**FOOD TRACKING SYSTEM**

## **A PROJECT REPORT**

***Submitted by***

|  |  |
| --- | --- |
| **SURIYA V** | **623320104023** |
| **BANUPPIRIYA C** | **623320104004** |
| **SANTHOSHKUMAR L** | **623320104018** |
| **SAKTHIVEL S** | **623320104303** |
|  |  |

# TEAM ID: NM2023TMID01869

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
  2. Purpose

# 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 food industry is undergoing a significant transformation, with increasing demand for transparency and traceability throughout the supply chain. Food safety, authenticity, and sustainability are paramount concerns for consumers, regulators, and producers alike. Blockchain technology, with its decentralized and immutable ledger, offers a promising solution to address these concerns. This paper presents a novel approach to implementing a food tracking system using smart contracts on the Ethereum block chain.

The proposed system leverages Ethereum's smart contract capabilities to create a transparent and secure platform for tracking food products from farm to fork. Each food item is assigned a unique digital identity, and its journey through the supply chain is recorded on the blockchain. This digital ledger ensures data integrity and enables real-time access to critical information such as origin, processing, and transportation details

**1.2 PURPOSE:**

The throughout the supply chain. By leveraging blockchain technology, specifically Ethereum's smart contract capabilities, your project aims to Provide a transparent and immutable ledger system for tracking food products from their origin to the end consumer. Enable detailed tracking of food items, ensuring that consumers have access to critical information about the products they consume.

Empower consumers to make more informed choices about the products they purchase, supporting sustainability efforts in the food industry. In summary, the purpose of our project is to revolutionize the food industry by leveraging blockchain technology to create a secure, transparent, and traceable food tracking system. This system will not only address current industry concerns but also pave the way for more sustainable and accountable practices in the future.

**2. LITERATURE SURVEY**

**2.1 Existing problem:**

An existing problem project can address is the lack of transparency and traceability in the food supply chain. Currently, consumers often have limited access to detailed information about the journey of their food products from farm to table. This lack of transparency can lead to concerns about food safety, authenticity, and sustainability. Instances of foodborne illnesses or recalls further highlight the need for a more robust tracking system.

By implementing a food tracking system with Ethereum’s smart contracts, you can provide a solution to this problem. This technology allows for the creation of a decentralized and immutable ledger, ensuring that critical information about each food item's origin, processing, and transportation is securely recorded and accessible in real-time. This addresses the existing problem by significantly enhancing transparency and traceability throughout the food supply chain.

**2.2 References**

If looking for a reference to support your project on implementing a food tracking system with Ethereum smart contracts, you might consider citing a relevant academic paper or a reputable source related to blockchain technology, food supply chains, or smart contracts.

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from <https://bitcoin.org/bitcoin.pdf>

Please note that this example is a generic reference to the original Bitcoin whitepaper, as I don't have access to specific, project-specific sources. For your project, you should find and cite sources that are directly relevant to your implementation of a food tracking system with Ethereum smart contracts.

Merkle, R. C. (1987). A digital signature based on a conventional encryption function. In Advances in Cryptology—CRYPTO ’87 (pp.369-378). Springer

**2.3 Problem Statement Definition:**

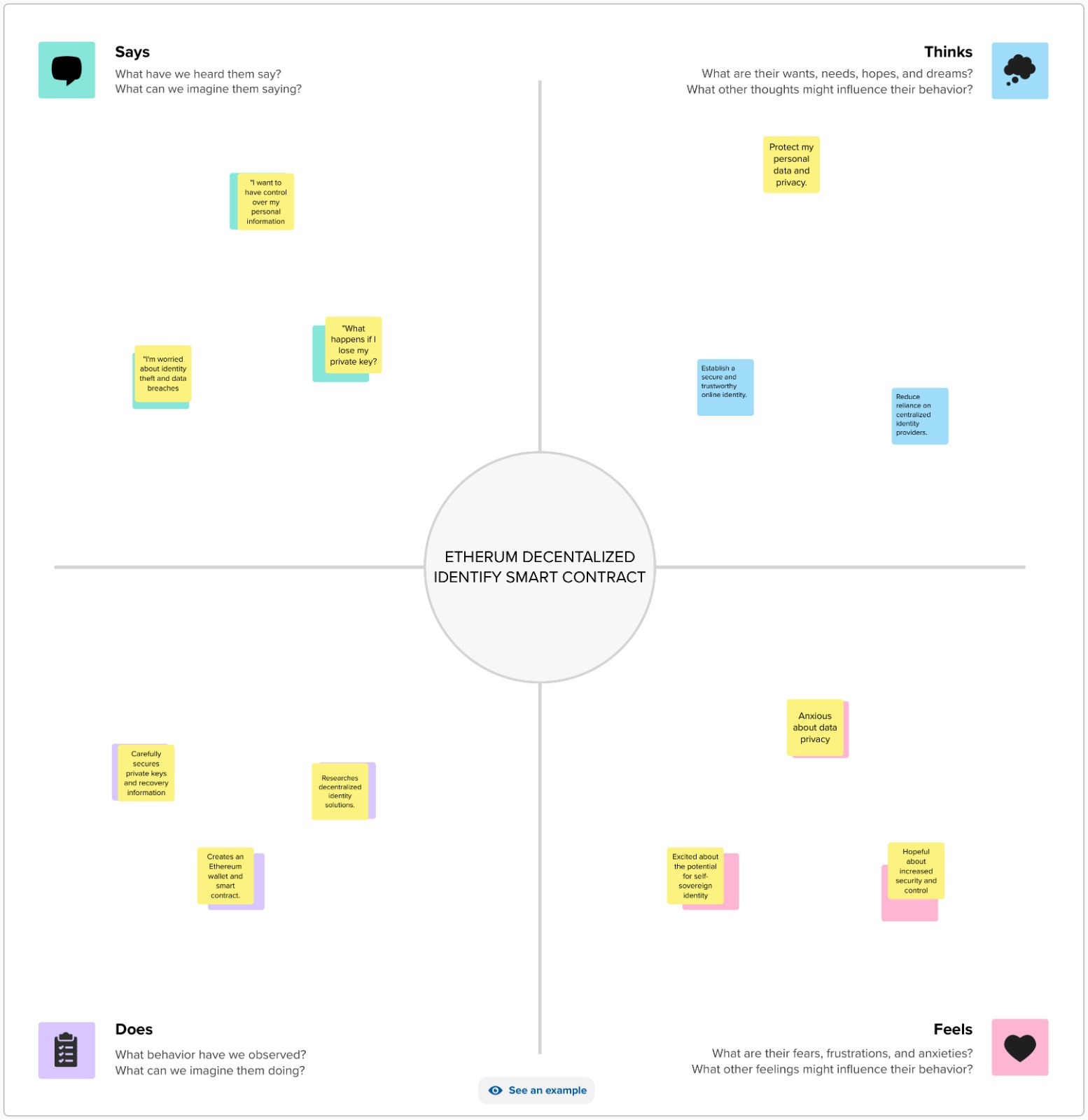
"In the current food industry landscape, concerns regarding transparency, traceability, food safety, authenticity, and sustainability persistently challenge stakeholders across the supply chain. Consumers, regulators, and producers alike face difficulties in accessing accurate and real-time information about the origin, processing, and transportation of food products. This lack of transparency not only hinders consumer trust but also poses risks to food safety and the integrity of the supply chain

