const [Wallet, setWallet] = useState("");
```

```
  const handleId = (e) => {
    setId(e.target.value)
  }

  const handleVaccineName = (e) => {
    setDrugName(e.target.value)
  }

  const handleManufacturer = (e) => {
    setManufacturer(e.target.value)
  }

  const handleDate = (e) => {
    setDate(e.target.value)
  }

  const handleBatch = (e) => {
    setBatch(e.target.value)
  }

  const handleQty = (e) => {
    setQty(e.target.value)
  }

  const handleCusAddr = (e) => {
    setCus(e.target.value)
  }


  const handleAddVaccine = async () => {
    try {
      let tx = await contract.addVaccine(Id.toString(), DrugName, Manufacturer, date,
Batch, Qty, Cus)
      let wait = await tx.wait()
```

```
      alert(wait.transactionHash)
      console.log(wait);
    } catch (error) {
      alert(error)
    }
}

const handleDrugId = (e) => {
  setTranId(e.target.value)
}

const handleNewOwner = (e) => {
  setOwner(e.target.value)
}

const handleTransfer = async () => {
  try {
    let tx = await contract.transferDrugOwnership(TranId.toString(), Owner)
    let wait = await tx.wait()
    console.log(wait);
    alert(wait.transactionHash)
  } catch (error) {
    alert(error)
  }
}

const handleVaccineDetailsId = (e) => {
  setBookId(e.target.value)
}

const handleDrugDetails = async () => {
  try {
    let tx = await contract.getVaccineDetails(BookId.toString())
```

```
      let arr = []
      tx.map(e => {
        arr.push(e)
      })

      console.log(tx);
      setBookDet(arr)
    } catch (error) {
      alert(error)
      console.log(error);
    }
  }

  const handleWallet = async () => {
    if (!window.ethereum) {
      return alert('please install metamask');
    }

    const addr = await window.ethereum.request({
      method: 'eth_requestAccounts',
    });

    setWallet(addr[0])

  }

 return (
  <div>
   <h1 style={{ marginTop: "30px", marginBottom: "80px" }}>Vaccination</h1>
     {!Wallet ?

       <Button onClick={handleWallet} style={{ marginTop: "30px", marginBottom:
"50px" }}>Connect Wallet </Button>
       :
```

```
      <p style={{ width: "250px", height: "50px", margin: "auto", marginBottom: "50px", border: '2px solid #2096f3' }}>{Wallet.slice(0, 6)}....{Wallet.slice(-6)}</p>
    }
  <Container>
   <Row>
    <Col style={{marginRight:"100px"}}>
     <div>


    <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleId} type="number" placeholder="Enter vaccine Id" value={Id} />
<br />

    <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleVaccineName} type="string" placeholder="Enter vaccine Name" value={DrugName} /> <br />

    <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleManufacturer} type="string" placeholder="Enter vaccine manufacturer" value={Manufacturer} /><br />

    <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleDate} type="number" placeholder="Enter vaccine manufacturing date" value={date} /><br />

        <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleBatch} type="string" placeholder="Enter Batch No" value={Batch} /><br />

        <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleQty} type="number" placeholder="Enter Quantity" value={Qty} /><br />
```

```
            <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleCusAddr} type="string" placeholder="Enter Customer Address"
value={Cus} /><br />


     <Button onClick={handleAddVaccine} style={{ marginTop: "10px" }}
variant="primary">Add vaccine</Button>
     </div>
    </Col>
          <Col >
            <div style={{ margin: "auto" }}>
              <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleVaccineDetailsId} type="number" placeholder="Enter Drug Id"
value={BookId} /><br />

              <Button onClick={handleDrugDetails} style={{ marginTop: "10px" }}
variant="primary">Get vaccine Details</Button>
              {BookDet ? BookDet?.map(e => {
                return <p>{e.toString()}</p>
              }) : <p></p>}
            </div>
          </Col>
   </Row>
   </Container>