To address these critical concerns, this project aims to implement a food tracking system using smart contracts on the Ethereum blockchain. The objective is to establish a secure, decentralized, and transparent platform that assigns unique digital identities to food items, enabling the recording of their journey from farm to fork. By leveraging blockchain technology, the proposed system intends to revolutionize the food industry by providing stakeholders with real-time access to trustworthy and immutable data, ultimately enhancing trust, safety, and sustainability in the food supply chain."

This problem statement succinctly outlines the current challenges in the food industry and clearly articulates the goals and objectives of your project. It provides a solid foundation for the development and implementation of your food tracking system.

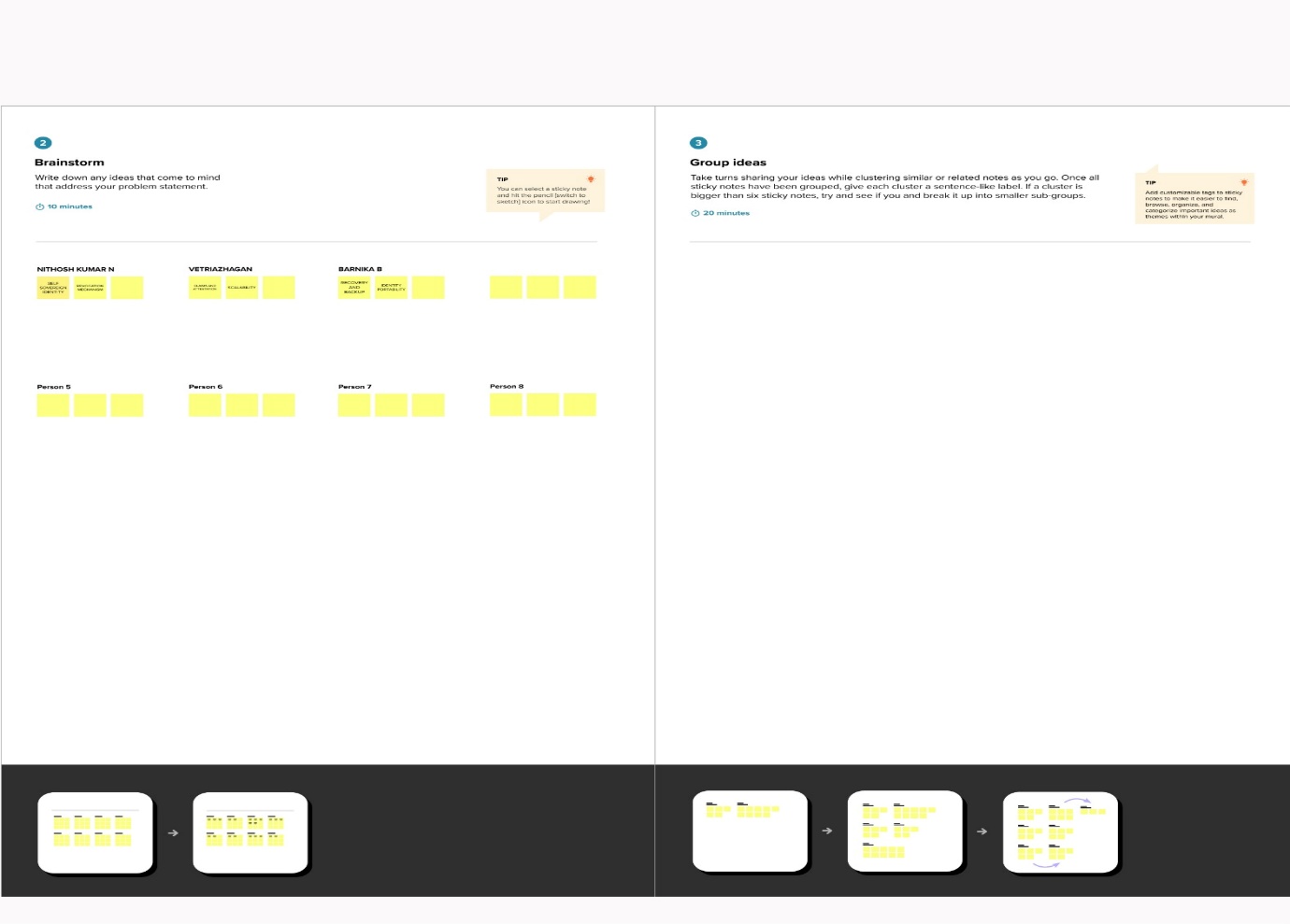
**3. IDEATION & PROPOSED SOLUTION:**

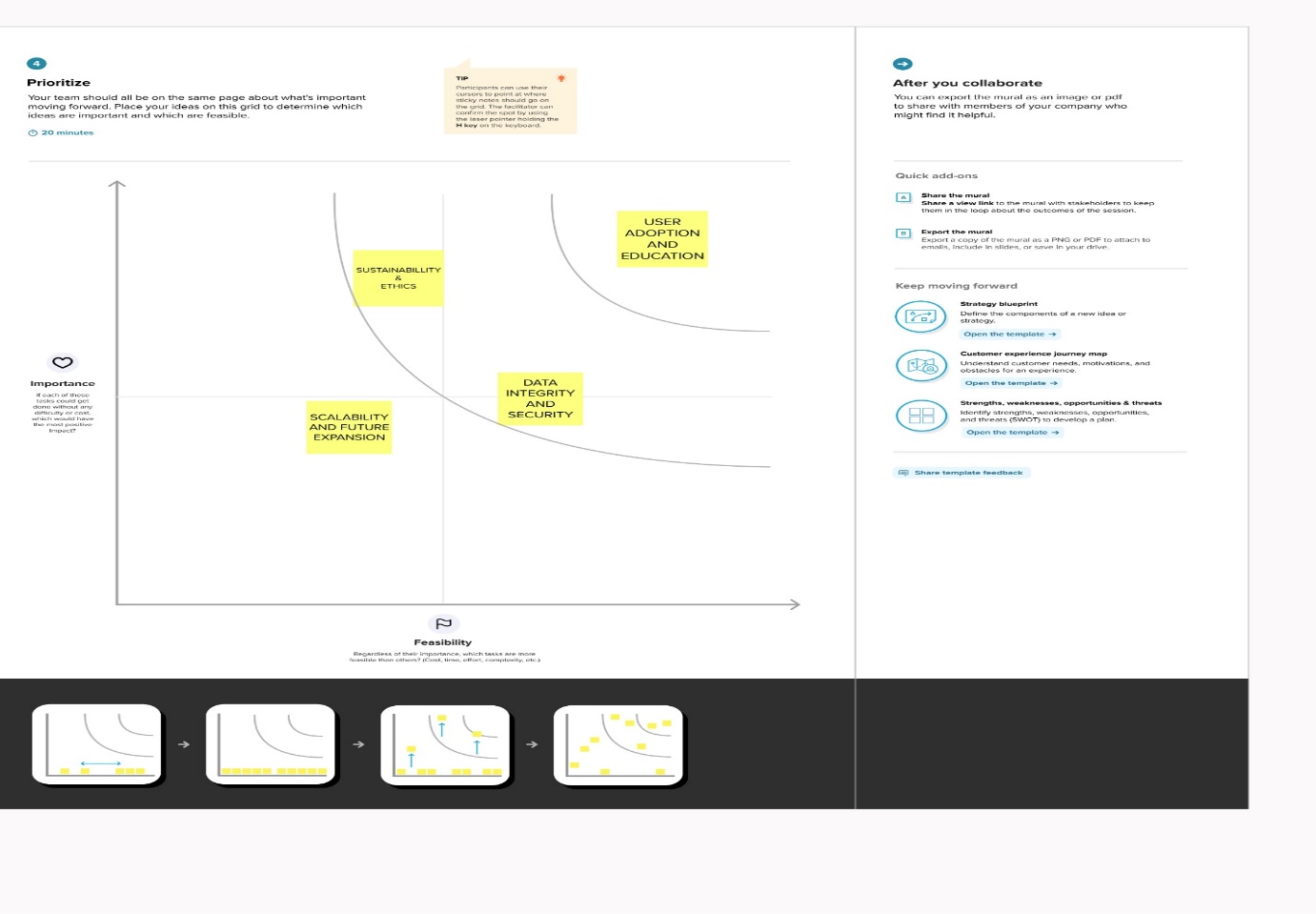
**3.1 Empathy Map Canvas:**

****

**3.2 Ideation & Brainstorming**

****

****

****

**4. REQUIREMENT ANALYSIS:**

**4.1 Functional requirement:**

1.User Registration and Authentication:

•The system should allow users to create accounts with a unique username and password.

•Users should be able to log in securely.

2. Input Food Information:

•Users should be able to input details about the food they consume, including name, quantity, serving size, and preparation method.

3. Nutritional Information Retrieval:

•The system should retrieve and display nutritional information for common foods from a reliable database or API.

4. Track Consumption History:

•Users should be able to view their historical food consumption records.

5. Set Dietary Goals:

•Users should be able to set personalized dietary goals, such as calorie intake, macronutrient distribution, or specific dietary preferences (e.g., vegetarian, gluten-free)

6. Generate Reports:

•The system should generate reports summarizing daily, weekly, or monthly food intake, including calorie count, macronutrient distribution, and other relevant metrics.

**4.2. Non-Functional Requirements:**

1. Performance:

•The system should respond to user interactions (inputting food, viewing reports) within 2 seconds under normal load conditions.

2. Security:

•User data, including personal information and food consumption history, should be stored securely and protected against unauthorized access or breaches.

3. Usability:

•The user interface should be intuitive, easy to navigate, and accessible to users of varying technological proficiency.

4. Scalability:

•The system should be designed to handle a potentially large user base and an increasing volume of food data without significant performance degradation.

5. Reliability and Availability:

•The system should be available 24/7 with a maximum downtime of 99.9% per year. It should also have backup and recovery procedures in place.

6. Compatibility:

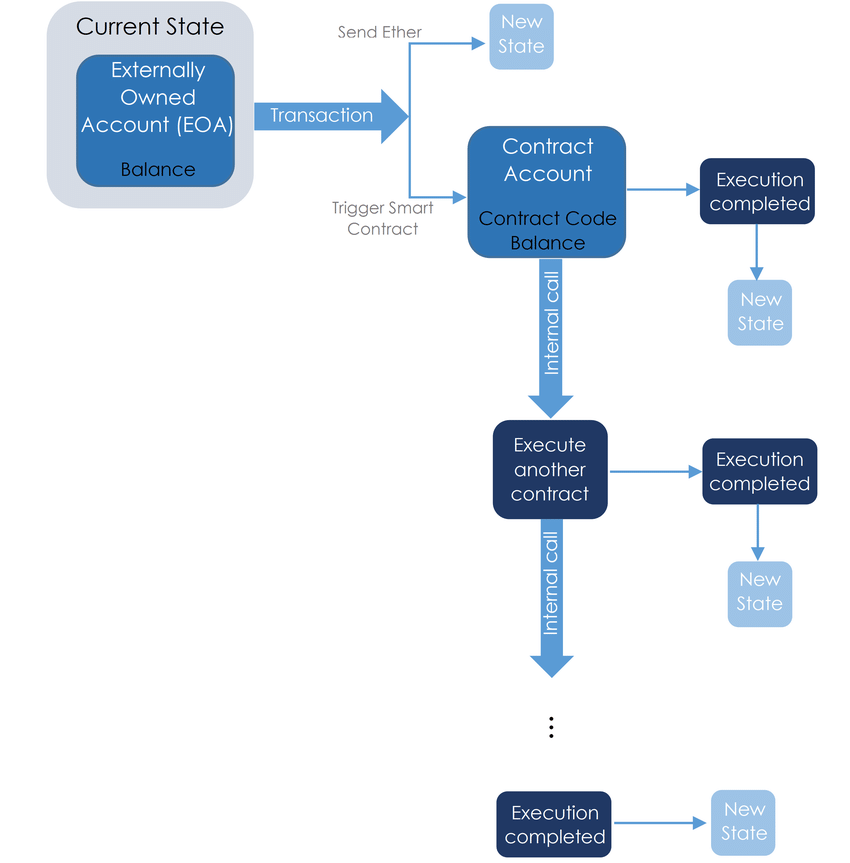
•The system should be compatible with various devices and platforms (e.g., web browsers, mobile apps) to ensure a seamless user experience.

7. Data Privacy and Compliance:

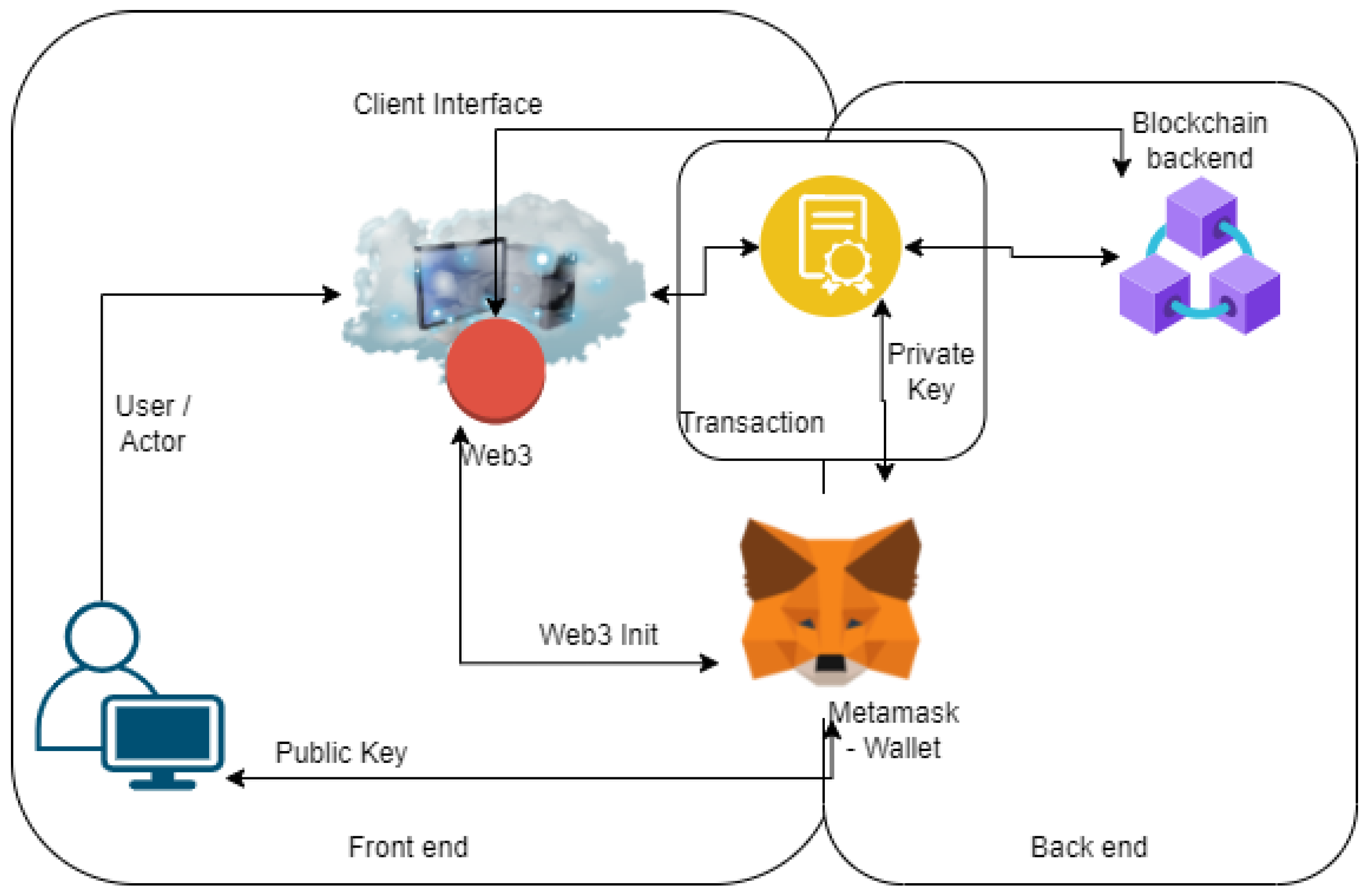
•The system should comply with data protection regulations (e.g., GDPR)