  </div>
 )
}

export default Home;
```

The code is a React.js component named "Home" that serves as a user interface for interacting with a blockchain-based vaccination system. Here are the key details of this code:

Import Statements:

The code imports the necessary modules, such as React, useState for managing state, Button, Container, Row, and Col components from the React Bootstrap library, and the 'contract' object from a 'connector' module.

Functional Component:

The 'Home' function is a functional component that represents the user interface for the vaccination system.

State Variables:

The component uses the useState hook to manage various state variables, including 'Id,' 'DrugName,' 'Manufacturer,' 'date,' 'TranId,' 'Owner,' 'BookId,' 'BookDet,' 'Batch,' 'Qty,' 'Cus,' and 'Wallet,' which store information and user input related to vaccine details, blockchain interactions, and wallet connections.

Event Handlers:

The component defines several event handler functions, such as handleId, handleVaccineName, handleManufacturer, handleDate, handleBatch, handleQty, handleCusAddr, handleAddVaccine, handleDrugId, handleNewOwner, handleTransfer, handleVaccineDetailsId, handleDrugDetails, and handleWallet. These functions manage user input and trigger blockchain interactions.

Blockchain Interactions:

The code includes functions like handleAddVaccine, handleTransfer, and handleDrugDetails that interact with the blockchain using the 'contract' object. These functions send transactions and retrieve data from the Ethereum blockchain related to vaccine details and ownership transfers.

Wallet Connection:

The code provides a "Connect Wallet" button that, when clicked, initiates a connection to the user's Ethereum wallet (e.g., MetaMask) by requesting account access. The user's wallet address is displayed once connected.

User Input Fields and Buttons:

The component renders input fields for various vaccine-related details, allowing the user to input data such as vaccine ID, name, manufacturer, manufacturing date, batch number, quantity, and customer address. Buttons are provided to trigger actions like adding a vaccine, transferring ownership, and retrieving vaccine details.

Display of Vaccine Details:

The component displays vaccine details returned from the blockchain in response to user queries, which include information about vaccine names, manufacturers, manufacturing dates, batch numbers, quantities, and customer addresses.

Overall, this code represents the frontend interface for a vaccination blockchain application, enabling users to interact with the Ethereum blockchain to manage vaccine data, transfer ownership, and retrieve vaccine details, all while ensuring wallet connectivity for secure transactions.

7. PERFORMANCE TESTING

8.1 PERFORMANCE METRICS

Transaction Throughput: Measure the number of vaccine-related transactions the system can handle per second. High throughput is essential for accommodating a large number of users and data interactions simultaneously, ensuring efficient vaccine tracking.

Latency: Track the time it takes for a vaccine-related transaction to be confirmed and recorded on the blockchain. Lower latency ensures a more responsive system, which is critical for real-time vaccine tracking**.**

Error Rates: Monitor the frequency of transaction failures or errors within the vaccine tracking system. Reducing error rates is crucial for ensuring the accuracy and reliability of vaccine data.

Scalability: Evaluate how the system scales as the volume of vaccine data and the number of users increase. Ensure that the system can handle growth without significant performance degradation, accommodating evolving vaccine distribution needs**.**

Resource Utilization: Assess the system's utilization of computational resources, memory, and network bandwidth. Efficient resource utilization helps optimize costs and maintain performance while minimizing resource wastage.

Gas Costs: Keep track of gas costs associated with vaccine-related transactions on the Ethereum blockchain. Optimizing gas costs is essential for cost-effective operation in the vaccine tracking system.

Response Time: Measure the time it takes for the frontend of the vaccine tracking system to respond to user actions. Lower response time leads to a more satisfying user experience, facilitating efficient vaccine data management**.**