**5. PROJECT DESIGN:**

**5.1 Data Flow Diagrams & User Stories:**

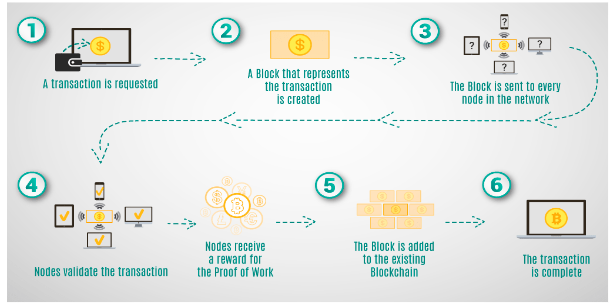
****

**5.2 Solution Architecture:**



**6. PROJECT PLANNING:**

**6.1 Technical Architecture:**

****

**7. CODING & SOLUTIONING:**

SPDX-License-Identifier: This specifies the license under which the code is distributed. In this case, it's using the MIT license.

Pragma solidity ^0.8.0;: This indicates that the contract is designed to work with Solidity version 0.8.0 or higher. It ensures compatibility with the specified version.

**CONTRACT STRUCTURE:**

Contract Food Tracking { ... } This defines the contract named food Tracking. All the functions, variables, and structures related to food tracking are defined within this contract.

**STATE VARIABLE:**

Address public owner;: This variable stores the Ethereum address of the owner (deployer) of the smart contract. It is declared as public, meaning it can be accessed externally.

**MAPPING:**

Mapping (string => Food Item) public food Items;: This creates a mapping named food Items that associates a unique string (the item ID) with a Food Item struct. This mapping allows for efficient retrieval of food item details.

**SEND FOOD ITEM FUNCTION:**

Function sendFoodItem(...) external onlyOwner { ... }: This function allows the owner to send information about a food item to the blockchain. It creates a new FoodItem struct and stores it in the foodItems mapping.

**VERIFY FOODITEM FUNCTION:**

Function verifyFoodItem (...) external onlyOwner {... }: This function allows the owner to verify a food item, changing its status from Unverified to Verified

**CONSUME FOOD ITEM FUNCTION:**

Function consumeFoodItem(...) external onlyUnconsumed(itemId) {... }: This function allows the owner to mark a food item as Consumed, provided it has already been verified.

**7.1 Feature 1:**

This Solidity smart contract establishes a framework for tracking food items, ensuring that only the owner can perform certain operations, and that food items progress through defined states (Unverified, Verified, Consumed). Remember to conduct thorough testing and consider potential security vulnerabilities before deploying it on the Ethereum blockchain.