Uptime and Availability: Calculate the percentage of time the vaccine tracking system is available and operational. High uptime ensures uninterrupted access to vaccine-related information for users and stakeholders.

Data Retrieval Speed: Evaluate how quickly the system can retrieve and display vaccine-related data to users. Faster data retrieval contributes to informed decision-making and more effective vaccine distribution**.**

Data Storage Efficiency: Analyze how efficiently the system stores vaccine-related data on the blockchain. Efficient data storage minimizes blockchain bloat, reduces costs, and ensures that vaccine data is managed optimally.

Security Audits and Vulnerabilities: Conduct regular security audits and penetration testing to identify and address vulnerabilities within the vaccine tracking system. Ensuring the safety of vaccine data and user information is paramount**.**

Compliance: Monitor compliance with data privacy and regulatory requirements in the context of vaccine tracking. Ensure that vaccine data is handled ethically and in accordance with relevant regulations, maintaining transparency and trust.

User Adoption and Engagement: Track user adoption rates and user engagement with the vaccine tracking system. High adoption and engagement indicate the system's value and effectiveness in facilitating vaccine distribution.

Response Time: Measures speed of addressing user feedback and support for vaccine tracking, enhancing user satisfaction and trust.

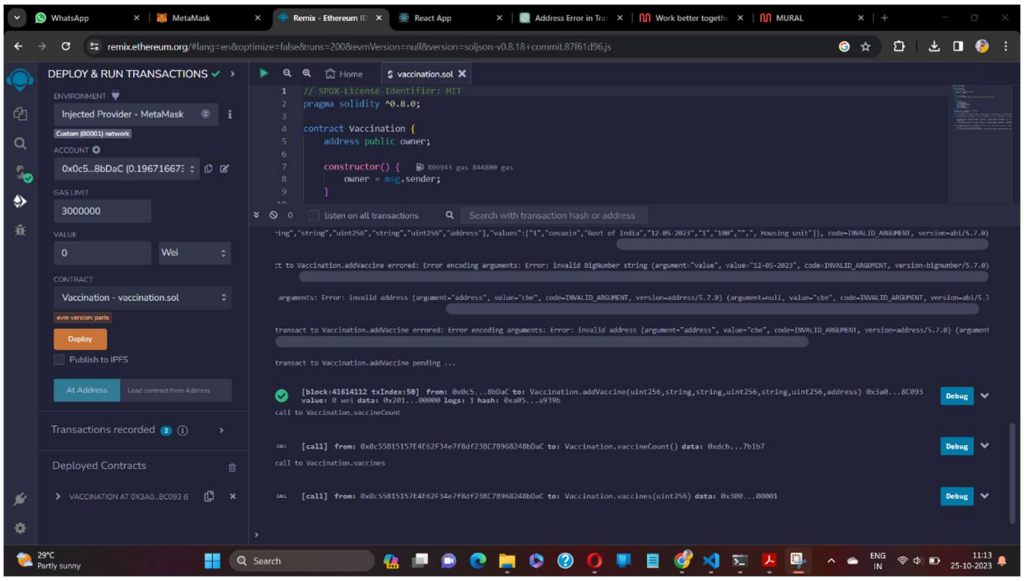
Documentation Quality: Evaluates the comprehensiveness of system documentation for effective onboarding and troubleshooting.

Regularly monitoring these performance metrics ensures the efficiency, effectiveness, and security of the vaccine tracking transparency project, meeting the evolving needs and expectations of users and stakeholders in vaccine distribution

9.RESULTS

9.1 OUTPUT SCREENSHOTS

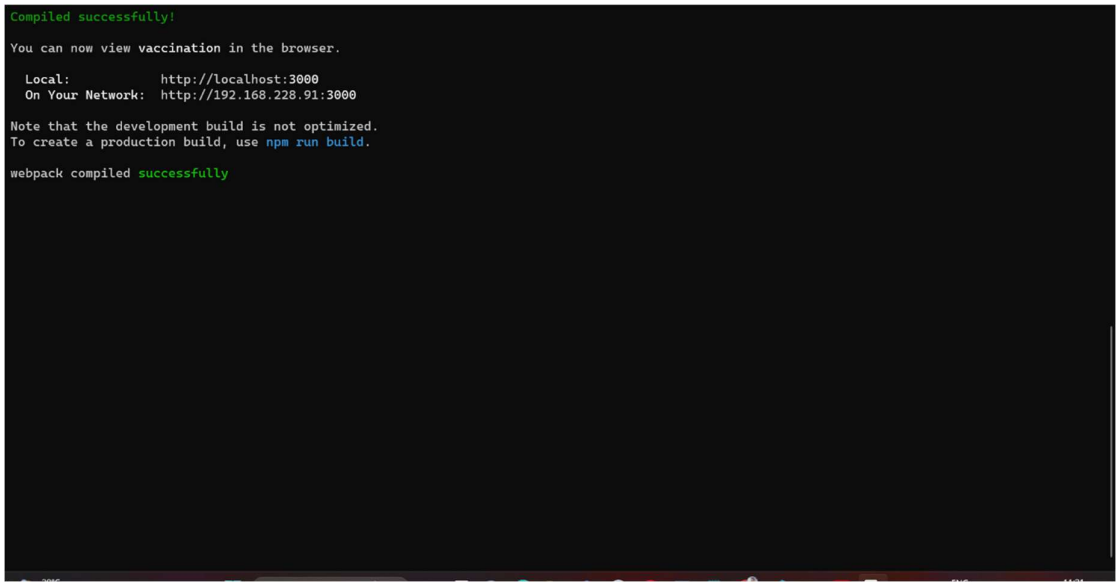
CREATING A SMART CONTRACT:



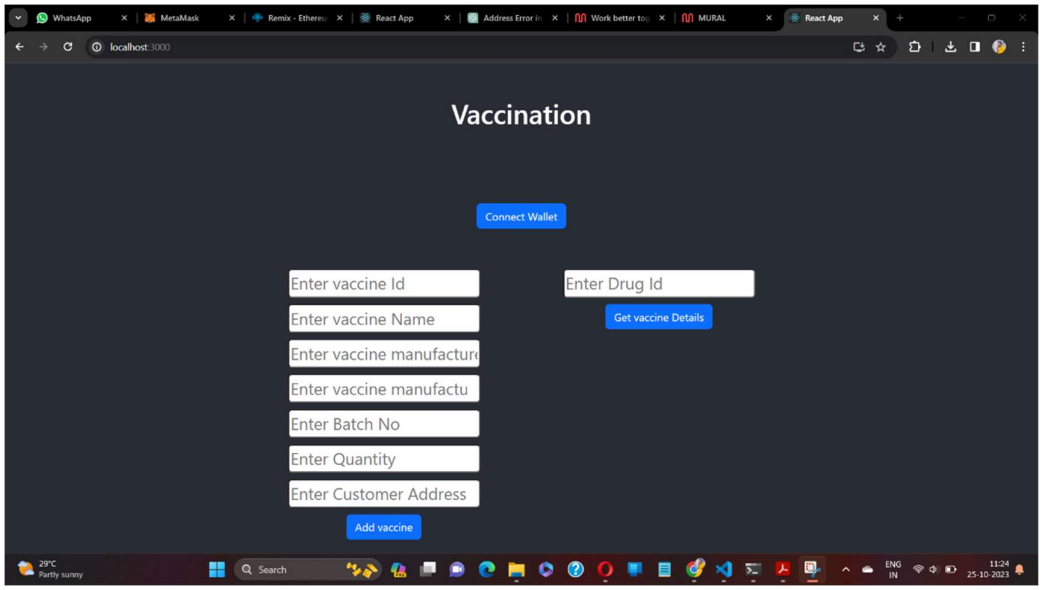
INSTALLING DEPENDENCY:

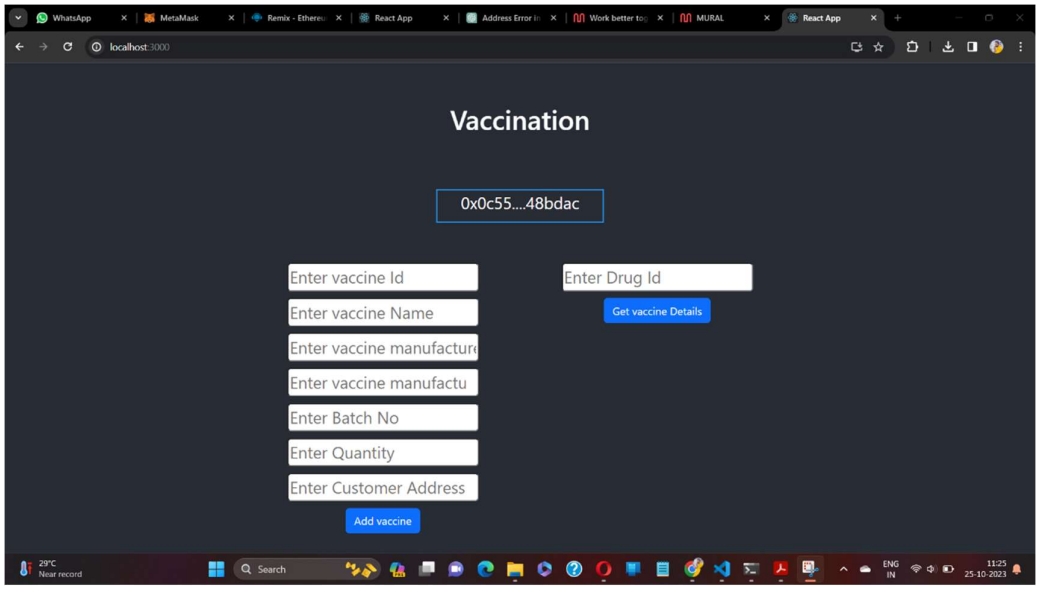
```
C:\Users\lohit\Downloads\10_Problem_Statement_10_ vaccination\vaccination\vaccination\src\Page>npm install