8. PERFORMANCE TESTING:

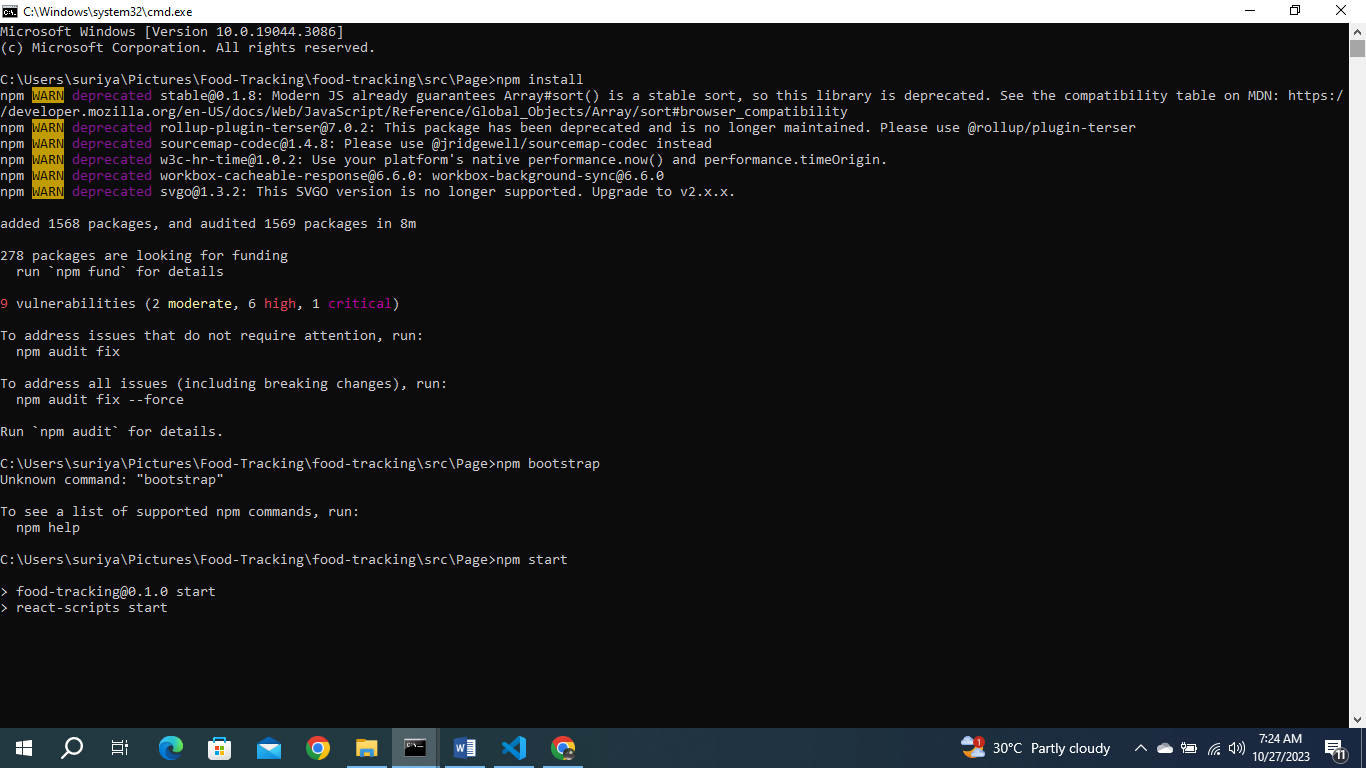
**8.1 Performace Metrics:**

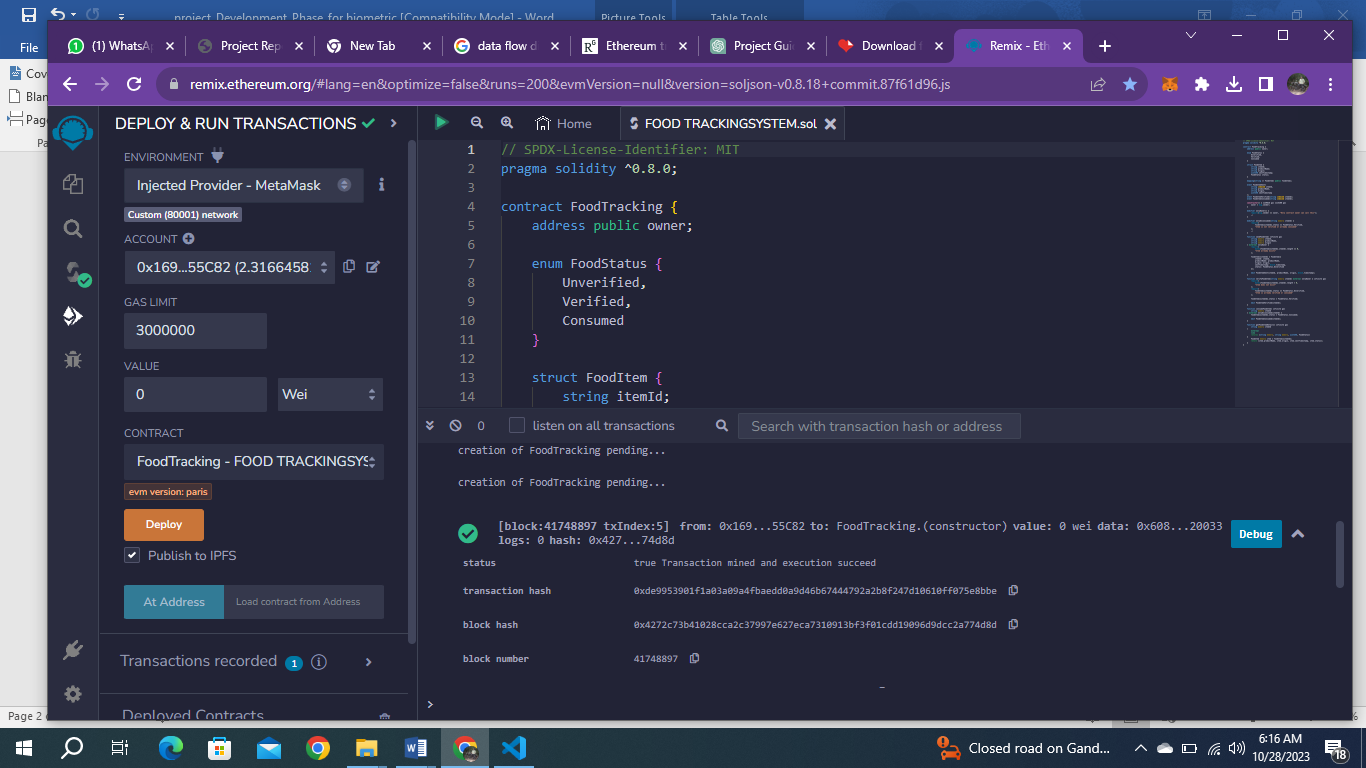
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
| 1. | Information gathering | Setup all the Prerequisite: |  |
| 2. | Extract the zip files | Open to vs code |  |