up to date, audited 1569 packages in 8s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

C:\Users\lohit\Downloads\10_Problem_Statement_10_ vaccination\vaccination\vaccination\src\Page>npm install bootstrap

up to date, audited 1569 packages in 4s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

C:\Users\lohit\Downloads\10_Problem_Statement_10_ vaccination\vaccination\vaccination\src\Page>
```

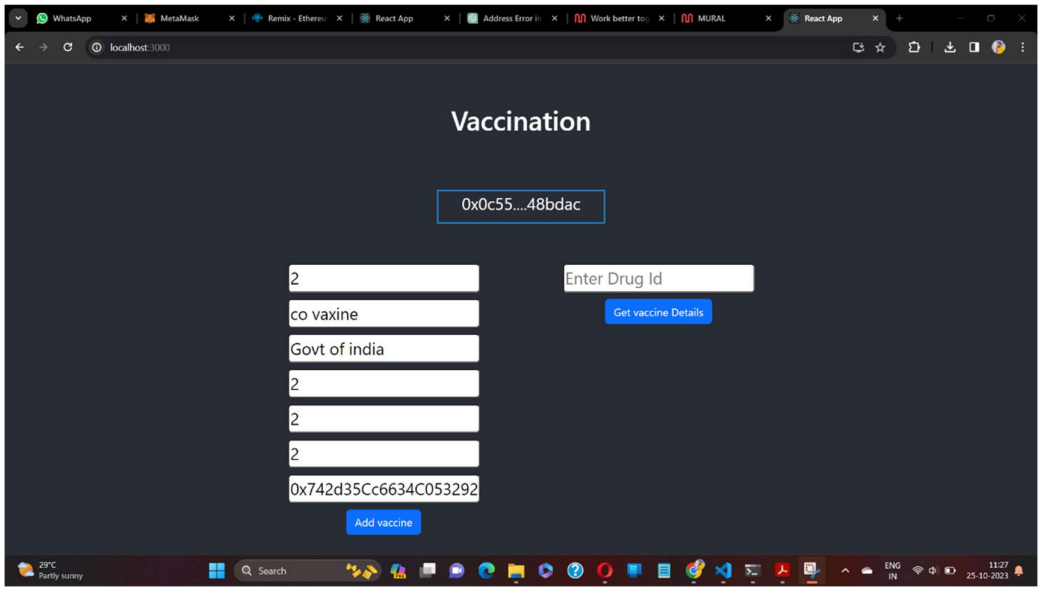
HOSTING THE SITE:

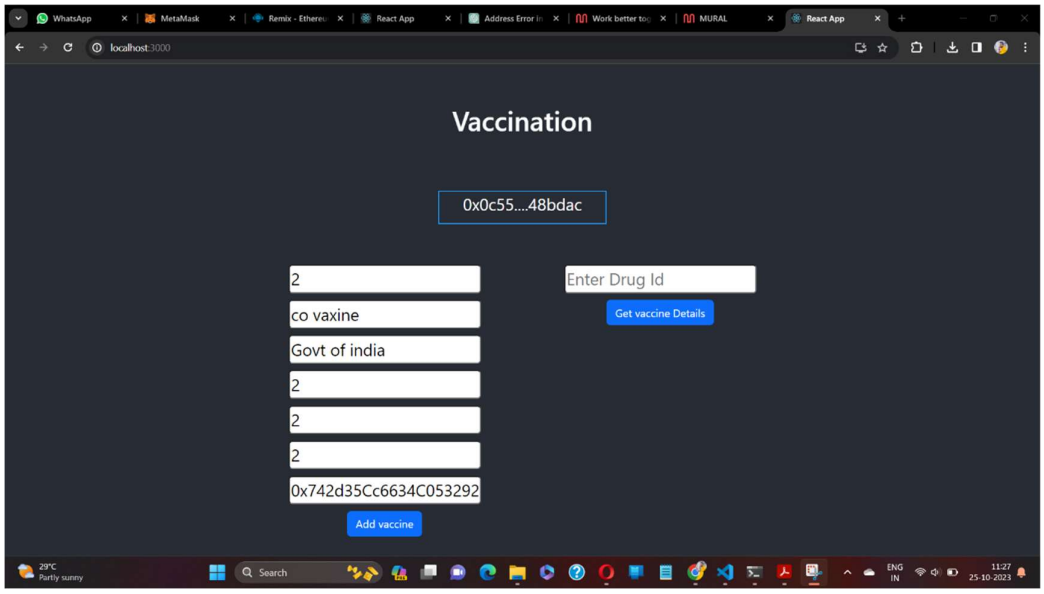


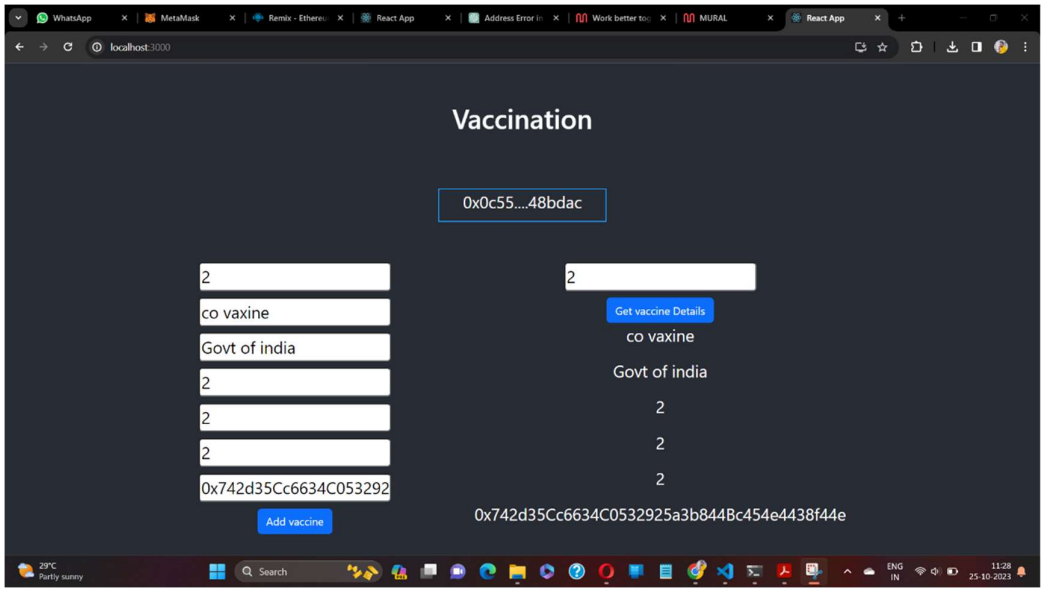
OUTPUT SCREENSHOT:




WhatsApp
MetaMask
Remix - Ethereu
React App
Address Error in
Work better to
MURAL
React App
localhost:3000
Vaccination
0x0c55....48bdac
Enter vaccine Id
Enter vaccine Name
Enter vaccine manufacture
Enter vaccine manufactu
Enter Batch No
Enter Quantity
Enter Customer Address
Add vaccine
Enter Drug Id
Get vaccine Details
29°C
Near record
Search
ENG
IN
11:25
25-10-2023


WhatsApp
MetaMask
Remix - Ethereu
React App
Address Error in
Work better to
MURAL
React App
localhost:3000
Vaccination
0x0c55....48bdac
2
co vaxine
Govt of india
2
2
2
0x742d35Cc6634C053292
Add vaccine
Enter Drug Id
Get vaccine Details
29°C
Partly sunny
Search
ENG
IN
11:27
25-10-2023


Vaccination
0x0c55....48bdac
2
co vaxine
Govt of india
2
2
2
0x742d35Cc6634C053292
Add vaccine
Enter Drug Id
Get vaccine Details


Vaccination
0x0c55....48bdac
2
co vaxine
Govt of india
2
2
2
0x742d35Cc6634C053292
Add vaccine
2
Get vaccine Details
co vaxine
Govt of india
2
2
2
0x742d35Cc6634C0532925a3b844Bc454e4438f44e

10. ADVANTAGES AND DISADVANTAGES

10.1 ADVANTAGES

Enhanced Vaccine Data Management: The project simplifies the management of vaccine-related data, providing a centralized platform for recording, tracking, and updating vaccine information. This streamlines the vaccine distribution process.

Real-Time Vaccine Information: Users have access to real-time data on vaccine availability, distribution status, and vaccine-related trends. This empowers them to make informed decisions promptly, contributing to more effective vaccine distribution.

Heightened Data Security: Leveraging blockchain technology and robust security measures, the system prioritizes data security and privacy. This significantly reduces the risk of data breaches, ensuring the integrity of vaccine data.

Transparency and Trust: Vaccine data recorded on the blockchain is transparent and tamper-proof, fostering trust among users, stakeholders, and the general public. This transparency is especially critical in vaccine distribution.

User-Friendly Interface: The user interface is designed to be intuitive and user-friendly, catering to individuals with varying levels of technical expertise. This accessibility ensures that a wide range of stakeholders can effectively use the system.

Compliance with Regulations: The system is built to comply with relevant data privacy and regulatory requirements. This ethical approach to data handling assures users that their information is being managed in a responsible and legal manner**.**

Scalability: The vaccine tracking transparency system is engineered to scale efficiently, accommodating an increasing number of users and vaccine records as the needs of vaccine distribution evolve. This adaptability ensures the system's continued effectiveness.

10.2 DISADVANTAGES

Implementation Complexity: Setting up and configuring the blockchain system, including smart contracts, can be technically complex and may require specialized expertise, potentially increasing the project's initial complexity and cost.

Blockchain Transaction Costs: Utilizing the Ethereum blockchain incurs transaction costs (gas fees), which can be a disadvantage for projects involving frequent or small transactions. This could affect the overall cost-effectiveness of the system.