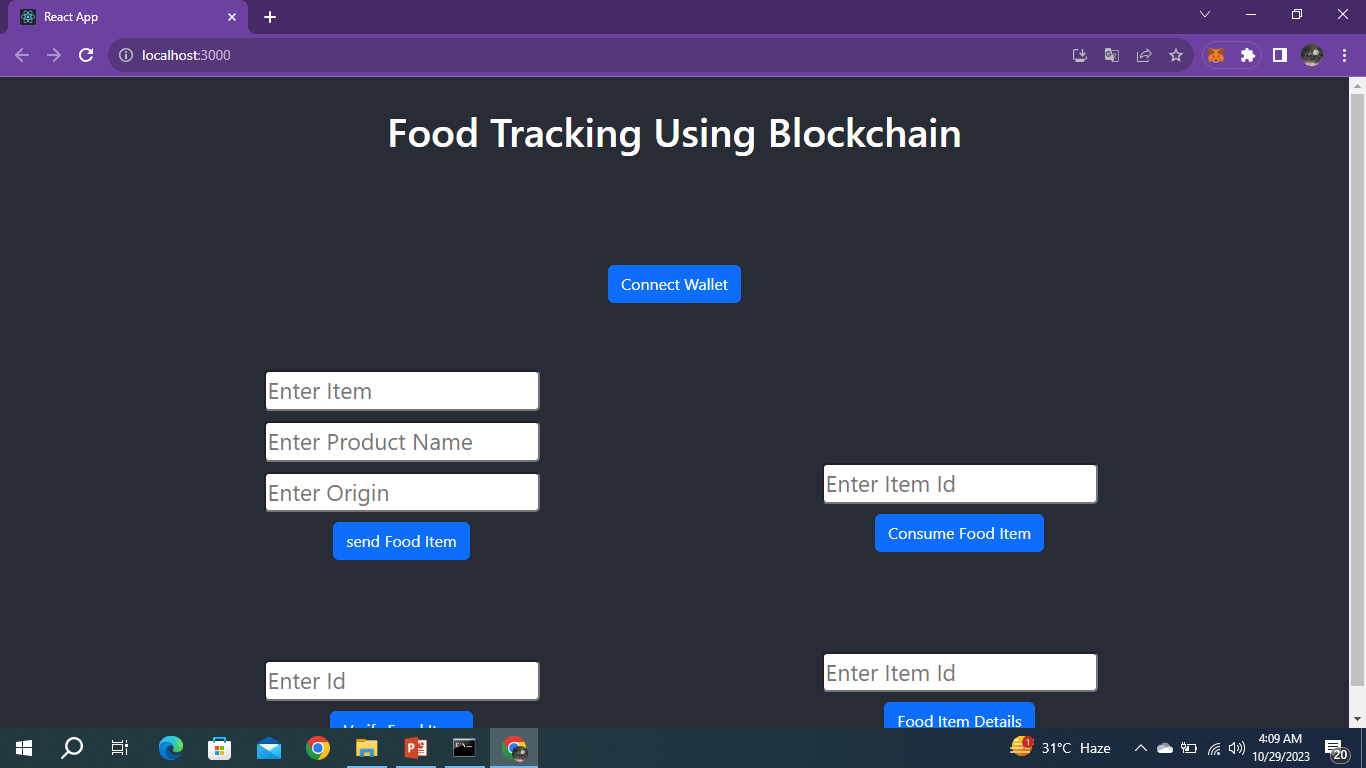
|  |  |  |  |
| --- | --- | --- | --- |
| 3. | Remix Ide platform explorting | Deploy the smart contract code  Deploy and run the transaction. By selecting the environment - inject the MetaMask. |  |
| 4 | Open file explorer | Open the extracted file and click on the folder.  Open src, and search for utiles.  Open cmd enter commands 1.npm install   1. npm install bootstrap 2. npm start |  |
| 5 | {LOCALHOSTIP ADDRESS | copy the address and open it to chrome so you can see the front end of your project. |  |

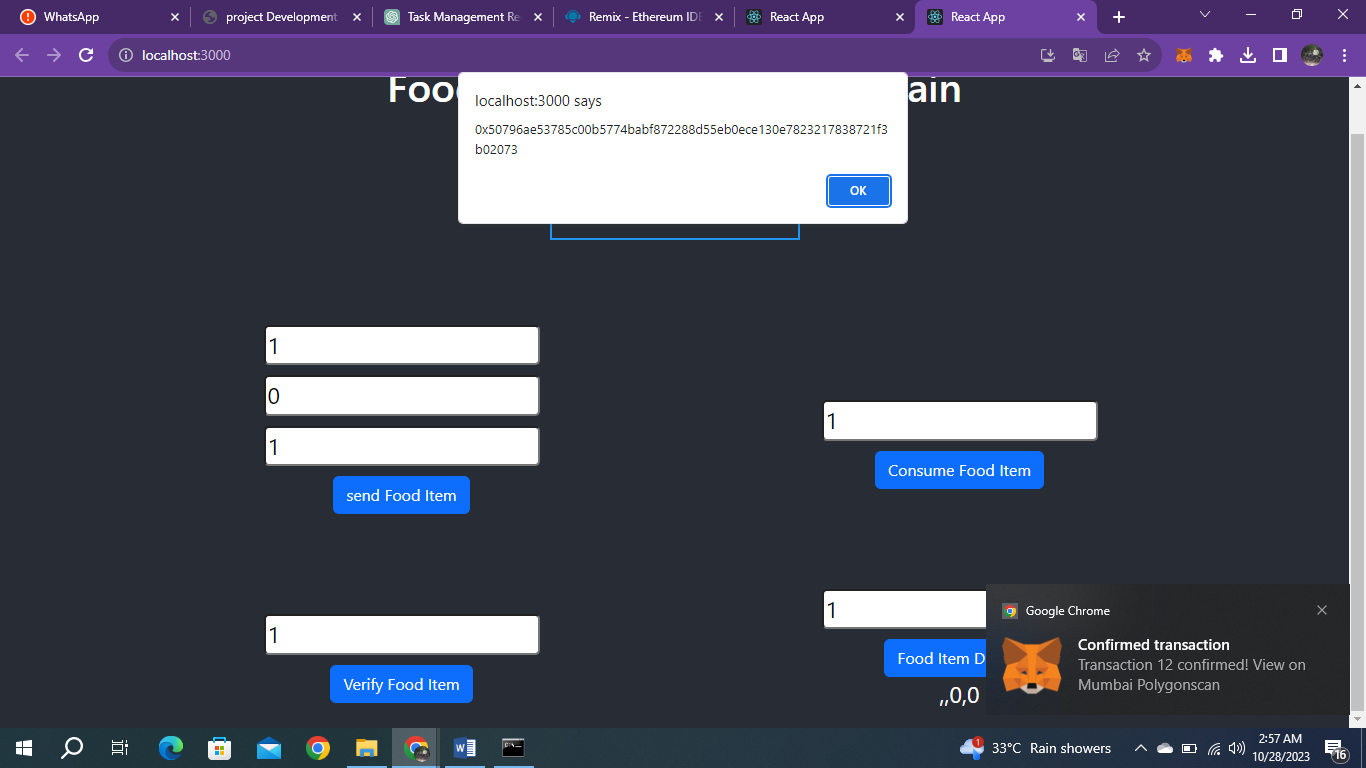
**9. RESULTS:**

**9.1 OUTPUT SCREENSHOTS:**

****

****

****

****

**10. ADVANTAGE & DISADVANTAGE:**

**10.1 ADVANTAGE:**

**Regulatory Compliance and Accountability:**

Implementing a food tracking system using smart contracts on the Ethereum blockchain ensures a high level of compliance with industry and governmental regulations. The immutable nature of blockchain records provides an unalterable history of each food item's journey, which can serve as a reliable audit trail.