Dependency on Blockchain Performance: The system's reliability is contingent on the performance and stability of the Ethereum blockchain. Network congestion, scalability issues, or other blockchain-related problems may impact the system's performance and responsiveness.

User Training Requirements: Users, including healthcare professionals and administrators, may need training to understand how to use digital wallets, interact with blockchain-based systems, and navigate the new technology. Training can be time-consuming and costly.

Data Validation Challenges: Ensuring the accuracy and consistency of data input can be challenging, requiring the implementation and maintenance of robust data validation mechanisms. Inaccurate data can compromise the system's trustworthiness.

Technical Support Demands: Technical issues and user queries related to blockchain and system operation may require responsive technical support. This can be resource-intensive, necessitating a dedicated support team or helpdesk.

Regulatory Adaptation: Changes in blockchain or healthcare regulations and standards may necessitate updates and adjustments to the system to remain compliant. Keeping up with evolving regulatory requirements can be challenging and may require ongoing legal and technical efforts.

11. CONCLUSION

In conclusion, the "Vaccine Tracking Transparency Blockchain" project represents a groundbreaking solution to address the critical imperative of ensuring transparency, trustworthiness, and efficiency in vaccine distribution. By harnessing the capabilities of blockchain technology, this project promises to revolutionize the way vaccines are monitored and administered, benefiting various stakeholders and the public health sector as a whole. The system's advantages are substantial, offering a streamlined approach to recording and verifying vaccine data, ensuring data integrity, and bolstering public confidence. Real-time access to vaccine information empowers users with critical insights into vaccine distribution, safety, and availability. It introduces a new era of trust and accountability in the vaccination process through secure and transparent data storage. However, the project is not without its complexities and challenges, including the intricacies of initial setup, Ethereum gas costs, and the need for user training. Nevertheless, with a well-structured technical foundation and a commitment to continuous development and support, these challenges can be effectively addressed. In the end, the "Vaccine Tracking Transparency Blockchain" project holds the promise of creating a safer and more secure future for public health. It has the potential to optimize vaccine distribution, reduce fraud, and enhance data-driven decision-making. As the system evolves, it is poised to be a beacon of progress in the healthcare sector, promoting a more reliable and efficient vaccination process for the benefit of healthcare providers, regulators, and the general public.

12. FUTURE SCOPE

Integration with IoT and Sensor Data: Expanding the project to integrate with Internet of Things (IoT) devices and sensors in healthcare facilities and storage units can provide real-time monitoring of vaccine conditions, ensuring temperature control and quality maintenance.

AI and Predictive Analytics: Implementing advanced data analytics, machine learning, and artificial intelligence can help predict vaccine demand, optimize distribution routes, and improve vaccination strategies based on historical and real-time data.

Global Vaccine Distribution: Extending the system to support international vaccine distribution efforts, enabling cross-border tracking and verification, and facilitating vaccine supply chain management on a global scale.

Vaccine Passport Integration: Integrating the system with digital vaccine passports to enable secure and verifiable proof of vaccination for international travel and access to various services.