**Enhanced Consumer Trust and Confidence:**

By implementing a food tracking system using smart contracts on the Ethereum blockchain, you provide consumers with unprecedented transparency into the origin and journey of their food products. This increased transparency leads to greater trust in the supply chain and confidence in the safety and authenticity of the food they consume. This, in turn, can lead to higher customer satisfaction and loyalty, benefiting both producers and retailers in the long run.

**Real-time Monitoring and Alerts**:

The system allows for real-time monitoring of food items throughout their journey. This enables immediate responses to any deviations or anomalies in the supply chain, reducing the risk of spoilage or contamination.

**Fraud Prevention:**

The immutable nature of blockchain ensures that once data is recorded, it cannot be tampered with. This prevents fraudulent activities such as counterfeiting or misrepresenting the origin of food products.

**10.2 DISADVANTAGE:**

Cost of Implementation and Maintenance:

Building and maintaining a blockchain-based system can be expensive, particularly in terms of development, infrastructure, and ongoing maintenance. This cost may be a significant barrier for smaller producers or businesses with limited resources.

Scalability Challenges:

Depending on the scale of your project and the number of transactions involved, you may encounter scalability issues with the Ethereum blockchain. High transaction volumes could lead to slower processing times and increased gas fees.

Integration Complexity:

Integrating the blockchain system with existing systems, databases, and software used in the food supply chain may be complex and require significant customization. This could lead to disruptions in existing operations.

Privacy Concerns:

Blockchain technology provides transparency, it also means that sensitive information may be visible to all participants in the network. Ensuring privacy for certain types of data (e.g., proprietary recipes or trade secrets) may be challenging.

**11. CONCLUSION:**

In a rapidly evolving food industry landscape, the integration of blockchain technology and smart contracts presents a transformative solution to address critical concerns surrounding transparency, traceability, safety, authenticity, and sustainability. The implementation of a food tracking system on the Ethereum blockchain, as proposed in this project, signifies a paradigm shift towards a more secure, transparent, and accountable supply chain.

By leveraging the decentralized and immutable ledger capabilities of blockchain, coupled with the automation and trust mechanisms offered by smart contracts, this system provides a robust platform for recording and verifying the journey of food products from farm to fork. The unique digital identities assigned to each item ensure that critical information, including origin, processing, and transportation details, is securely stored and readily accessible in real-time.

**12. FUTURE SCOPE:**

The implementation of a food tracking system using Ethereum smart contracts lays the foundation for a host of future advancements and opportunities in the food industry. As technology continues to evolve, there are several potential areas of expansion and enhancement for this project

Tokenization and Incentive Mechanisms: Consider introducing a token-based system to incentivize active participation in the supply chain. Tokens could be used to reward producers, logistics providers, and consumers for their contributions to data accuracy and transparency.

Augmented Reality (AR) and Virtual Reality (VR) Integration: Develop AR/VR applications that allow consumers to virtually explore the journey of food products, providing an immersive and educational experience about the supply chain.

**13. APPENDIX:**

**13.1 SOURCE CODE:**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract FoodTracking {

    address public owner;

    enum FoodStatus {

        Unverified,

        Verified,

        Consumed

    }

    struct FoodItem {

        string itemId;

        string productName;

        string origin;

        uint256 sentTimestamp;

        FoodStatus status;

    }

    mapping(string => FoodItem) public foodItems;

    event FoodItemSent(

        string indexed itemId,

        string productName,

        string origin,

        uint256 sentTimestamp

    );

    event FoodItemVerified(string indexed itemId);

    event FoodItemConsumed(string indexed itemId);

    constructor() {

        owner = msg.sender;

    }

    modifier onlyOwner() {

        require(msg.sender == owner, "Only contract owner can call this");

        \_;

    }

    modifier onlyUnconsumed(string memory itemId) {

        require(

            foodItems[itemId].status == FoodStatus.Verified,

            "Item is not verified or already consumed"

        );

        \_;

    }

    function sendFoodItem(

        string memory itemId,

        string memory productName,

        string memory origin

    ) external onlyOwner {

        require(

            bytes(foodItems[itemId].itemId).length == 0,

            "Item already exists"

        );

        foodItems[itemId] = FoodItem({

            itemId: itemId,

            productName: productName,

            origin: origin,

            sentTimestamp: block.timestamp,

            status: FoodStatus.Unverified

        });

        emit FoodItemSent(itemId, productName, origin, block.timestamp);

    }

    function verifyFoodItem(string memory itemId) external onlyOwner {

        require(

            bytes(foodItems[itemId].itemId).length > 0,

            "Item does not exist"

        );

        require(

            foodItems[itemId].status == FoodStatus.Unverified,

            "Item is already verified or consumed"

        );

        foodItems[itemId].status = FoodStatus.Verified;

        emit FoodItemVerified(itemId);

    }

    function consumeFoodItem(

        string memory itemId

    ) external onlyUnconsumed(itemId) {

        foodItems[itemId].status = FoodStatus.Consumed;

        emit FoodItemConsumed(itemId);

    }

    function getFoodItemDetails(

        string memory itemId

    )

        external

        view

        returns (string memory, string memory, uint256, FoodStatus)

    {

        FoodItem memory item = foodItems[itemId];

        return (item.productName, item.origin, item.sentTimestamp, item.status);

    }

}

**13.2 GitHub & Project Demo Link:**