Pharmaceutical Manufacturer Collaboration: Collaborating with pharmaceutical manufacturers to ensure end-to-end transparency in the vaccine production process, from ingredient sourcing to distribution, ensuring vaccine authenticity.

Research and Development Collaboration: Partnering with research institutions and public health organizations to share vaccine data for epidemiological research and the development of more effective vaccines.

Vaccine Inventory Management: Enhancing the system's features to include automated vaccine inventory management, with alerts for restocking and expirations, improving vaccine supply chain efficiency.

Patient Engagement: Developing patient-centric features such as vaccine appointment scheduling, reminders, and adverse event reporting through mobile applications, promoting proactive healthcare engagement.

Blockchain Interoperability: Exploring interoperability with other blockchain networks and healthcare systems to enhance data sharing capabilities and scalability.

Telemedicine Integration: Integrating telemedicine platforms to facilitate remote consultations and telehealth services related to vaccines, particularly for rural or underserved areas.

Decentralized Identity for Healthcare: Implementing decentralized identity solutions that give patients control over their healthcare data while ensuring compliance with healthcare regulations.

Environmental Impact Tracking: Extending the system to monitor and report the environmental impact of vaccine production and distribution, including energy consumption and waste management.

Collaboration with Healthcare Institutions: Partnering with healthcare institutions, hospitals, and clinics to ensure seamless integration and secure sharing of healthcare records and vaccination data.

Data Monetization for Researchers: Allowing users to monetize their vaccine and healthcare data by sharing it with researchers, contributing to medical research and public health efforts.

As the healthcare landscape evolves and embraces digital solutions, the "Vaccine Tracking Transparency Blockchain" project holds the potential to remain at the forefront of healthcare innovation. By adapting to emerging technologies and collaborating with healthcare stakeholders, the project can contribute significantly to the efficient, secure, and transparent management of vaccines and healthcare data, ultimately improving public health outcomes.

13. APPENDIX

SOURCE CODE

vaccination.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Vaccination {
    address public owner;

    constructor() {
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can perform this action");
        _;
    }

    struct Vaccine {
        string vaccineName;
        string manufacturer;
        uint256 manufacturingDate;
        string batchNumber;
        uint256 quantity;
        address customerAddress;
    }

    mapping(uint256 => Vaccine) public vaccines;
    uint256 public vaccineCount;
```

```
    event VaccineAdded(uint256 indexed vaccineId, string vaccineName, string manufacturer, uint256 manufacturingDate, string batchNumber, address customerAddress);

    function addVaccine(uint256 vaccineId, string memory _vaccineName, string memory _manufacturer, uint256 _manufacturingDate, string memory _batchNumber,uint256 _qty, address _customerAddress) external onlyOwner {


        vaccines[vaccineId] = Vaccine(_vaccineName, _manufacturer, _manufacturingDate, _batchNumber, _qty, _customerAddress);
        vaccineCount++;

        emit VaccineAdded(vaccineId, _vaccineName, _manufacturer, _manufacturingDate, _batchNumber, _customerAddress);
    }

    function getVaccineDetails(uint256 _vaccineId) external view returns (string memory, string memory, uint256, string memory,uint256, address) {

        Vaccine memory vaccine = vaccines[_vaccineId];
        return (vaccine.vaccineName, vaccine.manufacturer, vaccine.manufacturingDate, vaccine.batchNumber, vaccine.quantity, vaccine.customerAddress);
    }
}
```

connector.js

```
const { ethers } = require("ethers");

const abi = [
 {
  "inputs": [],
  "stateMutability": "nonpayable",
  "type": "constructor"
```

```
},
{
 "anonymous": false,
 "inputs": [
  {
   "indexed": true,
   "internalType": "uint256",
   "name": "vaccineId",
   "type": "uint256"
  },
  {
   "indexed": false,
   "internalType": "string",
   "name": "vaccineName",
   "type": "string"
  },
  {
   "indexed": false,
   "internalType": "string",
   "name": "manufacturer",
   "type": "string"
  },
  {
   "indexed": false,
   "internalType": "uint256",
   "name": "manufacturingDate",
   "type": "uint256"
  },
  {
   "indexed": false,
   "internalType": "string",
   "name": "batchNumber",
   "type": "string"
  },
```

```
  {
   "indexed": false,
   "internalType": "address",
   "name": "customerAddress",
   "type": "address"
  }
 ],
 "name": "VaccineAdded",
 "type": "event"
},
{
 "inputs": [
  {
   "internalType": "uint256",
   "name": "vaccineId",
   "type": "uint256"
  },
  {
   "internalType": "string",
   "name": "_vaccineName",
   "type": "string"
  },
  {
   "internalType": "string",
   "name": "_manufacturer",
   "type": "string"
  },
  {
   "internalType": "uint256",
   "name": "_manufacturingDate",
   "type": "uint256"
  },
  {
   "internalType": "string",
```

```
  "name": "_batchNumber",
  "type": "string"
 },
 {
  "internalType": "uint256",
  "name": "_qty",
  "type": "uint256"
 },
 {
  "internalType": "address",
  "name": "_customerAddress",
  "type": "address"
 }
],
"name": "addVaccine",
"outputs": [],
"stateMutability": "nonpayable",
"type": "function"
},
{
"inputs": [
 {
  "internalType": "uint256",
  "name": "_vaccineId",
  "type": "uint256"
 }
],
"name": "getVaccineDetails",
"outputs": [
 {
  "internalType": "string",
  "name": "",
  "type": "string"
 },
```

```
 {
  "internalType": "string",
  "name": "",
  "type": "string"
 },
 {
  "internalType": "uint256",
  "name": "",
  "type": "uint256"
 },
 {
  "internalType": "string",
  "name": "",
  "type": "string"
 },
 {
  "internalType": "uint256",
  "name": "",
  "type": "uint256"
 },
 {
  "internalType": "address",
  "name": "",
  "type": "address"
 }
],
"stateMutability": "view",
"type": "function"
},
{
"inputs": [],
"name": "owner",
"outputs": [
 {
```

```
  "internalType": "address",
  "name": "",
  "type": "address"
 }
],
"stateMutability": "view",
"type": "function"
},
{
"inputs": [],
"name": "vaccineCount",
"outputs": [
 {
  "internalType": "uint256",
  "name": "",
  "type": "uint256"
 }
],
"stateMutability": "view",
"type": "function"
},
{
"inputs": [
 {
  "internalType": "uint256",
  "name": "",
  "type": "uint256"
 }
],
"name": "vaccines",
"outputs": [
 {
  "internalType": "string",
  "name": "vaccineName",
```

```
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "manufacturer",
    "type": "string"
   },
   {
    "internalType": "uint256",
    "name": "manufacturingDate",
    "type": "uint256"
   },
   {
    "internalType": "string",
    "name": "batchNumber",
    "type": "string"
   },
   {
    "internalType": "uint256",
    "name": "quantity",
    "type": "uint256"
   },
   {
    "internalType": "address",
    "name": "customerAddress",
    "type": "address"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 }
]

if (!window.ethereum) {
```

```
 alert('Meta Mask Not Found')
 window.open("https://metamask.io/download/")
}

export const provider = new ethers.providers.Web3Provider(window.ethereum);
export const signer = provider.getSigner();
export const address = "0x3a0e6709e9131AE0dd86591318253E29f978C093"

export const contract = new ethers.Contract(address, abi, signer)
```

App.js

```
import './App.css';
import Home from './Page/Home'
function App() {
  return (
    <div className="App">
      <header className="App-header">
        <Home />
      </header>
    </div>
  );
}

export default App;
```

App.css

```
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}
```

```
@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

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();
```

Index.css

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

GitHub link:

https://github.com/Lohithpattu/Vaccine-Tracking-Transparent.git

Demo_video_link:

https://drive.google.com/file/d/17M1cXeadhvpaj4dGOxNBznfahmzDBzX7/view?usp=drivesdk